ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΑΤΡΙΒΗ
για την απόκτηση Διδακτορικού Διπλώματος
του Τμήματος Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων

Ιωάννη Ηλιάδη

ADoCSI: ΕΝΑΛΛΑΚΤΙΚΗ ΜΕΘΟΔΟΣ ΔΙΑΧΥΣΗΣ ΤΗΣ ΠΛΗΡΟΦΟΡΙΑΣ ΚΑΤΑΣΤΑΣΗΣ ΠΙΣΤΟΠΟΙΗΤΙΚΩΝ

(ADoCSI: ALTERNATIVE DISSEMINATION OF CERTIFICATE STATUS INFORMATION)

Συμβουλευτική Επιτροπή:

Πρόεδρος:

Στέφανος Γκρίτζαλης
Αναπληρωτής Καθηγητής
Πανεπιστημίου Αιγαίου

Μέλη:

Σωκράτης Κάτσικας,
Καθηγητής Πανεπιστημίου
Πειραιώς

Κωνσταντίνος Λαμπρινουδάκης
Επίκουρος Καθηγητής
Πανεπιστημίου Αιγαίου

Εξεταστική Επιτροπή:

Πρόεδρος:

Στέφανος Γκρίτζαλης
Αναπληρωτής Καθηγητής
Πανεπιστημίου Αιγαίου

Μέλη:

Σωκράτης Κάτσικας,
Καθηγητής Πανεπιστημίου
Πειραιώς

Σπυρίδων Λυκοθανάσης,
Καθηγητής Πανεπιστημίου
Πατρών

Νικήτας Νικητάκος,
Καθηγητής Πανεπιστημίου
Αιγαίου

Κωνσταντίνος Λαμπρινουδάκης
Επίκουρος Καθηγητής
Πανεπιστημίου Αιγαίου

Γεώργιος Καμπουράκης,
Λέκτορας
Πανεπιστημίου Αιγαίου

Δημήτριος Λέκκας,
Λέκτορας
Πανεπιστημίου Αιγαίου

Table of Contents

List of Tables

List of Figures

ΕΥΧΑΡΙΣΤΙΕΣ

Σα βγεις στον πηγαιμό για την Ιθάκη,
να εύχεσαι να είναι μακρύς ο δρόμος,
γεμάτος περιπέτειες, γεμάτος γνώσεις.

Κωνσταντίνος Καβάφης, *Ιθάκη*, 1911

Το συγκεκριμένο ταξίδι, εκτός από περιπέτειες και γνώσεις, προσφέρει στον ταξιδιώτη διαφορετικές και ενδιαφέρουσες οπτικές γωνίες της καθημερινότητας του. Η ερευνητική δραστηριότητα φωτίζει με διαφορετικά χρώματα την κάθε πτυχή της καθημερινότητας του ερευνητή και αυτό ίσως να είναι ένα από τα μεγαλύτερα οφέλη του ταξιδιού.

Για το ταξίδι, και τις νέες αυτές οπτικές γωνίες, ευχαριστώ τους γονείς μου που με μεγάλωσαν σωστά, τους συνάδελφούς μου από την ευρύτερη ακαδημαϊκή κοινότητα που μου προσέφεραν απλόχερα τη βοήθειά τους όποτε την ζήτησα, και ιδιαιτέρως τα μέλη της Τριμελούς Συμβουλευτικής Επιτροπής μου και τον Πρόεδρό της καθώς χωρίς αυτούς ούτε αυτό το ταξίδι θα ήταν δυνατό αλλά ούτε και άλλα, μικρότερα, ταξίδια που έλαβαν χώρα στο παρελθόν.

Ένα τμήμα της παρούσας έρευνας έγινε στα πλαίσια ενός χρηματοδοτούμενου από την Ε.Ε. έργου (Γενική Διεύθυνση III, συμβόλαιο #ETD/99/502536: Study on the Scalability of Certificate Revocation and Certificate Suspension and Proposals for Enhancements on the Respective Mechanisms).

ACKNOWLEDGEMENTS

When you set out on your journey to Ithaca,
pray that the road is long,
full of adventure, full of knowledge.

Konstantinos Kavafis, *Ithaca*, 1911

This specific journey offers the traveler different views of his everyday life, besides offering adventures and knowledge. Research activity sheds lights of different colour to every aspect of the researcher's everyday life and this is probably one of the biggest profits of the research journey.

For the journey, and these new views of my life, I thank my parents for doing a good work raising me up, my colleagues in the academic area for generously offering me their help whenever I asked for it, and especially the members of my Consulting Committee and its Chairman, as without them this journey would not be possible, nor would other, smaller past journeys be possible as well.

Part of this work was funded by the European Commission (Directorate General III, contract #ETD/99/502536: "Study on the Scalability of Certificate Revocation and Certificate Suspension and Proposals for Enhancements on the Respective Mechanisms").

ΠΕΡΙΛΗΨΗ

Οι Υποδομές Δημοσίων Κλειδιών (ΥΔΚ) χαίρουν πλέον ευρείας διάδοσης και αποδοχής. Ενώ οι ΥΔΚ παρέχουν λύσεις σε αρκετά ζητήματα, την ίδια στιγμή εισάγουν ένα νέο πρόβλημα που πρέπει να επιλυθεί: αυτό της διαχείρισης του κύκλου ζωής των πιστοποιητικών. Σε αυτή τη διατριβή επικεντρωνόμαστε στην ανάκληση πιστοποιητικών και τον τρόπο με τον οποίο η Πληροφορία για την Κατάσταση των Πιστοποιητικών (ΠΚΠ) μεταδίδεται στις ενδιαφερόμενες οντότητες.

Αρκετοί μηχανισμοί ΠΚΠ έχουν ήδη προταθεί, καθένας εκ των οποίων βελτιώνει κάποια πλευρά της διαδικασίας διάχυσης της Πληροφορίας Κατάστασης Πιστοποιητικών. Αυτό είναι ωφέλιμο για την έρευνα, διότι με αυτόν τον τρόπο η έρευνα εξελίσσει έναν τεχνολογικό τομέα. Βήμα με βήμα, βελτίωση επί βελτίωσης, προτάσεις και αντιπροτάσεις, δοκιμή και αποτυχία. Παρ' όλα αυτά, φαίνεται ότι δεν υπάρχει ακόμα ένα ενιαίο πλαίσιο για τη συγκριτική αξιολόγηση (ποιοτική ή/και ποσοτική) των διαφόρων μηχανισμών ΠΚΠ που έχουν ήδη προταθεί στην επιστημονική βιβλιογραφία. Θεωρούμε ότι ένα τέτοιο πλαίσιο αξιολόγησης θα μπορούσε να αποδειχθεί χρήσιμο στο να προχωρήσει η έρευνα στον τομέα αυτό, ειδικά τώρα που διάφοροι τέτοιοι μηχανισμοί έχουν ήδη προταθεί. Ένα πλαίσιο αξιολόγησης θα μπορούσε να αποδειχθεί χρήσιμο και εκτός των πλαισίων της επιστημονικής έρευνας, προκειμένου να επιλέγονται οι κατάλληλοι μηχανισμοί ΠΚΠ για την υλοποίηση μίας Υποδομής Δημοσίου Κλειδιού ανάλογα με τις απαιτήσεις της κάθε περίπτωσης.

Ένα άλλο ζήτημα σχετικά με τους προταθέντες μηχανισμούς ΠΚΠ είναι ότι εστιάζουν στη βελτίωση της απόδοσης των μηχανισμών, στην έγκαιρη πληροφόρηση, στη μείωση των απαιτήσεων σε εύρος ζώνης και στην ικανοποίηση νομικών απαιτήσεων. Όμως, υπάρχει ένας παράγοντας

σχετικά με τη λειτουργία των ΥΔΚ που σχεδόν πάντα ξεχνούμε να λάβουμε υπόψη: ο τελικός χρήστης. Οι ΥΔΚ απευθύνονται σε πληθώρα χρηστών, αλλά ο μέσος χρήστης δεν έχει μεγάλη εξοικείωση με τεχνικά ζητήματα. Δεν πρέπει να περιμένουμε από τον τελικό χρήστη να κατανοεί τον τρόπο λειτουργίας ενός μηχανισμού ΠΚΠ ώστε να τον χρησιμοποιήσει αποτελεσματικά. Επίσης δεν πρέπει να περιμένουμε από τον τελικό χρήστη να κατανοεί την ανάγκη για την αναζήτηση, ανάκτηση και επαλήθευση ΠΚΠ και να ενεργεί με βάση αυτήν την ανάγκη.

Φαίνεται ότι αυτή τη στιγμή, ο πιο αδύναμος κρίκος στην αλυσίδα των ΥΔΚ είναι ο τελικός χρήστης, ο οποίος ενδέχεται να (ή ενδέχεται και να μην) χρησιμοποιήσει τους διαθέσιμους μηχανισμούς ΠΚΠ για να επαληθεύσει την εγκυρότητα κάποιας ψηφιακά υπογεγραμμένης πληροφορίας ή την εγκυρότητα των στοιχείων επαλήθευσης ταυτότητας που του αποστέλλει κάποια άλλη οντότητα. Η έρευνα επί των μηχανισμών ΠΚΠ πρέπει να εστιασθεί και στη βελτίωση αυτού του τομέα, δηλαδή της διαφάνειας των μηχανισμών ΠΚΠ όσον αφορά στον τελικό χρήστη.

Σε αυτήν τη διατριβή, παρουσιάζουμε μία συστηματική κατάταξη των μηχανισμών ΠΚΠ και ένα πλαίσιο αξιολόγησης αυτών. Αξιοποιούμε, επίσης, το προαναφερθέν πλαίσιο αξιολόγησης ώστε να παρουσιάσουμε και μία συγκριτική αξιολόγηση των μηχανισμών ΠΚΠ που έχουν προταθεί στην επιστημονική βιβλιογραφία.

Κατόπιν εστιάζουμε στο ζήτημα το οποίο οι περισσότεροι μηχανισμοί ΠΚΠ δεν επιλύουν: αυτό της διαφάνειας του μηχανισμού ΠΚΠ απέναντι στον τελικό χρήστη. Πιστεύουμε ότι δεν πρέπει να είναι απαραίτητο για ένα χρήστη να κατανοεί τη λειτουργία ενός μηχανισμού ΠΚΠ για να τον χρησιμοποιήσει, ούτε πρέπει να είναι απαραίτητο να έχει εκπαιδευθεί σε

θέματα Ασφάλειας ώστε να μπορεί να εκμεταλλευθεί και χρησιμοποιήσει τις ΥΔΚ.

Αναπτύσσουμε και παρουσιάζουμε ένα πρωτότυπο μηχανισμό ΠΚΠ, τον οποίο ονομάζουμε ADoCSI (Alternative Dissemination of Certificate Status Information). Ο εν λόγω μηχανισμός χρησιμοποιεί Πράκτορες Λογισμικού ώστε να διαθέσει την Πληροφορία Κατάστασης Πιστοποιητικού και επίσης χρησιμοποιεί ορισμένες από τις ιδιότητες και τη λειτουργικότητα που προσφέρουν οι άλλοι προταθέντες μηχανισμοί ΠΚΠ. Πιστεύουμε ότι ο μηχανισμός ADoCSI επιλύει ορισμένα από τα ζητήματα που εγείρονται από τη χρήση των υπόλοιπων μηχανισμών ΠΚΠ. Σε κάθε περίπτωση, αυξάνει το επίπεδο διαφάνειας του μηχανισμού ΠΚΠ απέναντι στον τελικό χρήστη, παρέχοντας μία λύση στο προαναφερθέν πρόβλημα του «πιο αδύναμου κρίκου», ο οποίος είναι ο τελικός χρήστης, και από τον οποίο δεν πρέπει να περιμένουμε ούτε ότι διαθέτει υψηλά επίπεδα τεχνογνωσίας και ούτε ότι είναι ευαισθητοποιημένος σε θέματα Ασφάλειας Πληροφοριών.

EXECUTIVE SUMMARY

PKI seems to be here to stay. PKI does provide solutions to quite many problems but at the same time it introduces a new problem to be solved: certificate lifecycle management. In this thesis, we focus on certificate revocation and the way that Certificate Status Information (CSI) is being disseminated to the appropriate stakeholders.

Quite many CSI mechanisms have been proposed already, each one attempting to improve some aspect or aspects of the CSI dissemination process. This is good for research, simply because this is how research moves on. Step after step, improvement over improvement, counter proposition over proposition, trial and error. However, there does not seem to exist a unified framework for the comparative evaluation (be it qualitative and/or quantitative) of the various CSI mechanisms already proposed in literature. We argue that such an evaluation framework could prove to be useful in further advancing research in the domain, especially now that many different CSI mechanisms have already appeared in the literature. Such an evaluation framework could also prove to be useful in real life scenarios (i.e. outside the research lab), when someone has to decide on the CSI mechanism to use, depending on the needs of the particular case.

Another issue with the proposed CSI mechanisms is that they focus on improving performance and timeliness of information, downsizing bandwidth requirements, meeting legal requirements. However, there is one actor in the PKI scene one almost always neglects to take into account: the end user. PKI addresses to the masses, but the average end user is probably not tech savvy. One should not expect the end user to comprehend the inner workings of the CSI mechanism in order to use it effectively. One should not

probably expect as well the end user to appreciate the need for locating, retrieving and verifying CSI and to act upon that.

It seems that right now the weakest link in the chain of PKI is the end user who may (or may not) use the available CSI mechanisms to verify some signed piece of information or verify the authentication data some entity provides. CSI research should also focus on improving this aspect, i.e. the transparency of CSI mechanisms.

In this thesis, we present a taxonomy of CSI mechanisms and an evaluation framework for them. We also use our evaluation framework in order to present a comparative evaluation of the CSI mechanisms proposed in the literature. We believe our evaluation framework can be of use in further researching CSI mechanisms.

We then focus on the issue that most CSI mechanisms tend to neglect: that of CSI mechanism transparency. A user should not have to comprehend the mechanics of CSI mechanisms in order to use them and should not also be highly trained regarding security to be able to operate in the PKI world.

We develop a prototype for a CSI dissemination mechanism, which we call Alternative Dissemination of Certificate Status Information (ADoCSI). This mechanism uses Software Agents in order to disseminate CSI, and also uses some of the properties and functionality offered by the other CSI mechanisms. We believe that ADoCSI addresses some of the issues that emerge from the use of the other Certificate Status Information dissemination mechanisms. It certainly increases the level of transparency, thus providing a solution to the aforementioned "weakest link" problem, being the dependent entity, which one should not expect to have high levels of information security awareness.

1 INTRODUCTION

1.1 Statement of the Problem

Public Key Infrastructures seem to be taking off. It may take them longer than initially expected, but it seems that they will eventually be established as the primary tool for entity and data authentication.

In some countries, where the private sector initiative is strong, PKI is taking off because small PKI islands are being created within organizations interoperating with PKI islands of other organizations to offer interorganisational security mechanisms. In other countries, the driving force behind PKI development seem to be e-government services being developed right from the start with PKI services built in.

In any case, PKI seems to be here to stay. PKI does provide solutions to quite many problems but at the same time introduces a new problem to be solved: certificate lifecycle management. In this thesis, we focus on certificate revocation and the way that Certificate Status Information (CSI) is being disseminated to the appropriate stakeholders.

Quite many CSI mechanisms have been proposed already, each one attempting to improve some aspect or aspects of the CSI dissemination process. This is good for research, simply because this is how research moves on. Step after step, improvement over improvement, counter proposition over proposition, correction and rollback. However, there does not seem to exist a unified framework for the comparative evaluation (be it qualitative and/or quantitative) of the various CSI mechanisms already proposed in literature. We argue that such an evaluation framework could prove to be useful in further advancing research in the domain, especially now that many different CSI mechanisms have already appeared in the literature. Such an evaluation framework could also prove to be useful in real life scenarios (i.e. outside the

research lab), when someone has to decide on the CSI mechanism to use, depending on the needs of the particular case.

Another issue with the proposed CSI mechanisms is that they focus on improving performance and timeliness of information, downsizing bandwidth requirements, meeting legal requirements. However, there is one actor in the PKI scene one almost always neglects to take into account: the end user. PKI addresses to the masses, therefore the average end user is probably not tech savvy or security savvy. One should not expect the end user to comprehend the inner workings of the CSI mechanism in order to use it effectively. One should not probably expect as well the end user to appreciate the need for locating, retrieving and verifying CSI and to act upon that.

It seems that right now the weakest link in the chain of PKI is the end user who may (or may not) use the available CSI mechanisms to verify some signed piece of information or authenticate another entity. CSI research should also focus on improving this aspect, i.e. the transparency of CSI mechanisms. The end user should not have to comprehend their inner workings of CSI in order to use them and also should not have to be that much sensitive or trained regarding security in order to decide to use some CSI mechanism. It has to be more transparent.

1.2 Motivation

The following areas can benefit from our work:

1. Academic research on CSI. Work has been already performed on evaluating CSI by others; however, each such research attempt is based on a few evaluation criteria covering only a few aspects of a CSI mechanism. Existing evaluation efforts focus mainly on the qualitative or quantitative evaluation of the scalability of CSI mechanisms. Our evaluation framework is much more complete and thus can be used by academia to

evaluate new CSI mechanisms against others and to aid in the development of new CSI mechanisms. Furthermore, the taxonomy of CSI mechanisms we present covers most (if not all, to the best of our knowledge) published CSI mechanisms, thus being a very good CSI mechanisms' reference point for academia.

2. Selecting a CSI to use in an actual PKI implementation. The industry could use our taxonomy and our CSI mechanisms' evaluation (and the respective framework) in order to select what is the best CSI mechanism on a PKI case-by-case basis, thus not being limited to the obvious choice of CRLs due to lack of knowledge of the other existing mechanisms.

3. A new CSI mechanism with new, unique characteristics. ADoCSI can be used in order to efficiently provide CSI to users (dependent entities) in highly distributed environments without assuming what most CSI mechanisms assume: (a) that the users are always online, (b) that the users (dependent entities) have some grasp of information security therefore they know why and how they are supposed to use a CSI mechanism, (c) that the users (dependent entities) will act responsibly regarding CSI while operating in the PKI world, i.e. they will retrieve and assess CSI whenever this is needed without fail.

1.3 Challenges

Ideally, research should be performed in the following way: first, basic research is conducted on a subject leading to a theoretical basis for that subject; the results of basic research are then used to move on to applied research and develop notions, mechanisms and ideas that could actually be put into practice in real life. Finally, industry (or even academia) comes along with implementations based on the results of applied research.

(Kohnfelder 1978) introduced the notion of a certificate; at the same time he provided insight as to the way dependent entities should be notified in case a certificate is no longer valid for some reason (revocation) based on the most widely known revocation mechanism of that time: the black list of banking cards (cards that have been stolen or lost, therefore they should not be accepted as valid). The theoretical properties of a CSI mechanism (and the evaluation criteria of such a mechanism) were not discussed in Kohnfelder's thesis, since the main contribution of the thesis was the notion of the *certificate*.

This has led to CSI dissemination mechanisms' research being more sporadic than it should be; researchers usually presented a new CSI mechanism of theirs and -often but not always- provided a comparative evaluation of their mechanism to one or two of the other CSI mechanisms based on a limited set of criteria, most of the times that criteria being scalability.

It has proven to be an interesting task to build a theoretical basis for CSI mechanisms a posteriori (after many CSI mechanisms had already been proposed), comprising of the basic properties of a CSI mechanism (taxonomy criteria) and the merits of a CSI mechanism (evaluation criteria).

Furthermore, using Agents to support the CSI dissemination process (as our mechanism, ADoCSI, does) did provide relatively easy solutions to some of the problems other CSI mechanisms faced; however, it seems to have introduced at the same time many more issues in the area of securing the mechanism itself, because a lot more players get introduced in the process of CSI dissemination with Agents that may or may not be trustful. It has been interesting and challenging to research these issues.

1.4 Contribution of the Thesis

During the recent years, many CSI mechanisms have been proposed; enhancements to these mechanisms and variations of these mechanisms are constantly emerging in literature as well. What seems to be missing is a systemic approach to CSI dissemination, a taxonomy. A taxonomy, identifying the building blocks of CSI mechanisms and classifying them would provide more solid CSI research foundations and would also probably foster better research results. Our first contribution is a taxonomy of CSI mechanisms.

What is also missing from current research on CSI mechanisms is a thorough evaluation framework that can be used in order to comparatively evaluate them. Attempts to quantitatively and qualitatively evaluate some CSI mechanisms, few at a time, have already been performed but they were based on specific properties of the few CSI mechanisms they were considering, i.e. the researchers did not provide an adequate abstraction of the evaluation framework so that it can be used on other CSI mechanisms as well. Perhaps the lack of a taxonomy was also a factor that led to such narrowed down evaluations. Our second contribution, presented in this thesis, is an evaluation framework for CSI mechanisms. We also use our evaluation framework in order to present a comparative evaluation of the CSI mechanisms proposed in the literature and point out their strengths and weaknesses. We believe our evaluation framework can be of use in further researching CSI mechanisms.

We then focus on the issue that most CSI mechanisms tend to neglect: that of CSI mechanism transparency. A user should not have to comprehend the inner workings of CSI mechanisms in order to use them and should not also be highly trained in security issues to be able to use PKI. We describe the prototype of a mechanism (ADoCSI: Alternative Dissemination of Certificate

Status Information) that attempts, among others, to increase the level of CSI mechanism transparency, thus minimising the requirements for security expertise and awareness on behalf of the average PKI user. This mechanism (ADoCSI) is our third contribution.

1.5 Thesis Outline

In Section 2 we survey CSI mechanisms proposed in literature and present a taxonomy for them. An earlier version of this section has been presented in (Iliadis, Ioannis, Spinellis, Diomidis, Katsikas, Sokratis et al. 2000). In Section 3 we present our evaluation framework for CSI mechanisms and also use the proposed evaluation framework to provide the reader with a comparative evaluation of the CSI mechanisms proposed in the literature. Most of the work presented in Section 3 has been published in (Iliadis, John, Spinellis, Diomidis, Katsikas, Sokratis et al. 2000) and (Iliadis, Gritzalis, Spinellis et al. 2003).

In Section 4 we present our proposal for a more user-transparent CSI mechanism based on Software Agents, along with a comparative evaluation of our mechanism to the already proposed ones. An earlier version of the work presented in Section 4 has been published in (Iliadis, Gritzalis and Gritzalis 2003) while some of the mechanisms for securing the operation of ADoCSI (Section 4.3.5) have been published in (Gritzalis, Moulinos, Iliadis et al. 2001), (Gritzalis and Iliadis 1998) and (Iliadis, Gritzalis and Oikonomou 1998).

We conclude our thesis by providing a summary of our CSI mechanisms' comparative evaluation findings, an outline of own proposal regarding CSI (ADoCSI) and finally presenting some of the future research work that we will carry out on ADoCSI.

2 SURVEY OF RELATED WORK

2.1 Introduction

The adoption of public-key cryptography as a basis for electronic commerce and other security-related information technology applications has brought to the surface a number of issues related to the deployment of a large public-key infrastructure (PKI). Public-key certificates — public keys signed by a trustworthy entity — are subject to subsequent revocation. Certificates can be revoked if e.g. a user's private key has been compromised, the authority that issued the certificate ceases to certify a given user, or the authority's certificate is compromised. Timely availability of correct revocation information is essential to build trust in digital signatures and applications built on digital signatures and PKIs. The validity of certificates (Certificate Status Information — CSI) must therefore be communicated by the *CSI provider,* the entity that is responsible for maintaining CSI (a certification authority or a party authorised to act on its behalf), to the *dependent entity,* the entity that relies on data in the certificate to make decisions.

The mechanisms used for providing CSI have important implications on the management, security, and performance of the PKI that relies on them. As public-key certificates enter the mainstream, they will be issued and used in large numbers by less sophisticated users, and be relied on for a large number of important transactions. For these reasons, a PKI must be based on a CSI distribution mechanism that provides flexibility and transparency in the management side, timeliness and scalability on the performance side, while guaranteeing high levels of trust and security. According to a MITRE Report (Berkovits S. 1994), CSI dissemination can be the most costly process in a PKI implementation.

CSI mechanisms have recently been proposed that provide for instant revocation of certificates (Beno Libert and Quisquater 2003; Boneh, Ding, Tsudik et al. 2001; Vanrenen, Smith and Marchesini 2006; Yang, Sakurai and Rhee 2006). To achieve this, they require an online authority to collaborate with the certificate holder whenever he has to use his private key. They do provide some solutions to confront with the problem of possible unavailability of the aforementioned online authorities (e.g. threshold cryptography) but the issue still remains that the certificate holder needs to communicate with these online authorities to use his private key, e.g. sign a piece of document. When a certificate needs to be revoked, the certificate holder notifies the respective authority and that authority simply does not collaborate anymore with the certificate holder in order to produce new signatures. According to these mechanisms, CSI is embedded in a signed piece of information because the certificate was assuredly valid at the time of signature. These mechanisms transpose the problem of the dependent entity, i.e. the fact that the dependent entity has to go online and retrieve CSI, to a problem of the certificate holder, who has to be online in order to use his private key. It is an interesting problem shift but it is outside the scope of this thesis, therefore we have not considered such mechanisms in our study.

In this section, we present a method for categorising Certificate Status Information mechanisms, depending on their elementary functionality. We present the elementary functions a Certificate Status Information mechanism comprises of, and provide a way to analyse the capabilities of a CSI mechanism according to these functions. This taxonomy can be used as a guide for selecting CSI mechanisms to be used in large-scale PKI deployment efforts.

2.2 Methodology

CSI mechanisms are operated by one or more entities. These entities undertake the task of identifying the need to generate new CSI, generating CSI and storing it in a way that makes it available to entities who depend on CSI (dependent entities). The latter are entities that use the status information during the process of certificate validation, in order to authenticate or authorise the respective certificate holder.

The elementary functions a **CSI provider** performs are the following:

1. Identification of the need for revoking a certificate. The CSI provider has to provide to external entities the capability of communicating revocation requests. When the CSI operator receives such information, he proceeds with verifying their validity and, if they are valid, generates the respective Certificate Status Information. Specific mechanisms and procedures have to be in place, both for the external entities to communicate revocation requests to the provider and for the provider to verify the validity of such requests.
2. Generation of Certificate Status Information. A CSI provider uses the information collected in the previous stage in order to generate CSI. The CSI format used in this stage might not be the one that will be used for CSI dissemination to the dependent entities. The purpose of generating CSI at this stage is not to disseminate it to the dependent entities, but to maintain a trusted repository of CSI. The characteristics of the CSI generation function include the frequency of CSI generation, and the format and size of the produced CSI.
3. Storage of CSI. This function relates to the mechanisms used by the CSI provider in order to protect the generated CSI, and render it readily available to the dependent entities in a format they can interpret. These

mechanisms may be part of the CSI mechanism itself; alternatively, external mechanisms could be used in order to support the protection and availability of the generated CSI.

On the **dependent entity** side, the elementary CSI functionality comprises of:

1. Location function. Dependent entities need trusted information in order to locate CSI regarding a certificate they wish to validate. Also, dependent entities need to know the mechanisms they have to use in order to communicate with the CSI provider at the specified location.
2. Retrieval of CSI. This function concerns the mechanisms dependent entities use in order to retrieve trusted copies of CSI they are interested in.
3. Validation of CSI and of the respective certificates. Dependent entities verify the integrity and authenticity of CSI they receive. In addition, they verify that the CSI they received contains information on the certificate they wish to validate. Finally, they validate a certificate, based on the retrieved CSI.

CSI mechanisms can be categorised in three major categories, depending on the kind of information they disseminate:

1. CSI mechanisms that communicate negative CSI, i.e. CSI that includes information on revoked certificates
2. CSI mechanisms that communicate positive CSI, i.e. CSI that includes information on certificates that are still valid
3. CSI mechanisms communicating complete CSI, i.e. CSI that specifies explicitly whether a certificate has been revoked or if it is still valid

2.3 Taxonomy of CSI mechanisms

2.3.1 Mechanisms supporting negative CSI

2.3.1.1 Certificate Revocation List

Certificate Revocation Lists were the first CSI mechanism to be standardised (International Organization for Standardization 1994), (International Organization for Standardization 1995), (International Organization for Standardization 1996), (Housley, Ford, Polk et al. 1999). A CRL consists of time stamped pointers to revoked certificates, which have not yet expired. The certificate identifier used in this list is the unique serial number assigned to the certificate by the Certification Service Provider that issued the certificate. A certificate that appears in a CRL is no longer considered valid. As the list is digitally signed by the CSP that issued the CRL, it can be retrieved from an untrusted CSI repository such as an LDAP server or a Web server. Each certificate contains a reference to the CSI repository, typically in the form of a URI. CRLs are issued by CAs periodically, for the dependent entities to know whether they have in their possession the latest revocation information available. Therefore, the dependent entities are not capable of possessing fresh revocation information on a need-to-know basis. Another problem this mechanism faces is that the CRL grows as certificates get revoked. This can lead to a very large CRL which will be difficult to communicate to dependent entities and be installed by them in their local storage media, which may have a restricted storage space.

In order to deal with the aforementioned problems, two solutions have been proposed (International Organization for Standardization 1994), (International Organization for Standardization 1995), (International Organization for Standardization 1996), (Housley, Ford et al. 1999): Distribution Points and Delta-CRLs. With Distribution Point CRLs it is

possible to partition a full CRL into several smaller CRL partitions using a variety of partitioning criteria (e.g., certificate serial numbers, certificate issuing dates, revocation reasons, etc.). The collection of all these CRLs includes all the entries the full CRL contains.

According to Distribution Points, each certificate contains the (optional) *cRLDistributionPoints* X.509v3 certificate extension. The extension point to a valid URI (*DistributionPointName*) from which a specific CRL can be downloaded; this CRL is the one that will contain the revocation information for that specific certificate, once it is revoked. It has been proposed (Housley, Ford et al. 1999) to use *cRLDistributionPoints* as a means to segment CRLs based on the revocations reasons, i.e. to have the *cRLDistributionPoints* extension point to different locations (CRLs) each of which will contain the serial numbers of certificates revoked for specific revocation reasons.

This practice could lead to a security compromise, if at the same point in time the revocation of a certificate could be perceived differently by an entity, depending on the reason it has been revoked (Cooper 1998). Suppose *cRLDistributionPoints* is used to partition CRLs based on revocation reasons and a CA takes the next –logical– step to decrease the issuance frequency of CRLs containing revoked certificates due to subjects' affiliations changes (*affiliationChanged)* to one month and not one week, which is the issuance frequency for the rest of its CRLs. In such a case, if a malicious entity compromises the private key of the certificate holder then he may attempt to ask from the CA a certificate revocation for the compromised certificate (private key) due to a change in affiliation lest the certificate holder realises the key compromise and requests a revocation because of key compromise (*keyCompromise).* In such a case, the malicious entity in possession of the private key will be able to continue using the revoked certificate for the rest of the month. The solution to this is for CAs to drop the use of the

revocationReasons altogether or at least use only the *unspecified revocationReason*, unless the reason for the revocation of a certificate has been thoroughly substantiated.

Delta-CRLs (International Organization for Standardization 1994), (International Organization for Standardization 1996), (Housley, Ford et al. 1999) are the second major variation on full CRLs. Delta-CRLs intend to address the problem of using up too much of the available network resources when communicating the CRL either as a whole, or even in parts through Distribution Points. Delta-CRLs provide the means for constructing incremental CRLs, partitioning CSI according to time criteria. Whenever new revocations have taken place, the (new) CRL that dependent entities have to retrieve contains only this new certificate revocation information. A Delta-CRL must contain the *baseCRLNumber* extension, so the dependent entities will know which CRL it is that a specific Delta-CRL complements. The respective CRL must contain the *DeltaCRLIndicator* extension, pointing to the Delta-CRL.

Berry et al (Berry and Gresty 2001) propose a different way to manage the large volume of the network traffic CRLs entail. They propose a CA should have a high priority CRL repository that charges for access to CRLs, providing a high-quality service level agreement (SLA) based service to those who really need it. This high priority CRL repository is supposed to replicate its contents to low priority CRL repositories that do not offer SLA based service but at the same time, do not charge for access to CRLs.

CRLs do not specify a mechanism for collecting revocation requests from the aforementioned external entities. CSI providers must provide such a mechanism for requesting revocation of a certificate or notifying the CSI provider (or the CA) of information that relates or could lead to the revocation of a certificate. The mechanism specified in (Adams C. 1999) could

be used for these reasons. However, it is not mandated by the CRL mechanism itself.

Entities external to the CSI provider can request revocation of a certificate. These entities inform the provider whenever they have reasons to believe a specific certificate needs to be revoked. These entities can be either the certificate holders themselves or other entities. The CSI provider (usually the CA that issued the certificate) has to validate these revocation requests and proceed with revocation of the specific certificate.

CRLs are issued periodically. In the time between the issuance of two consecutive CRLs, the CSI that has been collected and validated from submitted revocation requests is stored in a format and repository the CSI provider finds suitable. Dependent entities never receive CSI in this format, and never access directly the repository where CSI is stored in this format; therefore they do not need to know either how to interpret CSI in this format or how to access the aforementioned repository. The CSI provider uses the information in his internal, protected repository of CSI in order to generate a CRL.

Protection of the CSI, when it is stored in the non-standard format in the CSI provider's protected repository, must be ensured by the CSI provider using protection mechanisms he considers suitable. Protection of the CSI, when it is CRL-formatted, is achieved by the digital signature of the CSI provider on the CRL itself.

Table 1: Certificate Revocation Lists	
Collection of revocation requests	No mechanism specified; (Adams C. 1999) could be used.
Generation	CA generates primary CSI in proprietary format.

of CSI	This CSI is used for the periodic issuance of CSI to be disseminated in the form of CRLs.
Storage of CSI	CSI provider maintains repository of CRLs.
CSI location function	URIs pointing to CRLs contained in certificates.
CSI retrieval function	Format: CRL. Retrieval through LDAP(S)/HTTP(S) or other protocol mentioned in the URI included in the certificate that points to the CRL
CSI and certificate validation function	Signature of CA on CRL. CA identification information in CRL. Certificate serial numbers contained in CRL (negative CSI assertions).

2.3.1.2 Sliding window Delta-CRL

The idea behind Sliding Window Delta-CRLs, as suggested in (Cooper 2000), is to have shorter lived, overissued CRLs and shorter lived Delta CRLs having the same validity period as the CRLs themselves, the difference being that Delta CRLs will be referring to older CRLs, thus creating a sliding window. To clarify the mechanics of Sliding Window Delta-CRLs, the following table presents validity periods for CRLs and Delta CRLs according to the traditional method and the sliding window method.

Table 2: Sliding Window Delta-CRLs – Validity Periods

Time	**Traditional Delta-CRLs**		**Sliding Window Delta-CRLs**	
	Base CRL	*Delta CRL*	*Base CRL*	*Delta CRL*
…				
00:00	thisUpdate=00:00	thisUpdate=00:00	thisUpdate=00:00	thisUpdate=00:00

	nextUpdate=04:00 CRL Number=96	nextUpdate=00:10 BaseCRLNumber=96	nextUpdate=04:00 CRL Number=96	nextUpdate=00:10 BaseCRLNumber=5
00:10		thisUpdate=00:10 nextUpdate=00:20 BaseCRLNumber=96	thisUpdate=00:10 nextUpdate=04:10 CRL Number=97	thisUpdate=00:10 nextUpdate=00:20 BaseCRLNumber=6
…				
03:50		thisUpdate=03:50 nextUpdate=04:00 BaseCRLNumber=96	thisUpdate=03:50 nextUpdate=07:50 CRL Number=112	thisUpdate=03:50 nextUpdate=04:00 BaseCRLNumber=21
04:00	thisUpdate=04:00 nextUpdate=08:00 CRL Number=97	thisUpdate=04:00 nextUpdate=04:10 BaseCRLNumber=97	thisUpdate=04:00 nextUpdate=08:00 CRL Number=113	thisUpdate=04:00 nextUpdate=04:10 BaseCRLNumber=22

Table 3: Sliding Window Delta-CRLs	
Collection of revocation requests	No mechanism specified; (Adams C. 1999) could be used.
Generation of CSI	CA generates primary CSI in proprietary format. This CSI is used for the periodic issuance of CSI to be disseminated in the form of CRLs/DeltaCRLs.
Storage of CSI	CSI provider maintains repository of CRLs/DeltaCRLs.
CSI location	URIs pointing to CRLs/DeltaCRLs contained in certificates.

function	
CSI retrieval function	Format: CRL. Retrieval through LDAP(S)/HTTP(S) or other protocol mentioned in the URI included in the certificate that points to the CRL/DeltaCRL
CSI and certificate validation function	Signature of CA on CRL/DeltaCRL. CA identification information in CRL/DeltaCRL. Certificate serial numbers contained in CRL/DeltaCRL (negative CSI assertions).

2.3.1.3 Freshest CRL

Adams et al (Adams C. 1998) propose the use of Delta-CRLs, issued on top of partitioned CRLs (CRL Distribution Points). Their method uses two certificate extensions: the standardised *crlDistributionPoint* X.509v2 CRL extension and a custom extension, which they call *Freshest Revocation Information Pointer* (FRIP). The latter points to a CRL or Delta-CRL, which has as a base the partitioned CRL pointed at by the *crlDistributionPoint*. The aforementioned CRL or Delta-CRL, which is called Freshest CRL (FCRL), can be issued very frequently and not have a fixed update granularity. Therefore, the dependent entities that need very fresh CSI could get hold of the latter, at the expense of downloading another CRL. One of the advantages of this method is that the implementation requires no major changes in the mechanisms already used by the CAs. The *crlDistributionPoints* certificate extension is a standardised extension (International Organization for Standardization 1994), (International Organization for Standardization 1995), (International Organization for Standardization 1996) and the Freshest Revocation Information Pointer is a custom extension that could be added to the certificates issued by the specific CA.

This mechanism differs from the standardised CRL mechanism only in the location function. There is additional location information that enables

dependent entities to locate a Freshest CRL, i.e. a Delta-CRL that is issued very frequently and in non-predefined time periods on top of a Distribution Point CRL.

Table 4: Freshest CRL	
Collection of revocation requests	No mechanism specified; (Adams C. 1999) could be used.
Generation of CSI	CA generates primary CSI in proprietary format. This CSI is used for the periodic issuance of CSI to be disseminated in the form of CRLs/DeltaCRLs/FreshestCRLs
Storage of CSI	CSI provider maintains repository of CRLs/DeltaCRLs/Freshest CRLs
CSI location function	URIs pointing to CRLs/DeltaCRLs contained in certificates
CSI retrieval function	Format: CRL. Retrieval through LDAP(S)/HTTP(S) or other protocol mentioned in the URI included in the certificate that points to the CRL
CSI and certificate validation function	Signature of CA on CRL CA identification information in CRL/DeltaCRL/FreshestCRL Certificate serial numbers contained in CRL/DeltaCRL/FreshestCRL (negative CSI assertions)

2.3.1.4 Redirect CRL

Adams *et al.* (Adams C. 1998) suggest the use of another CRL custom extension, the Redirect Pointer. This could be used in combination with

Distribution Points to allow for dynamic re-partitioning of the CRL. If a CRL fragment contained in a Distribution Point grows to be unmanageably large, then the CSI for a subgroup of the certificates contained in that CRL fragment could be moved into another, possibly new, Distribution Point. Adams *et al*. propose to install a pointer as a custom CRL extension in the original CRL fragment that points to the new CRL fragment and specifies the scope of certificates covered by that new CRL fragment. This would ensure that the dependent entities will be able to continue receiving CSI without any disruption. The advantage this method presents is the dynamic re-partitioning of the CSI space, which can become a necessity if the devices used to store CSI (regarding a specific, restricted group of certificate holders) have a limited storage capability.

This mechanism differs from the standardised CRL mechanism only in the location function. In this mechanism, CSI location information is not contained in total inside the certificate itself. In addition, this mechanism introduces a minor change in the generation of CSI.

Table 5: Redirect CRL	
Collection of revocation requests	No mechanism specified; (Adams C. 1999) could be used.
Generation of CSI	CA generates primary CSI in proprietary format. This CSI is used for the periodic issuance of CSI to be disseminated in the form of CRLs. CSI space is partitioned dynamically.
Storage of CSI	CSI provider maintains repository of CRLs.

CSI location function	URIs pointing to CRLs contained in certificates. Certain CRLs may contain URIs to other CRLs (segmentation of CSI space).
CSI retrieval function	Format: CRL. Retrieval through LDAP(S)/HTTP(S) or other protocol mentioned in the URI included in the certificate that points to the CRL
CSI and certificate validation function	Signature of CA on CRL. CA identification information in CRL. Certificate serial numbers contained in CRL (negative CSI assertions).

2.3.1.5 Indirect CRL

An Indirect CRL combines CRL entries from multiple CSPs into a single CRL. These CRLs are implemented using three extensions: the *issuingDistributionPoint* extension (International Organization for Standardization 1994), (International Organization for Standardization 1995), (International Organization for Standardization 1996), (Housley, Ford et al. 1999) for a CRL to identify itself as an Indirect CRL, the corresponding certificate extension that indicates which CA issued the original CRL (if different from the CA that issues the Indirect CRL), and a *certificateIssuer* extension contained in each CRL entry, pointing to the CSP that issued the corresponding revoked certificate.

Table 6: Indirect CRL	
Collection of revocation requests	No mechanism specified; (Adams C. 1999) could be used.

Generation of CSI	CA generates primary CSI in proprietary format. This CSI is used for the periodic issuance of CSI to be disseminated in the form of CRLs. CRLs may contain CSI for more than CSP
Storage of CSI	CSI provider maintains repository of CRLs.
CSI location function	URIs pointing to CRLs contained in certificates. Certain CRLs may contain URIs to other CRLs (segmentation of CSI space).
CSI retrieval function	Format: CRL. Retrieval through LDAP(S)/HTTP(S) or other protocol mentioned in the URI included in the certificate that points to the CRL
CSI and certificate validation function	Signature of CA on CRL. CA identification information in CRL. Certificate serial numbers contained in CRL and the respective *certificateIssuer* extensions binding a certificate revocation to the CSP that issued the certificate (negative CSI assertions).

2.3.1.6 Enhanced CRL Distribution Points

Hallam *et al.* (Hallam-Baker and Ford 1998) had proposed a separation of the location function from the validation function, using the CRL extension *StatusReferrals*. This extension can be used to convey information regarding the newly issued CRLs. A CRL that contains *StatusReferrals* extensions does not contain certificate status information. Such a CRL is used in order to provide the dependent entities with information on the location of the CRLs they are interested in.

Hallam *et al.* (Hallam-Baker and Ford 1998) also proposed the use of the *cRLScope* extension as a mechanism for implementing the validation function. Once a dependent entity locates a CRL through the use of *StatusReferrals* extensions, that entity can decide whether the located CRLs contain the required CSI based on the information contained in the *cRLScope* extension. Multiple *PerCAScope* entries could be used in order to provide for Indirect CRLs, or even as another mechanism for implementing Redirect CRLs.

This mechanism can reduce the unneeded downloading of CRLs that have not been updated yet and enables the user as well to find out whether a CRL has been issued ahead of time or not, without actually downloading the CRL itself.

Table 7: Enhanced CRL Distribution Points	
Collection of revocation requests	No mechanism specified; (Adams C. 1999) could be used.
Generation of CSI	CA generates primary CSI in proprietary format. This CSI is used for the periodic issuance of CSI to be disseminated (CRL) CSI space is partitioned dynamically
Storage of CSI	CSI provider maintains repository of CRLs
CSI location function	URIs pointing to CRLs contained in certificates A CRL may contain only URIs (StatusReferrals) to other, newly issued CRLs
CSI retrieval	Format: CRL. Retrieval through LDAP(S)/HTTP(S) or other protocol mentioned in the URI included in the certificate that

function	points to the CRL
CSI and certificate validation function	Signature of CA on CRL CA identification information in CRL Verification of CSI contents within the CRL using the cRLScope extension Certificate serial numbers contained in CRL (negative CSI assertions)

2.3.1.7 Augmented CRL

Lakshminarayanan et al (Lakshminarayanan and Lim 2006) propose an alternative mechanism based on CRLs to provide both for locally stored complete revocation information, in the form of a CRL, but at the same time, surpassing the high network load of base CRL downloads and the disadvantage of having CSI stored in many different files, like in CRLs and Delta CRLs. Their mechanism is based on CRLs and a new construct they call Augmented CRLs. Augmented CRLs contain revocation updates only, and once downloaded they can be locally integrated with the already downloaded, respective, CRL to produce a fresh full CRL. In addition, multiple, sequential, ACRLs can be downloaded if a full CRL already exists locally referring to the older of the downloaded ACRLs and then the dependent entity can construct locally a full CRL.

Their mechanism is based on the fact that an ACRL is a Delta-CRL that includes the CRL issuer's digital signature over the base CRL; the base CRL is issued at the same time as the ACRL.

Table 8: Augmented CRLs	
Collection of revocation	No mechanism specified; (Adams C. 1999) could be used.

requests	
Generation of CSI	CA generates primary CSI in proprietary format. This CSI is used for the periodic issuance of CSI to be disseminated in the form of CRLs and ACRLs
Storage of CSI	CSI provider maintains repository of CRLs and ACRLs
CSI location function	URIs pointing to CRLs/ACRLs contained in certificates
CSI retrieval function	Format: CRL, ACRLs. Retrieval through LDAP(S)/HTTP(S) or other protocol mentioned in the URI included in the certificate that points to the CRL
CSI and certificate validation function	Signature of CA on CRL, ACRL. Combined signature of CA on resulting, combined CRL, constructed locally by dependent entity (the signature alone of the CA on the locally constructed Full CRL is already included in the ACRL itself) CA identification information in CRL, ACRL Certificate serial numbers contained in CRL, ACRL (negative CSI assertions)

2.3.1.8 Efficient Fault-Tolerant Certificate Revocation

Wright et al (Wright Rebecca N., Lincoln Pitman Pressatrick D. and K. 2000), (Millen and Wright 1998) introduce the use of depender graphs for CSI dissemination. They propose a method for constructing depender graphs consisting of k dependers for each certificate and disseminating CSI for that certificate to the latter k dependers, which in their turn disseminate CSI to their own k dependers and so on. This mechanism focuses on a distributed

and fault-tolerant CSI location function. Regarding the CSI generation function, the certificate holder signs with the to-be-revoked private key a revocation notice; this is the (negative) CSI this mechanism disseminates. The CSI validation function is apparent since the revocation notice has been signed with the to-be-revoked private key. This mechanism does not detail the way the other CSI functions should operate.

When a certificate is sent from its holder to a user, the user should ask the certificate holder to register him as a depender for that certificate. The certificate holder may either register him, or refer him to a server, where he can obtain CSI for this certificate, or even refer him to some other depender who will forward CSI to this user, once it is available.

This mechanism is k-resilient, i.e. it can survive even if a dependent entity cannot receive CSI from k-1 of the entities it has registered with as a depender for a specific certificate due to connectivity problems or because the later entities have colluded in order not to disseminate CSI. CSI contains negative only information.

The advantage of this mechanism is that it can be quickly adopted in PKI trust models resembling the PGP trust model; at the same time, the main drawback is that applying this mechanism to PKIs that incorporate other trust models might be cumbersome, since it requires from the certificate stakeholders to participate actively in CSI dissemination.

Table 9: Efficient Fault-Tolerant Certificate Revocation	
Collection of revocation requests	No mechanism specified.
Generatio	The certificate holder signs with the to-be-revoked private key a

n of CSI	revocation notice; CSI consists of this notice.
Storage of CSI	CSI is only stored by dependent entities, once it is received; it is not stored at a central location.
CSI location function	Dependent entities must locate CSI for a particular certificate through the depender graph. Dependent entities must participate in the depender graph (register as dependers for a particular certificate) before receiving CSI for this particular certificate
CSI retrieval function	No mechanism specified
CSI and certificate validation function	CSI is self-validating since it consists of negative-only revocation information for a specific certificate, signed with the private key corresponding to the certificate in question.

2.3.2 Mechanisms supporting positive CSI

All the methods described in this section specify positive CSI, i.e. the information contained in CSI specifies which certificates are valid.

2.3.2.1 Positive CSI (Suicide Notes)

Rivest (Rivest 1998) argues that CRLs are not needed at all in order to convey CSI. He claims that CRLs are probably the wrong mechanism to use for disseminating CSI because they contain negative statements instead of positive ones and because it is the issuer and not the dependent entity that sets the requirements on the freshness of the CSI.

According to Rivest's mechanism, it is the certificate holder who should revoke his own certificate, by signing with his own private, compromised, key

a "suicide note" (SN). There should be a network of "Suicide Bureaus" (SB), which gather suicide notes from every possible source, and either replicate the information they hold or have a means to refer queries to each other.

When a dependent entity needs CSI it can ask for fresh CSI from the certificate holder; the latter, in turn, should ask a SB for a "certificate of health", stating that 'no evidence has been received that the key has been lost or compromised' (Rivest 1998). The dependent entity could set in this case requirements on the freshness of the "certificate of health" provided by the SB to the certificate holder and by the latter to the dependent entity.

According to Rivest, the dependent entity should be able to revoke a certificate by itself e.g. in case the dependent entity is a service provider and it notices that there is more than one entity that holds that certificate and makes, possibly illegitimate, use of it. In order to enable the dependent entity to revoke a certificate of a certificate holder, the latter could be asked to sign a "suicide note" before having the right to use the service. Thus, the dependent entity could send the suicide note to SBs whenever the dependent entity believes that the certificate is being used illegitimately by more than one entity or has been compromised, without having to communicate with the certificate holder and without the latter having to produce the suicide note at that time.

Table 10: Positive CSI	
Collection of revocation requests	Suicide Notes (SNs).
Generation of CSI	CSI providers do not generate primary CSI (Suicide Notes). A certificate revocation request (Suicide Note) serves as primary

	CSI too. SNs are used in order to generate certificates of health, upon request.
Storage of CSI	Cache repository of certificates of health (provisional).
CSI location function	The certificate holder provides the dependent entity with CSI, after having contacted a Suicide Bureau.
CSI retrieval function	Format: Certificates of Health. Communication of Certificates of Health from the certificate holder to the dependent entity could be performed using a number of different mechanisms (e.g. the in-band communication mechanism between certificate holder and dependent entity), however this is not specified in (Rivest 1998).
CSI and certificate validation function	Signature of Suicide Bureau on Certificate of Health. SB identification information on Certificate of Health. Certificate of Health concerns and identifies only one certificate. Positive CSI assertion.

2.3.2.2 Self-Controlled Positive CSI

(Zhou, J. Y., Bao, F. and Deng, R. 2003) and (Zhou 2003) propose a CSI mechanism (we will be referring to it as Self-Controlled Positive CSI) where the certificate holder issues positive CSI on its own behalf. Their mechanism is based on the hash chain values(Lamport 1981).

According to this mechanism, there is no need for the dependent entity to contact a trusted authority, such as the CA or an OCSP responder in order to verify the validity of a certificate. When an entity requests a certificate, it finds

a pseudorandom value r, stores it locally and then sends to the Certificate Authority the following information:

PK_U, D, $H^j(r)$, j, L, where

- PK_u is the public key of the entity requesting the certificate,
- D is the starting validity date of the certificate,
- $H^j(r)$ is the j^{th} application of the hash function H() on the value r
- j=T/L, where T is the maximum lifetime of the certificate and L is the time period for refreshing validity of the certificate

When the certificate holder wants to authenticate himself or sign some piece of information, and the next refresh point is at time D_e, he also includes in the signed data he sends $H^i(r)$, where $i=j - (D_e-D)/L$. This way, the certificate holder provides verification himself to the dependent entity that the certificate was valid at the time he produced that signature or authenticated against the dependent entity.

The certificate holder must protect both his private key and the original random value r. The authors propose that value r can be a short string that the certificate holder can memorize or otherwise keep at an off-line storage; in any case he should not keep it along with his private key since if the mechanisms protecting his private key were violated, the intruder would then get access to both his private key and the random value r. Should the private key of the certificate holder be compromised, then he should simply not disclose any more hash values out of the hash chain.

Dependent entities do not need to communicate with the CA or any other authority for that matter, to receive CSI. Time-limited CSI is included in the signed piece of information itself or in the authentication data itself. The main downside is that the certificate holder has to be security sensitive enough, so as to protect the random value r, i.e. not stored in the along with the private

key, perhaps memorizing it, doing her that extra little work when signing or authenticating and also when choosing the random value r at the time of certificate request.

As a further improvement, (Zhou, J., Bao, F. and Deng, R. 2003) describe the use of forward-secure signatures (Abdalla and Reyzin 2000; Bellare and Miner 1999) in their mechanism. For each time period L, a different private key is being produced. A similar mechanism, lacking however, the forward secure signatures has been proposed by JingFeng et al (Li, Zhu, Pan et al. 2005).

The authors' mechanism ensures non-repudiation and an efficient way to disseminate CSI, assuming the certificate holder protects the confidentiality of random value r using a different (and probably stronger) mechanism to the one protecting his private key. Another way to protect the value r, as suggested by the authors, is to hand over this responsibility to another entity and having that entity reveal the value $H^i(r)$ for each time period i to the certificate holder; the authors call this entity Home Base. However, this increases the number of players that need to cooperate for a user to be able to digitally sign or authenticate against another entity.

The mechanism proposed by the authors is an improvement over previous mechanisms (Itkis and Reyzin 2002) attempting to incorporate CSI in the signed information, without the need for contacting a trusted third party to download CSI. In previous similar mechanisms (Itkis and Reyzin 2002), a loss of synchronization between the signer and the HomeBase could render the signer incapable of producing any more valid signatures.

This mechanism contains a vulnerability; the user may decide to withhold the current hash value for a certain period (claiming his key was compromised) and then begin distributing fresh, valid hash values again, claiming his key was not compromised after all, it was only assumed lost. The

user may be able to repudiate some of his actions during the "gray" period when his key was supposedly compromised.

Table 11: Self-Controlled Positive CSI	
Collection of revocation requests	Revocation is controlled by the certificate holder, i.e. by not issuing any more values of the hash chain, thus there is no need for collection of revocation requests.
Generation of CSI	Certificate holder produces the i*th* value of a hash chain, in order to prove to the dependent entity that the private key was not compromised at the time of signature
Storage of CSI	CSI is stored in two places: (a) the certificate itself that contains the j^{th} value of the hash chain and (b) each signature that contains a value of the hash chain that corresponds to that specific point in time
CSI location function	No mechanism required, since the certificate holder communicates CSI to the dependent entity
CSI retrieval function	Format: j-i element of the hash chain of value r (contained in the signed data or authentication data itself) , where i is the number of times the value r has been hashed before given to the CA and j is the serial number of the current validity time interval
CSI and certificate validation function	Hashing j times the value $H^{i-j}(r)$ received by the certificate holder and verifying whether that equals $H^{i}(r)$

2.3.2.3 Freshness-constrained Revocation Authority

Stubblebine (Stubblebine 1995) proposes a revocation mechanism where revocation can be definite, and where the repositories of revocation information need not be trusted. According to this mechanism, the role of the Certification Authority (CA) is separated from the role of the Revocation Authority (RevA). The CA issues long-term certificates, which contain freshness constraints on the CSI the dependent entities will use in order to validate the certificates. Such a certificate also contains a pointer to the RevA that is responsible for issuing CSI regarding the specific certificate. The RevA issues frequently time stamped certificates, which are used in order to provide dependent entities with positive assertions regarding the validity of the certificate. Dependent entities themselves impose their own CSI freshness requirements, when certificate holders use their certificates in order to authenticate themselves. Another level of CSI freshness constraints can be imposed if there are higher-level CAs (Policy CAs – PCA) that issue certificates for the lower-level CAs. Policy CAs may impose their own CSI freshness requirements on the end-entity certificates, contained in the certificates of the lower-level CAs.

When a certificate holder attempts to authenticate himself, the dependent entity will check the CSI freshness requirements imposed by the issuing CA. In addition, the dependent entity will impose its own freshness requirements; it will then locate the RevA that corresponds to the certificate of the authenticating entity and retrieve a short-lived RevA certificate that meets these freshness requirements. The combined CSI freshness requirements, the timestamp on the certificate issued by the RevA and the current time are the information the dependent entity will use in order to verify the validity of the certificate presented by the certificate holder.

This method allows for flexible balancing of the authentication costs and level of protection on a per transaction basis. Furthermore, CSI (the short-lived, time stamped certificate) need not be communicated from a trusted repository. The RevA can replicate the frequently issued, short-lived certificates to non-trusted repositories. Therefore, this method is efficient even when the network infrastructure is not reliable. Moreover, time stamped CSI is more flexible compared to CSI that contains expiration dates since the former can be used in environments with different CSI freshness requirements.

Table 12: Freshness-constrained Revocation Authority	
Collection of revocation requests	No mechanism specified. (Adams C. 1999) could be used.
Generation of CSI	RevA generates primary CSI in proprietary format. This CSI is used to generate short-lived RevA certificates.
Storage of CSI	No CSI is stored except for the primary CSI. Cache repository of short-lived RevA certificates (provisional).
CSI location function	URI pointing to RevA is contained in the certificate.
CSI retrieval function	Format: Short-lived RevA certificates, communicated from the RevA. Although no specific mechanism is described in the authors' mechanism, all mechanisms widely used for certificate dissemination could be used (e.g. http(s), ldap(s))
CSI and certificate	CSI (short-lived certificates) are signed by RevA, so dependent entities need to verify RevAs signature on them.

validation function	Each short-lived certificate provides CSI for a specific certificate only.

The aforementioned method allows the delegation of the revocation service to an authority other than the CA, but at the same without depending on the RevA to specify revocation policies. CSI freshness requirements are specified both from the hierarchy of the CAs and from the dependent entities themselves.

2.3.2.4 Efficient long-term validation of digital signatures

Ansper et al (Ansper, Buldas, Roos et al. 2001) present an efficient notary protocol that can be used both for notirising signed information and also for disseminating CSI. The Notary accepts queries concerning a specific certificate and a specific digital signature (already computed by the signer) and returns answers as to the validity of the certificate and digital signature in question at the specified time. The authors employ Merkle hash trees to construct CSI, using certificates and digital signatures as primary data to be inserted in the hash tree. Their mechanism can be used to disseminate CSI and ensure the long-term validity of a specific digital signature even after the corresponding certificate has been revoked. The authors' mechanism is supposed to retrieve primary CSI from the CA directly.

The details of the queries and answers of this mechanism are as follows (Ansper, Buldas et al. 2001):

$$1.\mathrm{A} \rightarrow \mathrm{B} : CertA, SigA[x]$$
$$2.B \rightarrow N : CertA, SigA[x]$$
$$3.N \rightarrow b : Valid(CertA), SigA[x], P(StatusA, S), SigN\{D(S)\}$$
$$where\ \ StatusA = Valid(CertA), SigA\{X\}$$
$$P(Si, S)\ is\ membership\ proof\ of\ the\ statement\ Si \in S$$
$$S\ is\ the\ set\ of\ data\ items\ contained\ in\ the\ Merkle\ hash\ tree$$
$$D(S)\ is\ the\ label\ of\ the\ root\ vertex, a\ digest\ itself\ of\ the\ \mathrm{sec}\ ond - level\ vertexes$$

Figure 1: Efficient long-term validation of digital signatures

Table 13: Efficient long-term validation of digital signatures	
Collection of revocation requests	No mechanism specified. Other entities (CA) are supposed to take care of this.
Generation of CSI	Primary CSI regarding certificate is supposed to be generated and made available by other entities. This mechanism consists of a Notary mechanism, and generates CSI for a specific digital signature in time.
Storage of CSI	Primary CSI is supposed to be stored by the CA. This mechanism stores CSI (in Merkle hash trees) for a specific digital signature in time
CSI location function	No mechanism specified; the certificate holder could forward CSI to the dependent entity.
CSI retrieval function	The dependent entity submits a certificate and a specific digital signature and receives a response validating that specific digital signature
CSI and certificate validation function	The dependent entity cannot validate itself the received CSI. The received CSI can only be (re)validated by the Notary server itself.

This mechanism produces a special kind of CSI regarding a specific certificate and a specific digital signature, using the corresponding private key. Therefore, CSI being produced by the Notary should probably be called SSI (Signature Status Information). (Lekkas, Gritzalis and Mitrou 2005)

present another SSI mechanism and they also present some solid –legal– arguments favouring the production of SSI instead of CSI.

2.3.3 Mechanisms supporting complete CSI

CSI mechanisms presented in the following sections give explicit information on the validity of certificates in question, i.e. CSI that specifies explicitly whether a certificate has been revoked or if it is still valid.

2.3.3.1 Certificate Revocation Status (Novomodo)

The use of X.509v2 CRLs results in a communication overhead mainly from the CRL repository to the dependent entities. The Certificate Revocation Status (CRS) mechanism (Micali 1996) and its enhanced version, Novomodo, (Micali 2002) is a revocation mechanism that attempts to address that.

In CRS, the CA has to include in every certificate two pseudorandom 100-bit values: YES (Y) and NO (N). Initially, the CA has to decide on the CSI update granularity and calculate the number of CSI updates it will perform for the certificate it is going to issue, within the certificate's validity period. The CA produces two random or pseudorandom numbers Y_0 and N_0. If the number of CSI updates that are going to be performed for that certificate is i, the CA calculates Y by applying a hash function F to Y_0 i consecutive times. N is derived from N_0 by applying F once to N_0. Therefore, the certificate contains the following two values, in addition to its usual contents:

4. $N=F(N_0)$

5. $Y=F^i(Y_0)$

The CA communicates with the CSI repository regularly (the update granularity is predefined and CA-specific), and sends the following data:

1. a list of all the serial numbers of certificates that have been issued and are not expired yet, signed by the CA

2. for each such certificate, the CA also sends a 100-bit value K, where $K=N_0$ if the certificate has been revoked and K=Fi-j(Y0), if the certificate has not been revoked; j represents the number of CSI updates that have been performed since the issuance of the certificate.

The entity requesting CSI from the CSI repository will retrieve K. That entity also retrieves Y,N from the certificate and calculates:

1. $F^j(K)=Y$
2. $F(K)=N$

If (1) applies then the certificate has not been revoked, while if (2) applies then the certificate has been revoked. If neither of these two applies, the dependent entity should request from the CSI repository the signed list of all the serial numbers of certificates that have been issued and are not expired yet. If the certificate in question is in that list, the dependent entity should conclude that the CSI repository has not sent him the correct number K which has been sent to the repository by the CA. Depending on the integrity and authentication mechanisms used for the communication between the dependent entity and the CSI repository, the dependent entity should draw its conclusions about the reason why neither of the aforementioned conditions (1) and (2) applied.

The main advantage of this mechanism is that it significantly reduces the communication costs between the CSI repository and the dependent entity. Furthermore, the advantage this mechanism presents over others is that complete (i.e. positive or negative, depending on the case) statements are employed and that the CSI repository does not have to be trusted by the dependent entity. However, in comparison to CRLs, communication between the CSP and the public repository increases.

An addition to this mechanism (Micali 2002) is to have the CA give also *full revocation certificates* to the CSI repository. These certificates could contain a revocation timestamp of the certificate and the revocation reason. If the dependent entities would like to have more information on the revocation of a specific certificate they could request for a *full revocation certificate* from the CSI repository.

One of the downsides of CRS is that the dependent entity needs to perform i iterations of a hash function, resulting in potentially large computational costs for certificate validity verification. (Elwailly, Gentry and Ramzan 2004) present an improvement over that, using QuasiModo trees. Their improvement results in decreasing the maximum hash function iterations a dependent entity has to perform to $\log_2(i)+1$.

Table 14: Certificate Revocation Status	
Collection of revocation requests	No mechanism specified.
Generation of CSI	CA generates primary CSI in proprietary format. This CSI is used for the periodic issuance of CSI to be disseminated (K values and CA-signed list).
Storage of CSI	Y_0, N_0 values and CA-signed list.
CSI location function	No mechanism specified.
CSI retrieval	Format: Value K and CA-signed list (if requested by dependent entity), and full revocation certificates (if requested by

function	dependent entity). No specific retrieval function has been specified by the authors, however, widely used protocols (e.g. http(s), ldap(s) could be used)
CSI and certificate validation function	Indirect Integrity/Authenticity mechanism for value K; if K does not return expected results, CA-signed list provides integrity/authenticity verification of K. Each value K concerns only a specific certificate. Full revocation certificate can be verified because it is CA-signed.

2.3.3.2 Certificate Revocation Trees

The Certificate Revocation Tree (CRT) (Kocher 1998) consists of a binary hash tree, where each leaf corresponds to a statement about a set of individual certificates. Each leaf specifies a lower and an upper boundary for certificates. A certificate has been revoked if its serial number appears as a lower-boundary in one of the tree's leaves. It is clear that it is not necessary to browse the complete tree in order to determine whether a certificate has been revoked. Note however, that it is computationally expensive to add a newly revoked certificate to the tree, as it requires the full restructuring of the binary hash tree. In any case, the root of the tree is digitally signed by the CA, in order to protect the integrity of the CRT.

(Naor and Nissim 1998) propose an enhancement to CRTs called Authenticated Dictionaries, replacing the binary tree with a 2-3 tree. Insertions and deletions, according to their CRT improvement, require only changing a path in the tree, whose size is logarithmic to the size of the tree.

The enhancements proposed by (Naor and Nissim 1998) have been further improved by (Munoz, Jose L., Forne, Jordi, Esparza, Oscar et al. 2004) and (Munoz, Forne, Esparza et al. 2005) in order to construct unbalanced trees, rather than balanced ones. Before constructing the unbalanced trees, the

frequency of already performed CSI requests is calculated and then the certificates for which CSI is most often requested are placed in leaves that have shorter paths to the tree's root.

Table 15: Certificate Revocation Trees	
Collection of revocation requests	No mechanism specified.
Generation of CSI	CA generates primary CSI in proprietary format. This CSI is used for the periodic issuance of CSI to be disseminated (CRTs).
Storage of CSI	CRT
CSI location function	No mechanism specified.
CSI retrieval function	Leafs of a tree and the necessary hash values up to the root of the chain, along with signed root of the tree
CSI and certificate validation function	Verification of hashes for tree leafs and verification of hash and signature on tree's root.

2.3.3.3 Online Certificate Status Protocol (OCSP)

The Online Certificate Status Protocol (OCSP) (Myers, Ankney, Malpani et al. 1999) is a mechanism proposed by the IETF PKIX Working Group that allows dependent entities to query for CSI in a timely fashion. OCSP could be

used in conjunction with CRLs. It provides for an X.509 extension that can be used as a pointer to a CRL, in case OCSP-based CSI is unavailable at a certain point of time.

The responses to CSI queries returned by OCSP are digitally signed. The authority that runs the OCSP service can either be the CA itself, another entity that is designated by the CA as a CSI provider (CA Designated Responder) or an entity trusted by the dependent entity to provide CSI (Trusted Responder). A CA Designated Responder must possess a specially marked certificate, issued by the CA, which authorises it to provide CSI to requestors. The requests for OCSP service can be signed by the requestors themselves.

OCSP Responders query for the status of certificates (possibly by direct interaction with the Certification Authority that issued the certificate in question, or by collecting CRLs from that CA), and return digitally signed (but possibly outdated) status responses to dependent parties in a more timely fashion than CRLs. Depending on the allocation of certificates and CAs to responders, multiple queries may be necessary to validate a certificate chain.

OCSP includes the CSI location function inside the certificate itself. CAs that support the use of OCSP for disseminating CSI should include in the certificates they issue the AuthorityInfoAccess extension (Housley, Ford et al. 1999) , as a pointer to the location of the authority that provides OCSP service for the specific certificate.

The possible OCSP responses are the following three:

1. "*good*", meaning that the certificate in question is not revoked. In the current version of OCSP (Myers, Ankney et al. 1999), it is mentioned that this response does not indicate that the certificate has ever been issued or that the OCSP response was produced within the validity interval of the certificate. Further CSI will be provided through the use of response

extensions, which have not yet been specified. Therefore, at its current status, OCSP provides only negative CSI, like CRLs. However, since OCSP could be enhanced (and it seems that the future work on OCSP is supposed to lead to this goal) in order to support *complete CSI* (definite positive or negative assertions regarding the status of a certificate), we are including OCSP in the category of CSI mechanisms offering *complete CSI* (Section 2.3.3).

2. "*revoked*", this indicates that the certificate has been revoked or has been suspended ('suspension' in OCSP terminology is equivalent to the *certificateHold* revocation reason code in CRLs).
3. "*unknown*", this response indicates that the OCSP service is not aware of the certificate in question.

Dependent entities that request CSI from an OCSP service provider must be able to check the revocation status of the certificate of the service provider himself. According to (Myers, Ankney et al. 1999) CAs may choose to provide that functionality in the following three ways:

1. The CA could issue only short-lived certificates for OCSPs in order to avoid having them revoked,
2. the CA may choose to specify an extension in the certificate of the OCSP service provider that points to a CRL, or
3. the CA could choose not to specify any method for OCSP certificate status verification.

Dependent entities use the hash of the CA name (*issuerNameHash*) that issued a certificate for an entity, the hash of the public key contained in CA's certificate (*issuerKeyHash*) and the certificate's in question serial number (*serialNumber*) in order to form a CSI query addressed to the OCSP service provider. Using hashes, the amount of information communicated is less and

at the same time there is only a negligible chance of two sets of identification information (the hashes we mentioned above) to collide, if the hash function has a sufficiently large range and is collision-resistant (Bellare and Rogaway 1997). Using hashes, only an entity that already holds the certificate in question can create the appropriate hashes and request for CSI from the OCSP service provider.

Younggyo etc al (YoungGyo Lee 2006) propose an enhancement to OCSP, to minimize the potential impact of certificate holder's private key compromise, i.e. the potential impact before CSI is made available. They propose a key insulated signature scheme, where part of the CSI is computed by the signer itself and transmitted along with the signed information and the other part of the CSI is left for the dependent entity to retrieve for from the CA. Their mechanism, further enhances the one proposed by Dodis et al (Yevgeniy, Jonathan, Shouhuai et al. 2003). The user produces a random value r and computes $H^i(r)$; then the user sends to the CA the value $H^i(r)$ on his public key and the CA signs them both. The CA also sets the counter C_{before} to the value zero and stores it. Each time a user sends a signed piece of information or authenticates himself against another entity, he also sends the value $H^{i-j}(r)$, where j is the number of times it has already used his private key. The dependent entity performs an OCSP request to the CA, along with the value $H^{i-j}(r)$. The CA checks the certificates status, including the check $H^{i-j}(r)=C_{now}$ and then increments C_{now} by one. Clearly, if the user's private key is compromised, the malicious entity cannot use a private key, unless they also know that random value r. However, if the user protects the random value r in the same manner he protects his private key, then a compromise of the private key leads to a sense consequences.

Finally, (Koga, Ryou and Sakurai 2004) further enhance OCSP, proposing a way to do pre-production of OCSP Responses, as a means to enable OCSP to deal with Denial of Service attacks and OCSP response replay attacks.

Table 16: Online Certificate Status Protocol	
Collection of revocation requests	No mechanism specified.
Generation of CSI	CA generates primary CSI in proprietary format. This CSI is used by the OCSP service provider (direct access or replicated) to generate OCSP Responses, upon request.
Storage of CSI	No CSI is stored (except for the primary CSI, if this is replicated locally to the OCSP service provider).
CSI location function	URI pointing to OCSP service provider is contained in certificate; if OCSP-based CSI is not available, OCSP Response points to a CRL.
CSI retrieval function	Format: OCSP Queries (accompanied with hash chain value proposed by (YoungGyo Lee 2006) and OCSP Responses. Authentication of dependent entities (provisional).
CSI and certificate validation function	OCSP Responses signed by OCSP service provider. OCSP Responses contain CSI for a specific certificate only. OCSP do not contain complete CSI.

2.3.3.4 Certificate Re-issuance

Fox et al (Fox and LaMacchia 1999) claim that there is no need to develop new protocols and messages for online CSI since the existing mechanisms for requesting and producing certificates are adequate for the job. They propose

that the dependent entity, upon receiving some signed piece of information, forward the certificate in question to the CA and ask of her to issue a new certificate for the certificate holder and present it to the dependent entity, in case the certificate has not been revoked. According to their mechanism, the response by the CA should include the following:

1. a certificate corresponding to the certificate holder, following the same syntax as the original certificate
2. an X.509 extension (*ResponseSubject)* that points to the original certificate, containing the key identifier, certificate identifier, certificate issuer and certificate serial number
3. another X.509 extension (*ResponseStatus)* containing positive or negative assertions regarding the validity of the certificate.

The validity of the newly issued certificate should be narrow, i.e. a day or so.

Yum et al (Yum, Kang and Lee 2003) further enhance the Certificate Re-issuance CSI mechanism; they call their mechanism the Advanced Certificate Status Protocol (ACSP). According to ACSP, the dependent entity again accepts short-lived certificates as CSI, but they have managed to reduce the number of fresh certificates the CA must produce. The dependent entity first has to set for herself a recency requirement t_{max} regarding CSI. If, upon receiving communication from a certificate holder, less time than t_{max} has passed since the issuance date of the certificate, then there is no need to request any further CSI. This is one of the ways the authors propose to reduce the number of CSI tokens (i.e. certificates) the CA must produce.

The authors go on, proposing that the validity period of the certificate being sent by the CA in response to the CSI request should be from t_q to t_{end}, where t_q is the time that CSI was originally produced and t_{end} is the expiration

date of the original certificate. Either the CA or the dependent entity could also sent the new certificate to the certificate holder, in order for him to have a certificate with a more recent issuance date thus minimizing the number of CSI requests regarding his certificate.

One of the disadvantages of ACSP is that the CA must have its signing key (the one it uses to sign new certificates or renew existing ones) online, as this is needed in the CSI generation process. Clearly, this increases the risk for this key to be compromised.

The Certificate Re-Issuance mechanism, and its improved version (ACSP), resemble the Freshness-constrained Revocation Authority mechanism (see Section 2.3.2.3). However, the former is a mechanism providing *complete CSI* while the latter only provides *positive* CSI.

Table 17: Certificate Re-Issuance	
Collection of revocation requests	No mechanism specified.
Generation of CSI	CA generates primary CSI in proprietary format. When CSI needs to be distributed, a short-lived certificate with some extra attributes is being prepared as CSI
Storage of CSI	No CSI is stored (except for the primary CSI).
CSI location function	No mechanism specified
CSI retrieval	Format: Short-lived X.509v3 certificates with extra attributes (*ResponseSubject, ResponseStatus*). Retrieval mechanism (although

function	not specified) could be anyone out of those used to disseminate certificates (e.g. http(s), ldap(s))
CSI and certificate validation function	Verification of CA signature on short-lived certificate

2.3.4 Summary

We have presented the elementary CSI functions of a CSI provider and of a dependent entity, attempting to provide a theoretical background to CSI mechanisms in general. A previous version of this section has been published in (Iliadis, Ioannis, Spinellis, Diomidis et al. 2000).

The elementary CSI functions of a CSI provider are the following:

1. Identification of the need for revoking a certificate.
2. Generation of Certificate Status Information.
3. Storage of CSI.

The elementary CSI functions of a dependent entity are the following:

1. CSI Location function.
2. Retrieval of CSI.
3. Validation of CSI.

CSI mechanisms can be categorised in three major categories, depending on the kind of information they disseminate:

1. CSI mechanisms that communicate negative CSI, i.e. CSI that includes information on revoked certificates. CSI mechanisms that belong to this category are:
    a. Certificate Revocation List (along with CRL Distribution Points and Delta CRLs)
    b. Sliding Window Delta-CRL
    c. Freshest CRL
    d. Redirect CRL
    e. Indirect CRL
    f. Enhanced CRL Distribution Points
    g. Augmented CRL
    h. Efficient Fault-Tolerant Certificate Revocation
2. CSI mechanisms that communicate positive CSI, i.e. CSI that includes information on certificates that are still valid
    a. Positive CSI (Suicide Notes)
    b. Self-Controlled Positive CSI
    c. Freshness-constrained Revocation Authority
    d. Efficient long-term validation of digital signatures
3. CSI mechanisms communicating complete CSI, i.e. CSI that specifies explicitly whether a certificate has been revoked or if it is still valid
    a. Certificate Revocation Status (Novomodo)
    b. Certificate Revocation Trees
    c. Online Certificate Status Protocol (OCSP)
    d. Certificate Re-issuance

We have also presented how each of the aforementioned mechanisms implements the elementary functionality of a CSI mechanism, both the functionality required for the CSI provider (and the certificate holder) to operate and the functionality required for the dependent entity to operate.

3 EVALUATING CSI MECHANISMS

3.1 Introduction

The evaluation methods described in this paper strive to avoid unnecessary biases. The framework comprises of a set of qualitative and quantitative evaluation criteria, which can be applied to any mechanism that updates information on the status of certificates (Certificate Status Information — CSI). We use this model as a tool to identify potential problems in the mechanisms in a methodical way. Our evaluation framework splits the evaluation process into three main domains, namely management, performance and security.

Management of revocation mechanisms includes the way these mechanisms operate, the way information processing is being performed, the participating entities, and the timeframes within which operations must take place.

Performance of revocation mechanisms refers to the efficiency characteristics of those mechanisms. These characteristics include the timeliness of the mechanism, the freshness of information it delivers, the scalability and adjustability of the mechanism, and the capability to immediately generate information on the status of a certificate (emergency certificate status information).

The security aspect of revocation mechanisms covers issues related to protecting the operation of the mechanisms themselves, as well as of the information they produce. Certificate status information has to be protected while generated, communicated, and stored.

While designing the evaluation framework, we took into consideration the requirements imposed on the use of these mechanisms by the modus operandi required by the European Directive on a Community Framework for

Electronic Signatures (EU Parliament 1999), the EESSI Expert Team Report (Nilsson, Van Eecke, Medina et al. 1999) and the NIST PKI study (Berkovits S. 1994), (Cart and Password 1995).

The framework can be used to evaluate mechanisms that are operated by a Certification Service Provider (CSP) that issues "qualified certificates" (EU Parliament 1999), (Nilsson, Van Eecke et al. 1999). Certificates are considered to be ``qualified'' if they meet the requirements set forth in Annex I of the Electronic Signatures Directive (EU Parliament 1999) and are provided by a Certification Service Provider meeting the requirements laid out in Annex II of the aforementioned Directive. We have also considered the draft or final requirements and recommendations contained in (International Organization for Standardization 1994), (International Organization for Standardization 1995), (International Organization for Standardization 1996), (Housley, Ford et al. 1999), (Chokhani and Ford 1999), (Adams C. 1999), (Santesson, Polk, Barzin et al. 2001), (Myers, Ankney et al. 1999) and (Housley, Ford et al. 1999) . Most European countries have rewritten their legislation so that digital signatures produced with a secure signature creation device must be considered as a handwritten signature if that digital signature comes with a qualified certificate.

3.2 Evaluation Framework

The Electronic Signature EU Directive (Annex II) requires "the operation of a prompt and secure directory and a secure and immediate revocation service". Furthermore, the Directive requires that the authenticity and validity of the certificate are reliably verified, and that the verification result and the signatory's identity are correctly displayed. The EU Directive also requires that the date and time when a certificate is issued or revoked must be determined precisely. Finally, the last major requirement of the EU Directive

that relates to CSI mechanisms is that the use of these mechanisms must be in accordance with data protection legislation, and respect the privacy of individuals.

The evaluation framework we propose covers the aforementioned, legislative requirements; our framework also comprises of technical requirements, which the EU Directive and the related regulatory frameworks do not deal with.

The requirements our framework identifies fall in three main categories: *management*, *performance* and *security*. These requirements are, in fact, evaluation criteria that can be used both for evaluating existing CSI mechanisms, as well as for designing new ones. In the following paragraphs, these requirements are referred to as CSI Evaluation Criteria, or criteria for short. An earlier version of our evaluation framework has appeared in (Iliadis, Gritzalis, Spinellis et al. 2003).

3.2.1 Management

3.2.1.1 Feedback

Feedback on the CSI that has been retrieved is mostly a matter of user interface of PKI-aware applications or devices that handle CSI on behalf of the dependent entity, and not a matter of CSI mechanisms themselves. Dependent entities must receive information regarding the intermediate operation results of a CSI mechanism and the final output it produces (i.e., whether the certificate is still valid), in accordance with Annex II of the EU Directive on a Community Framework for Electronic Signatures. This feedback must indicate at least the following information:

1. Location information on the CSI that relates to the certificate the dependent entity attempts to validate. This information could be a URI (Berners-Lee T., Fielding R. and L. 1998),

2. if the CSI location has been successfully contacted,
3. if CSI has been retrieved,
4. if the validity (integrity and authenticity) of the retrieved CSI can be verified. Also, if this verification can be based (directly or indirectly) on information the dependent entity has declared as trusted (e.g., CA certificates that are stored locally), or if other, possibly untrusted, information is also needed (e.g., more CA certificates which cannot be validated based on the locally stored set of trusted CA certificates),
5. if the CSI that was retrieved corresponds to the certificate the dependent entity wishes to validate,
6. the status (revoked, suspended, not revoked) of the certificate the dependent entity wishes to validate.

If the CSI mechanism can provide the dependent entity with the information above, either at the beginning or at the end of its execution, or in fragments while the mechanism is operating, then the feedback criterion is met.

3.2.1.2 Transparency

Information systems nowadays are commonplace. Users of information systems are not necessarily computer experts. On the contrary, they are inexperienced computer users that are bound to lack information security awareness and training. Therefore, locating the CSI repository and verifying that the CSI contained in that repository is the one that corresponds to the certificate to be verified must be an automated procedure that requires little, if none at all, human interaction. Dependent entity interaction should be restricted, if possible, to requesting a validity check on a specified certificate. Going even further as far as abstraction is concerned, the dependent entity

should not be required to request a validity check on a certificate but on a digital signature the dependent entity sees on a document it received.

3.2.1.3 Delegation of revocation

The dependent entity could trust an authority, other than the CA, for generating CSI.

It could be either another CA or another authority that operates only as a Revocation Authority (RevA) and not as a Certification Authority. Moreover, it could be an authority, using a separate, distinct CA-issued key for signing CSI or it could be a distinct authority, local to the dependent entity, which is trusted by this entity.

In any case, the dependent entity must be able to verify the status of the keys used by that authority, based on certificates or other information it already considers trusted, before trusting and using CSI from that authority.

3.2.1.4 Delegation of CSI dissemination

The CA may delegate CSI dissemination to another authority. This second authority may need to be trusted by the dependent entities or not, depending on the operation of the particular CSI dissemination mechanism.

The dependent entity has to be able to verify the authenticity and integrity of CSI it retrieves from that authority: if CSI delivered to the dependent entity is contained in a CA-signed field, then the repository (e.g., an LDAP Directory) used by the CSI dissemination authority need not be trusted. However, if CSI delivered to the dependent entity is not already integrity-protected, then the authority disseminating CSI must digitally sign CSI before delivering it to the dependent entity and the dependent entity must be able to validate the respective certificates. However, even if CSI is contained in a CA-signed field, the authority that disseminates CSI may behave maliciously, withholding fresh CSI from the dependent entity. A CSI mechanism must use a distinct method to let the dependent entity verify that

the CSI it received is the freshest one (e.g., bounded revocation in CRLs, white-lists in CRS (Micali 1996)).

3.2.1.5 Delegation of certificate path validation

Certificate path validation can be delegated, in some CSI mechanisms, from the dependent entity to another entity. The dependent entity should be provided with the means to review and verify the results of the validation process (Annex II of the EU Directive on a Community Framework for Electronic Signatures). In addition to that, the entity that performs the certificate path validation should be trusted by the dependent entity.

The validation of a certificate path takes as input the certificate to be validated, a number of other certificates the dependent entity trusts, and CSI regarding these certificates. The output is status information regarding the certificate to be validated.

In some CSI mechanisms, one proceeds as follows to assert on the validity of a certificate: the dependent entity retrieves CSI regarding the certificate to be validated and uses as input the other certificates, the retrieved CSI and the certificate in question. Other CSI mechanisms can perform the certificate path validation on behalf of the dependent entity. In this case, the dependent entity uploads the certificate to be validated and the other certificates (or appropriate values that cryptographically identify in a unique way these certificates), to the authority that has access to the relevant CSI; this authority will then validate the certificate in question on behalf of the dependent entity. That authority then communicates the validity check result to the dependent entity. The dependent entity has to be able to verify the integrity and authenticity of the result returned by the CSI mechanism, using appropriate cryptographic mechanisms.

3.2.1.6 Referral capability

The CSI location function may lead the dependent entity to a CSI location that does not contain the requested CSI (CSI may be less fresh than requested, or it may not contain information regarding the specific certificate to be validated). In that case, the CSI mechanism could refer the dependent entity to another CSI location in order to retrieve the requested CSI.

3.2.1.7 Revocation Reasons

When validating the path, the certificate path validation function could consider the reasons for the revocation of a specific certificate contained in a certificate path. The validation function may output different results depending on the revocation reason. The semantics of revocation reasons and the logic the validation function uses in order to include this information, while validating a certificate path, is a matter of policy.

If the certificate path validation function occurs locally to the dependent entity, then the dependent entity must be able to set the policy for using revocation reasons in the validation function. If the certificate path validation function is delegated to another authority then the inclusion of revocation reasons in the logic of certificate path validation is a matter of the policy used by that authority. The dependent entity must be aware of that policy. Alternatively, the dependent entity could communicate its own policy regarding revocation reasons to the authority that performs the validation.

3.2.1.8 Notification of revocation or suspension

A subscriber, whose certificate is being revoked or suspended, should be notified; it might be necessary to inform other entities, such as a Revocation Authority or a Suicide Note collecting bureau, as well (mentioned as a possible policy requirement in (Chokhani and Ford 1999)).

The notification process should not be tied (in a synchronous way) to the CSI mechanisms themselves, because failure to locate appropriate contact

information for the certificate owner could lead could lead to disruption of the revocation mechanism (disruption either of the revocation request process or the process of disseminating CSI to dependent entities). The notification procedure must be implemented outside the revocation CSI mechanism itself, loosely linked to the CSI generation or CSI storage function of the mechanism.

3.2.1.9 Ability of dependent entity to evaluate the CSI mechanism before trusting it

The CSI mechanisms dependent entities use, to check the validity of a specific certificate, are those offered by the issuer of the aforementioned certificate. Dependent entities are asked to trust the CSI mechanisms offered by the certificate issuers, without having the right or capability to evaluate those CSI mechanisms. Therefore, dependent entities are asked to trust a specific CSI mechanism without knowing the way it works, i.e. without knowing whether they should trust it or not, and to what extend should they trust it.

The formalisation of certificate policies (Klobucarklobucar and Jerman-Blazicjerman-Blazic 1999) would allow dependent entities to express the minimum requirements they would expect a CSI mechanism to meet, before trusting it. Furthermore, the formalisation of certificate policies could automate the procedure of distinguishing between trusted and untrusted CSI mechanisms for a specific dependent entity.

3.2.1.10 Completeness

Some CSI mechanisms disseminate positive CSI, some others disseminate negative CSI and some others disseminate complete CSI (see Section 2.3). Moreover, some mechanisms include the reason why a certificate has been revoked along with the disseminated CSI and some others do not; the revocation reason may well affect the decision of the dependent entity to trust a certificate at a specific point in time, or not.

In our context, the *CSI mechanism completeness* is met only if a CSI mechanism provides *complete* (both positive and negative, depending on the status of the certificate) information, and also provides the reason why this certificate has been revoked.

3.2.2 Performance

3.2.2.1 Timeliness of CSI

Dependent entities should be able to locate and receive CSI in a timely fashion, to allow them to use such information in authenticating entities or verifying the signatures of entities. This feature, along with the Emergency CSI capability criterion (Section 3.2.2.4), satisfies the requirement for an "immediate revocation service", as this has been laid out in Annex II of the EU Directive on a Community Framework for Electronic Signatures (EU Parliament 1999).

Timeliness concerns the amount of time between the generation of primary CSI (i.e. CSI as this is stored at the internal repository of the Authority who is responsible for producing CSI) from the appropriate authority until CSI becomes available to dependent entities, in the format they are supposed to receive CSI. This time period does not include the actual dissemination of CSI to dependent entities, but only the time it takes for CSI to be made available to dependent entities.

As a side note, real-time timeliness is often considered as a must for PKI-based electronic transactions; however, this often seems not to be a required feature for PKI-based electronic transactions in practice (McDaniel and Rubin 2001). In environments where certificates are used to conduct money transactions (where the risk may be high in case of revocation), two factor authentication is the current modus operandi for quite many of the providers of money-related transactions. In some cases (e.g. e-banking), the proof of possession of a user's certificate is required only when the user

requests a money transfer or a payment. To actually enter the e-banking website the user has to use his account's username and password. Thus, due to two-factor authentication, real-time timeliness of the CSI mechanism is not considered as a de facto critical issue, in practice.

3.2.2.2 Freshness of CSI

This criterion concerns the maximum period of time between a request for CSI regarding a certificate and the most recent CSI generation regarding that certificate.

3.2.2.3 Bounded revocation

There should be an upper time limit for new, fresh CSI to be produced and made available (this does not include the actual dissemination of CSI to dependent entities, but only its availability). Dependent entities should reject CSI they receive, if the "date of fresher CSI generation" is in the past and not in the future.

Another requirement concerning the time of issuance of CSI that restricts the placement of CSI issuance in time even more can be of use in locating valid CSI and avoiding replay attacks. We call this *time-complete* revocation. In time-complete revocation, CSI is generated in specific moments in time, neither sooner nor later.

3.2.2.4 Emergency CSI capability

This requirement concerns the ability of the CSI authority to generate CSI and make it available, immediately after receiving a valid revocation request. However, this does not include the immediate dissemination of this CSI to dependent entities, but only the immediate generation of CSI. This feature, along with the timeliness criterion, satisfies the requirement for an "immediate revocation service" in Annex II of the EU Directive on a Community Framework for Electronic Signatures.

3.2.2.5 Scalability

This criterion concerns the efficiency of a CSI mechanism in terms of the processing, communication and storage burden its usage entails. This burden may be increasing in a slow or rapid pace as the number of CSI requests and certificate holders' increase.

The scalability of specific CSI mechanisms has been studied in literature during the recent years (Bruschi, Curti and Rosti 2003; Cooper 1999; Eichler and Muller-Rathgeber 2005; Forné 2000; H, K and S 1999; Hormann, Wrona and Holtmanns 2006; Kikuchi, Abe and Nakanishi 1999; Li, Zhao and Hou 2004; Munoz 2003; Munoz, Forne and Castro 2002; Munoz, J. L., Forne, J., Esparza, O. et al. 2004; Munoz, Forne, Esparza, Soriano et al. 2003; Munoz, Forne, Esparza and Soriano 2003; Papapanagiotou, Markantonakis, Zhang et al. 2005; Rojanapasakorn and Sathitwiriyawong 2004a; Rojanapasakorn and Sathitwiriyawong 2004b; Schwingenschlogl, Eichler and Muller-Rathgeber 2006; Shimshon and Jonathan 1998; Zheng 2003). Our goal is not to replicate that information, but to provide the reader with a concise qualitative comparative evaluation of the CSI mechanisms in terms of scalability.

3.2.2.6 Adjustability

The dependent entities (or the CA and the CSI authorities) should be able to adjust the location function (see Section 2.2) based on their own requirements for freshness, in order to create a balance between performance and protection, depending on risk assessment in each case. They should be able to do that because it is the dependent entities that take the risk by accepting this balance between performance and protection.

3.2.3 Security

The CSI mechanism criteria presented in this section correspond to the requirement (Annex II of the EU Directive on a Community Framework for

Electronic Signatures) for a "secure directory" and a "secure revocation service".

3.2.3.1 CSI disseminator authentication

The dependent entities must be able to verify the origin of CSI they receive. If authentication is not used, a malicious entity pretending to be a trusted CSI disseminator could disseminate false CSI to dependent entities, which appears to be valid.

3.2.3.2 CSI integrity and authenticity

The integrity of the CSI must be protected, when it is stored in the CSI repository, while it is transferred to the dependent entities and when it is stored in the dependent entities' local repository. The integrity protection mechanisms must ensure that a malicious entity cannot modify either the stored CSI or the CSI while in transit. If this happens, the dependent entity should be able to know that the received CSI is old, partial, or invalid in any way. Furthermore, the authenticity of CSI (i.e. that CSI has indeed been produced by the CSI authority it claims to have been produced) must be verifiable.

Note however, that this requirement may introduce another type of attack, i.e. the Denial of Service. The information that is transferred to the dependent entity must be integrity-protected. If a system such as OCSP is used, this means that the OCSP responder must produce a (mostly expensive) fresh digital signature on each response. On the contrary, when using CSI mechanisms that produce a single signature (or other computational-heavy process) on CSI regarding a group of certificates (e.g. CRLs), Denial of Service attacks from dependent entities, or entities pretending to be dependent ones, cannot easily be mounted.

3.2.3.3 CA compromise

There should be a mechanism, (e.g. an Authority Revocation List (ARL) that is similar to a CRL but enumerates revoked CA certificates) for the dependent entities to know whether a CA has been compromised. There should also be a mechanism to allow a CA to recover from compromise. The effects of a CA key being compromised should be minimised.

3.2.3.4 Revocation Authority (RevA) compromise

There should be a mechanism (e.g., similar to an Authority Revocation List as explained above) for the dependent entities to know whether the authority that revokes certificates (RevA) has been compromised. This mechanism must not be the same as the one used by dependent entities in order to receive CSI on certificates that belong to entities other than the RevA.

3.2.3.5 Contained functionality

If RevA is compromised, it should not be possible for the entities that gained control of the RevA to issue new certificates.

3.2.3.6 Availability

CSI mechanisms should be resilient against Denial of Service attacks. DoS could be used as a way to prevent negative CSI information reaching its intended destination (dependent entities), thus nullifying the revocation of the certificate holder's certificate. DoS could also be used in order to prevent positive CSI reaching the dependent entities, thus preventing some transactions from concluding due to supposedly revoked certificates (i.e. absence of positive CSI available).

3.3 Evaluation of CSI-retrieval mechanisms

In this section, we apply our evaluation framework to the CSI mechanisms presented in Section 2.3).

3.3.1 Management

3.3.1.1 Feedback

We believe that it is necessary to provide the dependent entity with feedback information. An assertion regarding the status of a certificate will be interpreted differently by the dependent entity, depending on the information that led to this assertion and the local policy. Existing standardisation efforts, which are related to the CSI mechanisms we examine, do not include suggestions for the user interface of such applications or devices.

It is also necessary to define a simple way to convey this information to the dependent entity, because of the complexity of the information as well as of the possible lack of understanding of nontrivial security and cryptography issues by the dependent entity. Standardisation efforts related to the mechanisms we examine should include high-level requirements for a user interface that provides such information to the dependent entities. This would result in an even higher level of user awareness on CSI mechanisms. Finally, this information could be provided to the dependent entity at no substantial operational cost for the CSI mechanism.

As mentioned in Section 2 ("Survey of Related Work"), CSI mechanisms can provide either negative, or positive, or complete CSI. The only mechanisms that could provide adequate feedback to dependent entities (if one interprets the Digital Signatures EU Directive (EU Parliament 1999) in a strict manner) are the ones that provide *complete CSI* (see Section 2.3.3), also providing the reasons for the revocation of a certificate to the dependent entity, besides a "revoked"/"not revoked" status check.

OCSP provides *complete CSI* (actually, it provides *partially complete CSI* because there is a third *CertStatus*, other than "good" and "revoked", the "unknown" one) and also includes the revocation reasons for a certificate. Therefore, only OCSP (partially) meets the feedback criterion.

CRS, provides *complete CSI*, if one also considers the enhancements that have been proposed for this mechanism (*full revocation certificates*, see Section 2.3.3.1)

Table 18 : Evaluating Feedback

CSI Mechanism	Feedback
CRL	✗
Sliding Window DeltaCRL	✗
Freshest CRL	✗
Redirect CRL	✗
Indirect CRL	✗
Enhanced CRL Distribution Points	✗
Augmented CRL	✗
Efficient Fault-Tolerant Certificate Revocation	✗
Positive CSI (Suicide Notes)	✗
Self-Controlled Positive CSI	✗
Freshness-constrained Revocation Authority	✗
Efficient Long-term Validation of Digital Signatures	✗
Certificate Revocation Status	✓
Certificate Revocation Trees	✗
Online Certificate Status Protocol (OCSP)	Partial
Certificate Re-issuance	✗

3.3.1.2 Transparency

Although this is an important feature of CSI mechanisms and does not introduce an important additional communication cost, it is not, yet, a matter of common practice among commercial or non-commercial CAs, neither has it

been a matter of extensive research. The fact that standardisation efforts, regarding CSI location and retrieval information in the certificates, are ongoing could be a reason for that.

The chain of security mechanisms protecting an Information System is only as secure as its weakest link. In the case of PKI, the weakest link seems to be the dependent entity. If she does not know or is not conscious enough to perform CSI location retrieval and validation then all security provided by CSI mechanisms is rendered null and void. CSI mechanisms reviewed in Section 2 (Survey of Related Work) do not meet the *transparency* criterion. None of them deals with the fact that the dependent entity must be able to appreciate the need to use CSI mechanisms in order to verify its electronic transactions and the corresponding certificates. None of those mechanisms also deals with the fact that the dependent entity must have some technical background in order to understand how they operate, and use them.

There seem to be two kinds of solutions to this problem:

1. Conduct security awareness seminars to all dependent entities so that they will know how to, and not neglect to, perform CSI location, retrieval and validation whenever that is necessary. This is the only complete solution to the problem, however it is not feasible.
2. Research further the interfaces that PKI-aware applications provide to users and make sure that CSI location, retrieval and validation can be performed as transparently as possible. Software agents could help with that, undertaking the responsibility for handling CSI on behalf of the user and locally interfacing on the user's PC with PKI-aware applications in order to fetch CSI for them and validate it, without the user knowing how it ever happened.

The mechanism we propose in Section 4.3 attempts to fill somehow this gap.

3.3.1.3 Delegation of revocation

CRLs and their variants support delegation of revocation. The CA can designate a distinct authority (or a unit of the CA itself, distinct to the certificate-issuing unit) to issue the various CRL types. The CA has to issue a certificate for that authority, which must contain the cRLSign key usage attribute. CRTs also support delegation of revocation; the cRLSign key usage attribute could also be used in the case of CRT delegation of revocation, with an extra attribute to declare that this is a CRT signing key and not a CRL signing key.

The Efficient Fault-Tolerant Certificate Revocation mechanism, the Positive CSI and the Self-controlled Positive CSI mechanism inherently support partial delegation of revocation. In all aforementioned mechanisms, CSI generation is –by design– delegated to the certificate holder. However, CSI generation cannot be delegated to any other entity, with the exception of the Positive CSI mechanism, where any dependent entity can revoke a certificate holder's certificate.

CRS, the Efficient Long-term Validation of Digital Signatures mechanism and the Certificate Re-issuance mechanism do not support delegation of revocation to other entities than the CA.

OCSP does not support delegation of revocation, because OCSP does not include a specific mechanism for generating CSI. It uses the CSI generated by the CA, retrieving it possibly from a database indicated by the CA, in order to construct the CSI to be sent to the dependent entities.

Table 19 : Evaluating Delegation of Revocation	
CSI Mechanism	**Delegation of Revocation**
CRL	✓
Sliding Window DeltaCRL	✓
Freshest CRL	✓
Redirect CRL	✓
Indirect CRL	✓
Enhanced CRL Distribution Points	✓
Augmented CRL	✓
Efficient Fault-Tolerant Certificate Revocation	Partial
Positive CSI (Suicide Notes)	✓
Self-Controlled Positive CSI	Partial
Freshness-constrained Revocation Authority	✓
Efficient Long-term Validation of Digital Signatures	✗
Certificate Revocation Status	✗
Certificate Revocation Trees	✓
Online Certificate Status Protocol (OCSP)	✗
Certificate Re-issuance	✗

3.3.1.4 Delegation of CSI dissemination

Delegation of CSI dissemination for mechanisms based on CRLs and their variants can be performed since the authenticity and integrity of CSI is verified with the help of the digital signature on the CRL. This signature is produced either by the CA, or by an entity possessing a certificate from the

CA where the cRLSign attribute has been set, designating this entity as trusted for CRL-based CSI generation.

However, since the authority that disseminates CSI and the respective repository may not be trusted, there has to be a specific policy for the issuance of these CRLs. Such a policy must ensure that the CRL provided by the CSI disseminating authority is always the one that corresponds to the needs of the dependent entities at the specific moment in time when the CSI query is performed. Furthermore, the aforementioned policy must ensure that dependent entities have the ability to identify whether the CSI disseminating authority is withholding a specific (fresher) CRL or CRL variant from them.

The Efficient Fault-Tolerant Certificate Revocation CSI mechanism does not support delegation of CSI dissemination; the root of a depender graph, i.e. the certificate holder or the CA that issued the certificate, disseminates CSI regarding the certificate in question as soon as this is available to all its registered dependers. They, in turn, disseminate CSI regarding the certificate in question to their registered dependers and so on. Note, however, that (Wright Rebecca N., Lincoln Pitman Pressatrick D. et al. 2000) do not specify in their mechanism the way to verify the authenticity and integrity of disseminated CSI; rather, they depend on the mechanics of the underlying CSI mechanism to be used by their depender graph scheme.

Positive CSI inherently supports delegation of CSI dissemination since it is the Suicide Bureaus and not the CAs themselves that distribute CSI, using the certificate holder as an intermediary (dependent entity asks the certificate holder for CSI and he turns to the Suicide Bureau requesting for a "certificate of health", which he delivers to the dependent entity). The same kind of inherent support for delegation of CSI dissemination applies to Self-Controlled Positive CSI, since according to this mechanism it is the certificate holders themselves that produce CSI every time they use their

certificate and they include the CSI along with the data that has been signed with their private key.

The Freshness-constrained Revocation Authority mechanism also supports delegation of CSI dissemination, since it is the Revocation Authority (and not the CA) that disseminates CSI, which in this case consists of short-lived certificates.

The Notary Authority employed by the Efficient Long-term Validation of Digital Signatures mechanism produces CSI on its own, regarding a specific signed piece of information, using a specific certificate. In that sense, it supports delegation of CSI information since it is the Notary that produces CSI (or SSI – Signature Status Information) rather than the CA itself.

CRS and CRTs support delegation of CSI dissemination to other entities, since the authenticity and integrity of CSI is verified offline, by the dependent entity, after having received it.

OCSP supports two levels of CSI dissemination delegation:

1. At the first level, the authority (CA Designated Responder (Myers, Ankney et al. 1999)) that disseminates CSI (OCSP signed Responses) must have a certificate issued by the CA for that purpose.
2. At the second level, the authority that disseminates CSI can be a third, distinct authority (Trusted Responder (Myers, Ankney et al. 1999)) whose public key is trusted by the dependent entity.

Neither the Certificate Re-Issuance mechanism (nor its improved version, the Advanced Certificate Status Protocol) support delegation of CSI dissemination since CSI is being produced by the CA, per certificate and on request by dependent entities.

Table 20 : Evaluating Delegation of CSI Dissemination	
CSI Mechanism	**Delegation of CSI Dissemination**
CRL	✓
Sliding Window DeltaCRL	✓
Freshest CRL	✓
Redirect CRL	✓
Indirect CRL	✓
Enhanced CRL Distribution Points	✓
Augmented CRL	✓
Efficient Fault-Tolerant Certificate Revocation	Partial
Positive CSI (Suicide Notes)	✓
Self-Controlled Positive CSI	✓
Freshness-constrained Revocation Authority	✓
Efficient Long-term Validation of Digital Signatures	✓
Certificate Revocation Status	✓
Certificate Revocation Trees	✓
Online Certificate Status Protocol (OCSP)	✓
Certificate Re-issuance	×

3.3.1.5 Delegation of certificate path validation

CRL-based mechanisms, the Efficient Fault-Tolerant Certificate Revocation mechanism, the Freshness-constrained Revocation Authority mechanism, CRS and CRTs and the Certificate Re-issuance mechanism do not support delegation of the certificate path validation function.

For the Positive CSI mechanism, there is no certificate path that needs validation since the CSI is a Suicide Note signed by the certificate holder itself and sent to the Suicide Bureau. No certificate path needs validation also in the case of the Self-controlled Positive CSI mechanism and the Efficient Long-term Validation of Digital Signatures mechanism.

OCSP partially supports delegation of the certificate path validation function. Extensions to the supported delegation have been proposed (Hallam-Baker 1999). Delegation of the certificate path validation function requires the availability of a large set of trust-related information (e.g., CA certificates, CRL or CSI in other formats) to the OCSP service provider. The aggregation of this kind of information could incur communication, maintenance or other costs.

Table 21 : Evaluating Delegation of Certificate Path Validation

CSI Mechanism	**Delegation of Certificate Path Validation**
CRL	✗
Sliding Window DeltaCRL	✗
Freshest CRL	✗
Redirect CRL	✗
Indirect CRL	✗
Enhanced CRL Distribution Points	✗
Augmented CRL	✗
Efficient Fault-Tolerant Certificate Revocation	✗
Positive CSI (Suicide Notes)	Not Applicable (there is no certificate path that needs validation)
Self-Controlled Positive CSI	Not Applicable (there is no certificate path that

	needs validation)
Freshness-constrained Revocation Authority	✗
Efficient Long-term Validation of Digital Signatures	Not Applicable (there is no certificate path that needs validation)
Certificate Revocation Status	✗
Certificate Revocation Trees	✗
Online Certificate Status Protocol (OCSP)	Partial
Certificate Re-issuance	✗

Some CSI mechanisms do not support delegation of certificate path validation while for some others such a feature is not applicable. The reason why the former do not support delegation of certificate path validation is that this feature has usually been considered out of the scope of the CSI mechanism itself, i.e. it is another mechanism by itself. Some certificate path validation mechanisms (A. Malpani 2002; Freeman T, Housley R., Malpani A. et al. 2007) have already been proposed in the literature.

3.3.1.6 Referral capability

CRLs and their variants do not inherently support the referral capability. If an LDAPv3 (Chadwick 2002) Directory is used as the CRL repository, the CSI provider can refer the dependent entity to other CRL repositories, by the means of LDAP referrals (Chadwick 2002). There are two exceptions to that; Redirect CRLs (Adams C. 1998) and Indirect CRLs inherently support referrals through the use of the Redirect Pointer extension.

OCSP supports this capability, through the *serviceLocator* request extension. If the OCSP service provider does not have CSI concerning the certificate the dependent entity enquires, it may refer the dependent entity to another OCSP service. Location information for that OCSP service is contained in the OCSP Response extension *serviceLocator*.

Table 22 : Evaluating Referral Capability	
CSI Mechanism	**Referral Capability**
CRL	✗
Sliding Window DeltaCRL	✗
Freshest CRL	✗
Redirect CRL	✓
Indirect CRL	✓
Enhanced CRL Distribution Points	✗
Augmented CRL	✗
Efficient Fault-Tolerant Certificate Revocation	✗
Positive CSI (Suicide Notes)	✗
Self-Controlled Positive CSI	✗
Freshness-constrained Revocation Authority	✗
Efficient Long-term Validation of Digital Signatures	✗
Certificate Revocation Status	✗
Certificate Revocation Trees	✗
Online Certificate Status Protocol (OCSP)	✓
Certificate Re-issuance	✗

3.3.1.7 Revocation Reasons

Some of the CSI mechanisms can disseminate information to the dependent entity regarding the reasons for the revocation of a certificate (e.g. *CRLReason* extension (Housley, Ford et al. 1999), (Myers, Ankney et al. 1999)). However, the use of those reasons as input information in a certificate path validation function has to be studied further (Fox and LaMacchia 1998), (Iliadis 1999).

These reasons should be used in the process of certificate path validation only if they can provide results that are complete, repeatable, and compliant with the policy of the dependent entity and the policy of the authority that executes the certificate path validation function. These requirements are currently not met by the mechanisms we examine because they do not provide the capability (to the dependent entity) to define a complete certificate validation policy and have the certificate validated against that policy, providing complete and repeatable results. The table that follows shows which mechanisms support at least the inclusion of the revocation reason within the disseminated CSI.

Note that CRS supports revocation reasons only if its improved version is used (see Section 2.3.3.1), where *full revocation certificates* are also disseminated.

Table 23 : Evaluating Revocation Reasons

CSI Mechanism	**Revocation Reasons**
CRL	✓
Sliding Window DeltaCRL	✓
Freshest CRL	✓
Redirect CRL	✓
Indirect CRL	✓
Enhanced CRL Distribution Points	✓
Augmented CRL	✓
Efficient Fault-Tolerant Certificate Revocation	✗
Positive CSI (Suicide Notes)	✗
Self-Controlled Positive CSI	✗

Freshness-constrained Revocation Authority	✗
Efficient Long-term Validation of Digital Signatures	✗
Certificate Revocation Status	✓
Certificate Revocation Trees	✗
Online Certificate Status Protocol (OCSP)	✓
Certificate Re-issuance	✗

3.3.1.8 Notification of revocation or suspension

The CSI mechanisms we examine do not inherently provide this kind of functionality. The certificate owner can find out if his certificate can be revoked only by looking up CSI for his own certificate.

However, this does not hold true for those CSI mechanisms where the certificate holder is always actively involved in the CSI dissemination process. These mechanisms are the Efficient Fault-tolerant Certificate Revocation and the Self-Controlled Positive CSI. In these cases, notification of revocation or suspension is not applicable, simply because it is the certificate holder himself producing and disseminating CSI.

Table 24 : Evaluating Notification of Revocation or Suspension

CSI Mechanism	**Notification of Revocation or Suspension**
CRL	✗
Sliding Window DeltaCRL	✗
Freshest CRL	✗
Redirect CRL	✗
Indirect CRL	✗
Enhanced CRL Distribution Points	✗

Augmented CRL	✗
Efficient Fault-Tolerant Certificate Revocation	N/A
Positive CSI (Suicide Notes)	✗
Self-Controlled Positive CSI	N/A
Freshness-constrained Revocation Authority	✗
Efficient Long-term Validation of Digital Signatures	✗
Certificate Revocation Status	✗
Certificate Revocation Trees	✗
Online Certificate Status Protocol (OCSP)	✗
Certificate Re-issuance	✗

3.3.1.9 Ability of dependent entity to evaluate the CSI mechanism before trusting it

The CSI mechanisms we examine do not provide this kind of functionality. Dependent entities cannot evaluate and/or choose among a set of CSI mechanisms offered, in order to get CSI from them according to an order of preference the dependent entity establishes.

3.3.1.10 Completeness

Only CSI mechanisms providing *complete* CSI (see Section 2.3.3) could possibly meet this criterion, provided that they also include in CSI the reasons why a certificate has been revoked.

Out of those CSI mechanisms, only OCSP includes in the CSI response the reasons why a certificate has been revoked. However, OCSP supports three kinds of CSI responses: *good*, *revoked* and *unknown*. Due to the *unknown* response, OCSP does not fully meet the completeness criterion.

CRS, provides *complete CSI*, if one also considers the enhancements that have been proposed for this mechanism (*full revocation certificates*, see Section 2.3.3.1)

Table 25 : Evaluating Completeness

CSI Mechanism	Completeness
CRL	✗
Sliding Window DeltaCRL	✗
Freshest CRL	✗
Redirect CRL	✗
Indirect CRL	✗
Enhanced CRL Distribution Points	✗
Augmented CRL	✗
Efficient Fault-Tolerant Certificate Revocation	✗
Positive CSI (Suicide Notes)	✗
Self-Controlled Positive CSI	✗
Freshness-constrained Revocation Authority	✗
Efficient Long-term Validation of Digital Signatures	✗
Certificate Revocation Status	✓
Certificate Revocation Trees	✗
Online Certificate Status Protocol (OCSP)	Partially
Certificate Re-issuance	✗

3.3.2 Performance

3.3.2.1 Timeliness of CSI

CRLs tend to have a rather low issuance frequency, otherwise the mechanism leads to traffic congestion. CRLs definitely do not provide *timely*

CSI, since dependent entities do not know of the new revocations that have been requested by the respective certificate holders within the validity period of a CRL, until the next CRL is issued. Delta CRLs and Sliding Window Delta CRLs reduce the time it takes for CSI to be made available to dependent entities, however, CSI still takes time to reach dependent entities (the time remaining between the revocation request and the issuance of the next Delta CRL). The same applies to all other CRL-based mechanisms (Redirect CRL, Indirect CRL, Enhanced CRL Distribution Points and Augmented CRLs) with the exception of the Freshest CRL mechanism. FCRL provides for timely CSI, since the CA can issue very frequently fresh CSI in the form of Freshest Delta CRLs; these are not issued in fixed time intervals; rather, they are issued as often as needed (e.g. if the CA considers that a new one has to be issued because a certificate has been revoked, and this revocation may be of interest to a lot of dependent entities).

The Efficient Fault-Tolerant Revocation mechanism provides CSI in a very timely fashion, since it is the certificate holder that produces CSI as soon as he realizes that there is a need to revoke his certificate, and makes this CSI available to members of the depender graph. The same applies in the case of the Positive CSI mechanism, where the certificate holder initiates the primary CSI production (Suicide Notes) and also provides dependent entities with CSI (Certificates of Health) retrieved from Suicide Bureaus at the time of the certificate-related transaction, respecting dependent entity's requirements regarding the freshness of CSI she wishes to receive. The same applies for Self-Controlled Positive CSI since the certificate holder himself distributes Positive CSI along with the signed data and thus he can simply stop distributing positive CSI as soon as he believes his private key has been compromised. The Freshness-constrained RevA mechanism also supports timely CSI because positive CSI is distributed in the form of short-lived

certificates; as soon as a revocation notice is submitted, RevA will stop producing these short-lived certificates and the signature of the certificate holder will no longer be validated.

Efficient Long-term Validation of Digital Signatures also supports timely CSI as validation data (for a specific signature of the certificate holder) can be produced per request of the dependent entity, semi-realtime.

CSI provided by CRS and CRT is not very timely; the computation time of CRS/CRT-based CSI prevents the CA from updating the CSI repository in very short time periods, thus preventing timely CSI.

OCSP, as an online query protocol giving out CSI regarding a specific certificate based on internal CSI stored in the CA upon revocation notification does provide timely CSI.

The Certificate Re-issuance mechanism cannot provide timely CSI because the validity period of the CSI it gives out cannot be short-lived enough due to the computational burden required for producing it.

Table 26 : Evaluating Timeliness

CSI Mechanism	**Timeliness**
CRL	✗
Sliding Window DeltaCRL	✗
Freshest CRL	✓
Redirect CRL	✗
Indirect CRL	✗
Enhanced CRL Distribution Points	✗
Augmented CRL	✗
Efficient Fault-Tolerant Certificate Revocation	✓

Positive CSI (Suicide Notes)	✓
Self-Controlled Positive CSI	✓
Freshness-constrained Revocation Authority	✓
Efficient Long-term Validation of Digital Signatures	✓
Certificate Revocation Status	✗
Certificate Revocation Trees	✗
Online Certificate Status Protocol (OCSP)	✓
Certificate Re-issuance	✗

3.3.2.2 Freshness of CSI

The freshness property of CSI for CAs in Europe is a required feature for actual operating PKIs due to legal requirements (EU Parliament 1999).

The dependent entity undertakes a specific risk by trusting (or not) a specific certificate; however, in most CSI mechanisms it is not the dependent entity who gets to impose specific requirements upon the freshness of CSI.

In the table that follows, we present the levels of freshness offered by CSI mechanisms. The scale notation we use is the following: a *low level* of freshness denotes that the CSI mechanism does not offer very fresh CSI, while a *medium* and a *high* level of freshness denotes that the CSI mechanism offers fresher CSI compared to the ones offering low levels of freshness.

Table 27 : Evaluating Freshness

CSI Mechanism	**Freshness**
CRL	Low
Sliding Window DeltaCRL	Low
Freshest CRL	Medium
Redirect CRL	Low

Indirect CRL	Low
Enhanced CRL Distribution Points	Low
Augmented CRL	Low
Efficient Fault-Tolerant Certificate Revocation	High
Positive CSI (Suicide Notes)	High
Self-Controlled Positive CSI	High
Freshness-constrained Revocation Authority	Medium
Efficient Long-term Validation of Digital Signatures	Medium
Certificate Revocation Status	Medium
Certificate Revocation Trees	Medium
Online Certificate Status Protocol (OCSP)	High
Certificate Re-issuance	Medium

CRLs have a –usually large– issuance period due to their size, thus the level of freshness they offer is low. If a certificate is issued within the time period between two CRL issuances, dependent entities will only be notified when it is time to issue the new CRL. This problem is alleviated by Delta-CRLs but the levels of freshness still remain low because even Delta CRLs cannot be issued that frequently. The same applies for Sliding Window Delta-CRLs, Redirect CRLs, Indirect CRLs, Enhanced CRL Distribution Points, Augmented CRLs,

Freshest CRLs offer medium levels of freshness at the expense of scalability and at the expense of time-completeness (see Section 3.2.2.3). Communication costs are increased because dependent entities requiring this very fresh CSI will be downloading irregularly issued Delta-CRLs all the time, or at the very least they will be querying to verify no such Delta-CRLs have been issued.

The Efficient Fault-Tolerant Certificate Revocation also offers high levels of freshness because CSI is generated by the dependent entity on the spot, after the dependent entity decides it's time to revoke the respective certificate. This applies also to the Positive CSI mechanism.

The Self-Controlled Positive CSI mechanism offers high levels of freshness, since CSI is embedded in the actual signed data, i.e. CSI always accompanies the signature.

The Freshness-constrained Revocation Authority presents medium levels of CSI freshness. The nature of the mechanism is such that it could offer high levels of CSI freshness to dependent entities if it were not for the very high processing costs of CSI (for each issued certificate another short-live certificate is issued periodically stating that the original certificate is still valid).

The Efficient long-term validation of digital signatures offers medium levels of freshness; it is supposed to draw CSI data directly from the CA (where proprietary-formatted CSI is being generated as soon as a revocation request comes in) but the new Merkle hash tree used by this CSI mechanism cannot be constructed very often both due to processing costs and due to communication costs this would entail if dependent entities were to check for CSI very often.

CRS uses a fixed time granularity of issuing CSI, like CRLs. This time granularity can be smaller than CRLs due to reduced processing cost and communication cost (communicating CSI to the dependent entity). The level of freshness it supports is medium. The same level of freshness is supported by CRT, for similar reasons.

OCSP supports high levels of freshness, if OCSP draws the primary proprietary-formatted CSI directly from the CA.

Certificate Re-issuance supports medium levels of freshness because the CSI it produces must have a minimum validity period of a day or so, due to CSI processing constraints (CSI production is cpu-intensive).

3.3.2.3 Bounded revocation

CRLs and their variants, OCSP support bounded revocation with the use of the *nextUpdate* CRL and Response extensions respectively. These CSI mechanisms could also support time-complete revocation. This is a matter of policy.

Efficient Fault-Tolerant Certificate Revocation, Positive CSI, Freshness-constrained Revocation Authority, Efficient Long-term Validation of Digital Signatures and CRT do not support bounded revocation

Self-Controlled Positive CSI supports bounded (and time-complete) CSI because CSI always accompanies the signed piece of data.

CRS by definition supports bounded and time-complete CSI since each $F^j(K)$ value is only valid for the time period that corresponds to i.

CRT can also support bounded revocation if the period of CRT issuance and validity is defined a priori.

Finally, Certificate Re-issuance supports bounded (and time-complete) CSI since the CSI (i.e. short-lived certificates) contains a specific validity period and, if the appropriate policy was to be applied, no other CSI regarding that certificate should be produced within that time period.

Table 28 : Evaluating Bounded Revocation

CSI Mechanism	Bounded Revocation
CRL	✓
Sliding Window DeltaCRL	✓
Freshest CRL	✓

Redirect CRL	✓
Indirect CRL	✓
Enhanced CRL Distribution Points	✓
Augmented CRL	✓
Efficient Fault-Tolerant Certificate Revocation	✗
Positive CSI (Suicide Notes)	✗
Self-Controlled Positive CSI	✓
Freshness-constrained Revocation Authority	✗
Efficient Long-term Validation of Digital Signatures	✗
Certificate Revocation Status	✓
Certificate Revocation Trees	✓
Online Certificate Status Protocol (OCSP)	✓
Certificate Re-issuance	✓

3.3.2.4 Emergency CSI capability

CRL-based mechanisms do not support Emergency CSI; even if Delta CRLs were used to issue Emergency CSI, this would break their bounded revocation functionality and their time-completeness (see Section 0). The same applies for CRS and CRT. Besides breaking the bounded revocation functionality and time-completeness, if Emergency CSI were to be supported on the aforementioned revocation schemes the computational burden (of preparing CRLs/CRSs/CRTs adhoc) and the communication burden (of dependent entities retrieving this CSI in its entirety each time a certificate was revoked) would be intolerable in a real life scenario.

Regarding OCSP, if the OCSP service locator retrieves CSI directly from the repository where the CSI authority stores it, then OCSP can render that CSI

immediately available to dependent entities, thus supporting Emergency CSI capability.

Efficient Fault-Tolerant Certificate Revocation and Self-Controlled Positive CSI support Emergency CSI since it is the certificate holder that sends CSI directly to dependent entities. Positive CSI supports Emergency CSI since the certificate holder produces a suicide note as soon as there is suspicion of certificate compromise and this suicide note is forwarded to the Suicide Bureaus and made available immediately to dependent entities.

The Freshness-constrained Revocation Authority and the Certificate Re-issuance mechanism do not support Emergency CSI since they may have already produced a short-lived certificate before the actual revocation request, stating that the certificate is valid for the validity period of the positive CSI (short-lived certificate).

The Efficient Long-term Validation of Digital Signatures supports Emergency CSI because it disseminates SSI (Signature Status Information) rather than CSI (see Section 2.3.2.4), thus if a certificate was used to sign some piece of data at a specific time when a revocation notice had already been forwarded to the CA, the Efficient Long-term Validation of Digital Signatures will not notarise that signature as valid.

Table 29 : Evaluating Emergency CSI capability

CSI Mechanism	**Emergency CSI capability**
CRL	✗
Sliding Window DeltaCRL	✗
Freshest CRL	✗
Redirect CRL	✗
Indirect CRL	✗

Enhanced CRL Distribution Points	✗
Augmented CRL	✗
Efficient Fault-Tolerant Certificate Revocation	✓
Positive CSI (Suicide Notes)	✓
Self-Controlled Positive CSI	✓
Freshness-constrained Revocation Authority	✗
Efficient Long-term Validation of Digital Signatures	✓
Certificate Revocation Status	✗
Certificate Revocation Trees	✗
Online Certificate Status Protocol (OCSP)	✓
Certificate Re-issuance	✗

3.3.2.5 Scalability

We will be grading each CSI mechanism in terms of scalability (communication, processing and storage costs) using the same grading scheme we used in Section 3.3.2.2.

In the case of CRLs, processing costs are low (all that is required is a signature on the list of serial numbers of revoked certificates) while the storage and communication are high; communication and storage costs increase linearly over time. The high communication and storage costs are somewhat reduced by the use of Delta-CRLs and Distribution Points, and even more by Sliding Window Delta-CRLs and Freshest CRLs but they still remain very high since a full CRL still needs to be downloaded at specific predefined points in time. Overall, these CSI mechanisms are of *low scalability.*

Redirect CRLs, Enhanced CRL Distribution Points and Augmented CRLs manage to split the high communication costs more evenly over the validity period of CRLs, this decreasing somewhat the communication costs.

However, storage costs are not reduced as much as communication costs. In total, their scalability level is medium.

The Efficient Fault-Tolerant Certificate Revocation mechanism presents low processing costs (CSI is being produced by the certificate holder himself). The communication and storage costs are at a medium level since dependent entities have to register in order to receive CSI for certificates they are interested in (push mechanism); this means that they will be getting also CSI for certificates they have not really got the need to verify anymore. The scalability level is medium.

The processing costs of Positive CSI and Freshness-constrained Revocation Authority mechanisms are at a medium level because dependent entities can, at their discretion, be asking for fresher CSI than the already produced one and the Suicide Bureaus and RevAs respectively will have to produce it; since CSI constitutes of a certificate, massively producing all the required certificates dependent entities require takes the processing costs to a medium level. However, the communication and storage costs of this mechanism are low since the transferred and stored CSI concerns only the certificates the dependent entity is interested in validating. In total its scalability is medium.

The processing costs of Self-controlled Positive CSI are low because CSI production is distributed among certificate holders. The communication and storage costs are also low because dependent entities receive CSI only for the certificates they are interested in, and the increase in the size of the original certificate this mechanism requires is unsubstantial. The scalability of the mechanism is high.

The Efficient Long-term Validation of Digital Signatures presents medium communication costs because it has to do insertions in the merkle hash tree it keeps for each CSI reply and also because it does not provide CSI for a specific certificate but for a specific signature that has been produced with a

specific certificate, thus the number of queries by dependent entities will be larger compared to other CSI mechanisms. However, the CSI reply is small in size; therefore the communication and storage costs are at a medium level. In total its scalability is medium.

CRS, CRT and OCSP provide for low communication and storage costs but processing costs are increased because the entity that produces CSI must do expensive CPU operations (produce new hash values for all issued certificates through hashing, full restructuring of a binary hash tree and produce a signed statement of the status of a certificate for every CSI query that comes in, respectively). In the case of CRS and CRT these expensive, in terms of CPU, operations take place when the validity period of a CRS or CRT ends and the required processing power required is not affected by the number of certificates that have actually been revoked, i.e. the process is always CPU intensive. In the case of OCSP, a signed response must be given for every CSI query, regardless whether the certificate has been revoked or not, thus the expensive operation (i.e. signing) is also high even if the number of (newly) revoked is low. However, the improvements presented by others (see Sections 2.3.3.1, 2.3.3.2 and 2.3.3.3) decrease the processing costs incurred by the mechanisms in their original versions. These mechanisms' scalability is high.

The Certificate Re-issuance mechanism presents medium level processing costs because the reply for CSI queries requires the online production of a new certificate. However, the number of CSI queries that have to be serviced with a new certificate is reduced thanks to the extensions proposed to this mechanism by other researchers (see Section 2.3.3.4). The communication and storage costs are low. In total the scalability of the mechanism is medium.

As a final note, all mechanisms where the CSI constitutes of a signed piece of information that concerns a specific certificates are vulnerable to Denial of Service attacks; all the attacker would have to do is request CSI on an

adequately high number of certificates; that would force the CSI mechanism to produce certificates (i.e. digitally sign) a high number of times, leading to a possible DoS. The only mechanism out of the aforementioned that deals with DoS is OCSP, thanks to the extensions proposed by some researchers (see Section 2.3.3.3).

Table 30 : Evaluating Scalability

CSI Mechanism	Scalability
CRL	Low
Sliding Window DeltaCRL	Low
Freshest CRL	Low
Redirect CRL	Medium
Indirect CRL	Low
Enhanced CRL Distribution Points	Medium
Augmented CRL	Medium
Efficient Fault-Tolerant Certificate Revocation	Medium
Positive CSI (Suicide Notes)	Medium
Self-Controlled Positive CSI	High
Freshness-constrained Revocation Authority	Medium
Efficient Long-term Validation of Digital Signatures	Medium
Certificate Revocation Status	High
Certificate Revocation Trees	High
Online Certificate Status Protocol (OCSP)	High
Certificate Re-issuance	Medium

3.3.2.6 Adjustability

The Freshness-constrained Revocation Authority and Positive CSI mechanisms support this criterion; in these mechanisms, dependent entities can require CSI with a specific level of freshness. Also the Certificate Re-Issuance mechanism (along with the extensions proposed by (Yum, Kang et al. 2003) supports adjustability; dependent entities can set freshness requirements to be met by the CSI provider.

All other CSI mechanisms cannot adjust the freshness of CSI they provide based on requirements set by the dependent entities.

Table 31 : Evaluating Adjustability

CSI Mechanism	Adjustability
CRL	✗
Sliding Window DeltaCRL	✗
Freshest CRL	✗
Redirect CRL	✗
Indirect CRL	✗
Enhanced CRL Distribution Points	✗
Augmented CRL	✗
Efficient Fault-Tolerant Certificate Revocation	✗
Positive CSI (Suicide Notes)	✓
Self-Controlled Positive CSI	✗
Freshness-constrained Revocation Authority	✓
Efficient Long-term Validation of Digital Signatures	✗
Certificate Revocation Status	✗
Certificate Revocation Trees	✗

Online Certificate Status Protocol (OCSP)	✗
Certificate Re-issuance	✓

3.3.3 Security

Features discussed in this section are the ones that would have to be met to comply with CSI security requirements set forth in Annex II of the EU Directive on a Community Framework for Electronic Signatures, concerning the "secure directory" and the "secure revocation service".

3.3.3.1 CSI disseminator authentication

CSI disseminator authentication is important for dependent entities, in order for them to verify they are getting CSI from a trusted source that will not try to deceive them (e.g. reveal less fresh CSI than what is available). This is mostly critical in CSI mechanisms that do not provide *time-complete* CSI (see Section 3.2.2.3).

CRL-based mechanisms do not enforce authentication of the CSI disseminator in any way. CSI disseminator authentication could be used (e.g. LDAP or HTTP over SSL with verification of the LDAP or HTTP Server certificate) but this is a choice the CRL-based mechanism implementers would do, rather than a mandatory feature described in CRL mechanisms themselves. The same applies for the Freshness-constrained Revocation Authority mechanism, the Efficient Long-term Validation mechanism, CRS, CRT, OCSP, Positive CSI and the Certificate Re-Issuance mechanism.

The Efficient Fault-tolerant Certificate Revocation mechanism does not provide a specific CSI retrieval function; therefore CSI disseminator authentication is not applicable in this case.

In the case of the Self-Controlled Positive CSI mechanisms, CSI is handed over to the dependent entities from the certificate holder as a signed token; therefore, CSI disseminator authentication is also not applicable.

Table 32: Evaluating CSI Disseminator Authentication	
CSI Mechanism	**CSI Disseminator Authentication**
CRL	✗
Sliding Window DeltaCRL	✗
Freshest CRL	✗
Redirect CRL	✗
Indirect CRL	✗
Enhanced CRL Distribution Points	✗
Augmented CRL	✗
Efficient Fault-Tolerant Certificate Revocation	N/A
Positive CSI (Suicide Notes)	✗
Self-Controlled Positive CSI	N/A
Freshness-constrained Revocation Authority	✗
Efficient Long-term Validation of Digital Signatures	✗
Certificate Revocation Status	✗
Certificate Revocation Trees	✗
Online Certificate Status Protocol (OCSP)	✗
Certificate Re-issuance	✗

3.3.3.2 CSI integrity and authenticity

In most cases the integrity and authenticity of CSI, while in transit or while stored at the dependent entities' local storage, is protected through the digital signatures of the CSI authority.

CRL-based mechanisms, the Positive CSI mechanism, the Freshness-Constrained Revocation Authority mechanism, the Efficient long-term validation mechanism, CRS, CRT, OCSP and the Certificate

Re-Issuance mechanism protect CSI integrity and authenticity through the signature of the entity that generated CSI.

The Efficient Fault-Tolerant Certificate Revocation mechanism protects CSI integrity and authenticity because CSI is signed with the private key corresponding to the certificate to be revoked. In addition, the Self-Controlled Positive CSI mechanism protects (positive) CSI as this is signed with the private key corresponding to the certificate that has not yet been revoked.

Table 33: Evaluating CSI Integrity & Authenticity

CSI Mechanism	**CSI Integrity & Authenticity**
CRL	✓
Sliding Window DeltaCRL	✓
Freshest CRL	✓
Redirect CRL	✓
Indirect CRL	✓
Enhanced CRL Distribution Points	✓
Augmented CRL	✓
Efficient Fault-Tolerant Certificate Revocation	✓
Positive CSI (Suicide Notes)	✓
Self-Controlled Positive CSI	✓
Freshness-constrained Revocation Authority	✓
Efficient Long-term Validation of Digital Signatures	✓
Certificate Revocation Status	✓
Certificate Revocation Trees	✓
Online Certificate Status Protocol (OCSP)	✓
Certificate Re-issuance	✓

3.3.3.3 CA compromise

Most of the mechanisms we examine do not meet this criterion. If the CA is compromised, the entity who gained control of the respective CA keys is able to revoke certificates or issue new ones at will, until the CA compromise information reaches the dependent entities through an Authority Revocation List (ARL) or another, possibly out-of-band mechanism (e.g., in case the CA private key is no longer available to the CA personnel and it is only available to the entity who illegally gained control of it).

The Efficient Fault-Tolerant Certificate Revocation mechanism is not prone to CA compromise, since it is the certificate holder who produces CSI. The same applies for the Self-Controlled Positive CSI mechanism (but not for the Positive CSI mechanism since a CA key compromise would enable the attackers to create a Suicide Bureau keypair and certificate and issue "bills of health" as if they were Suicide Bureaus).

Table 34: Evaluating CA Compromise

CSI Mechanism	**CA Compromise**
CRL	✗
Sliding Window DeltaCRL	✗
Freshest CRL	✗
Redirect CRL	✗
Indirect CRL	✗
Enhanced CRL Distribution Points	✗
Augmented CRL	✗
Efficient Fault-Tolerant Certificate Revocation	✓
Positive CSI (Suicide Notes)	✗
Self-Controlled Positive CSI	✓

Freshness-constrained Revocation Authority	✗
Efficient Long-term Validation of Digital Signatures	✗
Certificate Revocation Status	✗
Certificate Revocation Trees	✗
Online Certificate Status Protocol (OCSP)	✗
Certificate Re-issuance	✗

3.3.3.4 RevA compromise

For CRL-based mechanisms, the fact that the Revocation Authority key has been compromised can be communicated through an Authority Revocation List (ARL), thus minimizing the consequences of such a compromise. This is true, only in the case that the RevA is operated by a different entity (one having a different keypair) than the CA.

In the case of the Efficient Fault-Tolerant Certificate Revocation and Self-Controlled Positive CSI, compromise of RevA equals compromise of the certificate holder, since it is him that produces the CSI, therefore such a compromise will be communicated to dependent entities once the certificate holder produces and disseminates CSI.

CRS is not prone to RevA compromise since the entity having compromised the private key will not –supposedly– have also compromised the database with the initial values of the hash chains that is maintained by CRS. However, if the key used to sign the CRS root is compromised then an ARL could be used to disseminate this information.

The ARL-based mechanism minimizing the effects of a Revocation Authority key compromise being used in CRL-based CSI mechanisms could also be used for all other mechanisms, although it is not clearly described in the papers these mechanisms are presented.

Table 35: Evaluating RevA Compromise	
CSI Mechanism	**RevA Compromise**
CRL	✓
Sliding Window DeltaCRL	✓
Freshest CRL	✓
Redirect CRL	✓
Indirect CRL	✓
Enhanced CRL Distribution Points	✓
Augmented CRL	✓
Efficient Fault-Tolerant Certificate Revocation	✓
Positive CSI (Suicide Notes)	✓ (if ARLs are additionally used)
Self-Controlled Positive CSI	✓
Freshness-constrained Revocation Authority	✓
Efficient Long-term Validation of Digital Signatures	✓ (if ARLs are additionally used)
Certificate Revocation Status	✓ (if ARLs are additionally used)
Certificate Revocation Trees	✓ (if ARLs are additionally used)
Online Certificate Status Protocol (OCSP)	✓ (if ARLs are additionally used)
Certificate Re-issuance	✓ (if ARLs are additionally used)

3.3.3.5 Contained functionality

In the case of CRL-based mechanisms, if the Revocation Authority uses a keypair other than that of the CA, then compromise of the corresponding private key does not enable the entity that compromised the key to issue new certificates. The same applies for OCSP, if the authority that disseminates CSI is a Trusted Responder or a CA Designated Responder (Myers, Ankney et al. 1999), using a key other than that of the CA. However, previously revoked or

suspended certificates can be made valid again. The same applies for the Positive CSI, the CRT and the Certificate Re-Issuance mechanisms.

The Efficient Fault-Tolerant Certificate Revocation, the Self-Controlled Positive CSI, CRS mechanisms also meet this criterion, since the RevA key cannot be used to issue new certificates, and at the same time cannot be used to make valid previously revoked or suspended certificates.

Table 36: Evaluating Contained Functionality

CSI Mechanism	Contained Functionality
CRL	✓ (but revoked/suspended certificates can be made valid again)
Sliding Window DeltaCRL	✓ (but revoked/suspended certificates can be made valid again)
Freshest CRL	✓ (but revoked/suspended certificates can be made valid again)
Redirect CRL	✓ (but revoked/suspended certificates can be made valid again)
Indirect CRL	✓ (but revoked/suspended certificates can be made valid again)
Enhanced CRL Distribution Points	✓ (but revoked/suspended certificates can be made valid again)
Augmented CRL	✓ (but revoked/suspended certificates can be made valid again)
Efficient Fault-Tolerant Certificate Revocation	✓
Positive CSI (Suicide Notes)	✓ (but revoked/suspended certificates can be made valid again)
Self-Controlled Positive CSI	✓
Freshness-constrained Revocation Authority	✓ (but revoked/suspended certificates can be made valid again)
Efficient Long-term Validation of Digital Signatures	✓ (but revoked/suspended certificates can be made valid again)
Certificate Revocation Status	✓
Certificate Revocation Trees	✓ (but revoked/suspended certificates

	can be made valid again)
Online Certificate Status Protocol (OCSP)	✓ (but revoked/suspended certificates can be made valid again)
Certificate Re-issuance	✓ (but revoked/suspended certificates can be made valid again)

3.3.3.6 Availability

CRL-based mechanisms could split CSI in partitions available from different repositories (e.g. using the *cRLDistributionPoints* extension) in order to protect the availability of CSI. Furthermore, if LDAP Directories are used as CSI repositories, LDAPv3 (Chadwick 2002) replication and referral mechanisms could also be used to meet the same goal. Also, other network-level techniques like DNS round-robin (Katz E.D., Butler M. and R. 1994) could be used in order to have more than one repository (LDAP or other) store the same kind of CSI and make it available to dependent entities.

The Efficient Fault-Tolerant Certificate Revocation mechanism has been built from the start to provide highly available CSI.

The Self-Controlled Positive CSI mechanism disseminates CSI along with the actual data that generates the need for CSI verification, therefore availability is not an issue in this case.

The rest of the CSI mechanisms need to operate some sort of online network service to provide CSI. In this case, ensuring high availability can be more difficult since it is not only multiple repositories that need to be made available but also multiple instances of the related CSI service. OCSP could be made highly available, using the *serviceLocator* extension and the mirroring of the CSI repositories used by OCSP. (Koga, Ryou et al. 2004) further enhance OCSP, proposing a way to do pre-production of OCSP Responses, as a means to enable OCSP to deal with Denial of Service attacks and OCSP response replay attacks.

Suicide Bureaus could also be made high available using existing network mechanisms for high availability (e.g. DNS round-robin) but this would require replication of the Suicide Bureau services and the related CSI repositories to different servers. The same applies for the other mechanisms.

Table 37: Evaluating CSI Availability

CSI Mechanism	CSI Availability
CRL	✓
Sliding Window DeltaCRL	✓
Freshest CRL	✓
Redirect CRL	✓
Indirect CRL	✓
Enhanced CRL Distribution Points	✓
Augmented CRL	✓
Efficient Fault-Tolerant Certificate Revocation	✓
Positive CSI (Suicide Notes)	✓ (supports high availability, but at a high cost)
Self-Controlled Positive CSI	N/A
Freshness-constrained Revocation Authority	✓ (supports high availability, but at a high cost)
Efficient Long-term Validation of Digital Signatures	✓ (supports high availability, but at a high cost)
Certificate Revocation Status	✓ (supports high availability, but at a high cost)
Certificate Revocation Trees	✓ (supports high availability, but at a high cost)
Online Certificate Status Protocol (OCSP)	✓
Certificate Re-issuance	✓ (supports high availability, but at a high cost)

3.4 Summary

We provide the reader with a summary of our findings presented in Section 3. Older versions of the work presented in Section 3 have been published in (Iliadis, Gritzalis, Spinellis et al. 2003) and (Iliadis, John, Spinellis, Diomidis et al. 2000).

We have initially presented an evaluation framework for CSI mechanisms (Section 3.2). The three pillars of our evaluation mechanism are the management features of a mechanism, its performance and the security features it supports. The criteria our evaluation framework comprises of are the following:

- Management

1. Feedback
2. Transparency
3. Delegation of revocation
4. Delegation of CSI dissemination
5. Delegation of certificate path validation
6. Referral capability
7. Revocation Reasons
8. Notification of revocation or suspension
9. Ability of dependent entity to evaluate the CSI mechanism before trusting it
10. Completeness

- Performance

1. Timeliness of CSI
2. Freshness of CSI

3. Bounded revocation
4. Emergency CSI capability
5. Scalability
6. Adjustability

- Security

1. CSI disseminator authentication
2. CSI integrity and authenticity
3. CA compromise
4. Revocation Authority (RevA) compromise
5. Contained functionality
6. Availability

We proceeded with evaluating all CSI mechanisms presented in Section 2.3, using the aforementioned evaluation framework. We include in this section three tables (Table 38 : Evaluation of CSI Mechanisms according to Management Criteria, Table 39 : Evaluation of CSI Mechanisms according to Performance Criteria and Table 40 : Evaluation of CSI Mechanisms according to Security Criteria) presenting the overall evaluation of all CSI mechanisms according to the three sets of criteria our evaluation framework contains. There is no meaning in trying to declare a "best" or "worst" CSI mechanism since it is obvious that some CSI mechanisms succeed where others fail and vice versa. On the contrary, the user requirements in real-life PKI implementations can be so diverse that a CSI mechanism could be the right one for a case and the wrong one for another. However, we provide the reader with some remarks regarding the overall comparative evaluation of the CSI mechanisms based on our evaluation framework.

CRS and OCSP are the only mechanisms that meet the *feedback* and *completeness* evaluation criteria, provided that these mechanisms also employ the extensions that have been proposed for them (see Sections 2.3.3.1 and 2.3.3.3). Even though CRS and OCSP meet the *completeness* criterion and include the revocation reasons in the CSI replies, they do not include the revocation reasons in the status validation process.

Another management criterion met only by a few mechanisms is the *referral* one; only OCSP, Redirect CRLs and Indirect CRLs meet this criterion. Also, the *delegation of certificate path validation* criterion is only, partially, met by OCSP.

The *transparency, notification of revocation and ability to evaluate CSI mechanism* evaluation criteria are not met by any CSI mechanism. The lack of transparency means that all CSI mechanisms require some level of interaction with the dependent entity to function properly and they also require some level of knowledge and security consciousness regarding certificate revocation from the dependent entity, while this is rarely the case, i.e. dependent entities rarely are knowledgeable and sensitive enough about these issues.

The fact that no mechanism notifies the certificate holder that his certificate has been revoked (if the CSI mechanism allows an entity to revoke another entity's certificate) is also an issue since the certificate holder may still be using an invalid certificate to conduct business. Furthermore dependent entities, being the entities that actually take the risk by trusting a specific CSI mechanism and the results it offers to them, should be able to evaluate it and agree to use it or ask for another one to be provided to them; this feature is not provided by any CSI mechanism as well.

The management criteria most mechanisms seem to meet are the *delegation of CSI dissemination* and (on a lesser scale) the *delegation of revocation*.

Table 38 : Evaluation of CSI Mechanisms according to Management Criteria

CSI Mechanism	Feedback	Transparency	Delegation of revocation	Delegation of CSI Dissemination	Delegation of certificate path validation	Referral capability	Revocation Reasons	Notification of revocation	Ability to evaluate CSI mechanism	Completeness
CRL	×	×	✓	✓	×	×	✓	×	×	×
Sliding Window DeltaCRL	×	×	✓	✓	×	×	✓	×	×	×
Freshest CRL	×	×	✓	✓	×	×	✓	×	×	×
Redirect CRL	×	×	✓	✓	×	✓	✓	×	×	×
Indirect CRL	×	×	✓	✓	×	✓	✓	×	×	×
Enhanced CRL Distribution Points	×	×	✓	✓	×	×	✓	×	×	×
Augmented CRL	×	×	✓	✓	×	×	✓	×	×	×
Efficient Fault-Tolerant Certificate Revocation	×	×	Partial	Partial	×	×	×	N/A	×	×
Positive CSI (Suicide Notes)	×	×	✓	✓	Not Applicable (there is no certificate path that needs validation)	×	×	×	×	×

Self-Controlled Positive CSI	×	×	Partial	✓	Not Applicable (there is no certificate path that needs validation)	×	×	N/A	×	×
Freshness-constrained Revocation Authority	×	×	✓	✓	×	×	×	×	×	×
Efficient Long-term Validation of Digital Signatures	×	×	×	✓	Not Applicable (there is no certificate path that needs validation)	×	×	×	×	×
Certificate Revocation Status	✓	×	×	✓	×	×	✓	×	×	✓
Certificate Revocation Trees	×	×	✓	✓	×	×	×	×	×	×
Online Certificate Status Protocol	Partial	×	×	✓	Partial	✓	✓	×	×	Partially

(OCSP)										
Certificate Re-issuance	×	×	×	×	×	×	×	×	×	×

It seems that the most scalable CSI mechanisms, offering fresh CSI at the same time, are the Self-Controlled Positive CSI and OCSP, followed by CRS, CRT and Positive CSI.

The timeliness is met by most mechanisms except CRL-based ones; the only CRL-based mechanism that meets the timeliness criterion is Freshest CRL.

The only mechanisms that meet the *adjustability* criterion are Positive CSI, Freshness-constrained Revocation Authority and Certificate Re-issuance, thus giving a chance to the entity that actually takes a risk by trusting CSI it is provided with (the dependent entity) to adjust its freshness.

Table 39 : Evaluation of CSI Mechanisms according to Performance Criteria

CSI Mechanism	**Timeliness of CSI**	**Freshness of CSI**	**Bounded Revocation**	**Emergency CSI capability**	**Scalability**	**Adjustability**
CRL	×	Low	✓	×	Low	×
Sliding Window DeltaCRL	×	Low	✓	×	Low	×
Freshest CRL	✓	Medium	✓	×	Low	×
Redirect CRL	×	Low	✓	×	Medium	×
Indirect CRL	×	Low	✓	×	Low	×
Enhanced CRL Distribution Points	×	Low	✓	×	Medium	×

Augmented CRL	×	Low	✓	×	Medium	×
Efficient Fault-Tolerant Certificate Revocation	✓	High	×	✓	Medium	×
Positive CSI (Suicide Notes)	✓	High	×	✓	Medium	✓
Self-Controlled Positive CSI	✓	High	✓	✓	High	×
Freshness-constrained Revocation Authority	✓	Medium	×	×	Medium	✓
Efficient Long-term Validation of Digital Signatures	✓	Medium	×	✓	Medium	×
Certificate Revocation Status	×	Medium	✓	×	High	×
Certificate Revocation Trees		Medium	✓	×	High	×
Online Certificate Status Protocol (OCSP)	✓	High	✓	✓	High	×
Certificate Re-issuance	×	Medium	✓	×	Medium	✓

CSI disseminator authentication is not supported, or rather not enforced, in any of the CSI mechanisms; external authentication mechanisms could be used but this is at the discretion of the implementers of the system hosting the CSI mechanism.

All CSI mechanisms protect with one way or the other *CSI integrity* and most of them meet the *Revocation Authority Compromise* criterion, provided that Authority Revocation Lists are used.

The only mechanisms that meet the *CA compromise* criterion (i.e. minimise the impact of a CA key compromise) and the c*ontained functionality* criterion (minimise the impact of a Revocation Authority key compromise) are the Efficient Fault-Tolerant Certificate Revocation and the Self-Controlled Positive CSI. Other mechanisms could also partially meet the *contained*

functionality criterion, i.e. revoked/suspended certificates could be made valid again.

Table 40 : Evaluation of CSI Mechanisms according to Security Criteria

CSI Mechanism	CSI disseminator authentication	CSI integrity	CA compromise	Revocation Authority compromise	Contained functionality	Availability
CRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Sliding Window DeltaCRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Freshest CRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Redirect CRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Indirect CRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be	✓

					made valid again)	
Enhanced CRL Distribution Points	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Augmented CRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Efficient Fault-Tolerant Certificate Revocation	N/A	✓	✓	✓	✓	✓
Positive CSI (Suicide Notes)	×	✓	×	✓ (if ARLs are additionally used)	✓ (but revoked/suspended certificates can be made valid again)	✓ (supports high availability, but at a high cost)
Self-Controlled Positive CSI	N/A	✓	✓	✓	✓	N/A
Freshness-constrained Revocation Authority	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓ (supports high availability, but at a high cost)
Efficient Long-term Validation of Digital Signatures	×	✓	×	✓ (if ARLs are additionally used)	✓ (but revoked/suspended certificates can be made valid again)	✓ (supports high availability, but at a high cost)
Certificate Revocation Status	×	✓	×	✓ (if ARLs are additionally used)	✓	✓ (supports high availabil

						ity, but at a high cost)
Certificate Revocation Trees	×	✓	×	✓ (if ARLs are additionally used)	✓ (but revoked/suspended certificates can be made valid again)	✓ (supports high availability, but at a high cost)
Online Certificate Status Protocol (OCSP)	×	✓	×	✓ (if ARLs are additionally used)	✓ (but revoked/suspended certificates can be made valid again)	✓
Certificate Re-issuance	×	✓	×	✓ (if ARLs are additionally used)	✓ (but revoked/suspended certificates can be made valid again)	✓ (supports high availability, but at a high cost)

4 ADoCSI

4.1 Introduction

The explosion of Internet usage and the emergence of e-commerce and e-business introduced a completely new modus operandi regarding electronic transactions. The characteristics of this modus operandi are briefly presented in the following list:

1. transacting entities do not belong in a closed, controlled group (in contrast to e.g. bank customers and banks),
2. transactions do not occur within private networks (such as the banking network or other private, corporate networks); instead, public, unsafe and uncontrolled networks are used,
3. a dynamic, constantly increasing, group of entities has the capability to offer services over these public networks; these entities usually do not possess the technical skills and know-how to perform a detailed analysis of the security requirements concerning the services they provide,
4. users do not necessarily comprehend the underlying complexity of the transaction systems they use; most of them are not sensitive or cautious enough when it comes to security decisions (Smith 2003; Spruit M. 1998; Stanton, Stam, Mastrangelo et al. 2005).

Contemporary electronic transactions take place in open networks, where there is no trusted communication path and where communicating entities may not know each other before conducting business. These facts lead into an *authentication gap*. It is not easy for security unaware entities, unfamiliar to each other, communicating over untrusted networks to authenticate themselves. Public Key Infrastructure (PKI) has been presented as the way to fill this gap.

Initially, Certificate Status Information (CSI) mechanisms were a technical requirement to the operation of PKI. However, legal efforts, including the European Directive on Electronic Signatures (EU Parliament 1999) and the respective harmonisation on national levels, have elevated this requirement to a legal one. This change is being reflected to Certification Practice Statements and Certificate Policies of operational Certification Authorities.

There are advantages and disadvantages in using any one specific mechanism out of those presented in Section 2.3. However, all these CSI mechanisms require some level of security awareness, from the dependent entity. The latter has to be able to comprehend and then use the CSI location and validation functions in order to verify the status of a certificate it has been presented with (see Sections 3.2.1.1 and 3.2.1.2).

Entities transacting over the Internet, both dependent ones and certificate holders, are rarely security experts. One should not expect from them to comprehend the security measures put in place in order to protect their transactions. Neither the certificate holders (authenticating entities or signers) nor the dependent entities should have to contribute manually to the location and validation of Certificate Status Information (CSI). That should not be the job of certificate holders or dependent entities, as it is not the job of real-world merchants to locate or validate credit card revocation information.

Certainly, one could argue that it should be the responsibility of certificate holders to provide dependent entities with the most current CSI (Rivest 1998). Rivest also claims that it should be the responsibility of the dependent entities to set the requirements CSI they are presented with has to meet. Although entities that offer services over the Internet can very well be security-aware, much more than the certificate holder on an average, that does not make them security experts. Furthermore, dependent entities as well as certificate holders are usually not so knowledgeable, motivated and perceptive (Smith 2003;

Spruit M. 1998; Stanton, Stam et al. 2005) regarding security controls. They don't always follow good security practice. Therefore, certificate holders and dependent entities are bound to make security mistakes or overlooks, consciously or unconsciously, leading to direct or indirect failing. Human failing is in many cases the reason for the malfunction of a security mechanism, the incorrect use of a security service or a security breach (Smith 2003; Spruit and Looijen 1996; Stanton, Stam et al. 2005).

Furthermore, PKI users sometimes have to make security decisions in order to facilitate certificate path validation. The number of decisions users have to take and the number of questions they have to answer has to be restricted; also the way these questions are posed must be studied carefully, otherwise security layer transparency (see Sections 3.2.1.2 and 3.3.1.2) will cease to exist and users may fall back to a position where they should strive to understand the mechanics of CSI mechanisms. This often leads computer users to direct or indirect failing (Spruit M. 1998) or 'authorisation fatigue'.

Moreover, client applications have to be aware of the mechanisms required in order to conduct CSI queries. This includes certificate path validation, certificate status checking and on the ability to parse and appropriately evaluate the various certificate attributes related to the CSI. This results to fat clients, i.e. client applications that must include all this functionality. There are several drawbacks to this. A solution seems to be for client applications to delegate certificate path validation and CSI querying to a specialized client application that provides these services is to all client applications requiring them.

PKI seems to be gaining users' confidence; however, PKI-enabled services are not yet security-transparent enough. Users do not always comprehend the complexities of security mechanisms, and they should not be obliged to, after all. *Transparency* of security mechanisms is a solution to these problems.

However, a transparent mechanism for the dissemination of CSI must not lack the security features provided by the other non-transparent mechanisms. We claim an agent-based mechanism could meet those requirements. An agent-based mechanism responsible for locating and validating CSI, leveraging this burden from dependent entities and certificate holders.

In this paper, we present the prototype of an alternative mechanism for disseminating certificate status information, which we call ADoCSI (Alternative mechanism for the Dissemination Of Certificate Status Information (Iliadis, Gritzalis and Gritzalis 2003), (Iliadis 1999). Our mechanism is based on Software Agents, and its use promotes transparency in PKI-enabled services over the Internet.

4.2 Agents

Agents have recently been (and still are) a major research topic, both for the academia and for the industry, for various reasons (Petrie 1997). The researchers, trying to define accurately their view and specifications of Agents, have come up with a variety of alternative names or adjectives. A short review of those follows (Nwana 1996), (Chess, Grosof, Harrison et al. 1995), (Cugola, Ghezzi, Picco et al. 1997), (Franklin and Graesser 1997; Perdikeas, Chatzipapadopoulos, Venieris et al. 1999), (Perdikeas, Chatzipapadopoulos et al. 1999):

1. *Static.* Agents that do not transport themselves to execution environments other than the one they were originally,
2. *Mobile.* These agents can transport themselves to other execution environments in order to communicate locally with other agents and at the same time they can preserve the external state they had in the previous location,

3. *Deliberative.* Agents that contain internal reasoning and collaborate with other agents in order to achieve their goals,

4. *Autonomous.* Agents that have states and goals and do not need user feedback or interaction (except perhaps an initial user feedback) in order to carry out their tasks and reach their goals,

5. *Co-operative.* Agents that co - operate with other agents in order to reach their goals,

6. *Interface* (also called User). Agents that sit between another agent and a user. The user communicates with the interface agent only, and the latter transforms the user's goals into formal statements and procedures and assigns those to the other agents; the interface agent launches agents on behalf of the user, co-ordinates them and provides the results of their work back to the user,

7. *Heterogeneous.* Agents that encompass several of the characteristics of the other agents and can talk to other agents using a standard agent communication language, thus providing for interoperability between agents.

This wide spectrum of names results in semantic overlap and confusion. Furthermore, since there are no criteria for 'agenthood', some researchers came up with software constructs which they called Agents, even though they do not have any of the functionality that characterises Agents (Petrie 1997). Genesereth (Genesereth and Ketchpel 1994) argues that the criterion for 'agenthood' is a behavioural one; a software application is an agent if and only if it communicates with other software applications, using an Agent Communication Language (ACL). However, the lack of a widely accepted standard practice that could act as a reference point renders these taxonomies rather inaccurate or at least not in accordance with other ones.

We should clarify at this point that in this paper we are not interested in any Artificial Intelligence properties Agents may have. We are only interested in the distributed functionality they offer, because we can use this functionality in order to meet some of the CSI dissemination criteria (Iliadis, John, Spinellis, Diomidis et al. 2000) that other CSI dissemination mechanisms do not meet. The agents we describe in this document are mobile, deliberative, heterogeneous and (some of them) interface agents. However, since we are not interested in establishing a new name for the agents we discuss, we will be referring to them as CSI Agents or simply Agents.

The agents we will describe must use a standard way of talking to each other. One of the examples of languages used by agents to communicate is the Knowledge Query and Manipulation Language (KQML) (Labrou and Finin 1997), (Finin, Weber, Wiederhold et al. 1993), which describes programmatic content and the Knowledge Interchange Format (KIF) (Genesereth and Fikes 1992), (Chess, Grosof et al. 1995) which can be used to form knowledge representations, express tasks and goals. Another example is the more recent Agent Communication Language (ACL) of the Foundation for Intelligent Physical Agents (FIPA) (Erdur and Dikenelli 2002).

The agents we will be using must meet the following requirements:

1. they must be able to suspend execution and resume it at another execution environment,
2. they must retain their state, when transporting themselves to other execution environments,
3. they must be able to create child agents and deploy them,
4. they must be able to select a network location, out of a list of locations, with the least network congestion,

5. they must be able to communicate the retrieved information back to their owner or to their owner's application that spawned the agent.

We should note here that we are not going to examine the mechanics of the implementations of Agent infrastructures. Implementations exist (Frost and Cutkosky 1996), (IBM 2002) that meet the requirements set in this section, as far as Agent design and operation is concerned.

4.3 ADOCSI

In this section, we present the prototype of an alternative mechanism for disseminating CSI. We shall be calling this mechanism *ADoCSI* (Alternative Dissemination of Certificate Status Information). The Agents described in this mechanism exchange CSI, in one (or more) of the formats described in Section 2.3. Our mechanism makes extensive use of CSI mechanisms' properties and functionality we have presented in Section 3.2, and provides at least one additional feature: the *transparency* in disseminating CSI (see Section 3.2.1.2).

Let us consider the entities that take part in our mechanism:

1. **CA**: The three CAs in Figure 2 (page 132) offer the standard services a CA should offer. In addition to that, they develop CA-CSI Agents and User-CSI Agents, which are used throughout ADoCSI for the dissemination of CSI
2. **Agent Meeting Places** (AMP) (Chess, Grosof et al. 1995) : They are also called Agent Platforms (AP) (Foundation for Intelligent Physical Agents 2004). These provide the physical infrastructure where agents can be transported to and communicate with other agents that are already there. In ADoCSI, AMPs must also possess an X.509v3 certificate, which will be used in order to authenticate the AMPs against the Agents that visit them. If CAs decide to have their CA-CSI Agents installed in more than one

AMP, the agents sent by the dependent entities in order to retrieve CSI can choose to visit the AMP that is currently less occupied, i.e. it currently serves the least number of CSI requests and the communication lines leading to that AMP are less congested to others. AMPs could be operated and controlled by CAs themselves or they could be operated by a separate Trusted Third Party.

3. **Directory Facilitator Agent (DF)**: These agents are resident (static) within the AMPs. They keep a complete and up - to - date register (Genesereth 1993) of the services offered by Agents residing in the AMP and the methods other Agents should use in order to communicate with the former. In ADOCSI, Directory Facilitators also undertake the task of perform verification of the digital signatures on Agents and also maintain a registry of CSI available on the AMP by the CA-CSI Agents the AMP hosts. Also, if a CA-CSI Agent contains pointers to other AMPs where CA-CSI Agents of the same CA exist, it lets the DF know about it so that the DF can forward that information too, to User-CSI Agents, if needed.
4. **Dependent entity**: An entity that wishes to verify the validity of the certificate of another entity. The latter may have tried to authenticate (authenticating entity) against the dependent entity in order to access the services offered by the dependent entity, or the latter may have received offline, signed communication (e.g. e-mail) from another entity (signer). In both cases, the dependent entity wishes to verify the validity of the certificate presented by the aforementioned entities (authenticating entity or signer).
5. **Authenticating Entity**: An entity that attempts to authenticate against the dependent entity in order to gain access to the services offered by the latter.

6. **Signer**: An entity that has sent to the dependent entity a signed piece of information.

7. **Certification Authority Certificate Status Information (CA-CSI) Agent**: The CA develops this Agent and includes in it the latest available CSI, in one (or more) of the formats discussed in Section 2.3. CA-CSI Agents have short validity periods. Entities communicating with CA-CSI Agents can verify the validity period of such an Agent because the CA-CSI Agent contains a timestamp (time of CSI inclusion in the Agent) and a validity time period, and the Agent is signed with the CA private signing key. CAs issue new Agents if their validity period is close to the expiration date or if there is fresher CSI to be made available to dependent entities. Once a new Agent is issued, it is sent to the AMP. In case the Agents also serve CSI that is very frequently updated (e.g. Freshest CRLs), then this CSI can be stored externally, in the AMP, in a location where Agents can have access to. User-CSI Agents communicate with CA-CSI Agents at the AMPs in order to locate, validate and retrieve the CSI they are interested in. If CA-CSI Agents can deliver CSI in more than one ways or formats, then they negotiate with the requesting agents in order to deliver the CSI in the manner that meets best the CSI requirements set forth by the requesting agents, like CSI freshness and maximum volume of information. In order to increase the availability of the system, CAs may decide to have their CA-CSI Agents installed in more than one AMP, thus allowing for load-balancing of the communication traffic due to CSI requests.

8. **User Certificate Status Information (User-CSI) Agent**: The CAs also develop these Agents, sign them and hand them over to dependent entities, i.e. any entity that wishes to use the agent-based scheme in order to retrieve CSI. This Agent is also signed by the dependent entity every time the Agent is sent to locate and retrieve CSI. This signature is used

throughout the Agent's life in order to verify its integrity and origin. As soon as User-CSI Agents retrieve the CSI they are looking for, they return to the dependent entity that launched them in order to inform the entity of the retrieved CSI. However, if the dependent entity is not available at that time (e.g. the dependent entity my have gone temporarily offline, for some reason) then User-CSI Agents can temporarily suspend their execution and have themselves stored in a persistent storage at the AMP. The AMP will revive the User-CSI Agents at a later, predefined, amount of time so that they can continue their journey. When User-CSI Agents discuss with CA-CSI Agents in order to retrieve the CSI they were sent to find, they negotiate with them on the way CA-CSI Agents will deliver CSI. The negotiation parameters the User-CSI Agent uses are the ones he was informed about, from the Interface Agent, upon beginning his journey from the dependent entity to the AMPs.

9. **Interface Agent**: The Interface Agent (IA) interacts with the User-CSI Agent and the applications that the dependent entity is using. The Interface Agent launches a new User-CSI Agent whenever an application used by the dependent entity needs fresh CSI. Furthermore, it is the Interface Agent that installs the retrieved CSI at the local repository of the dependent entity, perform any certificate path validation needed and collaborate with the dependent entity's locally running applications in order to inform them of any progress or changes concerning the requested or retrieved CSI. Upon initial installation at the dependent entity's workstation (or other device) the Interface Agent pauses certain questions to the dependent entity in order to establish a set of parameters for CSI location, retrieval and validation. The IA instructs User-CSI Agents to retrieve the necessary CSI following some, or all, of the aforementioned parameters while being in its journey.

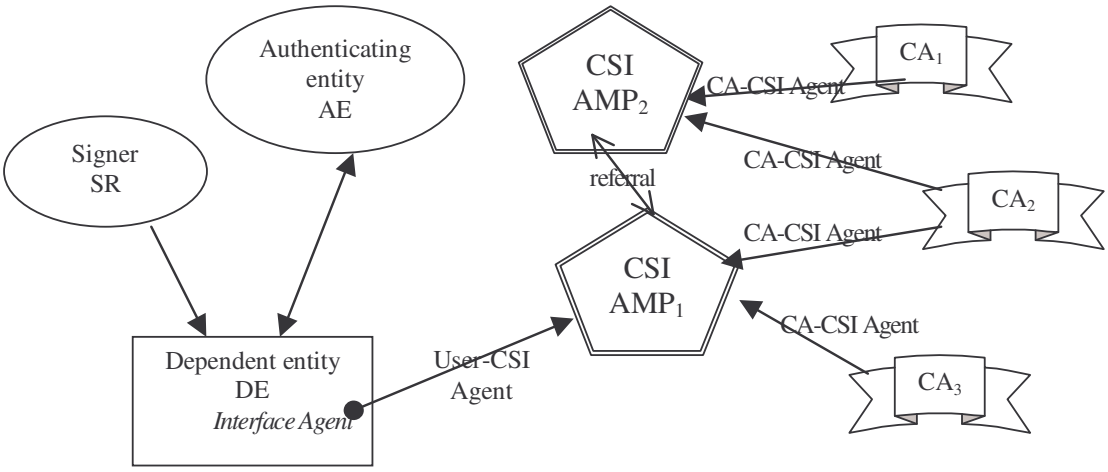

Figure 2: ADOCSI Infrastructure

4.3.1 AMP Location Function

The AMP location function can be implemented in the following redundant ways:

1. location information is stored at the CA, and the User - Agent will have to retrieve it before transporting itself to the AMP,
2. location information is stored in the CA-CSI Agent and it is given to the User-CSI Agent at the AMP,
3. location information is stored at the certificate of the authenticating entity (or signer) and the Interface Agent will have to retrieve it and deliver it to the User-CSI Agent before deploying it,
4. location information is stored in the CA certificate and the Interface Agent will have to retrieve it and deliver it to the User-CSI Agent before deploying it,
5. location information is stored in the User-CSI Agent.

The location of the AMP or AMPs that host the CSI-Agent of a CA can be stored in a repository maintained by that CA, such as the Directory or an existing highly available infrastructure like the Domain Name System (DNS). The location information should be signed by the CA in order to enable the

dependent entity to verify its integrity, unless the communication protocol supported by the CA repository includes integrity protection. If DNS were to be used, DNSSEC (Arends, Austein, Larson et al. 2005a; Arends, Austein, Larson et al. 2005b) could offer the required integrity protection. If this kind of location function is implemented, the User-CSI Agent will have to lookup the location information at the CA repository (be it a Directory of DNS/DNSSEC) and verify its integrity, before transporting itself to the AMP. A possible enhancement could be caching the location information in the local user environment, in order to avoid having the User-CSI Agent lookup the location information every time, before transporting itself to the AMP. The way to implement this location function using LDAP is obvious, since LDAP custom attributes could be used. In the case of DNS, the AMP location function could be implemented taking advantage existing DNS functionality: DNS A Records (Mockapetris 1987). The CA should operate a DNS Server that is authoritative for the domain it hosts its services in (as most, if not all, CAs already do) and should add to that DNS Server A records constructed in the following manner: *unique_id_of_cert.FQDN_of_CA*, where:

- *unique_id_of_cert* would be the serial number of the certificate issued by the CA, and
- *FQDN_of_CA* would be the Fully Qualified Domain Name of the CA.

The IP address inserted in the aforementioned DNS A record would be the IP address of the AMP that hosts the CA-CSI Agent containing CSI for the particular certificate. This AMP location method would be very economic since the existing DNS caching scheme would make it very scalable and highly available. A similar idea regarding DNS (and its use in directly locating CSI in the OCSP format) was proposed in (Egelman, Zaritsky and Jones) but it required a single, global, Root DNS Server for all CAs to be in

place for the mechanism to operate, rendering the proposed mechanism not very scalable and difficult to implement for practical reasons.

However, a CA may decide to stop trusting a specific AMP for hosting the CA-CSI Agent. In this case, the User-CSI Agent who is using the cached location information will not know about it. This is one of the cases where the short lifetime of CA-CSI Agents could prove to be useful. If a CA decides to trust no longer an AMP for hosting the CA-CSI Agent, the CA will no longer update the CA-CSI Agent in that AMP, therefore even if a User-CSI Agent visits the specific AMP, it will not meet a valid, i.e. non-expired, CA-CSI Agent.

A variation of the aforementioned AMP location function can be implemented by the CA in case it decides that online communication between the dependent entities and the CA should be avoided or at least minimised. The location of the AMP or AMPs that host the CA-CSI Agent can be stored inside the CA-CSI Agent himself. Certainly, the dependent entity (User-CSI Agent) would have to retrieve this information the first time from the CA itself. Once this is done and the User-CSI Agent knows which AMP to visit initially, it will cache at the local storage area of the dependent entity that information and update it whenever the CA-CSI Agent gives the User-CSI Agent new AMP location information. This implementation, however, obliges the CA to have always a CA-CSI Agent available at that specific AMP, even if that CA-CSI Agent only distributes a new AMP location to the visiting User-CSI Agents.

AMP location information can also be stored in the X.509v3 certificate delivered by the authenticating entity (or signer) to the dependent entity. The Interface Agent can retrieve that information from the certificate and deliver it to the User-CSI Agent, before sending the latter to the AMPs in order to retrieve CSI on the aforementioned certificate. In this case, the User-CSI

Agent does not have to find the AMP location from the CA or the CA-CSI Agent, therefore the communication and processing costs related to the actions of User-CSI Agent are decreased. However, the costs associated with distributing the certificates themselves are increased, because the certificates need to contain this additional information, i.e. the location of the AMP(s). If no other AMP location function except this were implemented by the CA then the dependent entity could be rendered prone to Denial of Service attacks; the location information could only be changed by the CA whenever a new certificate is issued (or an existing one renewed), therefore, if all AMPs mentioned in the certificate ceased offering services, or if the CA stops trusting them and therefore stops sending updated CA-CSI Agents to those AMPs, the User-CSI Agent of the dependent entity would not be able to locate an AMP to retrieve CSI from.

A variation of the aforementioned location function is to store the location information in the CA certificate. In this case, the costs associated with the communication and processing done by the User-CSI Agent are also decreased, and costs associated with distributing the certificates are not that increased since it is only the CA certificate that contains additional information, not all the certificates issued by the CA. However, the dependent entity is even more vulnerable to Denial of Service attacks, because the validity period of a CA certificate is longer than that of any other certificate issued by that CA.

A way to deal with possible DoS attacks related with the last two types of location functions would be to have the first type of location function operational as well. If the third or fourth type of location function fails (DoS), the first type of location function can provide the dependent entity with the AMP location that is needed.

User-CSI Agents should choose themselves the best AMP location function to use. This choice should be based on the following criteria:

1. which location functions are available,
2. which location function was used the last time the Agent was called to retrieve CSI,
3. whether the last AMP location function executed successfully or not, and
4. user preferences regarding the location function to be used.

Finally, the location information could be stored inside the User-CSI Agent itself. The Agent would not need to locate and retrieve the AMP location information. Its operation will be simpler and the size of the User-CSI mobile code application will be smaller, therefore communications costs related to the transport of the User-CSI Agent will be reduced.

4.3.2 CSI Location, Validation and Retrieval

Once the User-CSI Agent is in the AMP, it contacts the Directory Facilitator in order to locate the CA-CSI Agent that holds the CSI he is interested in. The User-CSI Agent has to form a query in a predetermined Agent Communication Language and send it to the Directory Facilitator, who will reply to the User-CSI Agent using the same ACL as well. If the CA-CSI Agent that contains the information the User-CSI Agent is interested in, does not reside in that AMP then the DF should refer the User-CSI Agent to another AMP.

DF maintains a database with all the CA-CSI Agents that reside in the AMP and the respective CSI they hold. Whenever a CA-CSI Agent is transported to the AMP by its respective CA, it has to contact the DF in order to inform the latter of the CSI it holds. The DF should also update the aforementioned database by polling the resident CA-CSI Agents. This ensures that even if the DF was unavailable at the time a new or updated

CA-CSI Agent reached the AMP and tried to contact the DF or if a CA-CSI Agent was removed abruptly (e.g. accidental cease of operation, perhaps due to a software bug) from the AMP, the DF will become aware of that and will have the database updated with the relevant information. A CA-CSI Agent may also have to be removed from the AMP because the CSI it contains is no longer valid or because the Agent has expired; if the CA-CSI Agent does not notify the Directory Facilitator for some reason, the latter will know about it in the next polling period.

Having located the CA-CSI Agents they seek, User-CSI Agents communicate with them and perform a series of queries, using a predetermined ACL. Since a CA-CSI Agent could deliver CSI in a number of different formats (see Section 2.3), User-CSI Agents will negotiate with the CA-CSI Agents in order to have the latter deliver the CSI in a way that meets best the requirements set by the dependent entity. The negotiation parameters that can be used include:

1. *Freshness of CSI*. The User-CSI Agent can negotiate on the freshness of the CSI it wants to retrieve, depending on the requirements set forth by the dependent entity and the capability of the CA-CSI Agent to deliver more (or less) fresh CSI. If the CA-CSI Agent is not able to deliver CSI as fresh as the User-CSI Agent requests, it could either deliver less fresh CSI or refer the User-CSI Agent to another CA-CSI Agent (probably at another AMP) that could probably deliver more fresh CSI. CAs could send at a normal rate new CA-CSI Agents to some AMPs (CA-CSI Agents containing relatively fresh CSI, that would normally be accepted by most dependent entities) and also send very frequently CA-CSI Agents to some few AMPs (CA-CSI Agents containing the freshest CSI possible),
2. *Volume of CSI*. The volume of the CSI received by a User-CSI Agent may vary, depending on the mechanism the CA-CSI Agent is using to deliver

that information. There are mechanisms (e.g. CRL) where the communicated CSI contains always more information than necessary (i.e. CSI on certificates that the dependent entity is not interested in),

3. *Estimated time of CSI delivery to the dependent entity*. The time that is estimated to take for the CSI to be delivered to the dependent entity varies, depending on the volume of the CSI, bandwidth constraints and dependent entity online availability. The User-CSI Agent should consider these factors and estimate the time it will take to carry the information back to the dependent entity. Then, the User-CSI Agent could negotiate with the CA-CSI Agent on the format of CSI to be delivered, to meet in the best possible way the requirement for timely delivery of CSI to the dependent entity,
4. *Delegation of certificate path validation*. This validation can be performed either by the CA-CSI Agent or the Interface Agent, depending on the mechanism used for delivering the CSI to the User-CSI Agent,
5. *Bounded revocation*. The User-CSI Agent may choose to prefer one CSI dissemination mechanism to another because the former supports bounded revocation (see Section 3.2.2.3).

Once the CA-CSI Agent delivers some CSI to the User-CSI Agent the latter verifies that the delivered CSI concerns the certificate it has been instructed to find CSI on. If it is so, the User-Agent transports itself back to the dependent entity. If the latter is not connected to the network at that point of time, the User-CSI Agent should be allowed to remain dormant at the AMP for a predetermined maximum amount of time, waiting for the dependent entity to connect back to the network. When the User-CSI Agent resumes operation (the AMP resumes operation of a dormant User-CSI Agent), and presuming that the dependent entity is back online, it should check again for fresher CSI and head back to the dependent entity. If the User-CSI Agent resumes

operation and the dependent entity is still offline, then it should be allowed to remain dormant again; however, the AMP should impose an upper limit of times a User-CSI Agent is allowed to suspend and resume operation, in order to avoid DoS attacks (e.g. multiple User-CSI Agents being sent by malicious entities and then going offline, in order to fill up the AMPs persistent storage space).

The ability of User-CSI Agents to suspend and resume operation at the AMP can also be exploited by dependent entities in order receive very fresh and timely CSI regarding certificates used in critical applications. For this functionality to be implemented, User-CSI Agents should visit the appropriate AMP and let the Directory Facilitator know that they wish to go dormant (suspend execution) and be waken up (resume execution) when a new CA-CSI Agent arrives from the specific CA they are interested in. The Directory Facilitator would allow a User-CSI Agent to go dormant for the aforementioned predefined maximum period of time (and number of times), thus the User-CSI Agent will have the chance to receive CSI regarding the specific certificate(s) and deliver it to the dependent entity in the timeliest possible manner.

4.3.3 Certificate Path Validation

Validation of a certificate path can be performed by the Interface Agent or it can be delegated to the CA-CSI Agent. In any case, the dependent entity should be able to verify the authenticity of the received CSI. If the certificate path has been validated by the CA-CSI Agent, the dependent entity should be able to verify that the received, processed, CSI is authentic; if the certificate path is to be validated by the Interface Agent the dependent entity should be able to verify the authenticity of the certificate path information received.

4.3.4 Interface Agent

The Interface Agent (IA) act as an intermediary between the User-CSI Agent and the PKI-aware applications a dependent entity uses. The IA has access to the certificate and CSI repository of the dependent entity (residing on the dependent entity's filesystem), which it updates as soon as new CSI becomes available. Furthermore, PKI-aware applications that reside at dependent entity's computer system communicate with the IA in order to request CSI on specific certificates. It is the IA that forms the appropriate CSI query, using a predetermined ACL, describing the CSI that has to be retrieved and the retrieval negotiation parameters. In addition, the IA may inform the User-CSI Agent on the location of the AMP to visit (see Section 4.3.1), if that information is stored in the certificate of the authenticating entity or signer, or in the respective CA certificate. After that, the IA spawns a User-CSI Agent and assigns it the task to look for the appropriate CSI, i.e. find the relevant CA-CSI Agent and discuss with him in order to retrieve the CSI. Once the User-CSI Agent returns to the dependent entity location, it communicates with the IA in order to deliver the CSI that has been retrieved. The IA must be able to identify the User-CSI Agent, once it comes back, verify his integrity, recall the query the Agent was assigned to execute and examine whether the User-CSI Agent did return with the requested CSI. If certificate path validation has not been delegated to the CA-CSI Agent, the IA will perform the certificate path validation or leave this task for the application itself, if the application supports certificate path validation. The IA will install the new CSI in the certificate and CSI repository of the dependent entity and inform the application of the status of the certificate in question.

4.3.5 ADoCSI Security

In this section, we present the threats ADoCSI faces, and the respective security services and mechanisms that have to be in place in order to deal

with these threats. The threats ADoCSI has to deal with are the following (Gritzalis and Spinellis; Jansen 2000; Wayne Jansen and Karygiannis 1999):

1. *Unauthorised modification or replacement of Agents.* An entity other than the CA could manage to modify a CA-CSI Agent before it reaches the AMP, or deploy one that has been developed by that entity itself, in order to have that CA-CSI Agent give incorrect or false CSI to User-CSI Agents. The same applies for User-CSI Agents.

2. *Unauthorised modification of information contained in the Agents.* CA-CSI and User-CSI Agents carry information, namely CSI. Moreover, User-CSI Agents carry information regarding the CSI they are looking for, on their way to AMPs. Protection of the integrity of both kinds of information is an issue. If CSI (or CSI query) integrity violation goes undetected, dependent entities will receive invalid CSI. If CSI query information is altered by third entities without them being detected, dependent entities will receive valid CSI, which might not include the CSI they were interested in, thus lead to believe that a certificate is still valid while it may be not.

3. *AMP masquerade.* An entity may attempt to masquerade as an AMP and attract User-CSI Agents. If a User-CSI Agent communicates with a (valid) CA-CSI Agent at an AMP that is controlled by an unauthorised, untrusted and possibly malicious entity, then that entity may manage to take control of the communication between the User-CSI Agent and the CA-CSI Agent and provide the former with false, inaccurate or partial CSI. Even if the two aforementioned Agents employ integrity mechanisms for the protection of their communication, while they are in the AMP, the AMP can always observe the Agents' actions, monitor the data they generate and manipulate the communication between them.

4. *Denial of Service.* Denial of Service attacks could be launched against AMPs, thus preventing dependent entities from obtaining CSI.

5. *Replay attacks.* A malicious entity might attempt to capture a User-CSI Agent while it is transported back to the dependent entity and send another User-CSI Agent in its place, an Agent that was sent at an earlier date by the same dependent entity. If this is achieved, the dependent entity might be led into believing that the Agent it received (and the information it carries) is authentic, and thus be lead to believe that a certain certificate is still valid while it may be not, because the received CSI will be old and possibly invalid. If that CSI has not been generated with a mechanism that supports bounded revocation, then the dependent entity will consider the invalid CSI it has received as valid. A malicious entity could also attempt to capture CA-CSI Agents on their way to an AMP and replace them with valid, but old, CA-CSI Agents of the same CA. If this replay attack is successful, then the User-CSI Agents would retrieve old, and possibly invalid, CSI. Finally, If an entity takes control of an AMP it could attempt to store in that AMP old CA-CSI Agents, with the purpose of disseminating old, and possibly invalid, CSI.
6. Malicious Agents. Signing the Agents with the private keys of the Agent developer or the entity that is sending the Agent does not guarantee that the Agent will not act in a malicious way either towards the AMP or towards other Agents.
7. Malicious AMPs. An AMP could interfere somehow with a CA-CSI Agent's actions in order to disclose false CSI to interested User-Agents.

The respective mechanisms that have to be put in place in order to deal with the aforementioned threats are analysed in the following sections.

4.3.5.1 Unauthorised modification or replacement of Agents

To deal with this threat, the integrity and origin of the CA-CSI Agents should be verifiable both by AMPs and by User-CSI Agents. Respectively, the integrity and origin of User-CSI Agents must be verifiable by AMPs and

CA-CSI Agents. This can be accomplished if Agents are signed (Gong and Schemers 1998; Zhang 1997) by the CA that sent them to the AMP (CA-CSI Agents), or to the dependent entity (User-CSI Agents). Both the AMP and the User-CSI/CA-CSI Agents should already trust that CA and have a trusted copy of that CA's certificate. We assume that the dependent entity launching the User-CSI Agent already trusts the CA that issued the CA-CSI Agent since the dependent entity is performing a CSI query on a certificate issued by that certificate. As for AMP (and the DF Agent residing on the AMP), this Agent must have received the CA certificate in a trusted way, upon agreeing with the CA to service its CA-CSI Agents.

The DF of that AMP and the User-CSI Agent must use the respective CA public key and verify the signature on the CA-CSI Agent. The AMP must contain the CA certificates of the CAs it has agreed to receive CA-CSI Agents from, in a repository local to the AMP. The User-CSI Agent must also contain the CA certificates that will be needed to verify the digital signatures on the CA-CSI Agents the User-CSI Agent will communicate with.

An entity other than the dependent entity may attempt to modify or replace the User-CSI or the CA-CSI Agent at the following points in time:

1. while the Agent is distributed to the dependent entities by the CA that developed it,
2. while the Agent is transported by the dependent entity to an AMP,
3. while the Agent is returning to the dependent entity from an AMP, or
4. while the Agent is distributed by the CA to the AMP.

The CA that developed the User-CSI Agent should sign that Agent as well, in order to deal with the aforementioned threats. If the signature of that CA is on the User-CSI Agent, the dependent entity, the DF and the CA-CSI Agent will be able to verify the integrity of the Agent once they receive it.

Another way to protect the unauthorised tampering of the CA-CSI Agent is to use threshold cryptography (Desmedt 1993) (Zhang 1998) (Canetti and Goldwasser 1999) (Shoup 2000) (Abdalla, Miner and Namprempre 2001) (Borselius, Mitchell and Wilson 2001) (Zhou, Schneider and Renesse 2002) for CA-CSI Agents, so even if they are compromised they cannot give false info. In this case, CA-CSI Agents should carry with them the AMP Locations where the other CA-CSI Agents deployed by that CA exist, so they can threshold sign the response to the User-CSI Agent. Thus, one would have to tamper with the CA-CSI Agents on all these AMPs (or at least tamper with the Agents in as many AMPs are needed in order to produce a threshold-crypto based signature on the CSI to be delivered).

Furthermore, a number n of CAs could decide to launch CA-CSI Agents that contain CSI for all n CAs, and these CA-CSI Agents can carry a threshold-crypto private key (i.e. a share of the key) commonly produced by the n CAs, or certified through a cross-certification process. This would result in some sort of cross-certification for revocation purposes only for those n CAs. A CA-CSI Agent receiving a CSI Query from a User-Agent should broadcast this request to t CA-CSI Agents (where t<= n, n = the number of cousin CA-CSI Agents, t = number of CA-CSI Agents necessary to threshold sign something), and collect the signed response. Even better, the User-CSI Agent should broadcast the request to the CA-CSI Agents (AMP should provide the list of CA-CSI cousin Agents to User-CSI Agent) and receive the threshold signatures (Jing, Liu, Feng et al. 2003).

In addition, collaborating CA-CSI Agents that threshold sign their responses can be ordered to renew their key shares in periodic intervals (Herzberg, Jakobsson, Jarecki et al. 1997) while retaining the same public/private keypair, so that even if an attacker manages to break into one

or more of those CA-CSI Agents and get ahold of the key share, this attack is invalidated.

Another way to minimise the risk of compromised CA-CSI Agents without requiring them to contact CA-CSI Agents on other AMPs would be to have them use reverse hash chains (Gritzalis, Moulinos et al. 2001; Jingfeng, Yuefei, Heng et al. 2005). CA-CSI Agents' certificates should contain the initial hash value *IV* (IV=Hash^{z}(pseudo*random_number*), where z is the maximum number of validity periods of the CA-CSI Agent) of a hash chain and give out to User-CSI Agents the current period's validity value r (r=Hash^{y-z}(*IV*), where y is the serial number of the current validity period) and CA's should "refresh" the CA-CSI Agents when validity period is over.

4.3.5.2 Unauthorised modification of information contained in the Agents

The CA-CSI Agents transport the CSI from CAs to AMPs. The integrity and origin of CSI contained in CA-CSI Agents can be verified because the Agent and its contents are signed by verifying the CA. This signature is not redundant even if the CSI carried by the Agent contains a CA signature as well because it ensures that the CSI contained in the Agent has not been replaced with older, valid at that time, CSI.

Protecting the integrity of CSI request information contained in User-CSI Agents on their way to AMPs can be accomplished by having dependent entity sign the Agents, before transporting them to the AMP. The AMP must verify both signatures (CA signature and signature of dependent entity on the User-CSI Agent) before allowing the execution of the User-CSI Agent.

When the User-CSI Agent has gathered all the CSI it needs, it transports itself back to the dependent entity, either immediately or whenever the latter is connected again to the network. The integrity of the Agent may be compromised while in transport from the AMP back to the dependent entity.

To counter this threat, the User-CSI Agent must request from the CA-CSI Agent to sign itself (both the User-CSI Agent and the CSI it has gathered) with a private key corresponding to a certificate issued by the CA-CSI Agent's CA. The CA-CSI Agent must obviously carry with it this private key. The issue of signing an Agent along with its state (and data it carries) has already been dealt with (Gong and Schemers 1998) so the issue that remains is the confidentiality of the private key carried by the CA-CSI Agent and used in order to sign User-CSI Agents in an attempt to ensure their integrity while being in the trip back to their dependent entities. The solution to this problem lies in the use of U*ndetachable Signatures* (Kotzanikolaou, Burmester and Chrissikopoulos 2000; Sander and Tschudin 1998). For CA-CSI Agents to use Undetachable Signatures, the CA must equip them with two functions f() and $f_{signed}()$, where $f_{signed}()$ is the original signature function $s()$ encrypted with f(). CA-CSI Agents applying these two functions on message m produce as a result the signature $s(m)$, without revealing the original private key of the signature function s. Message m can only have a predefined constrained format and range of values because the function f() is linked to these constrains.

Verification of the origin of the CSI itself a dependent entity receives may be possible or not, depending on the CSI mechanism (see Section 2.3) that the User-CSI and CA-CSI Agents decided to use in order to exchange CSI. However, this is covered by the mechanism described in the previous paragraph that verifies the integrity of the User-CSI Agent, including the information it carries with him.

4.3.5.3 AMP masquerade

In order to prevent AMP masquerade, both the CAs and the dependent entities should encrypt their Agents with the public key of the AMP they intend to send their Agents to, before transporting them. Thus, the AMP will

be able to decrypt and execute the Agents only if it is the AMP that the CAs or dependent entities intended to send their Agents to.

4.3.5.4 Denial of Service

CA-CSI Agents should be sent by the respective CAs to more than one AMPs. Therefore even if an AMP cannot perform as expected (e.g. the communication line that leads to the AMP is congested due to the DoS attack, or the User-CSI Agents that reside at the AMP perform too many CSI requests to the CA-CSI Agents, thus consuming much of the processing power of the AMP), a User-CSI Agent may select another AMP to retrieve CSI from. To accomplish this, CA-CSI Agents must carry with them the list of AMPs where cousin CA-CSI Agents can be located, in order to give this information to User-CSI Agents trying to locate another AMP to transport themselves to.

Another countermeasure that can be used in order to deal with DoS attacks would be to minimise the Central Processing Unit (CPU) time quota that is available to every User-CSI Agent residing in the AMP (Gorrieri and Marchetti 1998), thus reducing the possibility that a malicious User-CSI Agent could use too many system resources.

4.3.5.5 Replay attacks

Dependent entities should include a nonce (Number used ONCE) (Schneier 1996) in their User-CSI Agents in order to prevent User-CSI Agent replays.

CA-CSI Agents are timestamped and they also contain a validity period, which is short. If an unauthorised entity attempts to replay an old CA-CSI Agent to an AMP then the AMP will dismiss that CA-CSI Agent as expired. The timestamp should be cryptographically verifiable. One way to do this is through the use of one-way hash chains (Jingfeng, Yuefei et al. 2005), (Gritzalis, Moulinos et al. 2001).

When a CA deploys a new CA-CSI Agent to one or more AMPs, the CA should produce M_i where i= the 5-minute period (seconds elapsed since Unix time) when this message chain was produced and RV is a Random Value.

$$M_i = hash_i(TimePeriod_i) \| M_{i-1} \| Cert_{CA-CSI} \| RandomValueRV \forall i \in [b..a]$$

The CA should then sign M_i, where i= the last 5-minute period when this message chain can be used to authenticate the CA-CSI Agent and store the resulting signed information (PPAT - Privacy Protected Authentication Token (Gritzalis, Moulinos et al. 2001)) and the random value RV in the CA-CSI Agent, before deploying it.

User-CSI Agents can verify the validity of CA-CSI Agents by asking for the appropriate hash chain leaf to be revealed, the one corresponding to the precise time period the conversation between CA-CSI Agent and User-CSI Agent takes place.

User-CSI Agents would not retrieve CSI from expired CA-CSI Agents contained in a (potentially compromised) AMP because they would realise that the CA-CSI Agents have expired (by verifying the timestamp and validity period they contain) and therefore they should not trust the CSI they carry.

4.3.5.6 Malicious Agents

Having Agents signed by the entities that sent them provides accountability for the actions that the Agent will perform, on behalf of the entity that sent the Agent but does not preclude that Agents could behave maliciously. Protecting the AMP and Agents from other Agents has already been a topic of research both in academia and in the commercial world.

Solutions to this problem include the use of tamper-resistant hardware (Wilhelm, Staamann and Buttyan 1998), (Wilhelm 1999) and the use of

language semantics in order to verify the safety of a specific application (Gong, Mueller, Prafullchandra et al. 1997), (Necula 1998) either dynamically, at run-time, or statically.

Furthermore, access control mechanisms can be employed in order to restrict access to the resources or services offered by an Agent (Gong and Schemers 1998).

Limiting the percentage of AMP resources an Agent can use (Gorrieri and Marchetti 1998) can also reduce the potential impact of malicious actions by an agent.

Furthermore, confining the use and transportation of Agents inside network domains (Gritzalis and Iliadis 1998; Iliadis, Gritzalis et al. 1998), and enforcing security policies for their operation could contribute to a controlled execution environment for these Agents.

Finally, the integration of application development platforms that can be used for the development of Agents could lead into an integration of the respective security functionality. Such an integration could give to the developers more tools to address the security issues that arise from the operation of Agents (Gritzalis and Iliadis 1998).

Agents could also be subject to probabilistic auditing (Popescu, Crispo and Tanenbaum 2003). Dependent entities (or CAs) could send User-CSI Agents to an AMP looking for some CSI, which they already have in their possession, just to verify that the CA-CSI Agent will indeed hand them over the CSI it is supposed to. Interface Agents could launch User-CSI Agents for that job during idle times, i.e. times when the dependent entity requires no CSI to be fetched.

4.3.5.7 Malicious AMPs

A possible countermeasure against malicious AMPs would be to empower AMPs to control CA-CSI Agents' behaviour, and provide undeniable proof to the respective CAs that they have been doing so. One way to do that could be to have CA-CSI Agents include a hash of their response messages addressed to User-CSI Agents, inside the message itself, before signing it. The aforementioned hash should be calculated upon the current CSI response, and should include the hash of the previous CSI response, thus forming a hash chain. Therefore, if a CA-CSI Agent sends out erroneous CSI for some reason, it can be traced back through the hash chain. Furthermore, the AMP should be responsible for actively monitoring this hash chain to detect in time any discrepancies in CA-CSI Agent behaviour. Finally, the hash chain should be transferred to the respective CA, in order to verify that the CA-CSI Agent did send out correct CSI; if this does not hold true, then obviously either the AMP tampered with the CA-CSI Agent, or at least the AMP did not bother to detect malpractice on behalf of the CA-CSI Agent. In any case, the CA should stop sending CA-CSI Agents to that specific AMP until the issue is resolved. The hashes inserted by CA-CSI Agents in their responses should have the following format:

$$CA-CSI\,\mathrm{Re}\,ply_i = Sign_{CA-CSI}(CSI_i \,||\, Hash(CA-CSI\,\mathrm{Re}\,ply_{i-1}) \,||\, i)$$

(Esparza, Soriano, Munoz et al. 2003) propose a mechanism to detect malicious hosts based on limiting and monitoring the execution time of Agents. Dependent entities would have to limit the time of execution of their User-CSI Agent on an AMP and check the execution and transmission times of their User-CSI Agents and stop using a specific AMP the execution and transmission times reported back by the AMP (through the User-CSI Agent) are not coherent.

(Hohl 2000) generalises on the ideas of State Appraisals (Farmer, Guttman and Swarup 1996), Server Replication (Minsky, van Renesse, Schneider et al. 1996) and Execution Traces (Vigna 1998) referring to the notion of *reference states* that can be verified by the owners of Agents. However, in all the aforementioned cases there is some factor hindering the implementation: high bandwidth requirements, the need to involve the dependent entity more than it would like or could and a double size AMP infrastructure (twice AMPs as normal would be needed).

(Esparza, Fernandez, Soriano et al. 2003) propose two mechanisms for protecting Agents from malicious hosts (in our case, AMPs): agent watermarking and agent fingerprinting. In the case of agent watermarking, the User-CSI Agent attaches a predefined watermark to the data it receives from each CA-CSI Agent while it discusses with CA-CSI Agents on an AMP. This watermark has been included in the User-CSI Agent by the Interface Agent before letting him go on its trip to AMPs and the Interface Agent expects to see this watermark in the CSI the User-CSI Agent returns with. In the case of agent fingerprinting, the watermark placed by the User-CSI Agent on CSI received from CA-CSI Agents is dynamic and it depends on the actual AMP the User-CSI Agent resided when it was discussing with the CA-CSI Agent that provided CSI.

An additional countermeasure to protect Agents (CA-CSI) against the actions of a malicious AMP is to have CA-CSI Agents use Undetachable Signatures (for more info see Section 4.3.5.2) in order to deliver CSI to User-CSI Agents instead of regular signatures. However, this is a bandwidth-consuming countermeasure.

Finally, another mechanism that can be used to trace malicious actions by an AMP is software reflection (Spinellis 2000). Since both User-CSI Agents and CA-CSI Agents are being developed by a CA, these agents could use the

aforementioned mechanism in order to produce a hash of their code in memory and compare it with the respective hash value stored at some repository hosted by the CA. If the values differ, then tampering of the agents has occurred and these agents must notify their owners. The User-CSI Agent should go back to the dependent entity and notify the Interface Agent accordingly and the CA-CSI Agent should notify the CA.

4.3.5.8 Privacy of Query

A dependent entity might not want to let a CA-CSI Agent or an AMP know for which certificate is it looking CSI for. The reason for this is that the dependent entity may not want to let the CA-CSI Agent or the AMP know the identity of entities that communicate with the dependent entity and how often does this happen.

There is a way to prevent the CA-CSI Agent (and the AMP) from knowing which entity's certificate does the dependent entity seek CSI for, unless the CA-CSI Agent does contain that CSI. If that is achieved, then the risk of having a compromised AMP or CA-CSI Agent giving false CSI to User-CSI Agents could be reduced as well, because the AMP or CA-CSI Agent will not know what CSI the User-Agent is looking for, unless the AMP contains such CSI. One way to achieve that (Riordan and Schneier 1998) would be to form the CSI query in the following way:

Query: H(N, CertID, CaCertID),

where H() is a hash function, and CertID is an identifier for the certificate the dependent entity is looking CSI for. It could be the public key of that certificate, the hash of that public key, the hash of the serial number of the certificate and the respective Distinguished Name (CCITT 1998) contained in the certificate, or any other construct that can identify uniquely a certificate issued by a specific CA. Furthermore, the query should include another certificate identifier, identifying the CA that issued the certificate, the

dependent entity is looking CSI for. Finally, in order to avoid dictionary attacks from an AMP that would want to find out which certificate is the dependent entity looking CSI for, a nonce (N) should be included as well. This nonce could be the nonce included in the Agent in order to protect the entities that receive the Agent from replay attacks (see Section 4.3.5.5).

The User-CSI Agent will be retrieving the respective certificate identifier values from the CA-CSI Agent and will be calculating the respective query values until he finds one that matches. If this is the case, the User-CSI Agent can retrieve CSI from the CA-CSI Agent at that time. If the CA-CSI Agent or all the CA-CSI Agents that reside in the AMP, do not contain the CSI that the User-CSI is looking for, then the latter will not reveal to the CA-CSI (or to the AMP) which certificate is it he was looking CSI for.

Another alternative so that the CA-CSI Agent will not find out the identity of the entity the dependent entity is looking CSI for, would be to use blind signatures (Menezes, Oorschot and A. 1997). The User-CSI Agent would require all the CSI available by the CA-CSI Agent (or at least a superset of the CSI it is looking for), and then if it found the CSI it is looking for it would ask from the CA-CSI Agent to blind sign it. Certainly, this may lead to the CA-CSI Agent repudiating his actions, since he can claim the User-CSI Agent acted maliciously and gave him a wrongful CSI statement to blind sign, so it is only applicable in environments where the User-CSI Agent can be considered absolutely trustworthy.

Another alternative to the issue of privacy is to have the User-CSI Agent request CSI for a number of different certificates, among which lies the one he is really interested for, i.e. have the User-CSI Agent perform a *range CSI query*. However, that may also reveal some information since consecutive certificate serial numbers may belong to a specific group of persons, thus revealing a potential relation between the dependent entity and that group of persons.

(John Solis and Tsudik 2006) introduce the notion of a *range query* and *permuted ordering* especially for CRT queries in order to deal with the shortcomings of the *range query* itself.

4.4 UML Design

In this section, we provide the reader with UML Use Cases and Activity Diagrams, presenting the core ADoCSI functionality in a concise manner. Throughout the diagrams, we will be using the following abbreviations, for reasons of economy:

- Interface Agent: IA
- User-CSI Agent: UCSIA
- CA-CSI Agent: CACSIA
- Directory Facilitator Agent: DFA
- Cert: digital certificate
- User prefs: User preferences regarding CSI location, processing and storage. These preferences are collected by the Interface Agent upon initial installation on the system of the dependent entity; during that initial installation, IA poses a series of questions to the dependent entity in order to store its preferences regarding CSI.

We provide a Use Case diagram presenting ADoCSI functionality on the system of the dependent entity (Figure 3) and another Use Case diagram presenting ADoCSI functionality on the Agent Meeting Places (Figure 4).

We also provide UML Activity Diagrams in order to present the flow of ADoCSI Agent's actions. In Figure 5 we present the flow of actions leading to the deployment of a User-CSI Agent. The external event initiating a User-CSI Agent deployment, as this is depicted in Figure 5, is a PKI-aware application residing on the system of the dependent entity, which received signed data

and needs to have the respective certificate validated. The PKI-aware application forwards the respective certificate to the resident Interface Agent.

In Figure 6 we present the flow of actions performed by the User-CSI Agent during the quest for CSI. Finally, in Figure 7 we present the deployment of CA-CSI Agents by CAs and in Figure 8 we present the monitoring actions of an AMP's Directory Facilitator Agent.

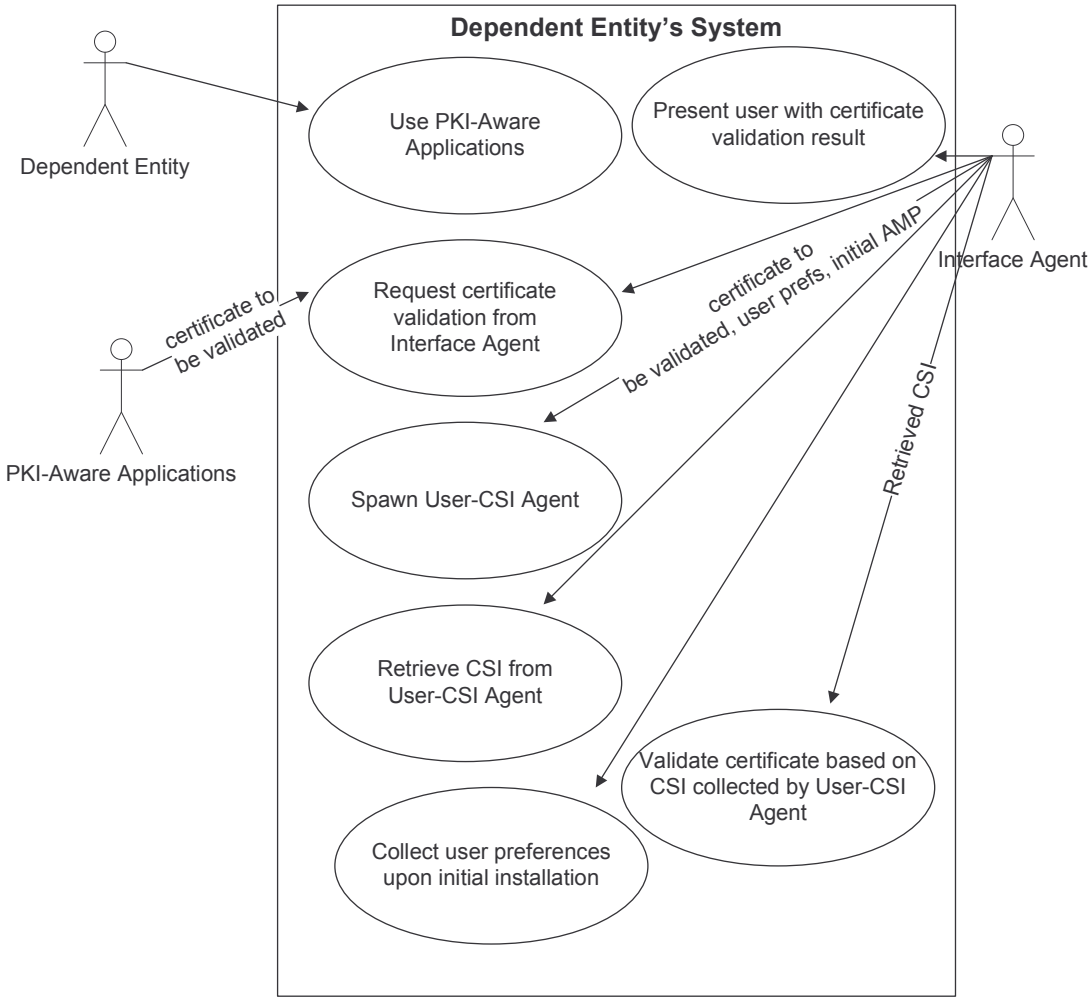

Figure 3: Use Case of ADoCSI functionality on Dependent Entity's system

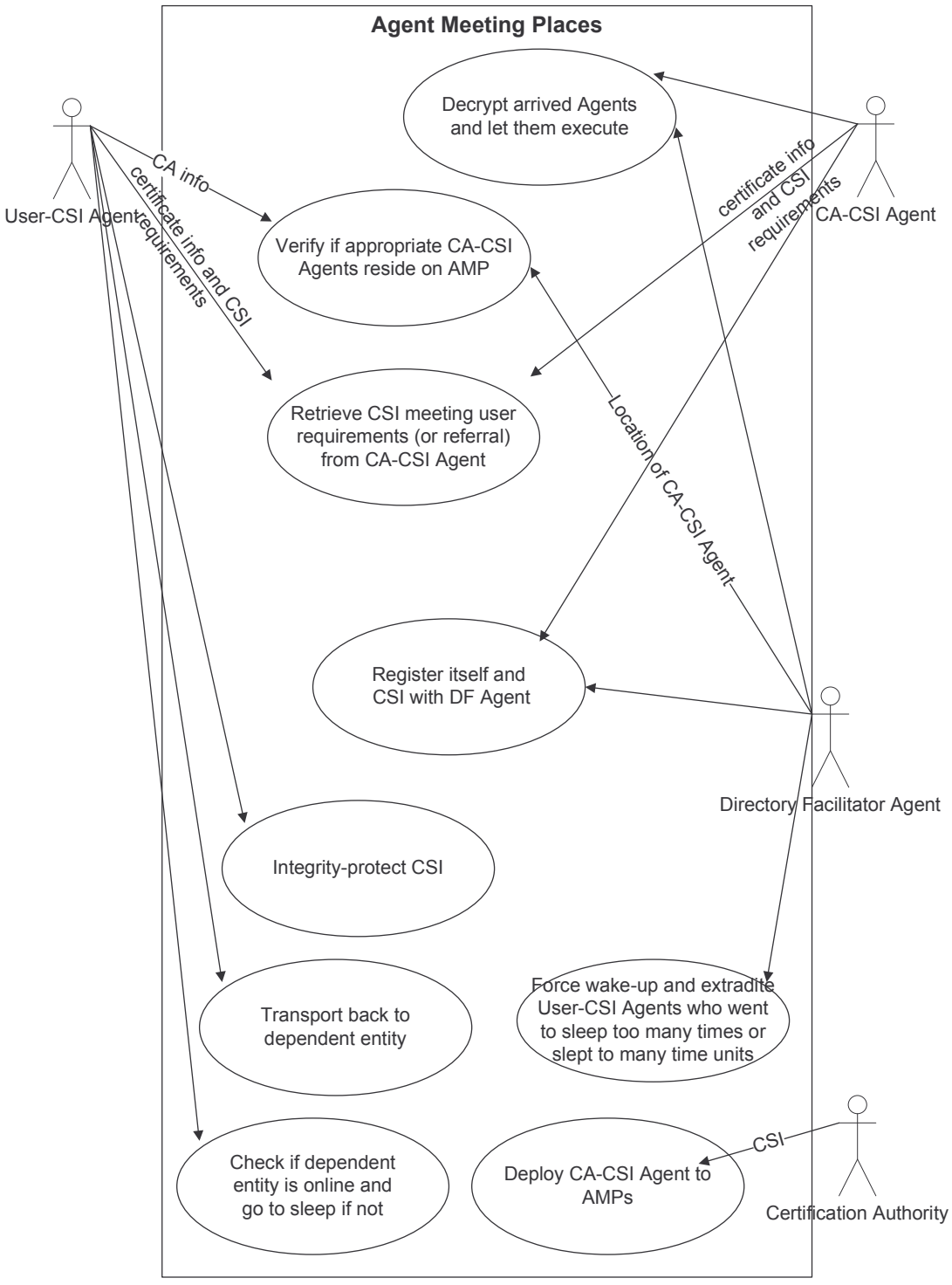

Figure 4: Use Case diagram of ADoCSI functionality on AMPs

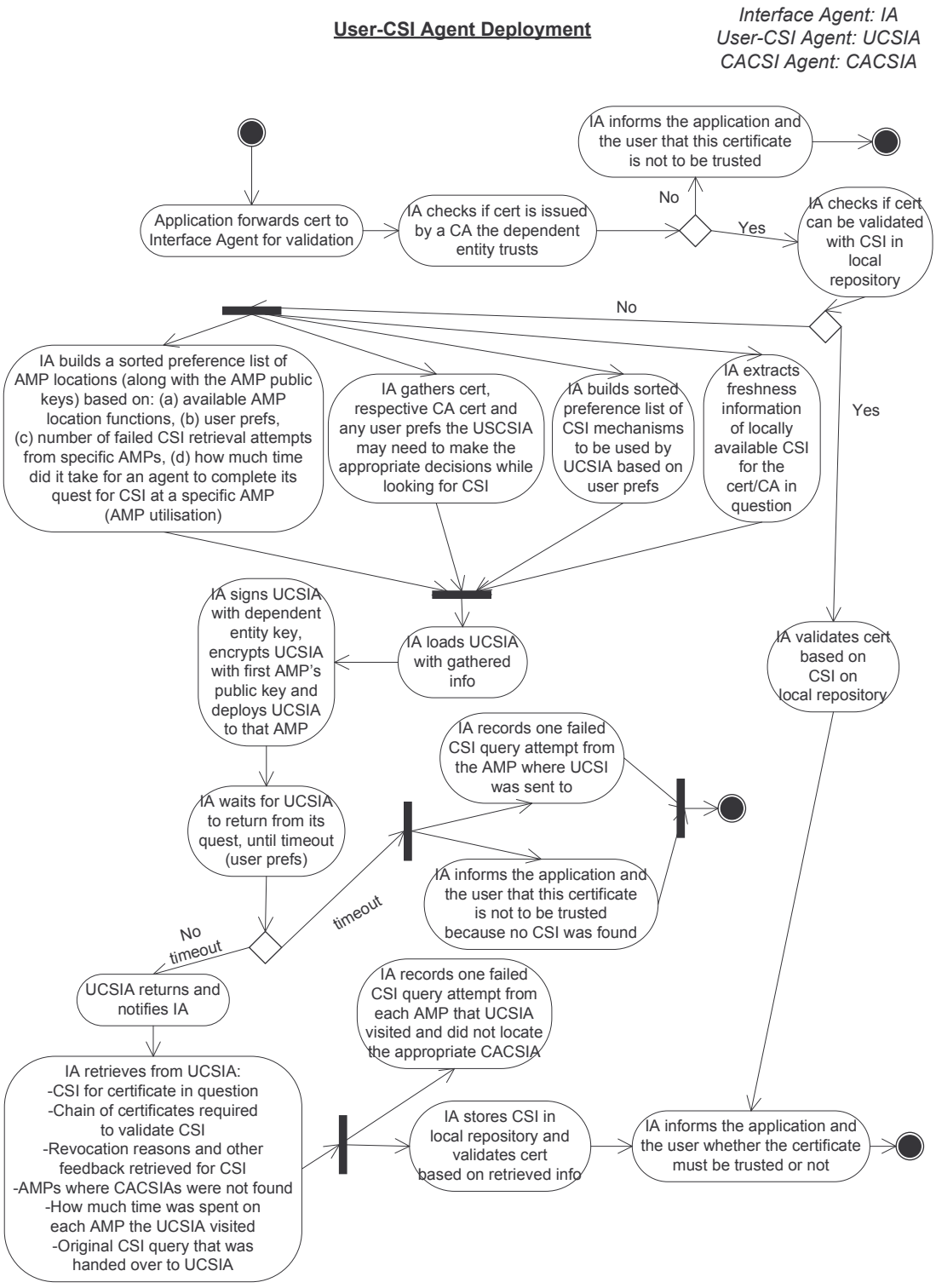

Figure 5: UML Activity Diagram of User-CSI Agent Deployment

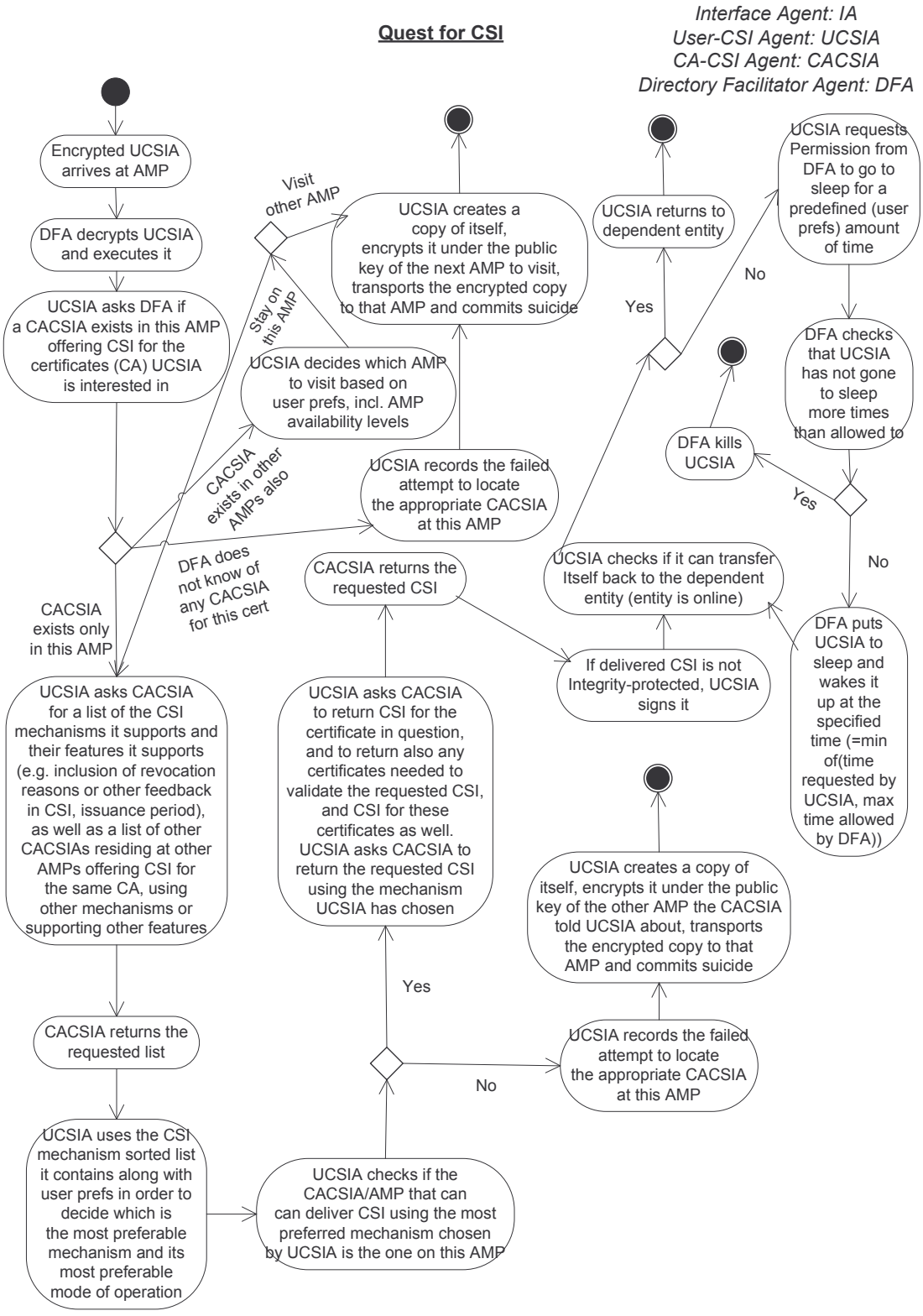

Figure 6: UML Activity Diagram of User-CSI Agent's quest for CSI

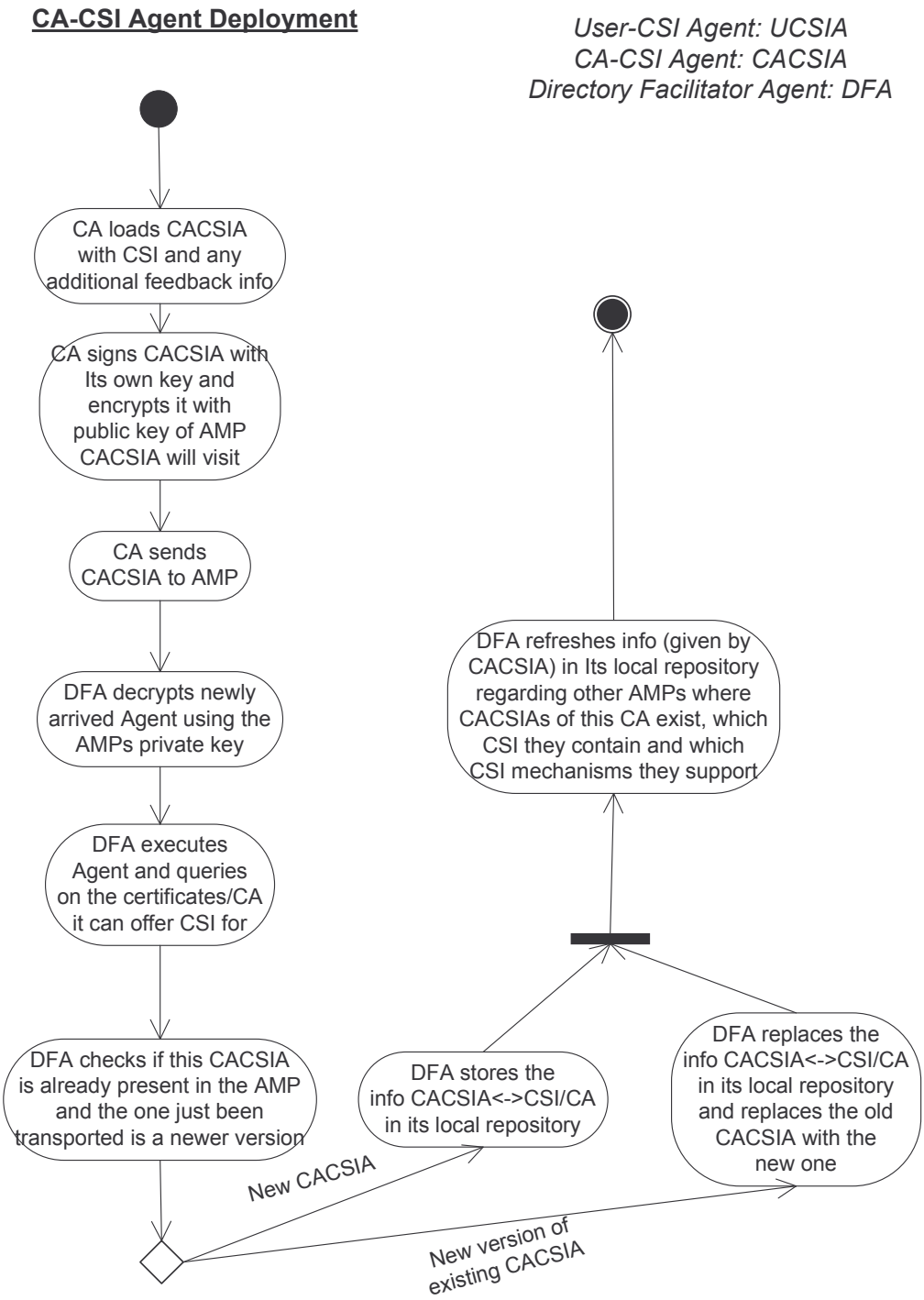

Figure 7: UML Activity Diagram of CA-CSI Agent Deployment

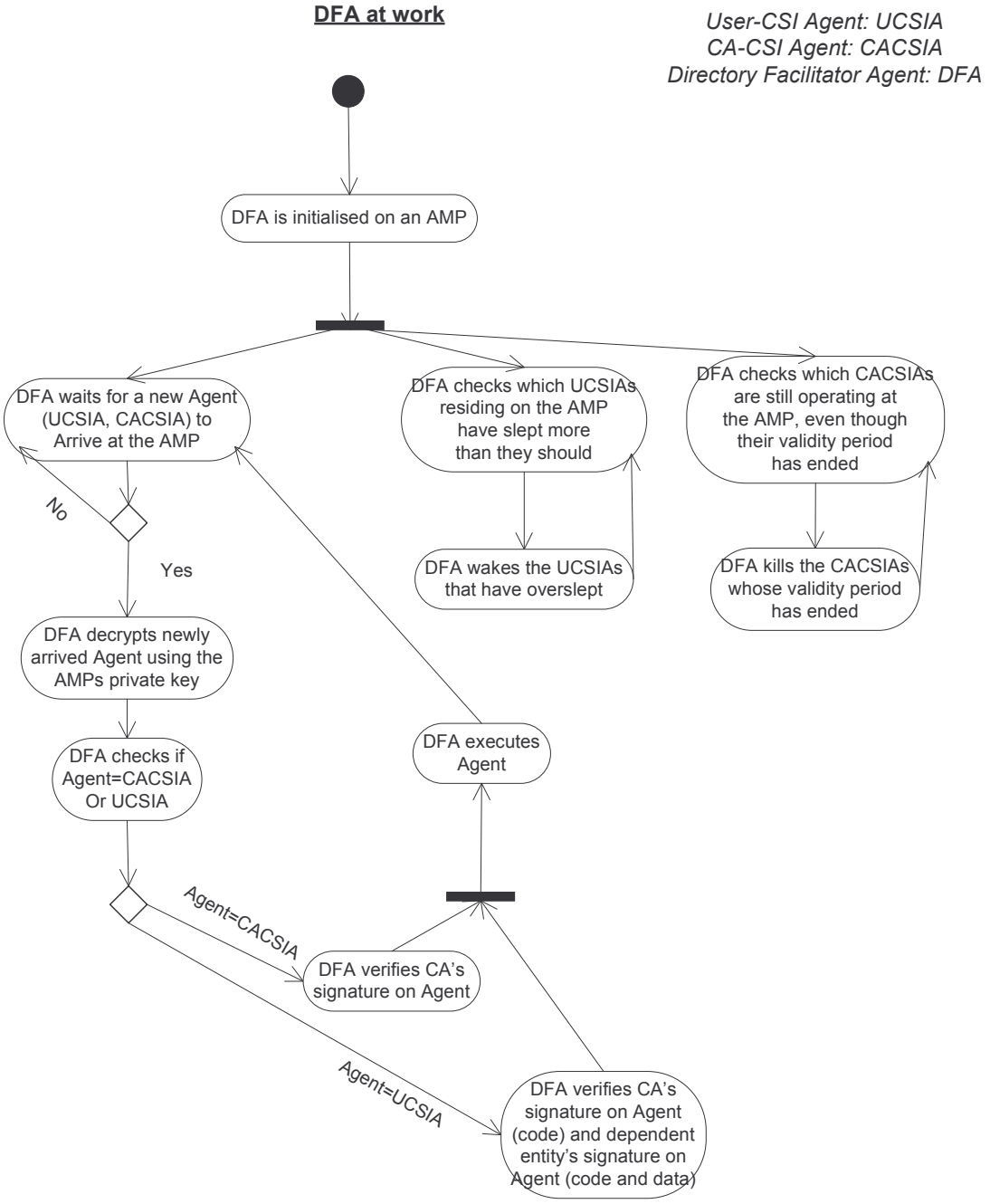

Figure 8: UML Activity diagram of Directory Facilitator Agent monitoring activity at an AMP

4.5 Proof of Concept

An early version of ADoCSI was implemented as an initial proof of concept, in the framework of UoA MSc students' assignments that I was supervising.*

Two prototype implementations were undertaken. The one using IBM Aglets and the other using Java sockets and remote method invocation. User-CSI Agents were being encrypted with the AMPs (DF) public key so that AMP could be avoided. In addition, CSI given by the CA-CSI Agents were digitally signed before handed off to the User-CSI Agent.

The experiment conducted included two typical Pentium4-based PCs acting as AMPs hosting CA-CSI Agents with each CA-CSI Agent belonging to the same CA but containing different CSI (Distribution Point CRLs) and another typical Pentium4-based PC acting as the dependent entity's PC, from which the User-CSI Agent was launched by the Interface Agent.

Two test scenarios were executed: in the first one, the dependent entity was continuously online so the User-CSI Agent could return immediately home and hand over the retrieved (or not) CSI. In the second scenario, the dependent entity went offline right after the Interface Agent launched the User-CSI Agent and came back online a little while after. In both scenarios, the CA-CSI Agents residing at the AMPs only supported CRL as a format for carrying CSI. Besides that, the CSI queries were formed so that some User-CSI Agents were returning with negative status information (i.e. certificate exists in CRL) while others were returning with "positive" status information (i.e. certificate does not exist in CRL). The network between the two AMPs could support a bandwidth of 100Mbps while the bandwidth of the connection between the entity's PC and the two AMPs was 1024kbps.

---

* Lest I forget to give them credit, the MSc students were the following: C. Kioulafas, S. Crystallides, N. Tsagkaris, A. Kikiles, G. Plataniotis and S. Ratsiatos.

In any case, the User-CSI Agent was encrypted before leaving dependent entity's PC with the public key of the AMP, to prevent AMP masquerading. Also, CSI being given by CA-CSI Agents to User-CSI Agents was signed by the former before handed off to the latter.

User-CSI Agent's round-trip times were measured. The average round-trip of the User-CSI Agent was 5,77 seconds. These measurements included the time that the dependent entity was offline for a total of 20 seconds approximately. If we take this query out (the one that took place when the dependent entity went offline and the User-CSI Agent had to wait for her to come back), then the average round-trip time for the User-CSI was 4,65 seconds.

4.6 ADoCSI Comparative Evaluation

In this section, we evaluate ADoCSI against the evaluation criteria presented in Section 3.2 and we also provide a comparative evaluation of ADoCSI against the other CSI mechanisms presented in Section 2.3.

ADoCSI seems to be dealing with some of the issues that remain unsolved for most if not all of the other CSI mechanisms (e.g. feedback criterion, transparency criterion). However, ADoCSI introduces new problems as well, namely security of the mechanism itself. This is due to the fact that ADoCSI is based on the use of Agents while all the other CSI mechanisms are based either on standalone applications running on the dependent entity's workstation or device, or on client server protocols. The new problems ADoCSI introduces, due to its agenthood, have been presented along with their solutions in Section 4.3.5.

In the next few paragraphs, we present the evaluation of ADoCSI against our evaluation criteria and the comparative evaluation of ADoCSI against the other CSI mechanisms we have presented in this thesis.

4.6.1 ADoCSI Evaluation - Management Criteria

ADoCSI meets the *feedback* criterion; the Interface Agent instructs the User-CSI Agent to negotiate with CA-CSI and attempt to retrieve *complete* (see Section 3.2.1.10) CSI, which is a prerequisite for the *feedback* information. Also, the Interface Agent instructs the User-CSI Agent to collect all information necessary in order to provide adequate information to the user (see Section 3.2.1.1); when then User-CSI Agent returns, the Interface Agent presents to the dependent entity all collected information in a concise and comprehensible way. ADoCSI is the only one mechanism, along with CRS, that fully meets the feedback criterion.

ADOCSI also meets the *transparency* criterion. Neither certificate holders nor dependent entities have to perform any task in order to locate and retrieve the CSI they need. The Interface Agent interacts with all PKI-aware applications on the workstation of the dependent entity and instructs the User-CSI Agent to retrieve the necessary CSI in order to hand it back to the aforementioned PKI-aware applications and inform the user, if needed. Probably the only point at which human interaction is needed is when the Interface Agent and the User-CSI Agent are installed at the local repository of the dependent entity. At that time, the Interface Agent should ask the dependent entity a series of appropriately formed high-level questions in order to create a profile for the use of the ADOCSI mechanism. These questions will define the values of the negotiation parameters that User-CSI Agents will use when communicating with the CA-CSI Agents. The aforementioned questions must be formed in such a way that the respective answers will provide the necessary feedback to the Interface Agent, and at the same time the dependent entity will not become confused about the questions asked and the answers he has to provide. Furthermore, the questions must be

formed in such a way that the dependent entity will be motivated to take the time to answer them.

The other CSI mechanisms presented in Section 2.3 do not meet the transparency criterion. All of those mechanisms require that either the dependent entities or the certificate holders (authenticating entities or signers), or even both, be motivated, knowledgeable and perceptive enough in order to carry out the necessary procedures to locate and retrieve the CSI they need. ADOCSI does not have such a requirement neither from the dependent entities nor from the certificate holders.

ADoCSI also meets the *delegation of revocation* criterion. The User-CSI Agent trusts the information it is given by the CA-CSI Agent because the CA-CSI Agent has taken appropriate steps to ensure the CSI integrity and authenticity (see Sections 4.3, 4.3.5.2) and the User-CSI Agent can verify that. Whether the revocation was performed by the CA itself or by another entity, this remains out of the scope of interest of the User-CSI Agent. The CA that implemented the CA-CSI Agent has implemented the appropriate mechanism inside the latter in order to draw CSI from the entity, whichever it may be, that produces it. For the very same reason, ADoCSI meets the *delegation of CSI dissemination* criterion. These two criteria (*delegation of revocation* and *delegation of CSI dissemination*) are also met by many of the CSI mechanisms presented in Section 2.3.

ADoCSI does not meet the *delegation of certificate path validation* criterion although it is not the dependent entity doing the certificate path validation, but the Interface Agent, having gathered any necessary CSI to do certificate path validation that the User-CSI Agent brought back from his trip. Since the Interface Agent is acting under the direct instructions of the dependent entity and within the workstation/device of the dependent entity, we shall not be

considering the delegation of the certificate path validation to the Interface Agent as such.

ADOCSI meets the *referral capability* criterion. If a User-CSI Agent cannot find the appropriate CSI at a specific AMP, he is referred to another AMP either by the Directory Facilitator or by the CA-CSI Agent (the CSI carried by the CA-CSI Agent may contain CSI referral information, like a Redirect CRL). This criterion is only met by three other CSI mechanisms out of those we presented in Section 2.3.

ADoCSI potentially meets the *revocation reasons* and the completeness criterion. The Interface Agent can instruct the User-Agent to bring back home *complete CSI (see Section 3.3.1.10)*, along with revocation reasons but the CA-CSI Agent deployed by the CA may have not equipped the CA-CSI Agent with revocation reasons information. If the User-CSI Agent returns with complete CSI along with revocation reasons information then the Interface Agent can conduct certificate path validation, including in the mix the revocation reasons, according to the policy set forth by the dependent entity at the original time of the Interface Agent installation.

ADoCSI can meet the *notification of revocation or suspension* criterion; the Interface Agent simply has to ask the User-CSI Agent to check for revocation of the dependent entity's certificate each time the User-CSI Agent goes on a trip to fetch CSI. No other CSI mechanism out of those presented in Section 2.3 meets this criterion.

ADoCSI meets the *CSI mechanism evaluation* criterion; the Interface Agent, depending on the preferences stated by the depended entity when the IA was installed, instructs the User-Agent to negotiate with CA-CSI Agents on receiving CSI in specific formats that meet the high-level requirements set forth by the dependent entity. No other CSI mechanism out of those presented in Section 2.3 meets this criterion.

Table 41 : Evaluation of ADoCSI and other CSI Mechanisms - Management Criteria

CSI Mechanism	Feedback	Transparency	Delegation of revocation	Delegation of CSI Dissemination	Delegation of certificate path validation	Referral capability	Revocation Reasons	Notification of revocation	Ability to evaluate CSI mechanism	Completeness
CRL	×	×	✓	✓	×	×	✓	×	×	×
Sliding Window DeltaCRL	×	×	✓	✓	×	×	✓	×	×	×
Freshest CRL	×	×	✓	✓	×	×	✓	×	×	×
Redirect CRL	×	×	✓	✓	×	✓	✓	×	×	×
Indirect CRL	×	×	✓	✓	×	✓	✓	×	×	×
Enhanced CRL Distribution Points	×	×	✓	✓	×	×	✓	×	×	×
Augmented CRL	×	×	✓	✓	×	×	✓	×	×	×
Efficient Fault-Tolerant Certificate Revocation	×	×	Partial	Partial	×	×	×	N/A	×	×
Positive CSI (Suicide Notes)	×	×	✓	✓	Not Applicable (there is no certificate path that needs validation)	×	×	×	×	×

Self-Controlled Positive CSI	×	×	Partial	✓	Not Applicable (there is no certificate path that needs validation)	×	×	N/A	×	×
Freshness-constrained Revocation Authority	×	×	✓	✓	×	×	×	×	×	×
Efficient Long-term Validation of Digital Signatures	×	×	×	✓	Not Applicable (there is no certificate path that needs validation)	×	×	×	×	×
Certificate Revocation Status	✓	×	×	✓	×	×	✓	×	×	✓
Certificate Revocation Trees	×	×	✓	✓	×	×	×	×	×	×
Online Certificate Status Protocol	Partial	×	×	✓	Partial	✓	✓	×	×	Partially

(OCSP)										
Certificate Re-issuance	×	×	×	×	×	×	×	×	×	×
ADoCSI	✓	✓	✓	✓	×	✓	Potentially	✓	✓	Potentially

4.6.2 ADoCSI Evaluation - Performance Criteria

ADoCSI can meet the *timeliness* and *freshness* criteria; The Interface Agent can instruct the User-CSI Agent to negotiate with the CA-CSI Agent in order to retrieve timely and fresh CSI. CAs, on the other hand, should be timely equipping their CA-CSI Agents with fresh data. There is even a business argument there for CAs: CAs could be timely equipping their CA-CSI Agents with fresh CSI and be also equipping their CA-CSI Agents with high latency and not-so-fresh CSI (e.g. CRL); the certificate holders who wish to render very fresh CSI regarding their certificate in a timely manner to dependent entities will be paying a little extra to get their certificate from the CA. When dependent entities' User-Agents negotiate with CA-CSI Agents for timely delivery of fresh CSI regarding a specific certificate, the CA-CSI Agents will be handing this kind of CSI if the respective certificate holders have paid that little extra for their certificate; if not, they will be handing over to the User-CSI Agents not-so-fresh CSI coming from a high latency mechanism as CRLs.

Moreover, dependent entities can specify different levels of CSI freshness, depending on the importance of the certificates they want their User-Agents to retrieve CSI for, depending on the resources they wish to avail to the CSI retrieval for a specific group of certificates and also depending on the consequences of not having timely CSI on a specific group of certificates. Therefore, ADOCSI also meets the *adjustability* criterion. There are only three other mechanisms out of those presented in Section 2.3 that meet the

adjustability criterion (Positive CSI, Freshness-constrained CSI and Certificate Re-issuance). However, none of those mechanisms provides such a transparent and fine-grained method for adjusting the CSI location function depending on the circumstances.

ADOCSI also meets the emergency CSI criterion. If emergency CSI is issued it can be sent either directly to the CA-CSI Agents that already reside at the AMPs (e.g. using one CSI format out of those described in Section 2.3), or new CA-CSI Agents can be deployed to AMPs that contain the emergency CSI. In any case, the emergency CSI will be made readily available to User-CSI Agents. The Interface Agents (depending on the preferences the dependent entity set at the time of their installation on the dependent entity's workstation/device) can choose specific certificates or group of certificates for which they wish to have emergency CSI and ask User-Agents to negotiate on getting Emergency CSI for these certificates, if available. Again, there is a business argument to be made here for CAs, i.e. they can sell this as an extra chargeable service to their certificate holders. In conclusion, ADOCSI meets the emergency CSI criterion, but at the same time it minimises the respective operational costs this might entail, since the User-CSI Agents do not look for and retrieve emergency CSI on all certificates but only on those the dependent entities specify.

ADOCSI could support bounded revocation, if all the CSI mechanisms used by the CAs in order to feed CSI into CA-CSI Agents supported bounded revocation. That would mean that ADoCSI Agents could not take advantage of the mechanisms that do not support *bounded revocation*, namely Efficient Fault-Tolerant Certificate Revocation, Positive CSI, Freshness-constrained Revocation Authority and Efficient Long-term Validation of Digital Signatures. If CAs do not use these mechanisms, ADOCSI can also support bounded revocation. It is a matter of policy regulation by CAs. We conclude

that ADoCSI partially supports bounded revocation. Interface Agents could ask dependent entities for their respective preference on bounded revocation (whether they wish the User-CSI Agents to negotiate on CSI delivery using bounded revocation CSI mechanisms and the extra information this entails) but they should not give the option to the dependent entity to select bounded revocation as mandatory otherwise some User-CSI Agents / CA-CSI Agents negotiations may fail and User-CSI Agents will come back home empty handed.

The *scalability* of ADOCSI depends on a number of factors, such as the number of the available AMPs, the number of Agents each AMP can host and the number of AMPs to which the CAs send their CA-CSI Agents. The architecture of ADoCSI is such that CAs can create AMPs of their own but also third parties (other than the CAs) may choose to host AMP services and let Agents visit them, probably charging CAs for allowing their CA-CSI Agents to visit their AMPs. In addition, the dependent entity may choose to receive CSI in a format that is more (or less) bandwidth consuming and insert that parameter too in the preferences of its Interface Agent, which are later translated to negotiation parameters and passed over from the Interface Agent to the User-CSI Agent. Also, the dependent entity could choose not to send a User-CSI Agent immediately, when CSI is needed, but wait for a predefined amount of time or wait until there is need for CSI regarding a predefined number of certificates, before sending a User-CSI Agent to retrieve the necessary CSI. Thus, ADoCSI has the potential to scale very well, as far as communication costs are concerned. For the same reasons, it also scales very well as far as storage costs are concerned. The processing cost of ADoCSI might be high, though, if a CA decides to offer CSI to its CA-CSI Agents in a multitude of ways in order to accommodate for different needs. The CAs must select which CSI mechanisms they will be using to equip their CA-CSI

Agents with CSI also based on the processing power they can provide, in order to prevent the CSI processing cost reach their processing power limits. Due to the potentially high processing costs for CAs, we rate ADoCSI with a *scalability* grade of *medium.*

ADoCSI provides for timely dissemination of fresh CSI accompanied by fine-grained CSI mechanism adjustability levels, a combination of services that cannot be offered by other CSI mechanisms. There is, though, a prerequisite: the CA policy accommodates ADoCSI needs as discussed in this section.

Table 42 : Evaluation of ADoCSI and other CSI Mechanisms - Performance Criteria

CSI Mechanism	Timeliness of CSI	Freshness of CSI	Bounded Revocation	Emergency CSI capability	Scalability	Adjustability
CRL	×	Low	✓	×	Low	×
Sliding Window DeltaCRL	×	Low	✓	×	Low	×
Freshest CRL	✓	Medium	✓	×	Low	×
Redirect CRL	×	Low	✓	×	Medium	×
Indirect CRL	×	Low	✓	×	Low	×
Enhanced CRL Distribution Points	×	Low	✓	×	Medium	×
Augmented CRL	×	Low	✓	×	Medium	×
Efficient Fault-Tolerant Certificate Revocation	✓	High	×	✓	Medium	×
Positive CSI (Suicide Notes)	✓	High	×	✓	Medium	✓
Self-Controlled Positive CSI	✓	High	✓	✓	High	×

Freshness-constrained Revocation Authority	✓	Medium	×	×	Medium	✓
Efficient Long-term Validation of Digital Signatures	✓	Medium	×	✓	Medium	×
Certificate Revocation Status	×	Medium	✓	×	High	×
Certificate Revocation Trees		Medium	✓	×	High	×
Online Certificate Status Protocol (OCSP)	✓	High	✓	✓	High	×
Certificate Re-issuance	×	Medium	✓	×	Medium	✓
ADoCSI	✓ (provided CA is willing to offer it)	High (provided CA is willing to offer it)	✓ Provided the CA policy allows it	✓	Medium	✓

4.6.3 ADoCSI Evaluation – Security Criteria

CSI retrieval occurs within the execution boundaries of the AMP. User-CSI Agents contact CA-CSI Agents in order to locate the CSI they look for and retrieve it. Authentication of the CA-CSI Agents is achieved by verifying the digital signature CAs have put on these Agents. User-CSI Agents must carry with them the respective CA certificates in order to verify the aforementioned digital signatures. Furthermore, authentication of the AMPs where User-CSI Agents reside and execute can be performed, if dependent entities encrypt those Agents with the public key of the AMP they are going to send them to. Therefore, the *CSI disseminator authentication criterion is met*. The other CSI mechanisms do not meet this criterion on their own; they could, though, if external mechanisms were in place for authenticating the entity that

disseminates CSI, before a dependent entity could access the CSI that entity has to offer.

Verifying the integrity of the CSI is being performed by the User-CSI Agents and by the Interface Agent, once the former arrive at the dependent entity, by verifying the CA and/or CA-CSI signature on the CSI, depending on the CSI mechanism that has been used for transferring CSI. Therefore, the *CSI integrity and authenticity* criterion is also met.

The effects of having a CA key compromised depend directly on the method used by the CA to generate the CSI, and therefore depend directly on the format of the CSI CA-CSI Agents carry with them. Taking into consideration that CA-CSI Agents would probably carry CSI in more than one formats, in order to accommodate for the varying needs and requirements of the dependent entities, we assert that ADOCSI does not meet criterion. If a CA private key is compromised, then the entity that holds the compromised key will be able to issue new certificates and revoke old ones, if the dependent entities are not made aware of the CA key compromise through an Authority Revocation List or otherwise.

ADoCSI meets the *Revocation Authority compromise* criterion and the *Contained functionality* criterion. The CA whose corresponding Revocation Authority has been compromised should send a CA-CSI Agent to all AMPs it collaborates with, carrying Emergency CSI concerning the compromised Revocation Authority key. Furthermore, if CAs produced CSI using more than one CSI mechanisms out of those we presented in Section 2.3 and equipped their CA-CSI Agents with CSI being produced by all these available CSI producing mechanisms, then CA-CSI Agents could verify the validity of CSI they have in one format (e.g. CRL) based on the respective CSI they carry in another format (e.g. CRS).

The availability of ADOCSI depends directly on the number of operating AMPs and the number of AMPs the CAs send their CA-CSI Agents to. If there are enough AMPs and the CAs send their Agents to a sufficient number of AMPs, then the availability of ADOCSI is ensured. The cost for ensuring high availability is the frequent communication between a CA and many AMPs; that cost, however is low since an Agent is a small application and the CSI is only sent once (along with the Agent) to the AMP, instead of being sent to all dependent entities that request it.

In conclusion, ADOCSI also rates high on the security scale, according to the evaluation criteria we have presented in Section 3.2.3. We should mention, though, that ADOCSI faces more threats than the other mechanisms, since it is the only mechanism that involves the transport and remote execution of Agents. In Section 4.3.5 we point out these threats and the respective countermeasures.

Table 43 : Evaluation of ADoCSI and other CSI Mechanisms - Security Criteria

CSI Mechanism	**CSI disseminator authentication**	**CSI integrity**	**CA compromise**	**Revocation Authority compromise**	**Contained functionality**	**Availability**
CRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Sliding Window DeltaCRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid	✓

					again)	
Freshest CRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Redirect CRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Indirect CRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Enhanced CRL Distribution Points	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Augmented CRL	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓
Efficient Fault-Tolerant Certificate Revocation	N/A	✓	✓	✓	✓	✓
Positive CSI (Suicide Notes)	×	✓	×	✓ (if ARLs are additionally used)	✓ (but revoked/suspended certificates can be made valid again)	✓ (supports high availability, but at a high cost)
Self-Controlled Positive	N/A	✓	✓	✓	✓	N/A

CSI						
Freshness-constrained Revocation Authority	×	✓	×	✓	✓ (but revoked/suspended certificates can be made valid again)	✓ (supports high availability, but at a high cost)
Efficient Long-term Validation of Digital Signatures	×	✓	×	✓ (if ARLs are additionally used)	✓ (but revoked/suspended certificates can be made valid again)	✓ (supports high availability, but at a high cost)
Certificate Revocation Status	×	✓	×	✓ (if ARLs are additionally used)	✓	✓ (supports high availability, but at a high cost)
Certificate Revocation Trees	×	✓	×	✓ (if ARLs are additionally used)	✓ (but revoked/suspended certificates can be made valid again)	✓ (supports high availability, but at a high cost)
Online Certificate Status Protocol (OCSP)	×	✓	×	✓ (if ARLs are additionally used)	✓ (but revoked/suspended certificates can be made valid again)	✓
Certificate Re-issuance	×	✓	×	✓ (if ARLs are additionally used)	✓ (but revoked/suspended certificates can be made valid again)	✓ (supports high availability, but at a high cost)
ADoCSI	✓	✓	×	✓	✓	✓

4.7 Summary

In this section, we presented an alternative mechanism for disseminating CSI, which we call ADoCSI. We have shown that ADoCSI meets the same high standards as the other CSI mechanisms presented in Section 2.3 and in some cases it meets even higher standards, by comparatively evaluating ADoCSI against the aforementioned CSI mechanisms. Our evaluation was based on the evaluation framework we presented in Section 3.2.

ADoCSI can probably be more suitable nowadays over other mechanisms due to the nature of contemporary network communications: transacting entities that do not belong to closed, controlled user group, possibly roaming and not necessarily equipped either with technical know-how or educated as far as information security is concerned. ADoCSI possesses some characteristics (e.g. adequate feedback, adjustability) that distinguish it from the rest of the CSI mechanisms presented in Section 2.3.

One of the most important, and unique, features of ADoCSI is that it meets the transparency criterion. Entities using PKI are not always knowledgeable, motivated and perceptive towards security procedures. ADoCSI could be of use, because it relieves these entities from the burden of carrying out security related tasks in order to locate and retrieve Certificate Status Information.

Another major advantage of ADOCSI over the other CSI dissemination mechanisms is that ADOCSI has a combination of online and offline features. User-CSI Agents can be transported from the dependent entity to an AMP that is currently available. If that Agent needs to visit another AMP in order to retrieve CSI and that other AMP is not available at that time, the Agent can stay at the first AMP until the other AMP is available again.

The fact that the User-CSI Agent can stay at an AMP for a predefined amount of time before transporting itself to the next destination can prove to be beneficial for the dependent entities as well. A dependent entity may

disconnect itself from the network or network connectivity between the dependent entity and an AMP may fail temporarily. If a User-CSI Agent has gathered all the CSI it has been sent to retrieve, it will go dormant and return to the dependent entity as soon as she is back online.

ADoCSI makes extensive use of mobile, deliberative, heterogeneous and interface agents that talk to each other using a standard Agent language, like KQML and KIF, or FIPA.

ADOCSI is a CSI dissemination mechanism that is rich in features. However, that does not impose any restrictions on the way CAs, dependent entities or certificate holders work. Dependent entities and certificate holders do not need to have a profound understanding of the mechanics of ADOCSI and do not have to disrupt their normal operation, whatever that may be, in order to locate, retrieve and use CSI.

5 CONCLUSIONS

The issue of certificate revocation has started to attract the interest of the research sector (both the academic and the industrial one), in the past few years. As a result, many CSI mechanisms have been proposed, each one operating in a totally different way. In addition, many researchers have come up with many minor or major variations and extensions to the aforementioned CSI mechanisms.

We identified the need to build a theoretical framework for CSI mechanisms, a basis for their taxonomy, in order to be able to study CSI mechanisms in a more methodological manner. Our first contribution is that we have identified the theoretical building blocks of CSI mechanisms (see Section 2.2) and used these building blocks to present a taxonomy of the CSI mechanisms that have emerged in the scientific literature (see Section 2.3).

We have also identified the need to comparatively evaluate CSI mechanisms. This was not initially feasible because the functionality of each mechanism differed very much to the functionality of almost any other CSI mechanism. That is why we tried to create a qualitative evaluation framework that could be used to evaluate all aspects of a CSI mechanism, no matter what its functionality looked like. This was our second contribution and we presented it in Section 3.2.

5.1 Comparison of Related Work

We have used our evaluation framework to comparatively evaluate the CSI mechanisms that emerged in the scientific literature and the extensions and variations of those mechanisms that have appeared in the scientific literature. This gave us a much more clear view on the advantages and disadvantages of each CSI mechanism. Our comparative evaluation of CSI mechanisms also

revealed to us that most CSI mechanisms lacked specific qualities, like *adequate feedback, referral capability, completeness, adjustability* while there were other qualities missing from all CSI mechanisms, like *transparency and notification of revocation.* We presented our comparative evaluation of CSI mechanisms in Section 3.3.

The findings of our aforementioned comparative evaluation gave us a direction towards the next step in our research. We consider the aforementioned qualities that most (if not all) proposed CSI mechanisms lack are crucial for the operation of a CSI mechanism. The *feedback capability* and the *completeness capability* should be supported by a CSI mechanism in order to adequately inform a dependent entity of all necessary information regarding the status of a certificate, how was this status information obtained, when was it produced, who initiated the last change in that status information and why, as well as other related information. Only then will the dependent entity be able to make an informed decision as to whether to trust a specific signature (or electronic transaction, or any other occurrence of that certificate's use). Besides the practical need for these capabilities, Annex II of the EU Directive on a Community Framework for Electronic Signatures also presents these capabilities as a *required* CSI mechanism functionality.

Without the *referral capability*, the scalability of CSI mechanisms is limited and without the *adjustability capability* dependent entities (i.e. entities that depend on data contained in the certificate and on CSI to decide whether to trust or not to trust a digital signature) cannot be free to take as much (or less) risk they want to take, by trusting a specific signature. Without *adjustability* dependent entities can either fully trust a digital signature or not trust it at all, while in real life the notion of trust rarely works that way.

Finally, all proposed CSI mechanisms lack the *transparency* capability, thus assuming that dependent entities possess high levels of awareness regarding information security issues and can make informed decisions on such matters.

5.2 ADoCSI: A new Approach to CSI Dissemination

We continued our research, trying to identify a new CSI mechanism that could provide both the qualities and advantages of existing CSI mechanisms and also include the ones that were missing from existing CSI mechanisms. We assumed this could be achieved with the use of Software Agents and thus we designed the prototype of such a CSI mechanism, an extensible alternative mechanism for disseminating certificate status information, based on Software Agents, which we called ADoCSI. This was our third contribution and we presented it in Section 4.

ADoCSI reutilises existing CSI mechanisms' functionality and provides an interoperability layer between dependent entities, CAs and entities that produce the actual revocation information, and also provides a way for these entities to "meet" together, negotiate on their needs with respect to CSI and decide on a commonly agreed manner to satisfy these needs.

We comparatively evaluated ADoCSI to the other CSI mechanisms and verified that our mechanism possesses most of the already proposed mechanisms' qualities and also possesses some qualities that these other mechanisms do not possess (see Section 4.6). ADoCSI supports the capabilities that most (or all, depending on the capability) other CSI mechanisms lack, like *adequate feedback*, *referral capability*, *completeness*, *adjustability*, *transparency* and *notification of revocation*

One of the most important features of ADoCSI is that it supports *transparency*. Entities using PKI are not always knowledgeable or motivated enough to pay attention to security procedures. ADoCSI relieves these entities

from the burden of carrying out security related procedures in order to locate and retrieve Certificate Status Information and provides them in a timely manner with adequate and concise CSI whenever there is such a need.

ADoCSI is extensible. New features can be developed and integrated in ADoCSI (e.g. in the CA-CSI Agents and/or the User-CSI Agents) without disrupting the operations of dependent entities or the CAs.

In addition, ADoCSI is resilient to unreliable networks, because it can operate both in an online and offline manner, i.e. if the dependent entity goes offline for some reason, the User-CSI Agent will switch to dormant mode at the AMP and wait for the dependent entity to come back online; then the User-CSI Agent will transport itself back to the dependent entity along with the information it gathered.

Most of the existing CSI dissemination mechanisms restrict the way Certification Authorities are supposed to operate. CAs cannot re-partition the CSI space, or transport their CSI repositories elsewhere, because dependent entities will not be able to track those changes easily. ADoCSI lifts that restriction. CAs can re-partition their CSI space, transport the CSI repositories and also maintain CSI in a number of different formats.

ADoCSI provides for adjustable CSI. Dependent entities can retrieve CSI that meets their own freshness, timeliness or other requirements, without even having to understand the mechanics of the CSI dissemination process and without having to manually intervene to the process of CSI dissemination. The only other mechanism offering fine-grained adjustability is the Freshness constrained Revocation, but it achieves that at the cost of low scalability.

The cost for implementing and deploying a base infrastructure for ADoCSI is probably higher than that of any other CSI mechanism. However, once ADoCSI is deployed, it can prove to be a very economic mechanism.

Dependent entities and certificate holders would not have to have a profound understanding of CSI mechanisms to use ADoCSI, as is the case with some of the existing mechanisms. Furthermore, neither of them has to disrupt their normal working practice in order to locate and retrieve the necessary CSI. CSI is delivered to the dependent entities transparently. This ensures CSI delivery to the entities that need it. This ensured CSI delivery is beneficiary both for the entities who need CSI and for the CAs themselves, who do not enjoy seeing their direct clients (certificate holders) or indirect clients (dependent entities) trust the "wrong", i.e. invalid, certificates in their transactions due to their ignorance because this invalidates their *raison d' être*.

The new functionality ADoCSI introduces comes at a price. The introduction of Software Agents in the area of CSI mechanisms lead to a series of security issues emerging, regarding the use of Software Agents. We have presented solutions to these issues in Section 4.3.5

ADoCSI makes the following, reasonable, assumptions: there will be a sufficient number of AMPs and each AMP will be able to sustain life for a sufficient number of CA-CSI and User-CSI Agents. Furthermore, ADoCSI assumes that all Agents involved in the scheme will be using the same Agent Communication Language. Although these assumptions do not constitute technical implementation obstacles per se, there might be other obstacles in actually implementing ADoCSI in a wide scope production, real-life scenario.

Commercial CAs might refuse to have their CA-CSI Agents residing at the same AMP as another CAs Agents, or in the AMP developed by an organisation that is somehow related with another CA. Furthermore, if a CA has a large user base (certificate holders' base) that CA may refuse to have its CA-CSI Agents serve CSI requests from User-CSI Agents developed by other CAs. In general, real life full-scale implementation of ADoCSI expects CAs to

develop multi-level collaboration, in order to deliver CSI. This kind of collaboration takes time to materialise in the business world.

Taking into consideration the above, a degenerate version of ADoCSI might prove to be the most economic and efficient one. We refer to the case where each CA is operating its own AMP(s), hosting only its own CA-CSI Agents, while the Directory Facilitator at the AMPs know the location of the AMPs of all other CAs. This is a very close analogy to the private network of the banking system. A dependent entity would have to send its User-CSI Agent to an AMP it already knows the location of (in banking terms, "Acquirer" AMP/CA), and the latter would refer it to the AMP that contains the appropriate CSI (in banking terms again, "Issuer" AMP/CA).

5.3 Future Work

The CSI mechanisms we have presented in Section 2.3 do not take into consideration the reasons for the revocation of a certificate when performing certificate path validation. ADoCSI supports taking revocation reasons into account, but for the time being the way each revocation reason will affect the outcome of the certificate validation is up to the dependent entity to decide (such preferences are set by the dependent entity at the time of the initial installation of the Interface Agent). However, the Interface Agent may very well be asking too much from the dependent entity, i.e. to have the necessary expertise to assert on the way she wants the various revocation reasons to affect future certificate validation processes. Even an experienced computer user, with an adequate level of information security awareness, could feel puzzled in front of such a puzzle. We believe it can be of high value to research the way various revocation reasons should affect the outcome of certificate validations and establish a best practice scenario.

During the process of certificate path validation, none of the aforementioned mechanisms, including ADoCSI, check for (or could check for) the existence of more than one revoked certificates containing the same public key but at the same time belonging to different certification paths. Obviously, the aforementioned certificates should belong to the same entity, but it could also be the case of two or more entities that had -on purpose- the same key pair, certified by different CAs.

Assume that certificates C_k and C'_k belong to the respective certificate chains C and C′, depicted in the table below, and contain the same subject public key. Should C_k be revoked, that does not mean that C'_k will be revoked as well, as it belongs to another certificate chain. The fact that C_k has been revoked should affect C'_k in a different way, depending on the reason for the revocation of C_k (Fox and LaMacchia 1998). Currently, PKI does not support the revocation, or any other action upon a subject's certificate, if another certificate containing the same public key is revoked.

$C_1, C_2, \ldots C_{k-1}, C_k, \ldots C_n,$
$C'_1, C'_2, \ldots C'_{k-1}, C'_k, \ldots C'_n$

Table 44: Certificate chains

For the sake of the example, we assume that CRL is the CSI mechanism used in our case. C_k could be revoked due to the following reasons:

1. The public key contained in C_k should not be trusted anymore because it has been compromised. The CRL entry should contain the reason 'keyCompromise'. In this case, C'_k should be revoked as well, because it contains the same, compromised public key.
2. The binding between the public key contained in C_k and the subject name contained in C_k should no longer be trusted, because the affiliation of the certificate holder has changed, there is a new certificate that supersedes C_k

or because the certificate holder will not be using anymore C_k. The respective reason codes that should be used in the CRL entry are: 'affiliationChanged', 'superseded' and 'CessationOfOperation'. If C'_k - contains information that bind the person denoted by the subject name in C'_k to the same affiliation as is the case with C_k then C'_k should be revoked as well. If C_k was revoked because it was superseded then C'_k might not need to be revoked. Finally, if C_k was revoked because the certificate holder will not be using that certificate anymore, then C'_k should be revoked or should not be revoked depending on the reason why the certificate holder will not be using C'_k anymore.

3. C_{k-1} (certificate holder k-1 is the issuer of certificate Ck) is no longer in position to vouch for the validity of Ck. This would be due to a compromise of the key contained in C_{k-1}. The reason code that should be used in the CRL entry is: 'cACompromise'. In this case, C'_k should not be revoked, because C'_{k-1} (and every certificate higher in the hierarchy of C') is not affected because of the compromise of C_{k-1}.

We have demonstrated that there are cases when the revocation of a certificate belonging to one certificate chain may provoke the revocation of a certificate that belongs to another certificate chain but bears the same public key. However, the current, supported certificate path validation mechanisms and CSI location mechanisms do not support such actions. Even if such actions were supported buy the aforementioned mechanisms, there would still be questions to be answered regarding the revocation of two or more certificates containing the same public key. One of the issues that would have to be tackled with, is what would happen in case a certificate needs to be revoked for a reason that is not included in the list of reasons proposed by (International Organization for Standardization 1994), (International Organization for Standardization 1995), (International Organization for

Standardization 1996), (Housley, Ford et al. 1999). It could be the case that a CA (e.g. C_{k-1}) has realised that an entity (e.g. C_k) has been misidentified, either because of a procedural registration error or because the entity has provided false identification to the CA during the registration phase. In this case, none of the existing proposed revocation reasons do not match the actual revocation reason. Another issue that would have to be tackled with, is the actions to be taken with respect to C'_k in case C_k appears in a CRL with the value certificateHold in the entry extension reasonCode. A certificate could be suspended (certificateHold) for many, different reasons and it would not be clear in this case what actions should be taken either by the entities that own the certificates in the certification chain C' or by the dependent entities that wish to use certificates (e.g. C'_k, C_k) that belong to these chains.

ADoCSI could possibly provide the means to deal with the aforementioned problems, since the Agents involved in ADoCSI do have access to a common pool of CSI resources (the set of all AMPs and CA-CSI Agents). User-CSI Agents could seek the necessary information in this pool in order to verify the validity of a certificate, and check whether a public key is contained in more than one certification paths.

Another interesting future topic for further research on ADoCSI is how should ADoCSI be deployed inside closed networks or user groups, that communicate also with the external world, e.g. inside a corporation. ADoCSI would not prove to be efficient for use by employees of a corporation, who communicate also with people outside the corporation. The reason why is that all dependent entities inside the corporate network would be sending User-CSI Agents all the time in order to retrieve CSI and that might reduce the availability of valuable corporate network resources. On the contrary, a traditional scheme like CRL scales better in this case because it can be cache proxied, therefore only the first dependent entity actually downloads the CRL

from the Internet. However, ADoCSI presents other advantages (e.g. transparency) that would be useful even in a corporate environment. An idea in order to efficiently deploy ADoCSI within such an environment would be to have a Proxy AMP at the corporate network borders that would capture all outgoing User-CSI Agents, detain them for a limited period of time and ask them to discuss with the other detained User-CSI Agents if they can optimise the CSI search and retrieval process, for instance if only 1-2 User-CSI Agents can go out there and retrieve the CSI all corporate users requested and then come back to the Proxy AMP to redistribute this CSI to the appropriate User-CSI Agents. That would save network bandwidth because fewer User-CSI Agents would have to go out and in of the corporate network, but also (and mainly) because there would be no duplication of CSI being transferred in the company network. Further research on that could prove to be interesting.

6 REFERENCES

A. Malpani, R. H., and T. Freeman, 2002. PKIX-SCVP: Simple Certificate Validation Protocol (SCVP). PKIX Working Group Internet Draft, IETF

Abdalla, M., S. Miner and C. Namprempre, 2001. Forward-Secure Threshold Signature Schemes, Lecture Notes in Computer Science 2020

Abdalla, M. and L. Reyzin, 2000. A New Forward-Secure Digital Signature Scheme, Lecture Notes in Computer Science 1976

Adams C., F. S., 1999. Internet X.509 Public Key Infrastructure Certificate Management Protocols, Request for Comments 2510, IETF

Adams C., Z. R., 1998. A General, Flexible Approach to Certificate Revocation, Entrust Technologies

Ansper, A., A. Buldas, M. Roos and J. Willemson, 2001. Efficient Long-Term Validation of Digital Signatures, Lecture Notes in Computer Science 1992

Arends, R., R. Austein, M. Larson, D. Massey and S. Rose, 2005a. DNS Security Introduction and Requirements. IETF RFC 4033.

Arends, R., R. Austein, M. Larson, D. Massey and S. Rose, 2005b. Resource Records for the DNS Security Extensions, IETF RFC 4034

Bellare, M. and S. K. Miner, 1999. A Forward-Secure Digital Signature Scheme, Lecture Notes in Computer Science 1666

Bellare, M. and P. Rogaway, 1997. Collision-Resistant Hashing: Towards Making UOWHFs Practical. Proceedings of Advances in Cryptology - CRYPTO'97, Springer-Verlag.

Beno Libert and J.-J. Quisquater, 2003. Efficient revocation and threshold pairing based cryptosystems. Proceedings of 22nd Annual Symposium on Principles of Distributed Computing, Boston, Massachusetts, ACM Press.

Berkovits S., C. S., Furlong J. A., Geiter J. A., and Guild J. C., 1994. Public Key Infrastructure Study: Final Report, Produced by the MITRE Corporation for NIST, MITRE Corporation

Berners-Lee T., Fielding R. and M. L., 1998. RFC 2396: Uniform Resource Identifiers (URI): Generic Syntax, IETF

Berry, T. and D. W. Gresty, 2001. A Proposed Architecture for Efficient Searching of Certificate Revocation Lists. Proceedings of Second Annual Network Symposium, Liverpool, UK, Liverpool John Moores University.

Boneh, D., X. Ding, G. Tsudik and C. M. Wong, 2001. A Method for Fast Revocation of Public Key Certificates and Security Capabilities. Proceedings of 10th USENIX Security Symposium.

Borselius, N., C. J. Mitchell and A. Wilson, 2001. Undetachable Threshold Signatures, Lecture Notes in Computer Science 2260

Bruschi, D., A. Curti and E. Rosti, 2003. A quantitative study of Public Key Infrastructures, Computers & Security 22(1)

Canetti, R. and S. Goldwasser, 1999. An Efficient Threshold Public Key Cryptosystem Secure Against Adaptive Chosen Ciphertext Attack, Lecture Notes in Computer Science 1592

Cart, V. and F. Password, 1995. A Public Key Infrastructure for US Government Unclassified but Sensitive Applications, NIST Report

CCITT, 1998. CCITT Recommendations X.500-X.521, Data Communication Networks Directory

Chadwick, D. W., 2002. Internet X. 509 Public Key Infrastructure Operational Protocols-LDAPv3. IETF Internet Draft.

Chess, D., B. Grosof, C. Harrison, D. Levine, C. P. Pressarris and G. Tsudik, 1995. Itinerant Agents for Mobile Computing, IEEE Personal Communications 2(5)

Chokhani, S. and W. Ford, 1999. Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework. IETF RFC 2459.

Cooper, D. A., 1998. A Closer Look at Revocation and Key Compromise in Public Key Infrastructures, Proceedings of the 21st National Information Systems Security Conference

Cooper, D. A., 1999. A Model of Certificate Revocation. Proceedings of 15th Annual Computer Security Applications Conference (ACSAC), Anaheim, California, USA, IEEE Computer Society.

Cooper, D. A., 2000. A more efficient use of delta-CRLs. Proceedings of 2000 IEEE Symposium on Security and Privacy, Oakland, California, USA, IEEE Computer Society.

Cugola, G., C. Ghezzi, G. P. Picco and G. Vigna, 1997. Analyzing mobile code languages, Mobile Object Systems Lecture Notes in Computer Science 1222

Desmedt, Y., 1993. Threshold cryptosystems, Lecture Notes in Computer Science 718

Egelman, S., J. Zaritsky and A. Jones, Improved Certificate Revocation with OCSP

Eichler, S. and B. Muller-Rathgeber, 2005. Performance analysis of scalable certificate revocation schemes for ad hoc networks. Proceedings of 30th Conference on Local Computer Networks, Sydney, Australia, IEEE Computer Society.

Elwailly, F. F., C. Gentry and Z. Ramzan, 2004. QuasiModo: Efficient certificate validation and revocation. Proceedings of Public Key Cryptography 2004, Springer Verlag.

Erdur, R. C. and O. u. Dikenelli, 2002. A FIPA-Compliant Agent Framework with an Extra Layer for Ontology Dependent Reusable Behaviour, Lecture Notes in Computer Science 2457

Esparza, O., M. Fernandez, M. Soriano, J. L. Munoz and J. Forne, 2003. Mobile agent watermarking and fingerprinting: Tracing malicious hosts. Proceedings of Database and Expert Systems Applications, Prague, Czech Republic, Springer-Verlag.

Esparza, O., M. Soriano, J. L. Munoz and J. Forne, 2003. A protocol for detecting malicious hosts based on limiting the execution time of mobile agents. Proceedings of Eighth IEEE International Symposium on Computers and Communication, Kemer Antalya, Turkey, IEEE Computer Society.

EU Parliament, 1999. Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures

Farmer, W., J. Guttman and V. Swarup, 1996. Security for Mobile Agents: Authentication and State Appraisal, Lecture Notes in Computer Science 1146

Finin, T., J. Weber, G. Wiederhold and M. Genesereth, 1993. Draft Specification of the KQML Agent Communication Language, DARPA Knowledge Sharing Initiative, External Interfaces Working Group

Forné, J. C. C. a. J., 2000. A model to evaluate certificate revocation. Proceedings of 4th World Multiconference on Systemics, Cybernetics and Informatics (SCI2000) Orlando, Florida, USA.

Foundation for Intelligent Physical Agents, 2004. Agent Management Specification, Foundation for Intelligent Physical Agents Specification

Fox, B. and B. LaMacchia, 1998. Certificate Revocation: Mechanics and Meaning, Lecture Notes in Computer Science 1465

Fox, B. and B. LaMacchia, 1999. Online Certificate Status Checking in Financial Transactions: The Case for Re-issuance, Lecture Notes in Computer Science 1648

Franklin, S. and A. Graesser, 1997. Is it an Agent, or Just a Program?: A Taxonomy for Autonomous Agents. Proceedings of 3rd International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag.

Freeman T, Housley R., Malpani A., Cooper D. and P. W., 2007. Server-Based Certificate Validation Protocol (SCVP). IETF RFC 5055.

Frost, H. R. and M. R. Cutkosky, 1996. Design for Manufacturability via Agent Interaction. Proceedings of ASME Design Engineering Technical Conferences and Computers in Engineering Conference, Irvine, CA.

Genesereth, M., 1993. An agent-based approach to software interoperability, Technical Report Logic-91-6, Logic Group, CSD, Stanford University.

Genesereth, M. R. and R. E. Fikes, 1992. Knowledge Interchange Format, Version 3.0 Reference Manual, Technical Report Logic-92-1, Computer Science Department, Stanford University, 1992

Genesereth, M. R. and S. P. Ketchpel, 1994. Software agents, Communications of the ACM 37(7)

Gong, L., M. Mueller, H. Prafullchandra and R. Schemers, 1997. Going Beyond the Sandbox: An Overview of the New Security Architecture in the Java Development Kit 1.2. Proceedings of USENIX Symposium on Internet Technologies, Monterey, California, USENIX Association.

Gong, L. and R. Schemers, 1998. Signing, Sealing, and Guarding Java Objects, Lecture Notes in Computer Science 1419

Gorrieri, R. and G. Marchetti, 1998. Applet Watch-Dog: A monitor controlling the execution of Java applets. Proceedings of IFIP TC11 14th international conference on Information Security Vienna, Austria, Austrian Computer Society.

Gritzalis, D., K. Moulinos, J. Iliadis, C. Lambrinoudakis and S. Xarhoulakos, 2001. PyTHIA: Towards Anonymity in Authentication. Proceedings of the IFIP 16th International Conference on Information Security, Paris-France, Kluwer Academic Press

Gritzalis, S. and J. Iliadis, 1998. Addressing security issues in programming languages for mobile code. Proceedings of DECA '98 Workshop of Database and Expert Systems Applications, IEEE Computer Society Press

Gritzalis, S. and D. Spinellis, 1997. Addressing Threats and Security Issues in World Wide Web Technology. Proceedings of Joint Working Conference IFIP TC-6 and TC-11 "Communications and Multimedia Security III", Athens, Greece, Chapman & Hall.

H, K., A. K and N. S, 1999. Performance Evaluation of Public-key Certificate Revocation System with Balanced Hash Tree. Proceedings of International Workshop on Security (IWSEC), Aizu, Japan, IEEE Computer Society Press.

Hallam-Baker, P., 1999. OCSP Extensions. IETF Internet Draft.

Hallam-Baker, P. and W. Ford, 1998. Internet X. 509 Public Key Infrastructure Enhanced CRL Distribution Options. IETF Internet Draft.

Herzberg, A., M. Jakobsson, S. a. Jarecki, H. Krawczyk and M. Yung, 1997 Proactive public key and signature systems Proceedings of the 4th ACM conference on Computer and communications security Zurich, Switzerland ACM Press

Hohl, F., 2000. A framework to protect mobile agents by using reference states,

Hormann, T. P., K. Wrona and S. Holtmanns, 2006. Evaluation of certificate validation mechanisms, Computer Communications 29(3)

Housley, R., W. Ford, W. Polk and D. Solo, 1999. Internet X.509 Public Key Infrastructure Certificate and CRL Profile. IETF RFC 2459.

IBM, 2002. Aglets Software Development Kit. available at http://www.trl.ibm.co.jp/aglets/ and http://aglets.sourceforge.net/

Iliadis, I., D. Spinellis, S. Katsikas and B. Preneel, 2000. A Taxonomy of Certificate Status Information Mechanisms. Proceedings of Information Security Solutions Europe, Barcelona-Spain, European Forum for Electronic Business

Iliadis, J., 1999. On the Dissemination of Certificate Status Information, University of London, Royal Holloway and Bedford New College, Department of Mathematics, MSc Thesis. Supervised by C. Mitchell

Iliadis, J., S. Gritzalis and D. Gritzalis, 2003. ADoCSI: Towards a Transparent Mechanism for Disseminating Certificate Status Information, Computer Communications 26(16)

Iliadis, J., S. Gritzalis and V. Oikonomou, 1998. Towards Secure Downloadable Executable Content: The Java Paradigm. Proceedings of SAFECOMP '98 EWICS & IFIP 17th International Conference on Computer Safety, Reliability and Security, Heidelberg, Germany, Springer Verlag.

Iliadis, J., S. Gritzalis, D. Spinellis, D. D. Cock, B. Preneel and D. Gritzalis, 2003. Towards a Framework for Evaluating Certificate Status Information Mechanisms, Computer Communications 26(16)

Iliadis, J., D. Spinellis, S. Katsikas, D. Gritzalis and B. Preneel, 2000. Evaluating certificate status information mechanisms. Proceedings of the 7th ACM Conference on Computer and Communication Security. S. J. a. P. Samarati, ACM Press

International Organization for Standardization, 1994. ISO/IEC 9594-8, Open Systems Interconnection - The Directory: Authentication Framework.

International Organization for Standardization, 1995. Technical corrigenda to Rec. X.500 International Organization for Standardization, Geneva, Switzerland/IEC 9594 resulting from Defect Reports 9594/128.

International Organization for Standardization, 1996. Final Text of Draft Amendments DAM 4 to International Organization for Standardization, Geneva, Switzerland/IEC 9594-2, DAM 2 to International Organization for Standardization, Geneva, Switzerland/IEC 9594-6, DAM 1 to International Organization for Standardization, Geneva, Switzerland/IEC 9594-7, and DAM 1 to International Organization for Standardization, Geneva, Switzerland/IEC 9594-8 on Certificate Extensions.

Itkis, G. and L. Reyzin, 2002. SiBIR: Signer-Base Intrusion-Resilient Signatures. Proceedings of 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, Springer-Verlag.

Jansen, W. A., 2000. Countermeasures for mobile agent security, Computer Communications 23(17)

Jing, J., P. Liu, D. Feng, J. Xiang, N. Gao and J. Lin, 2003. ARECA: a highly attack resilient certification authority. Proceedings of 1st Workshop on Survivable and Self-regenerative systems: in association with 10th ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, ACM Press.

Jingfeng, L., Z. Yuefei, P. Heng and L. Shengli, 2005. A distributed certificate revocation scheme based on one-way hash chain for wireless ad hoc networks, IEE Mobility Conference 2005. The Second International Conference on Mobile Technology, Applications and Systems

John Solis and G. Tsudik, 2006. Simple and Flexible Revocation Checking with Privacy. 6th International Workshop on Privacy Enhancing Technologies. Cambridge, UK, Springer-Verlag. 4258

Katz E.D., Butler M. and M. R., 1994. A Scalable HTTP Server: The NCSA Prototype. Proceedings of First International WWW Conference, Geneva, Switzerland.

Kikuchi, H., K. Abe and S. Nakanishi, 1999. Performance evaluation of certificate revocation using k-valued hash tree. Proceedings of 2nd International Workshop on Information Security, Springer-Verlag.

Klobucarklobucar, T. and B. Jerman-Blazicjerman-Blazic, 1999. A formalisation and evaluation of certificate policies, Computer Communications 22(12)

Kocher, P. C., 1998. On Certificate Revocation and Validation, Lecture Notes in Computer Science 1465

Koga, S., J. C. Ryou and K. Sakurai, 2004. Pre-production methods of a response to certificates with the common status - Design and theoretical evaluation. Proceedings of Public Key Infrastructure, Samos, Greece, Springer-Verlag.

Kohnfelder, L., 1978. Towards a Practical Public-Key Cryptosystem, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, BSc Thesis. Supervised by L. Adleman

Kotzanikolaou, P., M. Burmester and V. Chrissikopoulos, 2000. Secure Transactions with Mobile Agents in Hostile Environments. Proceedings of 5th Australasian Conference on Information Security and Privacy, Springer-Verlag.

Labrou, Y. and T. Finin, 1997. A Proposal for a new KQML Specification, University of Maryland Baltimore County, Baltimore, MD Technical Report, TR-CS-97-03

Lakshminarayanan, A. and T. L. Lim, 2006. Augmented certificate revocation lists. Proceedings of 11th Australasian Conference on Information Security and Privacy, Melbourne, Australia, Springer-Verlag.

Lamport, L., 1981. Technical Note: Password Authentication with Insecure Communication, Communications of the ACM 24(11)

Lekkas, D., S. Gritzalis and L. Mitrou, 2005. Withdrawing a declaration of will - Towards a framework for digital signature revocation, Internet Research 15(4)

Li, B. H., Y. L. Zhao and Y. B. Hou, 2004. Performance optimizations for certificate revocation. Proceedings of Workshop on IP Operations and Management Proceedings (IPOM) - Self-Measurement & Self-Management of IP Networks & Services, Beijing, China, IEEE Computer Society.

Li, J. F., Y. F. Zhu, H. Pan and D. W. Wei, 2005. A new public key certificate revocation scheme based on one-way hash chain. Proceedings of 6th International Conference on Advances in Web-Age Information Management, Hangzou, China, Springer-Verlag.

McDaniel, P. and A. Rubin, 2001. A response to "can we eliminate certificate revocation lists?" Proceedings of 4th International Conference on Financial Cryptography, Anguilla, British West Indies, Springer-Verlag.

Menezes, A. J., P. C. V. Oorschot and V. S. A., 1997. Handbook of applied cryptography, CRC Press.

Micali, S., 1996. Efficient certificate revocation. The MIT Pitman Pressress. Cambridge, MA, USA, Massachusetts Institute of Technology, Laboratory for Computer Science

Micali, S., 2002. NOVOMODO: Scalable Certificate Validation And Simplified PKI Management. Proceedings of 1st Annual PKI Research Workshop, Dartmouth College, USA, NIST.

Millen, J. K. and R. N. Wright, 1998 Certificate Revocation the Responsible Way Proceedings of Conference on Computer Security, Dependability, and Assurance: From Needs to Solutions IEEE Computer Society.

Minsky, Y., R. van Renesse, F. Schneider and S. Stoller, 1996. Cryptographic support for fault-tolerant distributed computing. Proceedings of Seventh ACM SIGOPS European Workshop, Connemara, Ireland, ACM Press.

Mockapetris, P., 1987. Domain Names - Implementation and Specification, IETF RFC 1035

Munoz, J. L., 2003. Design and Evaluation of Certificate Revocation Systems. Dept of Telematics Engineering, Technical University of Catalonia, Catalonia, PhD Thesis. Supervised by J. F. Munoz

Munoz, J. L., J. Forne and J. C. Castro, 2002. Evaluation of certificate revocation policies: OCSP vs. Overissued-CRL. Proceedings of 13th International Workshop on Database and Expert Systems Applications (DEXA'02), Aix-en-Provence, France, Springer-Verlag.

Munoz, J. L., J. Forne, O. Esparza and M. Rey, 2005. Efficient certificate revocation system implementation: Huffman merkle hash tree (HuffMHT). Proceedings of 2nd International Conference on Trust, Privacy, and Security in Digital Business, Copenhagen, Denmark, Springer-Verlag.

Munoz, J. L., J. Forne, O. Esparza and M. Soriano, 2004. Certificate revocation system implementation based on the Merkle hash tree, International Journal of Information Security 2(2)

Munoz, J. L., J. Forne, O. Esparza and M. Soriano, 2004. CERVANTES - A certificate validation test-bed. Proceedings of 1st European PKI Workshop, Samos, Greece, Springer-Verlag.

Munoz, J. L., J. Forne, O. Esparza, M. Soriano and D. Jodra, 2003. Evaluation of revocation systems with a JAVA test-bed. Proceedings of 14th International Workshop on Database and Expert Systems Applications, Prague, Czech Republic, Springer-Verlag.

Munoz, J. L., J. Forne, O. Esparza and N. Soriano, 2003. A test-bed for certificate revocation policies. Proceedings of 2003 IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing, Victoria, Canada, IEEE Computer Society Press.

Myers, M., R. Ankney, A. Malpani, S. Galperin and C. Adams, 1999. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. RFC 2560.

Naor, M. and K. Nissim, 1998. Certificate revocation and certificate update. Proceedings of the Seventh Usenix Security Symposium, USENIX Association

Necula, G. C., 1998. Compiling with proofs [Ph D dissertation], Pittsburgh, USA: Carnegie Mellon University

Nilsson, H., P. Van Eecke, M. Medina, D. Pinkas and N. Pope, 1999. Final Report of the EESSI Expert Team, EESSI–European Electronic Signature Standardization Initiative, July

Nwana, H. S., 1996. Software Agents: An Overview, Knowledge Engineering Review 11(3)

Papapanagiotou, K., K. Markantonakis, Q. Zhang, W. G. Sirett and K. Mayes, 2005. On the performance of certificate revocation protocols based on a Java Card certificate client implementation. Proceedings of IFIP TC11 20th International Information Security Conference: Security and Privacy in the Age of Ubiquitous Computing, Chiba, Japan, Springer-Verlag.

Perdikeas, M. K., F. G. Chatzipapadopoulos, I. S. Venieris and G. Marino, 1999. Mobile agent standards and available platforms, Computer Networks-the International Journal of Computer and Telecommunications Networking 31(19)

Petrie, C. J., 1997. What's An Agent... And What's So Intelligent About It?, Internet Computing, IEEE 1(4)

Popescu, B. C., B. Crispo and A. S. Tanenbaum, 2003. A certificate revocation scheme for a large-scale highly replicated distributed system. Proceedings of 8th International Symposium on Computers and Communication, Kemer-Antalya, Turkey, IEEE Computer Society Press.

Riordan, J. and B. Schneier, 1998. Environmental Key Generation Towards Clueless Agents, Lecture Notes in Computer Science 1419

Rivest, R. L., 1998. Can We Eliminate Certificate Revocations Lists?, Lecture Notes in Computer Science 1465

Rojanapasakorn, A. and C. Sathitwiriyawong, 2004a. A performance study of over-issuing delta-CRLs with distribution points. Proceedings of 18th International Conference on Advanced Information Networking and Applications, Fukuoka, Japan, IEEE Computer Society.

Rojanapasakorn, A. and C. Sathitwiriyawong, 2004b. A simulation study of over-issuing Delta-CRLS with distribution points. Proceedings of IEEE Conference TENCON 2004, Analog and Digital Techniques in Electrical Engineering, Chiang Mai, Thailand, IEEE Computer Society Press.

Sander, T. and C. F. Tschudin, 1998. Protecting Mobile Agents Against Malicious Hosts, Lecture Notes in Computer Science 1419

Santesson, S., W. Polk, P. Barzin and M. Nystrom, 2001. Internet X. 509 Public Key Infrastructure Qualified Certificates Profile. IETF RFC3039.

Schneier, B., 1996. Applied Cryptography, John Wiley & Sons.

Schwingenschlogl, C., S. Eichler and B. Muller-Rathgeber, 2006. Performance of PKI-based security mechanisms in mobile ad hoc networks, International Journal of Electronics and Communications 60(1)

Shimshon, B. and H. Jonathan, 1998. A Comparison of Certificate Validation Methods for Use in a Web Environment. Bedford, Massachusetts, MITRE Corporation

Shoup, V., 2000. Practical Threshold Signatures, Lecture Notes in Computer Science 1807

Smith, S. W., 2003. Effective PKI Requires Effective HCI. Proceedings of Workshop on Human-Computer Interaction and Security Systems, Fort Lauderdale, Florida.

Spinellis, D., 2000. Reflection as a mechanism for software integrity verification, ACM Transactions on Information and System Security 3(1)

Spruit M., 1998. Competing against human failing Proceedings of IFIP TC11 14th International Conference on Information Security, Vienna, Austria, Austrian Computer Society.

Spruit, M. E. M. and M. Looijen, 1996. IT security in Dutch practice, Elsevier Science Computers and Security 15(2)

Stanton, J. M., K. R. Stam, P. Mastrangelo and J. Jolton, 2005. Analysis of end user security behaviors, Computers & Security 24(2)

Stubblebine, S. G., 1995 Recent-secure authentication: enforcing revocation in distributed systems Proceedings of the 1995 IEEE Symposium on Security and Privacy IEEE Computer Society

Vanrenen, G., S. Smith and J. Marchesini, 2006. Distributing security-mediated PKI, International Journal of Information Security(5)

Vigna, G., 1998. Cryptographic Traces for Mobile Agents, Lecture Notes in Computer Science 1419

Wayne Jansen and T. Karygiannis, 1999. Mobile Agent Security - NIST Special Publication 800-19, Computer Security Division, NIST

Wilhelm, U. G., 1999. A technical approach to privacy based on mobile agents protected by tamper-resistant hardware. Department d' Informatique, Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland, PhD Thesis. Supervised by A. Schiper

Wilhelm, U. G., S. Staamann and L. Buttyan, 1998. On the problem of trust in mobile agent systems. Proceedings of Symposium on Network and Distributed System Security, San Diego, California, Internet Society.

Wright Rebecca N., Lincoln Pitman Pressatrick D. and M. J. K., 2000. Efficient fault-tolerant certificate revocation. Proceedings of 7th ACM conference on Computer and Communications Security, Athens, Greece, ACM.

Yang, J. P., K. Sakurai and K. H. Rhee, 2006. Distributing security-mediated PKI revisited. Proceedings of European Conference on Public Key Infrastructures (EuroPKI), Springer-Verlag.

Yevgeniy, D., K. Jonathan, X. Shouhuai and Y. Moti, 2003. Strong Key-Insulated Signature Schemes. Proceedings of 6th International Workshop on Theory and Practice in Public Key Cryptography: Public Key Cryptography, Springer-Verlag.

YoungGyo Lee, J. A., Seungjoo Kim and, Dongho Won, 2006. A PKI System for Detecting the Exposure of a User's Secret Key. Proceedings of Third European PKI Workshop: Theory and Practice, EuroPKI 2006, Turin, Italy, Springer-Verlag.

Yum, D. H., J. E. Kang and P. J. Lee, 2003. Advanced certificate status protocol, Computer Network Security Lecture Notes in Computer Science 2776

Zhang, K., 1998. Threshold Proxy Signature Schemes, Lecture Notes in Computer Science 1396

Zhang, X. N., 1997. Secure Code Distribution, IEEE Computer 30(6)

Zheng, P. F., 2003. Tradeoffs in certificate revocation schemes, Computer Communication Review 33(2)

Zhou, J., F. Bao and R. Deng, 2003. Validating Digital Signatures without TTP's Time-Stamping and Certificate Revocation, Proceedings of 2003 Information Security Conference, Bristol, UK, October

Zhou, J. Y., 2003. Efficient signature validation based on a new PKI. Proceedings of 4th International Conference on E-Commerce and Web Technologies, Prague, Czech Republic.

Zhou, J. Y., F. Bao and R. Deng, 2003. An efficient public-key framework. Proceedings of 5th International Conference on Information and Communications Security, Huhehaote, China.

Zhou, L., F. B. Schneider and R. V. Renesse, 2002. COCA: A secure distributed online certification authority, ACM Transactions on Computer Systems 20(4)