FFI-rapport 2009/01546

Model and specification for analyzing the scalability of a Public Key Infrastructure

Eli Winjum and Anders Fongen

Norwegian Defence Research Establishment (FFI)

1 October 2009

FFI-rapport 2009/01546

1070

P: ISBN 978-82-464-1648-9
E: ISBN 978-82-464-1649-6

Keywords

Infrastruktur for offentlige nøkler

Informasjonssikkerhet

Digitale sertifikater

Digitale signaturer

Skalerbarhet

Approved by

Eli Winjum	Project Manager
Vidar S. Andersen	Director

Summary

Most security mechanisms rely on cryptographic keys and other secret values. Key management is crucial. A Public Key Infrastructure (PKI) is commonly established as a basis for key management. So far, PKIs have been designed for wired systems where resource consumption has not been a limiting factor. Both NATO and the Norwegian Defence plan to deploy PKIs. For the tactical domain, however, PKI and PKI-dependant applications should not be planned without knowledge of the communication capacities required to operate a PKI. The goal of our study is to provide such knowledge. As far as we know, neither academic nor military research has published studies on this topic.

This report serves as a reference for subsequent analyzes of the impact of PKI usage under varying conditions. Based on a high-level description of the NATO PKI, the report models and specifies a generic PKI. The model encompasses user scenarios and traffic imposed by the PKI as well as by user applications. Previous publications model and specify the underlying wireless ad hoc network.

Sammendrag

De fleste sikkerhetsmekanismer er basert på kryptografiske nøkler og andre hemmelige verdier. Sikker nøkkelhåndtering er derfor kritisk. Infrastrukturer for offentlige nøkler – *Public Key Infrastructure* (PKI) – blir ofte satt opp som basis for nøkkelhåndtering. Slike infrastrukturer har hittil blitt utarbeidet for systemer i faste trådbaserte kommunikasjonsnett hvor kommunikasjonskapasitet ikke er en begrensende faktor. Både Nato og det norske Forsvaret planlegger å bygge ut PKI. Bruk av PKI og PKI-avhengige applikasjoner over trådløse taktiske kommunikasjonsnett bør imidlertid ikke planlegges uten kunnskap om hvilke kommunikasjonskapasiteter en PKI krever. Formålet med studien vår er å framskaffe slik kunnskap. Så langt vi kjenner til, har hverken akademisk eller militær forskning publisert kvantitative skalerbarhetsstudier av PKI.

Denne rapporten er et referansedokument for påfølgende analyser av effekten PKI vil ha på trafikkavviklingen under varierende vilkår. Rapporten modellerer en generisk PKI basert på NATO PKI. Modellen inneholder ulike bruksscenarioer. Trafikk forårsaket av PKI så vel som av ulike applikasjoner er modellert og spesifisert. Det underliggende kommunikasjonsnettet er et trådløst ad hoc-nett og er modellert og spesifisert i tidligere publikasjoner.

Contents

1 Introduction

1.1 Background

Future security schemes for network centric warfare should support operations characterized by dynamic organization and high mobility. Such scenarios demand seamless security solutions between end users. In addition to providing adequate security, security solutions should be "light-weight", scalable, distributed and flexible. Units from several countries, military as well as civil, may take part in operations. Seamless information sharing calls for interoperable security solutions. An objective of the FFI project *Fundamental Technologies and Trends in Information Security* (GOSIKT) is to study security technologies for system architectures with different bandwidth, battery, processing and storing capacities.

Most security mechanisms rely on cryptographic keys and other secret values. Key management is crucial. A Public Key Infrastructure (PKI) is commonly established as a basis for key management. So far, PKIs have been designed for systems based on wired networks. Resource consumption has not been a limiting factor. Recent research in key management has focused wireless and mobile systems, and several schemes have been proposed [11].

Whereas academic researchers seem to assume that traditional schemes are not suited for dynamic and mobile environments due to heavy resource requirements, such factors do not seem to concern planners of military information and communication technology (ICT) usage. Future military ICT systems, national as well as NATO systems, presume a PKI for key management. This is expressed explicitly[1]and implicitly[2].

Academic research includes theoretical and simulation-based performance studies of particular parts of PKIs. To our knowledge, however, no comprehensive quantitative analyses are conducted to investigate whether, or to which extent, traditional PKIs can be used in dynamic, mobile and resource-constrained ICT systems, like future military systems.

The purpose of this report is to prepare an analysis of the resource consumption of a generic X.509-based PKI. The study focuses on bandwidth consumption. A goal is to increase our knowledge about required communication capacities given PKI. We analyze different scenarios. Variables are parameters such as network topologies, traffic matrices, key length, key duration and certificate expiration time.

The PKI model presented in this report is based on the NATO PKI specifications and forms the basis for theoretical/mathematical scalability studies as well as simulation scenarios.

[1] For example through the specifications of NATO PKI

[2] For example through the specification of next generation IPSec in NATO where recommended protocols rely on PKI

1.2 Theoretical analysis

Theoretical scalability analyses will be performed on the basis of models such as scale free networks, described in [8].

1.3 Simulation-based analysis

Descriptions of the simulation model for the PKI scalability study are found in [3], whereas [4] documents the simulator.

1.4 Assumptions and restrictions

Electronic Key Management for NATO is currently subdivided into two separate infrastructures [22] :

- The NATO Electronic Key Management System (NEKMS) based on closed standards and secret data structures
- The emerging NATO Key Management Infrastructure (NPKI) based on open standards and public data structures.

In this context, we also mention Secure Communications Interoperability Protocol (SCIP), which is based on NEKMS, but not primarily a key management protocol. Both NEKMS and SCIP are out of scope for this report. SCIP may however be modeled and investigated later.

We assume that NATO will implement a public key infrastructure based on NATO PKI (NPKI) specifications [20]. The NPKI specifications are based on public/civil protocols specified by the Internet Engineering Task Force (IETF) [14].

This report does not provide a security analysis of PKIs as such. Our goal is to analyze the communications capacity needed to support a PKI based on NPKI/IETF specifications.

1.5 Structure of the document

A high-level description of PKIs defined by the Internet Engineering Task Force (IETF) and NATO is given in chapter 2. Chapter 3 presents the operational requirements of NPKI, which is relevant to our analysis. The chapter summarizes how we handle these requirements in our analysis. Different policies, schemes and protocols regarding use of digital signatures, certificate validation and other foundational functions, may imply different degrees of resource consumption. Chapter 4 discusses, describes, details and estimates important parameters for different aspects of the model. Chapter 5 describes scenarios for theoretical analysis and simulations, whereas chapter 6 describes the relevant traffic models. Chapter 7 summarizes a study of Commercial Off The Shelf (COTS) PKI products. The study was conducted to obtain relevant and realistic input data for theoretical analysis and simulations. Conclusive remarks are found in chapter 8.

2 Public Key Infrastructure

The PKIX working group [24] under IETF [14] has developed standards for general use of Public Key Infrastructures in the global Internet. PKIX defines *Public Key Infrastructure* (PKI) as *The set of hardware, software, people, policies and procedures needed to create, manage, store, distribute, and revoke public key certificates based on public-key cryptography*. The protocols and other specifications are based on the X.509-certificates [13] specified by the International Telecommunication Union-Telecommunication Standardization Sector (ITU-T) [12].

In [20] and [21], NATO recommends the IETF PKIX protocols for the NATO PKI (NPKI). Therefore, our analysis is based on civil IETF specifications and protocols. A description of a general PKI is found in [9].

In this chapter, we give a brief description of PKIX and NPKI.

2.1 PKIX – an architectural overview

2.1.1 Entities

IETF/PKIX describes types of entities that fill the roles of participants within a PKI [7]:

- *Certification authorities* (CAs) are the entities that issue certificates. A CA is the *issuing CA* with respect to the certificates it issues and is the *subject CA* with respect to the CA certificate issued to it. CAs may be organized in a hierarchy in which an organization's CA issues certificates to CAs operated by subordinate organizations, such as a branch, division, or department within a larger organization.
- *Registration authorities* (RAs) are the entities that establish enrollment procedures for end-user certificate applicants, perform identification and authentication of certificate applicants, initiate or pass along revocation requests for certificates, and approve applications for renewal or re-keying certificates *on behalf of a CA*. Subordinate organizations within a larger organization can act as RAs for the CA serving the entire organization, but RAs may also be external to the CA.
- *Subscribers*. Examples of subscribers, who receive certificates from a CA, include employees of an organization having its own CA, banking or brokerage customers, organizations hosting e-commerce sites, organizations participating in a business-to-business exchange, and members of the public receiving certificates from a CA issuing certificates to the public at large.
- *Relying parties*. Examples of relying parties include employees of an organization having its own CA who receive digitally signed e-mails from other employees, persons buying goods and services from e-commerce sites, organizations participating in a business-to-business exchange receiving bids or orders from other participating organizations, and individuals and organizations doing business with subscribers who have received their certificates from a CA

issuing certificates to the public. Within a given PKI, relying parties may or may not be subscribers as well.

- *Other participants*, such as certificate manufacturing authorities, providers of repository services, and other entities providing PKI-related services.

2.1.2 Architecture

Figure 2.1 shows a simplified view of the architectural model assumed by PKIX.

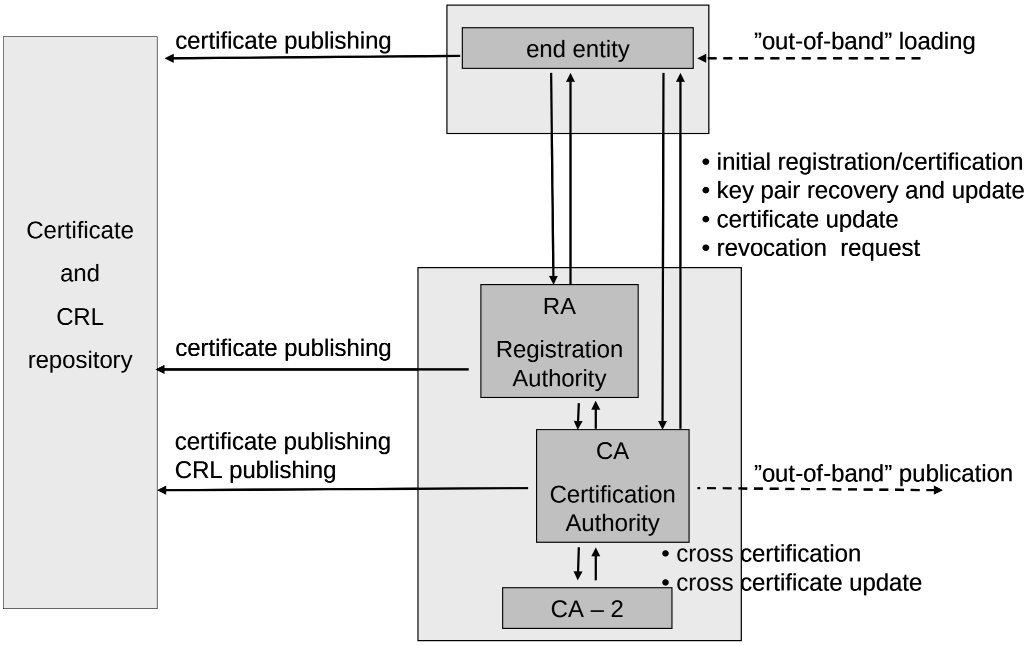


Figure 2.1 PKIX architectural model of PKI entities [2]

2.1.3 Other important terms

PKIX also defines [2]:

- *Certificate Policy* (CP) is a named set of rules that indicates the applicability of a public key certificate to a particular community or class of application with common security requirements. For example, a particular certificate policy might indicate applicability of a type of public key certificate to the authentication of electronic data interchange transactions for the trading of goods within a given price range.
- *Certification Practice Statement* (CPS) is a statement of the practices which a CA employs in issuing public key certificates.
- *Public Key Certificate* (PKC) is a data structure containing the public key of an end-entity and some other information, which is digitally signed with the private key of the CA which issued it.

2.2 NPKI – an organizational overview

Reference [20] describes how NATO adopts the PKIX entities and architectural model:

A CA is an entity authorized by NATO PKI Management Authority (NPMA) to create, sign, issue and manage public key certificates. The requirements described in the referenced policy document, applies to all NPKI CAs unless otherwise stated. Figure 2.2 shows the assumed structure of a NATO PKI. CAs are defined at three tiers. At tier 1, an offline root CA is defined. The root CA is operated by Military Committee Distribution and Accounting Agency (DACAN). Policy CAs are defined at tier 2. The CAs that actually issues certificates to subscribers are defined at tier 3. The tier 2 and tier 3 CAs are subordinate to the NPKI root CA.

A RA is an entity that enters into an agreement with a NPKI CA to implement the registration and authentication processes by collecting and verifying subscribers' identity and information that is to be entered into public key certificates.

A subscriber is the entity whose name appears as the *subject* in a certificate, and who asserts that it uses its key and certificate in accordance with the NPKI policy. Examples of subscribers include NATO military and civilian personnel, personnel from NATO as well as non-NATO nations and ICT products such as workstations, routers, servers, applications and other infrastructure components. Such components shall be under the cognizance of humans, who accept and are responsible for the certificates and associated keys.

A relying party is the entity who trusts the validity of the binding of the subscriber's name to a public key. A relying party may be a subscriber of NPKI or a subscriber of another PKI that has formally approved trust relationship with the NPKI, for example through cross certification. Other participants include NATO PKI Management Authority (NPMA) and PKI Adversary Cell (PAC).

Figure 2.2 shows the organizational structure of NPKI.

2.2.1 NPKI Certificate Policies

Two different certificate policies are specified for NPKI, policy A and policy B. Both policies separate *signing certificates* from *confidentiality certificates*:

- *Signing certificates*[3] are for the management and use of public keys for verification, authentication, non-repudiation and integrity. Policy A signing certificates are for all levels of NATO information.
- *Confidentiality certificates* are for the management and use of public keys for encryption key establishment. Policy A confidentiality certificates are for protection of information classified up to NATO SECRET across a secure network or protection of information classified up to NATO RESTRICTED across an unsecured network, including key transfer.

[3] Signing certificate is called *Identity certificate* in [22]

For both categories, policy B is intended for lower risk environments. NPMA shall on a case by case basis, determine the suitability for of policy B.

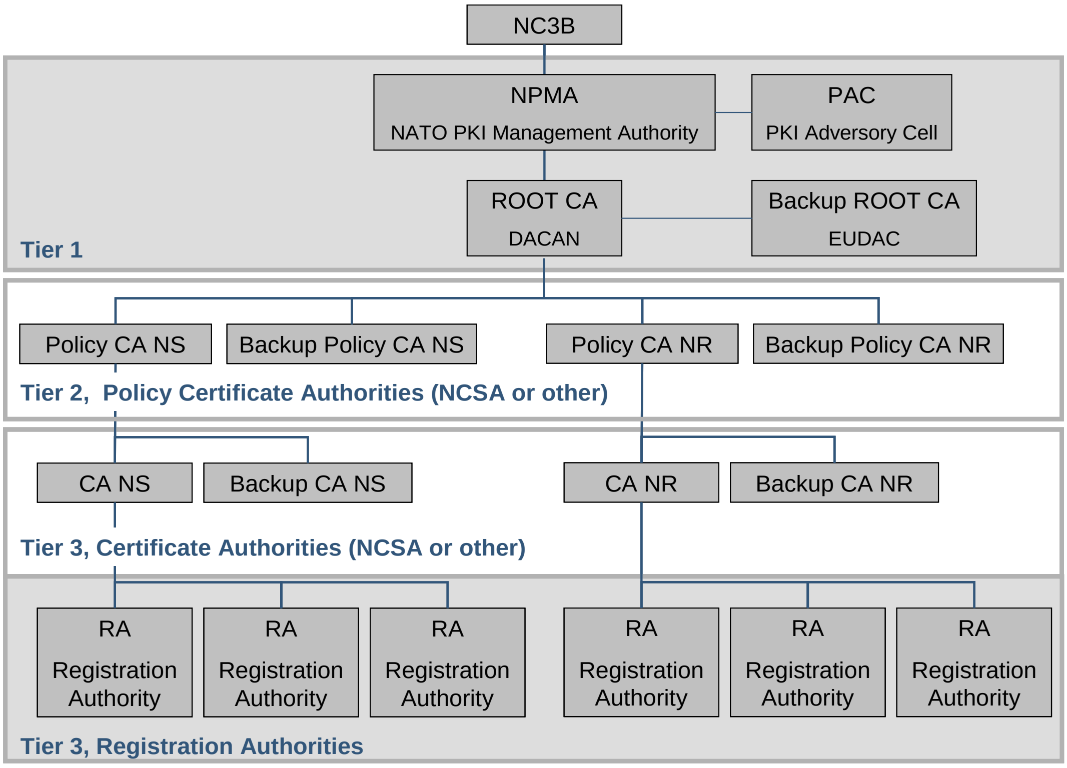

NC3B
NPMA
NATO PKI Management Authority
PAC
PKI Adversory Cell
ROOT CA
DACAN
Backup ROOT CA
EUDAC
Tier 1
Policy CA NS
Backup Policy CA NS
Policy CA NR
Backup Policy CA NR
Tier 2, Policy Certificate Authorities (NCSA or other)
CA NS
Backup CA NS
CA NR
Backup CA NR
Tier 3, Certificate Authorities (NCSA or other)
RA Registration Authority
RA Registration Authority
RA Registration Authority
RA Registration Authority
RA Registration Authority
RA Registration Authority
Tier 3, Registration Authorities

Figure 2.2 Hierarchical NATO PKI structure [20]

2.3 Main functionality

Major functions of a PKI are [2]:

- *Registration*. This is the process whereby a subject first makes itself known to a CA (or a RA). The subject provides its name and other attributes, which the CA verifies.
- *Initialization*. The subject gets the values needed to begin communicating with the PKI, for example the public key or PKC of the CA and the generated private/public key pair of the subject.
- *Certification*. This is the process whereby the CA issues a PKC for a subject's public key and returns the PKC to the subject, or posts it in a repository.
- *Key pair recovery*. A PKI may offer back up of (private) keys such that keys are recoverable in case of loss or access to previously-encrypted information is needed.
- *Key generation*. Depending on the CA's policy, the private-public key pair can be generated by the user or by the CA.
- *Key update*. Keys need to be updated or replaced regularly, for example, when the key has passed its maximum lifetime or the corresponding private key has been compromised.
- *Key expiry*. A PKI provides a facility to gracefully transition from a PKC with an existing key to a new PKC with a new key. This is particularly important when the key to be updated is that of the CA.

- *Key compromise*. This comprises the procedures to handle compromise of user's keys as well as compromise of the CA's keys.
- *Cross-certification*. A cross-certificate is a certificate issued by one CA to another CA. Cross-certification is typically used to make entities in one administrative domain communicate securely with entities in another. Cross-certification may also be issued from one CA to another within the same administrative domain[4]. Cross-certification can be issued in one direction or in both directions.
- *Revocation*. A PKC may need to be revoked prior to the expiration of the validity period. This may be due to for example change of name, change of association between a subject and a CA, compromise or suspected compromise of the corresponding private key. The X.509 recommendations [13] specifies only one facility to handle revocations, namely the Certificate Revocation List (CRL), which identifies revoked PKCs. CRLs are supposed to be distributed throughout the PKI periodically or aperiodically. PKIX does not require CAs to issue CRLs, but recognizes on-line methods of revocation notification to be applicable in some environments.
- *Certificate and revocation notice distribution and publication*. A PKI is responsible for the distribution of PKC and PKC revocation notice. Distribution of PKC includes transmission of the PKC to its owner, and may also include the publication of the PKC in a repository. Distribution of PKC revocation notices may involve posting CRLs in a repository, transmitting the notice to end-entities, or forwarding them to on-line responders.

2.3.1 Communications protocols

Specific protocols facilitate the functionality listed above. PKIX has defined:

- *Management protocols*. These protocols are required to support on-line interactions between PKI users and management entities. A set of functions that need to be supported by management protocols are registration, initialization, certification, key pair recovery, key pair update, cross-certification.
- *Operational protocols*. These protocols are required to deliver certificates and CRLs (or other status information) to certificate users.

The PKIX specifications define a set of standard messages. Note that on-line protocols are not the only way of implementing the functions listed above. There are off-line methods of achieving the same results. An example is hardware tokens that may implement many functions as part of the physical product.

2.4 Interactions

Based on functions and entities presented above, interactions can be outlined as shown in Figure 2.3 and Figure 2.4.

[4] PKIX specifications are ambiguous with regard to the use of the term *cross-certificate* for certificates issued between hierarchical ordered CAs (under the same root CA).

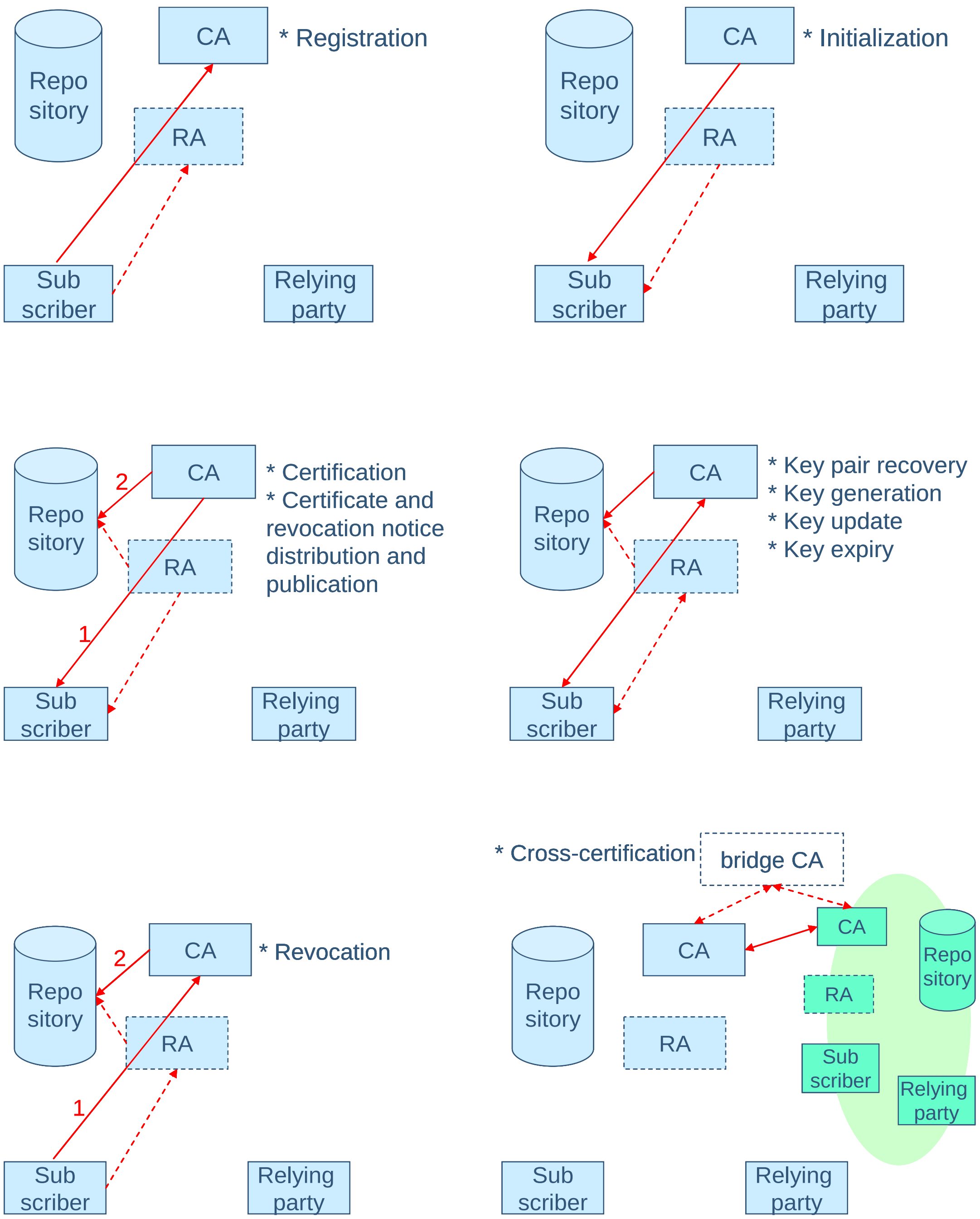


Figure 2.3 Management interactions between PKI entities

(Numbered arrows show a sequence of interactions. Dotted arrows show alternative interactions if RAs or bridge CAs are involved)

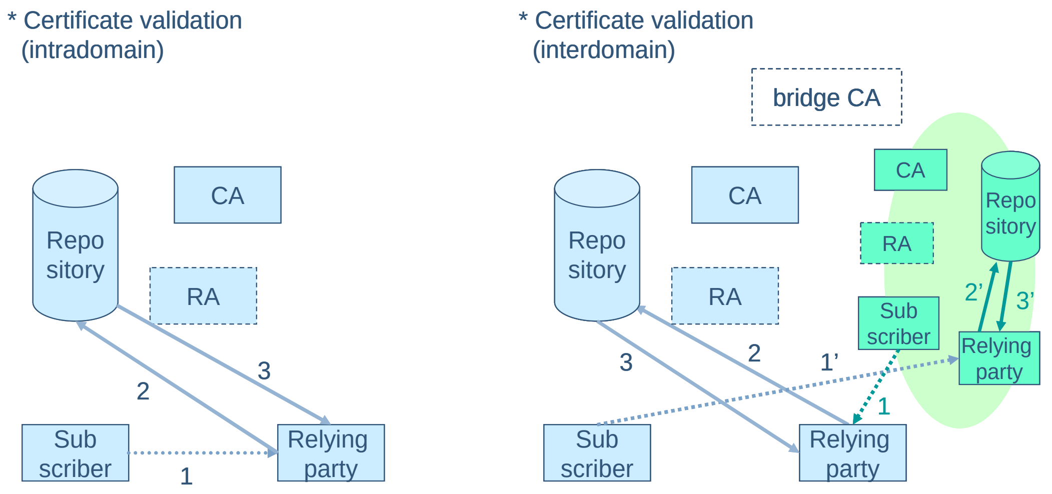


Figure 2.4 Operational interactions between PKI entities

(Numbered arrows show a sequence of interactions. Dotted arrows show alternative interactions if RAs or bridge CAs are involved)

Figure 2.3 shows interactions realized mainly by management protocols, whereas interactions outlined in Figure 2.4 are implemented by operational protocols. Note that the cross-certification shown in Figure 2.3, enables a relying party to request "own" repository to validate PKCs issued by a foreign CA.

For this study, we assume interactions between the entities by the following protocols defined by IETF:

- *Certificate Management Protocol* (CMP) [1], which specify relevant management messages
- *Online Certificate Status Policy* (OCSP) [16], which specify relevant operational messages to determine the status of a PKC without requiring CRL[5]
- *Lightweight Directory Access Protocol* (LDAP) [26] for repository and CRL management and look up.

In chapter 4.3, we present the specific protocol messages to be modeled.

[5] *Server-based Certificate Validation Protocol* (SCVP) [10], is a draft protocol, which allows a client to delegate certification path construction and certification path validation to a server. This protocol is more comprehensive than OCSP. This work, however, is not mentioned in [20].

3 Operational Requirements for NPKI

In this chapter, we present operational requirements for NPKI. These requirements are found in [20] and [21]. *Note that we consider requirements which may have an impact on communication resources, only. Therefore, this chapter does not give an overview of the operational requirements in general.* The last section summarizes how relevant requirements will be handled in our model.

A recent outline of operational requirements is found in [22]. This document refines the requirements from the above-mentioned documents.

3.1 Requirements and proposed deployment

Requirements and proposed deployment are found in [21]. Relevant to our work is the following:

- NPKI shall provide PKI support to individual users in locations and environments ranging from travelling individuals and tactical units to strategic headquarters, commands, agencies, and also in remote locations.
- NPKI shall verify with high assurance the source and integrity of electronic information processed and transmitted by NATO Communication and Information Systems (CIS) within NATO or exchanged with non-NATO nations (NNN) or International organizations (IO) for sensitive, political or military purposes, within one classification level and/or between interconnected CIS operating at different classification levels.

3.2 Functional requirements

Functional requirements are found in [21]. NPKI shall support the following services:

- Identification and authentication of end entities
- Integrity of end entities and transactions
- Encryption
- Non-repudiation of origin.

The jurisdiction of NPKI may include users and electronic entities in NNN/IO. Likewise, NPKI shall support the following interoperability mechanisms:

- Cross-certification of the NATO Root with an external PKI or designated interoperability point like gateway or bridge CAs
- Mutual recognition of the external PKI Root or designated interoperability point and the NATO Root
- Subordinating of national CA with the NATO PKI hierarchy.

3.3 Security requirements

Security requirements are found in [21]. The document mainly states that NPKI shall support requirements defined in the latest approved NPKI Certificate Policy [20], which is supposed to be a “living” document. Nevertheless, reference [21] states the following regarding revocation:

– Revocation status mechanisms shall include CRL and OCSP
– Revocation status distribution mechanisms shall include directories and OCSP, and should also include WEB and File download
– Revocation status distribution mechanisms shall not include FTP.

3.4 Certificate Policy

NPKI certificate policy is defined in [20].

3.4.1 Identities, identifications and authentication

Each subscriber shall have a clearly distinguishable and unique X.509 *Distinguished Name* (DN) in the subject name field and in accordance with [5]. The DN shall be in the form X.501 UTF Printable String and shall not be blank. The name shall be easily understandable for humans. Anonymity is not permitted within the NPKI. The provision of pseudonymity is neither explicitly permitted nor prohibited, but shall not be used in conjunction with non-repudiation.

Prior to the issuance of certificates, the issuing NPKI CA and subscriber shall mutually authenticate each other’s identity. Mechanisms described in CMP [5] are acceptable for proving the possession of a private key. Once every three year, face-to-face identification and authentication of individuals shall be implemented.

A request for re-key may only be made by the subscriber in whose name the keys have been issued. Therefore, all requests for re-key shall be authenticated by the NPKI CA, and the subsequent response shall be authenticated by the subscriber.

A NPKI CA shall authenticate a request for revocation of a certificate. Reference [20] states that appropriate revocation process shall be established and documented, but does not recommend a specific procedure. When the three year face-to-face period has expired, all certificates issued to the subscriber shall be revoked immediately.

3.4.2 Certificate application and application processing

An application for an individual to be a subscriber may be made by the individual or by an individual or body authorized to act on behalf of the prospective subscriber. All information exchanged between the applicant and the NPKI CA (and supporting RAs) shall be authenticated and protected from modification using mechanisms that corresponds with the requirements of the data to be protected by the certificate to be issued. Upon receiving a certificate request, the NPKI CA or RA verify the received information in accordance with [20], and build and sign the PKC.

Reference [20] states that certificates shall be processed in a timely fashion, while ensuring that all required steps are completed. In other words, there are no explicit time requirements.

3.4.3 Certificate issuance

The issuance and publication of a certificate by a NPKI CA indicates a complete approval of the certificate application. Notification is completed with the publication of the certificate within the directory.

3.4.4 Key pair and certificate usage

Whereas NPKI Root CA signs certificates to external CAs and subordinate NPKI CAs,
tier 2 NPKI CAs sign certificates to subordinate CAs. Only tier 3 NPKI CAs sign certificates for NPKI subscribers and shall not issue certificates to subordinate CAs. The certificate path from a subscriber to NPKI Root CA will then consist of at least three certificates.

Relying parties shall only trust certificates when they are being used for their intended purposes. It is the responsibility of the relying party to ensure that they check the most recent CRL information.

3.4.5 Certificate renewal

NPKI does not support certificate renewal.

3.4.6 Certificate re-key

Certificates which have not been revoked, may be re-keyed prior to their expiry. Certificates should be re-keyed in a timely fashion prior to their expiry. Prior to deployment, subscribers shall ensure that remaining certificate life prior to expiry is sufficient for the required mission.

3.4.7 Certificate modification

NPKI may support certificate modification, which occur when changes other than the public key are required.

3.4.8 Certificate revocation and suspension

The policy and procedures for certificate revocation and suspension lists requirements that may be challenging to fulfill in a tactical environment:

- Revocation requests shall be authenticated and authorized. Revocation of a subscriber certificate shall be published in the appropriate CRL.
- In case of known or suspected compromise of a subscriber's token, the subscriber shall notify the CA or a representative of the CA as soon as possible, but in all cases within 12 hours of the known or suspected compromise. This is means that the *revocation request grace period* is 12 hours.
- Any action taken because of a request for revocation of a certificate, shall normally be completed within at most 12 hours from the time of notification.

- When a certificate revocation is determined, the revocation shall be completed within 1 hour.[6]
- CRLs are periodically issued and posted to a repository, even if there are no changes or updates to be made. NPKI Root CA CRLs shall be published bi-weekly. NPKI at tier 2 and tier 3 shall publish their CRLs every 12 hours.
- Full propagation of a new CRL across the entire alliance must be completed before the expiry of the previous one. To allow for the latency of the network environment, NPKI CA's shall set the next update value within CRLs to a value which allows for the propagation of the CRL prior to the expiry of the previous one. A NPKI CA shall also ensure that its CRL issuance is synchronized with any directory synchronization to ensure the accessibility of the most recent CRL to relying parties, prior to the expiry of the previous CRL.
- When a certificate is revoked due to key compromise, the updated CRL shall be issued immediately within the 1 hour limit mentioned above.
- Online revocation/status checking is implemented by OCSP. An OCSP responder shall meet the same security and availability requirements as the certificate repository.
- Subscribers may choose between the use of any available revocation checking mechanism including WEB, file share, OCSP and CRLs. The requirements for validating certificate paths are the same regardless of which mechanism is chosen.
- CAs may use file shares or publish CRLs on a WEB server as additional mechanisms
- Revocation services shall be available 24 hours a day and 7 days a week continuous.

3.4.9 Certificate status services

Reference [20] states: "*For OCSP, certificate status services shall be capable of verifying the validity of certificates in an automated and transparent fashion*"[7]. The services shall be implemented such that high availability delivery of certificate status information is provided. This will require redundancy of implementation, including geographic and network diversity.

3.5 The impact of operational requirements on our NPKI model

The proposed deployment of NPKI in tactical environments justifies our analysis since wireless communication networks at tactical level have low and time varying communication capacity compared with fixed networks. It also justifies the use of the simulator described in [4].

3.5.1 Entities

- *NPKI CAs.* The management of multi-domain environments is a major functional requirement. Therefore, our model assumes more than one CA hierarchy. This means that we handle traffic to and from CAs outside our simulated PKI, but do not necessarily simulate several CAs directly. Traffic models are described in chapter 6.

[6] We presume that complete revocation means that a new CRL is fully propagated and received by all relevant CRL repositories.

[7] As far as we can see, OCSP can not *verify* the validity of certificates. See subsection 4.3.2.

- *NPKI RAs*. For simplicity, we omit NPKI RAs from the model. Since practically all CA functions may be delegated to a RA, there is no difference between a CA and a RA with regard to communication resource consumption. In real implementations, CAs and RAs may or may not be collocated.
- *NPKI subscribers*. To fulfill the functional requirements, we model NPKI subscribers as owners of one signing certificate and one confidentiality certificate. Also, a subscriber takes the role of a relying party with regard to certificates owned by other subscribers.
- *Relying parties*. In our model, relying parties are NPKI subscribers. Since every node in the simulated network is a NPKI subscriber, we omit relying party as a specific entity.

Therefore, we model two entities: *NPKI CA* and *NPKI subscriber*. We describe these entities in subsection 4.3.2.1.

3.5.2 Management functionality

PKIX management functions as described in subsection 2.3 and NPKI certificate policy as described in subsection 3.4, indicate that traffic imposed by management functions, is minor compared to traffic imposed by operational functions. At the other hand, there are critical time constraints related to certain types of management functions. Therefore, we select a few functions for our model, whereas other functions are supposed to be either preconfigured or negligible concerning the simulation results.

In a real NPKI, dynamical registration, initialization and certification may be required. Traffic imposed by these processes is supposed to have minor impact on communication resources. Also, it is reasonable to assume that all certificates are preconfigured[8]. Hence, we omit these functions from our model. Further, we assume that subscribers are preconfigured with the public keys of all CAs involved.

In a real NPKI, dynamical cross-certification may also be required. Traffic related to this process is supposed to have minor impact on communication resources and can be omitted. This means that we assume cross-certification between CAs to be preconfigured.[8]

For the same reasons we also omit traffic related to key pair recovery, key generation, key/certificate update and key/certificate expiry.

This means that the only management function to be modeled is related to key compromise and certificate/key revocation. These functions may initiate major and critical operational functions like the issuing and delivery of CRLs.

We describe the relevant management messages in subsection 4.3.

[8] Even if valid certificates are preconfigured in a real NPKI, the NPKI should probably be able to cope with dynamic certification, due to for example robustness. Further, dynamic functions should probably be subject to time constraints in case the operation of critical applications depends on valid certificates. Such questions, however, are out of scope for this analysis.

3.5.3 Operational functionality

PKIX operational functions as described in subsection 2.3 and NPKI certificate policy as described in subsection 3.4, indicate that traffic related to operational functionality imposes the major part of the NPKI traffic. There are also critical time constraints related to these functions.

Relevant operational functions are related to:

- NPKI CA's announcement and delivery of CRLs, both periodical and as response to key compromise and certificate/key revocation, see subsection 3.5.2
- NPKI subscribers' requests for certificate validation.

We discuss and describe the frequency of CRL announcements as well as validation requests in subsection 4.1. We describe relevant operational messages in subsection 4.3.

3.5.4 Summary

Figure 3.1A shows the entities and functions of the model, whereas Figure 3.1B shows a possible configuration for a multi-domain NPKI. Stippled entities are supposed to be outside the model, but we will estimate and handle traffic to and from these entities.
We describe different traffic models in chapter 6.

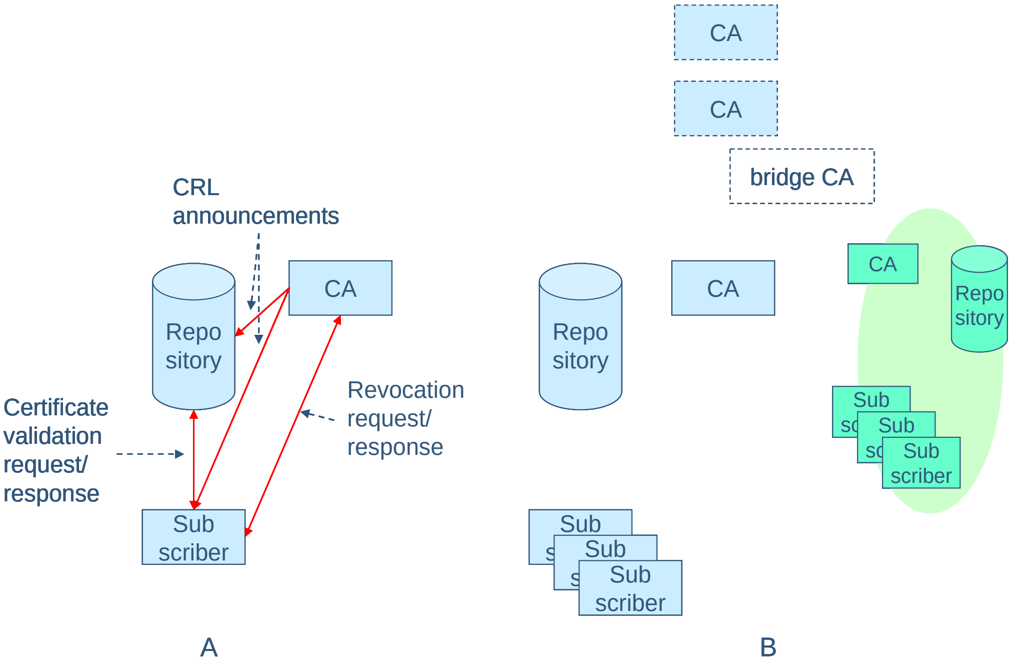


Figure 3.1 A) Entities and functions of the model

B) A possible PKI architecture for the model

4 The NPKI model

Different policies and schemes regarding certificate validation, cross-certification and the use of digital signatures may imply different resource consumption. In this chapter, we discuss, describe, detail and estimate important parameters for different aspects of the model.

4.1 Some policy considerations

We assume certificate policy A, as described in subsection 2.2.1.

4.1.1 Some definitions

Signature verification is the process where the binding between a signed object and the signature is verified. A successfully verified signature does *not* testify authenticity of the signature, only that the integrity of the message has been preserved after the signature has been applied.

Certificate status is an administratively declared property which decides if this certificate may be used, i.e. if the key pair associated with the certificate can be used for signing or encryption purposes. A certificate is annotated with a validity period, outside which the status is ”invalid”, but a certificate may also from administrative reasons be declared as invalid before the expiration time.

Certificate validation is the process to decide the status of a certificate, either by requesting the status from a status provider or through inspection of certificate revocation lists (CRLs)

4.1.2 Key generation and certificates

If using NPMA-approved algorithms, a prospective subscriber may generate its digital signature key pair as well as its confidentiality key management pair [20]. We assume, however, that a CA generates all key pairs.

PKCs shall be individually accountable [20]. For simplicity, we model only two certificates per subject name, one for signing and one for confidentiality. This means that each subscriber entity gets one DN and two certificates.

For the simulations, we assume:

– the repository (and directories) to be preconfigured with the certificates of all participating subscribers and CAs,

– all subscribers to be preconfigured with the public key of its own CA.

4.1.3 Validation schemes

The relevant validation schemes proposed for NPKI may be modeled as two extreme cases:

- *Full online certificate validation*[9]. CRLs are stored and maintained in one central repository available for subscribers' requests (pull-based).
- *Full CRL distribution.* CRLs are distributed to each subscriber (push-based).

A practical NPKI would probably not implement these variants, but something in between[10]. For analysis, however, it is meaningful to investigate characteristics of these extreme variants before we investigate solutions that may be more optimal. Such solutions may have different degrees of distributed repositories (directories) combined with on-line validation/status request, or a pull-based CRL distribution[11]. Both push-based and pull-based CRL schemes may utilize delta CRLs. Reference [8] describes various distribution schemes.

Our goal is not to optimize NPKI, but to provide knowledge about the resource consumptions imposed by some *main* strategies regarding architecture and policy. Therefore, we model at least three validation schemes: The two extreme cases and at least one combination scheme. We describe our traffic models and the relevant simulation parameters in chapter 6.

4.1.4 Use of certificates

Signing certificates. We model user-imposed traffic as application layer messages. We assume that every message shall be protected with regard to authenticity and integrity. We do not simulate *communication sessions* between two communication parties, see section 6.

When sending messages this means that a subscriber:
- signs each message with its a digital signature[12], and
- appends its signature, its signing certificate and certification path to each signed message.

When receiving messages, the subscriber:
- validates the certificate and certification path for each message. This strict policy will be the first option. Later we may utilize a more liberal policy, as described in [8].

Confidentiality certificates. In the first phase of our analysis, we do not model the use of confidentiality certificates. Further work may involve Security Architecture for the Internet Protocol (IPSec) and in particular Internet Key Exchange version 2 (IKEv2), which facilitates dynamic encryption key negotiations. Then, confidentiality certificates are more relevant.

[9] Since we utilize OCSP, the term *online certificate status* might be more correct, see also foot note 7.

[10] After conducting the experiment described in chapter 7, it seems, however, that COTS PKI products are designed for the first extreme case. Alternative configurations seem to require considerable effort.

[11] We are aware of research in distributed CA functionality. To our knowledge, this is a strategy which is not considered in PKIX and NPKI documents/specifications, and therefore out of scope for this analysis.

[12] Chapter 6 describes how the amount of offered traffic that should be signed, varies through the simulations.

4.2 Signatures, certificates and certificate revocation lists

4.2.1 The X.509 version 3 certificate

The NKPI CA shall issue X.509 version 3 certificates [19] in accordance with [5]. Subscriber hardware and software shall support the base certificate fields shown in Table 4.1. Certificate extensions specified in [5] may be supported.

Field	*Comment*
Version	Version of X.509 certificate, version 3 (2)
Serial Number	Unique serial number for certificate
Signature	NPKI CA signature to authenticate certificate
Issuer	Name of NPKI CA
Validity	Activation and expiry date for certificate
Subject	Subscriber's distinguished name
Subject Public Key Information	Algorithm ID, key

Table 4.1 Base X.509 certificate fields [5]

Several extensions are defined in [5], and a profile for NPKI CA certificates is specified in [20]. Table 4.2 shows the base certificate profile for CA certificates. This profile forms the basis for the CA certificates assumed in our model. Further, we assume that subscriber certificates embrace the same extensions as CA certificates at tier 2 and tier 3.

Extension	*Value*
NPKI Root CA (Tier 1)	
Authority Key Identifier	20 byte SHA-1 hash of the binary Distinguished Encoding Rules (DER) encoding of the Root CA's public key information
Subject Key Identifier	20 byte SHA-1 hash of the binary DER encoding of the Root CA's public key information
Basic Constraints **(Critical)**	Subject Type=CA Path Length Constraint=2
Key Usage **(Critical)**	Certificate Signing, Off-line CRL Signing, CRL Signing
Private Key Usage Period	See table 2 in [19]

Policy CA (Tier 2)

Authority Key Identifier	20 byte SHA-1 hash of the binary DER encoding of the Root CA’s public key information
Subject Key Identifier	20 byte SHA-1 hash of the binary DER encoding of the Root CA’s public key information
Basic Constraints **(Critical)**	Subject Type=CA Path Length Constraint=1
Key Usage **(Critical)**	Certificate Signing, Off-line CRL Signing, CRL Signing
Private Key Usage Period	See table 2 in [19]
Certificate Policies **(Critical)**	[1] Certificate Policy: Policy Identifier=1.3.26.1.9.1 [1,1] Policy Qualifier Info: Policy Qualifier Id=CPS Qualifier: http://www.infosec.nato.int/NPKI/CertP.pdf [1,2] Policy Qualifier Info: Policy Qualifier Id=User Notice Qualifier: Notice Reference: Organisation=NATO Notice Number=1 Notice Text=Limited Liability. See CertP-Responsabilite limitee. Voir Cert. [2] Certificate Policy: Policy Identifier=1.3.26.1.9.2 [2,1] Policy Qualifier Info: Policy Qualifier Id=CPS Qualifier: http://www.infosec.nato.int/NPKI/CertP.pdf [2,2] Policy Qualifier Info: Policy Qualifier Id=User Notice Qualifier: Notice Reference: Organisation=NATO Notice Number=1 Notice Text=Limited Liability. See CertP-Responsabilite limitee. Voir CertP.
CRL Distribution Point (CRLDP)	[1] CRL Distribution Point Distribution Point Name: Full Name: URL=ldap://nitcdsa.ncsa.nato.int/cn=NPKIRootCA,o=NATO?certificateRevocationList?base

CA (Tier 3)

Authority Key Identifier	20 byte SHA-1 hash of the binary DER encoding of the Root CA's public key information
Subject Key Identifier	20 byte SHA-1 hash of the binary DER encoding of the Root CA's public key information
Basic Constraints **(Critical)**	Subject Type=CA Path Length Constraint=0
Key Usage **(Critical)**	Certificate Signing, Off-line CRL Signing, CRL Signing
Private Key Usage Period	See table 2 in [19]
Certificate Policies **(Critical)**	See Policy CA (Tier 2)
CRL Distribution Point	See Policy CA (Tier 2)

Table 4.2 Base Certificate Profile for CA Certificates [20]

4.2.1.1 Size of certificate and signing structure

The choice of cryptographic algorithms has an impact on key length and consequently the size of digital signature and the size of the certificate. Each algorithm offers different parameter sizes. There are several algorithms approved for NPKI.

We base the size estimation on the study described in chapter 7, where we implemented Entrust, the COTS PKI application utilized by the Norwegian Defence.

For further analysis and simulations, we utilize a certificate size of *1200 bytes*. We use this size for all certificates types. Further details are found in section 7.2 and 7.3.

Our Entrust study shows that the size of a *signing structure* varies from one COTS product to another. The structure includes a signature and a certificate. Entrust ESP (Entelligence Service Provider) generates S/MIME-formatted signature objects[13]. Provided SHA-2 for hashing and RSA-2048 for signing, the size of the S/MIME signature object is *820 bytes*. Further details are found in section 7.4.

For further analysis and simulations, we utilize a single signature structure of *2000 bytes.* We assume a certificate chain of 4 certificates appended to each signed message: subscriber certificate together with the certificates of tier 1, tier 2 and tier 3 (root) CAs. Thus, signing a message means adding (4*1200 + 820) = *5620 bytes* to the message.

4.2.2 The X. 501 version 2 Certificate Revocation List

The NPKI CA shall issue X.509 version 2 CRLs [20] in accordance with [5]. CAs shall issue CRLs according to specified periods even though no certificates are revoked since the previous issuance. Table 4.3 shows the base fields of the X.509 CRL.

[13] S/MIME is based on the PKC#7 standard.

Field	*Comment*
Version	Version of X.509 certificate, version 3 (2)
Signature	Algorithm identifier
Issuer	Name of NPKI CA
This Update	Time
Next Update	Time
Revoked Certificates	A list of revoked certificates. The subsequent fields are per revoked certificate.
User Certificate	Unique serial number for certificate
Revocation Date	Time

Table 4.3 Base X.509 CRL fields [5]

Several CRL extensions are specified in [5]. There are extensions to the CRL as such, as well as extensions to the CRL entries (revoked certificates). To our knowledge, CRL profiles are not defined for NPKI. Table 4.4 shows available PKIX extensions to the CRL, whereas Table 4.5 shows the available extensions to each CRL entry.

Field
Authority Key Identifier
Issuer Alternative Name
Issuer CRL Number
Delta CRL Indicator
Issuing Distribution Point
Freshest CRL
Authority Information Access

Table 4.4 Available PKIX extensions to the CRL [5]

Field	*Comment*
Reason Code	The reason why this certificate is revoked
Invalidity Date	The date on which it was known that the private key was compromised or tthat the certificate otherwise became invalid.
Certificate Issuer	Relates to indirect CRLs and Issuing Distribution Point

Table 4.5 Available PKIX extensions to the CRL entries [5]

4.2.2.1 Size of CRL

Again, we base the size estimation on results from our Entrust study.

For further analysis and simulations, we utilize a CRL size of *700 bytes + 36 bytes per entry*. Further details are found in section 7.5 and 7.8.

4.3 Protocols and messages

4.3.1 Management

As stated in subsection 2.4, we assume the use of *Certificate Management Protocol* (CMP) [1]. This protocol is however not utilized in the COTS PKI products we have investigated. A vendor-specific protocol is used in conjunction with *Lightweight Directory Access Protocol* (LDAP). However, we will model the management functionality identified in subsection 3.5.2 with reference to the relevant CMP messages. We then identify corresponding traffic from our Entrust implementation to estimate "the size of the functionality" represented by selected CMP messages.

CMP is recommended by IETF and also for use in NPKI, for example in combination with LDAP or similar. Therefore, we assume this protocol to be the future choice. The general format of the CMP messages is given in Table 4.6. Table 4.7 shows the syntax of the common CMP message header, whereas available CMP message bodies are listed in Table 4.8. Table 4.9 shows the CMP message protection.

PKIMessage ::=	SEQUENCE {			
	header		PKIHeader	
	body		PKIBody	
	protection	0	PKIProtection	OPTIONAL
	extraCerts	1	SEQUENCE SIZE (1..MAX) OF CMPCertificate	OPTIONAL
}				
PKIMessages ::=	SEQUENCE SIZE (1..MAX) OF PKIMessage			

Table 4.6 General format of CMP messages

PKIHeader ::=	SEQUENCE {			
	pvno		INTEGER {cmp1999 (1), cmp2000 (2) }	
	sender		GeneralName	
	recipient		GeneralName	OPTIONAL
	messageTime	0	GeneralizedTime	OPTIONAL
	protectionAlg	1	AlgorithmIdentifier	OPTIONAL
	senderKID	2	KeyIdentifier	OPTIONAL
	recipKID	3	KeyIdentifier	OPTIONAL
	transactionID	4	OCTET STRING	OPTIONAL
	senderNonce	5	OCTET STRING	OPTIONAL
	recipNonce	6	OCTET STRING	OPTIONAL
	freeText	7	PKI freeText	OPTIONAL
	generalInfo	8	SEQUENCE SIZE (1..MAX) OF InfoTypeAndValue	OPTIONAL
}				
PKI free Text ::=	SEQUENCE SIZE (1..MAX) OF UTF8String			

Table 4.7 CMP message header

```
PKIBody ::=    CHOICE {
                  ir          0   CertReqMessages          Initialization Request
                  ip          1   CertRepMessage           Initialization Response
                  cr          2   CertReqMessages          Certification Request
                  cp          3   CertRepMessage           Certification Response
                  p10cr       4   CertificationRequest     PKCS # 10 Cert. Request
                  popdecc     5   POPODecKeyChallContent   Proof-of-possession Challenge
                  popdecr     6   POPODecKeyRespContent    Proof-of-possession Response
                  kur         7   CertReqMessages          Key Update Request
                  kup         8   CertRepMessage           Key Update Response
                  krr         9   CertReqMessages          Key Recovery Request
                  krp        10   KeyRecRepContent         Key Recovery Response
                  rr         11   RevReqContent            Revocation Request
                  rp         12   RevRepContent            Revocation Response
                  ccr        13   CertReqMessages          Cross-Certification Request
                  ccp        14   CertRepMessage           Cross-Certification Response
                  ckuann     15   CAKeyUpdAnnContent       CA Key Update Announcement
                  cann       16   CertAnnContent           Certificate Announcement
                  rann       17   RevAnnContent            Revocation Announcement
                  crlann     18   CRLAnnContent            CRL Announcement
                  pkiconf    19   PKIConfirmContent        PKI Confirmation Content
                  nested     20   NestedMessageContent
                  genm       21   GenMsgContent            PKI General Message Content
                  genp       22   GenRepContent            PKI General Message Response
                  error      23   ErrorMsgContent          Error Message Content
                  certConf   24   CertConfirmContent       Certificate Confirmation Content
                  pollReq    25   PollRecContent           Polling Request
                  PollRep    26   PollRepContent           Polling Response
}
```

Table 4.8 Available CMP message bodies

```
PKIProtection ::=    BIT STRING
ProtectedPart        SEQUENCE {
                        header          PKIHeader
                        body            PKIBody
}
```

Table 4.9 CMP message protection

To support the management functionality defined in subsection 3.5.2, we only need a small subset of the specified protocol messages in Table 4.8. We assume the following three messages:

- Number 11, rr, *RevReqContent* for revocation request
- Number 12, rp, *RevRepContent* for revocation response
- Number 18, crlann, *CRLAnnContent* for CRL Announcement.

Table 4.10 through Table 4.12 show the syntax of each message. Complete specifications and common data structures are found in [1].

```
RevReqContent ::=     SEQUENCE OF RevDetails
RevDetails ::=        SEQUENCE {
                          certDetails         certTemplate
                          crlEntryDetails     Extensions                   OPTIONAL
}
```

Table 4.10   Revocation Request (RevReqContent) message

```
RevRepContent ::=     SEQUENCE {
                          status              SEQUENCE SIZE (1..MAX) OF
                                              PKIStatusInfo
                          revCert             SEQUENCE SIZE (1..MAX) OF    OPTIONAL
                                              CertID
                          crls                SEQUENCE SIZE (1..MAX) OF    OPTIONAL
                                              CertificateList
}
```

Table 4.11   Revocation Response (RevRepContent) message

```
CRLAnnContent ::=     SEQUENCE OF CertificateList
```

Table 4.12   CRL Announcement (CRLAnnContent) message

For completeness, also *Certificate Request Message Format* (CRMF) [25] should be mentioned. This format relates to the process of creating a certificate. Since these processes are omitted from our analysis, CRMF is not as relevant as it might be in a real PKI implementation.

4.3.1.1 Management message sizes

Table 4.13 shows the message sizes derived from the Entrust study. Further details are found in section 7.8. As stated in section 4.3.1, these numbers do not stem from the listed CMP messages. The numbers are estimations of the functionality of these messages.

Message				*Estimated message size*
rr	11	RevReqContent	Revocation Request	6000 bytes
rp	12	RevRepContent	Revocation Response	6000 bytes
crlann	18	CRLAnnContent	CRL Announcement	700 bytes + 36 bytes per entry, see section 4.2.2.1

Table 4.13   Estimated size of assumed CMP messages

4.3.2 Operation

The *Online Certificate Status Protocol* (OCSP) enables applications to determine the (revocation) state of an identified certificate. The NPKI uses OCSP version 1. The protocol may be used to satisfy some of the operational requirements for providing more timely revocation information than possible with CRLs. OCSP may also be used to obtain additional status information. An OCSP client issues a status request to an OCSP responder and suspends acceptance of the certificate in question until the responder has validated [16].

The *Server-based Certificate Validation Protocol* (SCVP) is an alternative to OCSP. To our knowledge, however, this protocol is not mentioned in any NPKI documents. On the other hand, the protocol is still an internet draft. The primary goals of SCVP are to make it easier to deploy PKI-enabled applications by delegating path discovery and/or validation processing to a server, and to allow central administration of validation policies within an organization. SCVP can be used by clients that do much of the certificate processing themselves but simply want an non-trusted server to collect information for them. However, when the client has complete trust in the SCVP server, SCVP can be used to delegate the work of certification path construction and validation, and SCVP can be used to ensure that policies are consistently enforced throughout an organization [10]. In this context, the general protocol requirements for delegated path validation and delegated path discovery within PKIX should be mentioned [23]. Also, there is an IKEv2 extension to OCSP [17]. These extensions may be relevant in our future work, mentioned in subsection 4.1.4.

OCSP is a simple protocol and specifies three messages:

- *Request*. The message contains the following data:
  - *Protocol version*
  - *Service request*
    - *Target certificate.* List of certificates to be checked. The certificates are listed with their hash algorithm identifier, a hash of the issuer's name, a hash of the issuer's public key and the certificates serial number.
    - *Optional extensions*

  If the requestor digitally signs the Request Message, the message also contains the requestor's digital signature and its (list of) certificate.

  The actual formatting of the message could vary depending on the transport mechanism used. These include Hyper-Text Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP) and LDAP.
- *Response*. The basic response message contains the following data
  - *Version of the response syntax*
  - *Name of the responder*
  - *Responses for each of the certificates in a request.* The response for each of the certificates consists of a target certificate identifier and a certificate status value. Three definitive response indicators are defined: good, revoked, unknown.

    The "good" state indicates a positive response to the status inquiry. At a minimum, this response indicates the certificate is not revoked, but does not necessarily mean the certificate was ever issued, or that the time at which the response was produced is within the certificate's validity interval. The "revoked" state indicates the certificate has been revoked (either permanently or temporarily (on hold)). The "unknown" state indicates the responder doesn't know about the requested certificate.

Each single certificate response also contains a response validity interval and optional extensions. Response extensions may be used to convey additional information on assertions made by the responder regarding the status of the certificate such as positive statement about issuance, validity, etc

- *Optional extensions*
- *Signature algorithm Object IDentifier* (OID)
- *Signature computed across hash of the response.* All definitive Response messages shall be digitally signed. The key used to sign the Response message must belong to one of the following:
  - the CA who issued the certificate in question
  - a Trusted Responder whose public key is trusted by the requester
  - a CA Designated Responder (Authorized Responder) who holds a specially marked certificate issued directly by the CA, indicating the responder may issue OCSP responses for that CA.

This means the message also contains the responder's certificate and its (list of) certificate. Also in case of Response messages, the actual formatting could vary depending on the transport mechanism used.

- *Error.* A simple Error message is defined.

We model the request and response messages. Table 4.14 and Table 4.15 show the message syntax. Detailed specifications and data structures are found in [16].

OCSPRequest ::=	SEQUENCE {			
	tbsReqeust		TBSRequest	
	optionalSignature	0	EXPLICIT Signature	OPTIONAL
}				
TBSRequest ::=	SEQUENCE {			
	version	0	EXPLICIT Version	DEFAULT v1
	requestorName	1	EXPLICIT GeneralName	OPTIONAL
	requestList		SEQUENCE OF Request	
	requestExtensions	2	EXPLICIT Extensions	OPTIONAL
}				
Signature ::=	SEQUENCE {			
	signatureAlgorithm		AlgorithmIdentifier	
	signature		BIT STRING	
	certs	0	EXPLICIT SEQUENCE OF Certificate	
}				
Version ::=	INTEGER {			
	v1	0		
}				
Request ::=	SEQUENCE {			
	reqCert		CertID	
	singleRequstExtensions	0	EXPLICIT Extensions	OPTIONAL
}				
CertID ::=	SEQUENCE {			
	hashAlgorithm		AlgorithmIdentifier	
	issuerNameHash		OCTET STRING	
	issuerKeyHash		OCTET STRING	
	serialNumber		CertificateSerialNumber	
}				

Table 4.14 OCSP Request message

```
OCSPResponse ::=   SEQUENCE {
                      responseStatus           OCSPResponseStatus
                      responseBytes         0  EXPLICIT Response Bytes            OPTIONAL
}
OCSPResponseSta
tus ::=            ENUMERATED {
                      successful            0
                      malformedRequest      1
                      internalError         2
                      tryLater              3
                                            4  (not used)
                      sigRequired           5
                      unauthorized          6
}
```

Table 4.15 OCSP Response message

4.3.2.1 Operational message sizes

Table 4.16 shows the sizes derived from the Entrust study. Further details ar found in section 7.5.

Message	*Estimated message size*
OCSP Request	1400 bytes
OCSP Response	1400 bytes

Table 4.16 Estimated size of OCSP messages

4.4 NPKI Entities

For our analysis, it is sufficient to describe the NPKI entities as senders and receivers of messages. Further, we assume:

- Each node in the simulated tactical network contains a NPKI entity, a NPKI CA or a NPKI subscriber, respectively.
- Each NPKI entity has an asymmetric key pair, a certificate signed by a NPKI CA and the certificate chain up to root CA.
- Nodes containing a NPKI CA entity are assumed to control a certificate repository
- Nodes containing a NPKI CA entity are assumed to control a certificate status service
- The vast majority of nodes contain a NPKI Subscriber entity. These nodes also contain a general message entity. Hence, the majority of nodes send and receive two types of messages at application layer: PKI messages as described in subsection 4.3 and general messages.
- General messages reflect scenario-dependant information exchange. They are modeled as dummy messages. Size and frequency are described in chapter 6.
- A signature and a certificate (or a certificate chain) is attached to each general message to be signed. The amount of general messages to be signed, will vary.

4.4.1 NPKI CAs

A network node containing a NPKI CA entity is shown in Figure 4.1. The figure also lists the relevant messages.

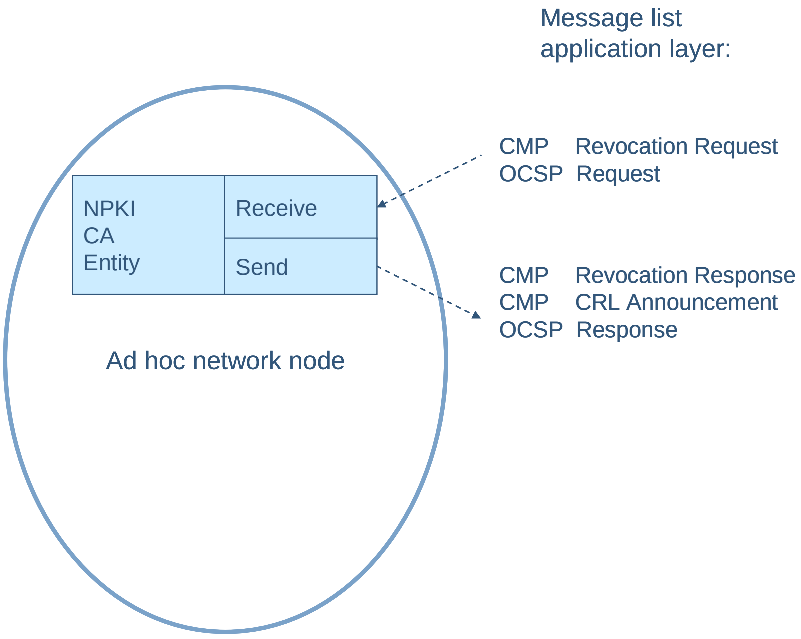


Figure 4.1 A network node containing a NPKI CA

4.4.2 NPKI Subscribers

A network node containing a NPKI subscriber entity and a general message entity is shown in Figure 4.2. The figure also lists the relevant messages.

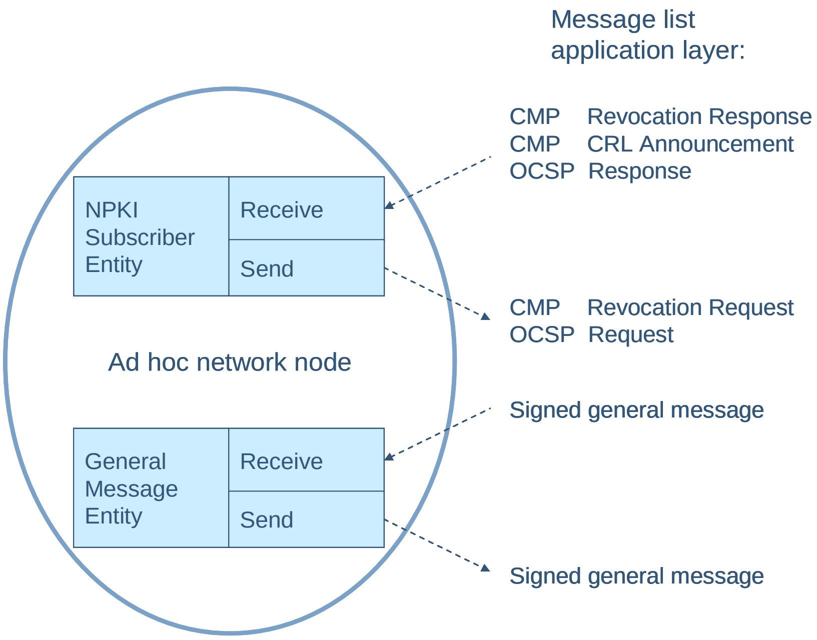


Figure 4.2 A network node containing a NPKI subscriber entity

5 Scenarios for analysis and simulations

This chapter outlines the scenarios specified for the forthcoming studies. Goals, assumptions and characteristics of each scenario are described.

5.1 Scenarios

In this subsection, we define two scenarios to be analyzed, whereas traffic models are specified in chapter 6.

Networks. We simulate 16 nodes per network (per wireless cell). All nodes share the same radio coverage area. Radio conditions are assumed to be perfect with signal levels fare above the RF background noise. Within the strategic (or deployed) network, we assume lossless infinite transmission capacity, see Figure 5.1and Figure 5.2. We assume that wired nodes are connected to the strategic network. See reference [4] for technical specification of the simulator.

Scenarios. We distinguish between scenarios by the number of ad hoc networks involved. The first scenario has one ad hoc network, whereas the second one encompasses three different networks simultaneously.

Variants and NPKI domains. For each scenario, we define three variants, distinguished by the number of NPKI domains involved. The first variant has no NPKI and is just for reference. The second one has *one single* NPKI domain and the third one encompasses *three different* NPKI domains at the same hierarchical level. In the multi domain scenario variant, we assume a root CA (trust anchor) at a hierarchical level above. This CA is, however, not an entity in our model.

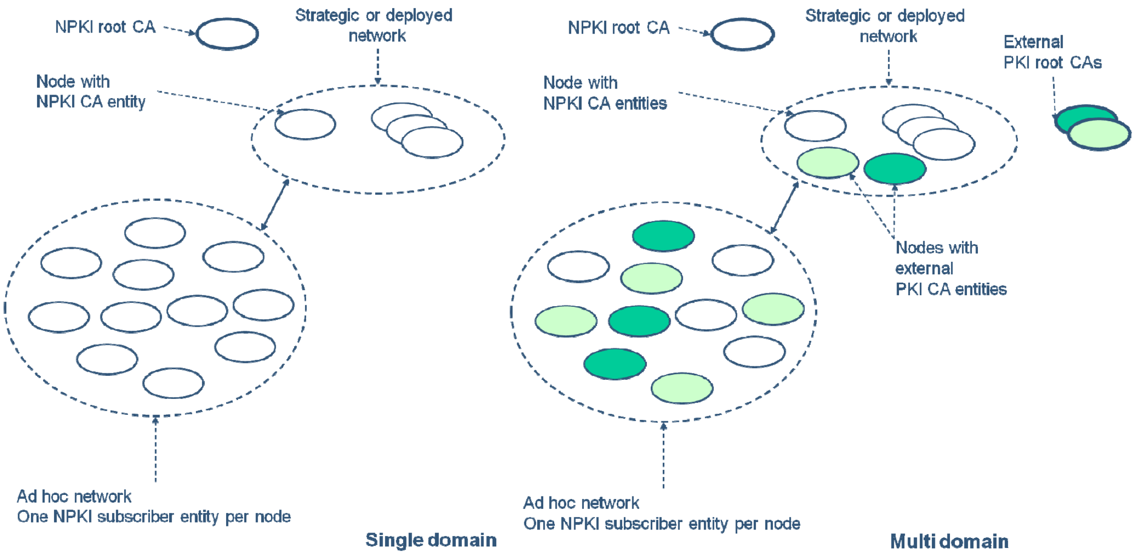
NPKI CAs. We assume centralized CAs physically situated outside the ad hoc network and one CA per NPKI domain. We assume that each CA controls and maintains one repository and one status service physically situated outside the ad hoc network(s).

Validation schemes. For both scenarios, we define two alternative policies, distinguished by the validation schemes described in subsection 4.1.3: one push-based and one pull-based scheme.

- Under push scheme in the single domain scenario variant, the CA distributes CRLs to each single subscriber regularly, and subscribers are supposed to validate all certificates locally. In the multi domain variant, a CA also distributes CRLs to all other CAs. Other CAs are supposed to forward these “foreign” CRLs to their intra domain subscribers
- Under pull scheme in the single domain variant, the CA is assumed to maintain its repository and OCSP status server dynamically, and subscribers are supposed to check all certificates online. In the multi domain variant, a CA distributes CRLs to all other CAs. Other CAs are supposed to update their status servers accordingly. Consequently, we assume it is sufficient for a subscriber to look up the intra domain OCSP status server even when “foreign” certificates are involved.

5.2 Scenario 1 – one network

Figure 5.1 shows scenario 1 with a single NPKI domain and with multiple domains. Characteristics are listed in Table 5.1.



NPKI root CA
Strategic or deployed network
Node with NPKI CA entity
Ad hoc network
One NPKI subscriber entity per node
Single domain
NPKI root CA
Strategic or deployed network
External PKI root CAs
Node with NPKI CA entities
Nodes with external PKI CA entities
Ad hoc network
One NPKI subscriber entity per node
Multi domain

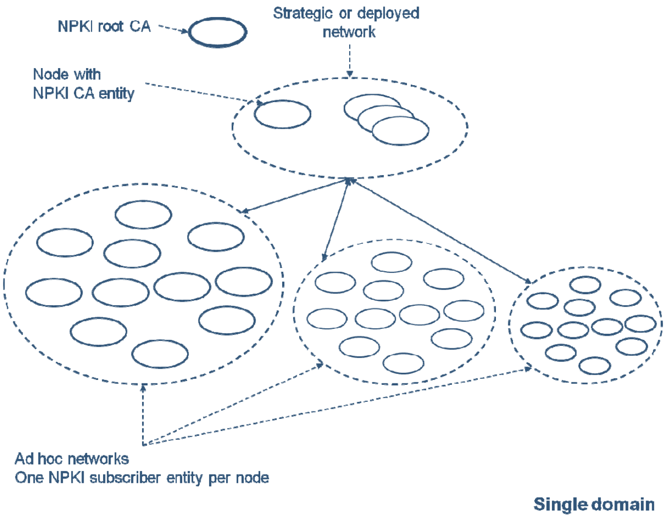
Figure 5.1 Scenario 1 – one network with single domain and multi domain variants

Variant	*Number of NPKI domains*	*Validation scheme*	*Goal*
1	0	(NA)	Study the effect of increasing user traffic Obtain reference values for one network
2	1	Push	Same offered user traffic as above. Study the effect of increasing user traffic under push scheme
		Pull	Same offered user traffic as above. Study the effect of increasing user traffic under pull scheme
3	3	Push	Same offered user traffic as above. Study the effect of adding CAs under push scheme. Three NPKI domains within a single network.
		Pull	Same offered user traffic as above. Study the effect of adding CAs under pull scheme. Three NPKI domains within a single network

Table 5.1 Scenario 1 – one network

5.3 Scenario 2 – three networks

Figure 5.2 shows scenario 2 with a single NPKI domain and with multiple domains. Characteristics are listed in Table 5.2.


NPKI root CA
Strategic or deployed network
Node with NPKI CA entity
Ad hoc networks
One NPKI subscriber entity per node
Single domain

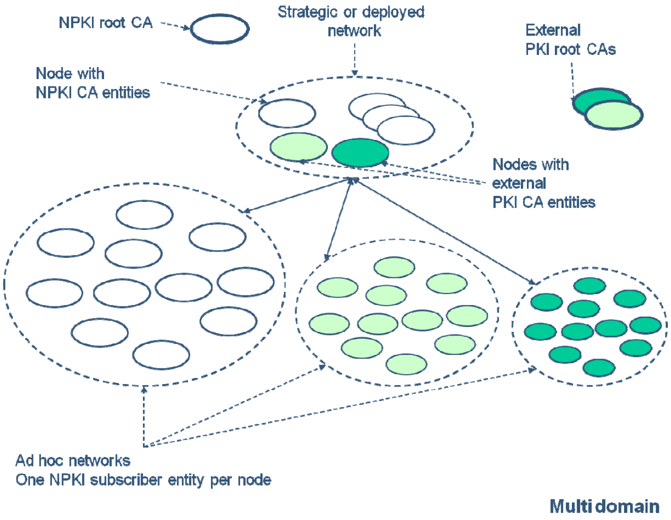

NPKI root CA
Strategic or deployed network
External PKI root CAs
Node with NPKI CA entities
Nodes with external PKI CA entities
Ad hoc networks
One NPKI subscriber entity per node
Multi domain

Figure 5.2 Scenario 2 – three networks with single domain and multi domain variants

Variant	*Number of NPKI domains*	*Validation scheme*	*Goal*
1	0	(NA)	Study the effect of increasing user traffic Obtain reference values for one network
2	1	Push	Same offered user traffic as above. Study the effect of increasing user traffic under push scheme
		Pull	Same offered user traffic as above. Study the effect of increasing user traffic under pull scheme
3	3	Push	Same offered user traffic as above. Study the effect of adding CAs under push scheme. Three NPKI domains, one per single network.
		Pull	Same offered user traffic as above. Study the effect of adding CAs under pull scheme. Three NPKI domains within a single network

Table 5.2 Scenario 2 – three networks

5.4 Identity and Naming Plan

The identity and naming plan is shown in Table 5.3.

Identity type	*Name space*	*ID*	*Network address*
Node	0, ... ,n	name	ID
NPKI domain	A, B, C	name	(NA)
NPKI CA entity	CA	[NPKI domain ID, name] *Examples:* A_CA, B_CA	[Node ID, NPKI CA entity ID] *Example:* 0_A_CA
NPKI subscriber entity	sub0, … , subn	[NPKI domain ID, NPKI CA entity ID, name] *Example:* A_CA_sub0	[Node ID, NPKI subscriber ID] *Example:* 0_A_CA_sub0,
General message entity	gmsg0,…, gmsgn	name	[Node ID, General message entity ID] *Example:* 0_gmsg0,
Message *)	[msg short name]0, …, [msg short name]n	Sender = NPKI CA entity: [NPKI CA entity ID, name (msg short name)] *Example*: A_CA_rp0 Sender = NPKI subscriber entity: (NPKI subscriber ID, name (msg short name)] *Example*: A_CA_sub0_rec2 Sender = general message entity: [General message entity ID, name (msg short name)] *Example*: 0_gmsg1_BMSl_3	(NA)
*)	*) *msg type*	*msg long name*	*msg short name*
Protocol messages:	CMP	Revocation Request	rr
	CMP	Revocation Response	rp
	CMP	CRL Announcement	crlann
	OCSP	Request	req
	OCSP	Response	resp
General messages:	Alarm and orders		AO
	Battle Management System	BMS local	BMSl
		BMS global	BMSg
	Internal message exchange		IME

Table 5.3 Names, identities and network addresses

6 Traffic models

The chapter starts with an overall description of the message sequences between entities. After describing offered NPKI traffic as well as offered user traffic, sequences are described in more detail.

6.1 Message sequences

In this subsection, we specify the relevant message sequences. The message sequences modeled are simple. On the other hand, sequences in single domain variants differ from sequences in multi domain variants and sequences under push-based scheme differ from sequences under pull-based scheme. Figure 6.1and Figure 6.2 show the sequences.

6.1.1 Single domain scenario variants

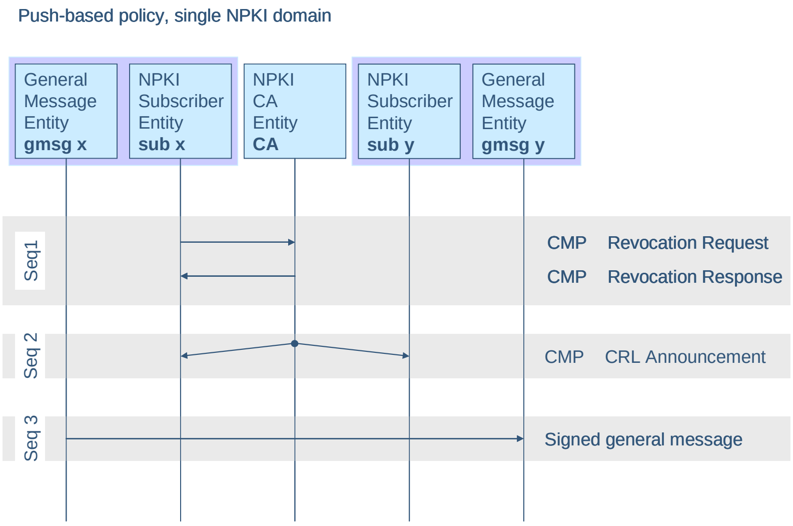
Under the push-based scheme shown in Figure 6.1, we model three sequences:

1. This one is trigged by an event. A subscriber requests the CA for revocation of another subscriber's certificate[14]. A CMP Revocation Request message received from any subscriber is followed by a CMP Revocation Response message from the CA to the requesting subscriber.
2. This one is generated regularly. The CA distributes its CRL to each subscriber by the CMP CRL announcement message. It is assumed that each subscriber maintains received CRLs in a local storage.
3. This one is trigged by an event. A subscriber sends a signed general message to another subscriber. The sending subscriber's certificate is attached to the message. It is assumed that the receiving subscriber validates the certificate by looking up its local CRL storage, and no further communication is required.

Under the pull-based scheme, we model two sequences:

1. This one is the same as described above
2. NA
3. This one is also trigged by an event. A subscriber sends a signed general message to another subscriber. The sending subscriber's certificate is attached to the message. A signed general message received from any subscriber is followed by an OCSP request message from the receiving subscriber to the CA. An OCSP request message from any subscriber is followed by an OCSP response message from the CA to the requesting subscriber.

[14] In a practical application, only certain authorized nodes should probably request revocation. For simulations, however, any node may request.


Push-based policy, single NPKI domain
General Message Entity **gmsg x**
NPKI Subscriber Entity **sub x**
NPKI CA Entity **CA**
NPKI Subscriber Entity **sub y**
General Message Entity **gmsg y**
Seq1
CMP Revocation Request
CMP Revocation Response
Seq 2
CMP CRL Announcement
Seq 3
Signed general message

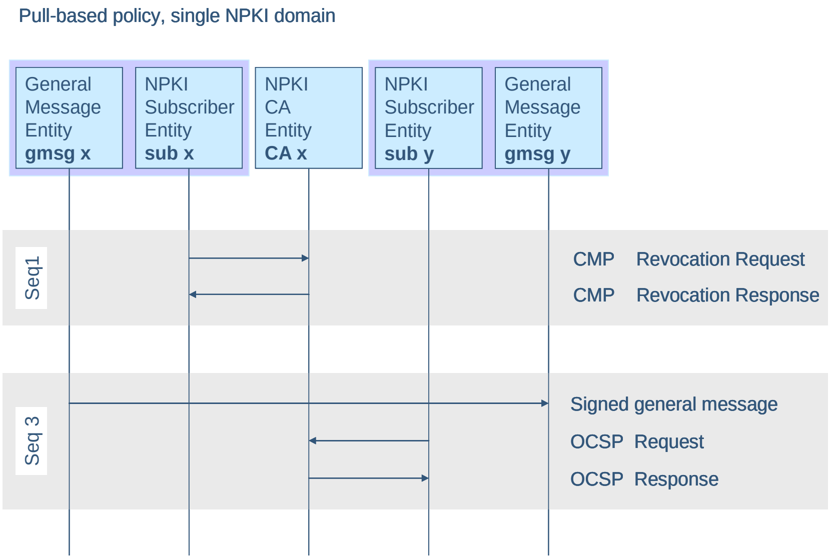

Pull-based policy, single NPKI domain
General Message Entity **gmsg x**
NPKI Subscriber Entity **sub x**
NPKI CA Entity **CA x**
NPKI Subscriber Entity **sub y**
General Message Entity **gmsg y**
Seq1
CMP Revocation Request
CMP Revocation Response
Seq 3
Signed general message
OCSP Request
OCSP Response

Figure 6.1 Message sequences under push-based and pull-based schemes in a single domain NPKI.

6.1.2 Multi domain scenario variants

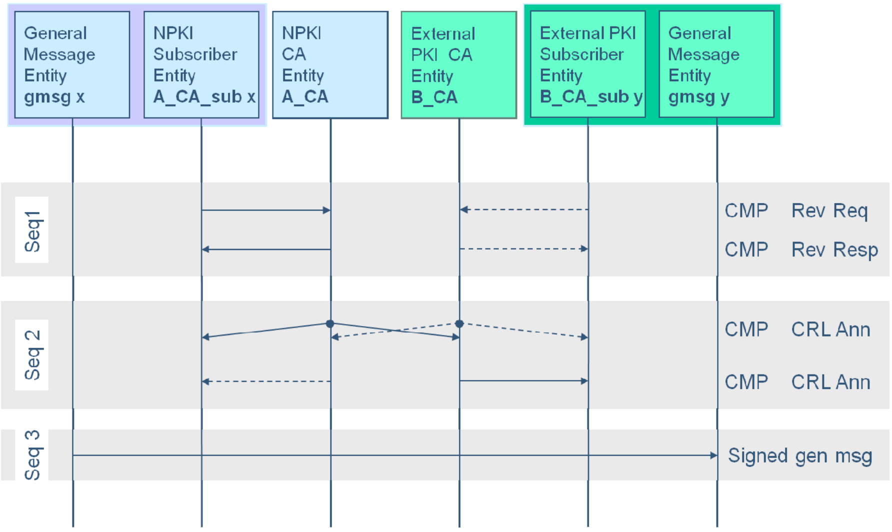
Under the push-based scheme shown in Figure 6.2, we model three sequences:

1. This one is the same as described for a single domain. The figure indicates that this may happen in other domains at the same time
2. This one is generated regularly. A CA distributes its CRL to each intra domain subscriber by the CMP CRL announcement message. In contrast to the single domain case, the CA also distributes its CRL to each of the other CAs involved. An CMP CRL announcement message from any "foreign" CA is forwarded to each intra domain subscriber
3. This one is the same as described for a single domain.

Under the pull-based scheme, we model two sequences:

1. This one is the same as described above
2. This one is generated regularly. A CA distributes its CRL to each intra domain subscriber by the CMP CRL announcement message. In contrast to the single domain case, the CA also distributes its CRL to each of the other CAs involved.
3. This one is trigged by an event as above. A subscriber sends a signed general message to another subscriber. The sending subscriber's certificate is attached to the message. A signed general message received from any subscriber is followed by an OCSP request message from the receiving subscriber to the intra domain CA. An OCSP request message from any subscriber is followed by a OCSP response message from the CA to the requesting subscriber. It is assumed that any CA maintains its repository dynamically, meaning that a subscriber can validate intra domain certificates as well as "foreign" certificates (certificate chains) by looking up its intra domain repository.

Push-based policy, multiple PKI domains (domain A and B)


General Message Entity gmsg x
NPKI Subscriber Entity A_CA_sub x
NPKI CA Entity A_CA
External PKI CA Entity B_CA
External PKI Subscriber Entity B_CA_sub y
General Message Entity gmsg y
Seq1
CMP Rev Req
CMP Rev Resp
Seq 2
CMP CRL Ann
CMP CRL Ann
Seq 3
Signed gen msg

Pull-based policy, multiple PKI domains (domain A and B)

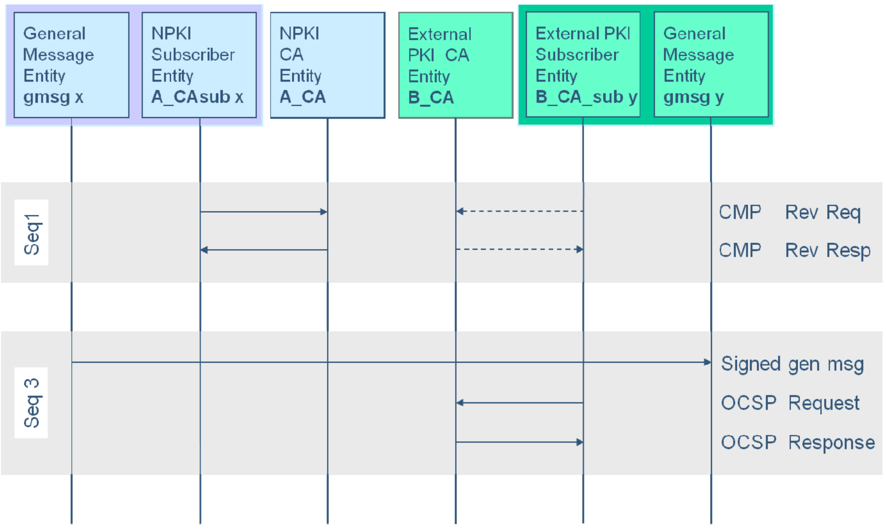

General Message Entity gmsg x
NPKI Subscriber Entity A_CAsub x
NPKI CA Entity A_CA
External PKI CA Entity B_CA
External PKI Subscriber Entity B_CA_sub y
General Message Entity gmsg y
Seq1
CMP Rev Req
CMP Rev Resp
Seq 3
Signed gen msg
OCSP Request
OCSP Response

Figure 6.2 Message sequences under push-based and pull-based schemes in a multi domain PKI.

6.2 Offered PKI traffic

6.2.1 Number of certificates involved

Even though, simulated traffic mainly involves local certificates, we assume all NPKI certificates within the scheme CA to be available to local NPKI subscribers through certificate repositories. As a consequence, CRL's to contain revoked certificates from a larger part of the NPKI domain.

We assume that 2000 certificates are available to the simulated NPKI subscribers, and that a CRL in average contains 10 % of the certificates available. This assumption is based on civilian use of PKI, described in [19].

In section 4.2.2.1, we estimated a CRL to the size of 700 bytes + 36 bytes per entry. This means that the size of a CRL will be (700 + 36*200) = *7900 bytes*.

In multi domain scenarios, we assume one or more external PKIs. This means that NATO root CA is cross certified with an external PKI. In these scenarios, we assume that 2000 certificates per external CA are available to the simulated NPKI subscribers, and that an external CRL in average contains 10 % of the certificates available.

We assume certificates are distributed as shown in Table 6.1:

Certificate subject	*Number of distinct certificates*	*Certificate storage*	*Revoked certificate storage*	
			Push scenarios	*Pull scenarios*
local NPKI subscriber	|node |	– “own” CA repository contains all certificates – local repository contains the subscriber’s own certificate	local repository contains all CRLs	“own” CA repository contains all CRLs
CA	|CA|	– “own” CA repository contains all certificates – local repository contains the certificate of “own” CA		
global NPKI subscribers	2000	“own” CA repository contains all certificates		
external PKI subscribers	2000 per external domain			

Table 6.1 Number of certificates

6.2.2 NPKI message parameters

From section 4.3 and section 6.1 we summarize PKI message parameters as shown in Table 6.2:

Message	*Priority*	*Size[15] in bytes*	*Traffic pattern*		*Interarrival distribution in seconds*
			Sender	*Receiver*	
CMP RevRec	3	fixed (6000)	any NPKI subscriber entity	*Traffic pattern*	exp(λ)
CMP RevRes	3	fixed (6000)	"own" CA	NPKI subscriber entity (sender of RevRec)	response to RevRec above
CMP CRL Ann	3	fixed (7900)	"own" CA	each NPKI subscriber entity within the CAs domain	exp(λ)
CMP CRL Ann (additional external CRLs in multi domain scenarios)	3	fixed (7900)	"own" CA (assuming that "own" CA has received CRL from an external CA)	each NPKI subscriber entity within the CAs domain	exp(λ)
OCSP Req	as the intiating user message	fixed (1400)	any NPKI subscriber	"own" CA	follows from general message, specified in Table 6.3
OCSP Res	as the intiating user message	fixed (1400)	"own" CA	NPKI subscriber entity (sender of OCSP Req)	response to OCSP Req above

Table 6.2 PKI message parameters

6.3 Offered user traffic

6.3.1 General message types involved

Information exchange between users is modeled by four types of general messages, one message type per application. In all scenarios, we assume the following user applications

- *Alarm and orders* (AO). This information mainly stems from sources outside the simulated network(s).
- *Battlefield Management System* (BMS). We assume an overall BMS with sub applications:
  - *BMS local*. The local BMS information stems from sources inside the simulated network, and is distributed to local receivers as well.

[15] Size refers to payload at OSI communication layer 5. Protocol overhead from lower communication layers will be added to the size.

- *BMS global*. The global BMS information stems mainly from sources outside the simulated network(s), and is distributed to local receivers. A part of the global BMS information, however, originates locally, and is distributed to senders outside the simulated network(s).

– *Internal Message Exchange* (IME). Conceptually this information represents voice communication[16] and/or simple text messages.

Four different message types may make the simulations more complex than necessary. Therefore, we may choose a simulation scenario, which involves IME messages, only.

6.3.2 General message parameters

The parameters of general messages are shown in Table 6.3.

Message	*Priority*	*Percentage of total offered user traffic*	*Size[17] in bytes*	*Traffic pattern*		*Interarrival distribution in seconds*
				Sender	*Receiver*	
AO	3	5 %	fixed (100)	wired general message entity outside the network	each general message entity within the network	exp(λ)
IME	2	40 %	fixed (100)	any general message entity within the network	any general message entity within the network	exp(λ)
BMS local	1	10 %	random (50, 400)	any general message entity within the network	each general message entity within the network	exp(λ)
BMS global	0	40 %	random (50, 400)	wired general message entity outside the network	each general message entity within the network	exp(λ)
		5 %	fixed (100)	any general message entity within the network	wired general message entity outside the network	exp(λ)

Table 6.3 General message parameters

[16] Our network simulator does not have the capability of simulating voice communication. IME may, however, be regarded as *Voice over IP* (VoIP).

[17] Size refers to payload at OSI communication layer 5. Protocol overhead from lower communication layers will be added to the size.

6.4 Message sequences revisited

The previous sections can be summarized as shown in Table 6.4. Initiators are randomly selected.

			Seq 1	*Seq 2*	*Seq 3*		
					Initiator		
Scenario	*Variant*	*Valid. scheme*	*Initiator*	*Initiator*	*AO*	*IME and BMS local*	*BMS global*
1	single domain	push	sub0..sub15	CA	external gmsg	gmsg0..gmsg15	GW or gmsg0..gmsg15
		pull	sub0..sub15		external gmsg	gmsg0..gmsg15	GW or gmsg0..gmsg15
	multi domain	push	sub0..sub15	A_CA..C_CA	external gmsg	gmsg0..gmsg15	GW or gmsg0..gmsg15
		pull	sub0..sub15		external gmsg	gmsg0..gmsg15	GW or gmsg0..gmsg15
2	single domain	push	sub0..sub47	CA	external gmsg	gmsg0..gmsg47	GW0..GW2 or gmsg0..gmsg47
		pull	sub0..sub47		external gmsg	gmsg0..gmsg47	GW0..GW2 or gmsg0..gmsg47
	multi domain	push	sub0..sub47	A_CA..C_CA	external gmsg	gmsg0..gmsg47	GW0..GW2 or gmsg0..gmsg47
		pull	sub0..sub47		external gmsg	gmsg0..gmsg47	GW0..GW2 or gmsg0..gmsg47

Table 6.4 Message sequences for simulations

6.5 Simulations

We select two types of user behaviour: First, a uniform traffic matrix, secondly a scalefree matrix based on [8]. We conduct simulations according to the scheme below. The simulations will have increasing complexity.

i. Scenario 1 – reference

ii. Scenario 1 – single domain – pull scheme

iii. Scenario 1 – single domain – push scheme

iv. Scenario 2 – reference

v. Scenario 2 – single domain – pull scheme

vi. Scenario 2 – single domain – push scheme

vii. Scenario 1 – multi domain – pull scheme

viii. Scenario 1 – multi domain – push scheme

ix. Scenario 2 – multi domain – push scheme

x. Scenario 2 – multi domain – pull scheme

7 Study of different COTS products in a PKI context

The quality of the simulation experiment relies on the accuracy and realism of the input parameters. Therefore, it is important to have a good impression on PKI operation and the volumes of generated network traffic. It is necessary to conduct a study using a configuration as similar as possible to the PKI planned by the Norwegian Ministry of Defence. Since the Norwegian Defence emphasizes the use of COTS software, it was an obvious decision to include COTS software in the test environment.

The conduct and results of such a study is presented in this chapter. The products that have been evaluated during the study are:

- Entrust ESP (Certificate Authority)
- Entrust Entelligence
- Microsoft Outlook
- Microsoft Internet Explorer
- Adobe Acrobat
- Mozilla Thunderbird
- Mozilla Firefox
- Java runtime library
- Corestreet VA
- Corestreet Desktop Validation Client

During the operation of an information network, message exchange is expected to happen far more often than the generation and revocation of keys/certificates. Consequently, our focus is the effect of *digital signatures*. The use of digital signatures has these side effects:

- The message volume increases due to signatures appended to the messages.
- The signatures received need validation, which cause network activity related to PKI validation services.

In addition to these observations, it is of some interest to investigate how the COTS products under study scale from a system administrative perspective. Although not relevant as parameter values for the simulation, properties of “administrative scalability” are meaningful for the evaluation of PKI scalability in a wider context.

7.1 The laboratory environment

The laboratory used to study the COTS products employed two ordinary laptop computers connected to a network. They were configured as follows:

1. The ”server”:
   a. Microsoft Windows Advanced Server 2003
   b. Entrust ESP (CA)
   c. Corestreet VA (OCSP Responder)

    d. Sun Directory Server
    e. Internet Information Server (for CRL distribution)
    f. Wireshark network analyzer

2. The ”client”:
    a. Microsoft Windows XP
    b. Adobe Acrobat
    c. Microsoft Outlook
    d. Mozilla Thunderbird
    e. Mozilla Firefox
    f. Entrust Entelligence Service Provider (ESP)
    g. Entrust Security Manager Administration (SMA)
    h. Corestreet Desktop Validation Client
    i. Jetty web server[18]
    j. Wireshark network analyzer

The Wireshark network analyzer was used to record the network traffic during PKI operations between the server and the client. Communication between programs on the same computer (over the local host adapter) was not recorded.

The following sections present the laboratory observations on a per-operation basis. This means that the relevant PKI operations will serve as a comparative context for the products that are involved in that type of operation.

7.2 CA's certificate profile

During the installation of the CA software, a specific certificate profile was chosen. The profile meets the requirements set by National Security Authority (NSM) for certificates in use in the Norwegian Ministry of Defence. These are the main properties of the certificates used for signatures:

- Separate key pairs for Signature and Encryption
- Key length: 2048 bits
- Signature algorithm: RSA over SHA-1 digest
- Key usage: Digital Signature, Key Encipherment (in different certificates)
- Extended Key Usage: Secure Email (OID 1.3.6.1.5.5.7.3.4)
- Authority Information Access: Uniform Resource Locator (URL) of OCSP responder
- CRL Distribution Point: URL of CRL retrieval service (HTTP)

Using this configuration, the sizes of certificates (DER-encoded binary X.509) were observed to be in the range of 1150 to 1250 bytes.

7.3 Issuing of keys and certificates

The process of key and certificate generation has two phases: (1) Generation of a key pair (public

[18] Jetty is an Open Source Web Server written in Java, available from http://www.mortbay.org/jetty/

and private key) and (2) Binding together the public key with identity information in a certificate, signed by the Certificate Authority (CA).

The key and certificate generation can take two approaches:

1. The key generation takes place in the client. The private key then never needs to travel over a network. The public key is sent to the CA for certification together with a ”proof-of-possession”, which is an evidence that the sender also possesses the private key (usually a signature).
2. The key generation takes place in the server, which transports the private key and the certificate back to the client under cryptographic protection.

Regardless of the approach taken, the Entrust CA always requires that a set of authorization codes are sent to the client through a side channel. These authorization codes are generated during user registration and support the cryptographic protection as well as the user authentication. Entrust offers to issue several key pairs for a registered user. It is a common requirement to use separate keys for signing and encryption. Our experimental conditions reflect this requirement. Entrust also offers key backup.

Even though the key pair is generated in the client (approach 1 from the list above), the client program (called Entrust Entelligence) must comply with the key profile set by the CA (for example bit length and choice of algorithm), so this approach does not imply any loss of administrative control over the key generation process.

The following certificate issuing scenarios were used:

1. Single certificate, keys generated in client, no key backup
2. Single certificate, keys generated in client, key backup in CA
3. Two certificates, keys generated in client, no key backup
4. Two certificates, keys generated in client, backup of one key
5. Three certificates, keys generated in client, no key backup
6. Three certificates, keys generated in client, backup of one key

The observed traffic volumes related to these scenarios were:

Scenario no.	*Bytes transferred*	*# packets*
1	20994	36
1	21054	37
2	25053	48
2	25053	48
3	25290	41
3	25236	40
4	27248	40
4	27248	40
5	29228	41
5	29228	41
6	31209	43
6	31209	43

Table 7.1 Observed network traffic during issuance of certificates

7.4 Message and file signing

Several of the client programs studied can sign messages and files, and verify signed messages upon receipt. Adobe Acrobat and Microsoft Outlook are examples of programs that manage signatures on PDF documents and e-mail messages, respectively.

The signing of a message (or file) involves only the locally stored private key, and does not need to generate network traffic. The observed traffic was associated with certificate validation when the signer validates its own certificate in advance to ensure the receiver can validate its signature. Certificate validation is discussed in section 7.5.

Use of signatures leads to increased size of files. The most commonly seen technique is to include the signer's certificate (possibly the entire certificate path) together with the signature value and some structural information on how to verify the signature and restore the original content of the file. Some observations on the tested software and file sizes were:

- *Adobe Acrobat*. A random document (original size 18605 bytes) increased its size with *11255* bytes during a signature process where Adobe validated the signature in advance using CRLs. It is not clear whether the CRL is stored as a part of the signature structure or not. When signing the same document with a certificate which was validated using OCSP (the AIA extension of the certificate pointing to the OCSP service), the size of the file grew with *25744* bytes. We observed that the OCSP response was included in the signature structure, as validation evidence to the receiver (since an OCSP response is time stamped and signed by a trusted authority). Using a different document sample, the size grew from 2663 kB to 2675 kB with the explicit options *not* to include validation evidence.
- *Microsoft Word 2003*. A random document (original size 100352 bytes) increased its size with *3584* bytes during the signature process.

– *Entrust Entelligence*. Offers signing of any file. The size of a jpeg file (original size 73591 bytes) grew with *1993* bytes during the signature process. A jpeg file was chosen since it yields little to a possible compression process (a compression stage is a likely part of a signature function).

– *Mozilla Thunderbird*: The e-mail program offers signatures on outgoing mail through S/MIME structure. The size of a message signature was observed to be *4432* bytes.

– *Microsoft Outlook 2003*: E-mail messages on the ordinary office mail system was signed, but the protocols between the client and the Outlook server were not studied for the purpose of finding signature sizes. Therefore, no data is known for signature sizes in Microsoft Outlook.

Client program	*Size of sample signature (bytes)*
Adobe Acrobat	11255 alt. 25744
Microsoft Word 2003	3584
Entrust Entelligence	1993
Mozilla Thunderbird	4432
Microsoft Outlook 2003	Not known

Table 7.2 Summary table of signature sizes from a selection of client programs

7.5 Signature verification - certificate validation

The program which receives a signed message or file, needs to make sure that:

1. the signature value is correctly calculated from the given public key and the digest value
2. the digest value represents the content of the file
3. the key used for signing is associated with the identity of the signer
4. the key used for signing is authorized for this use
5. the signature on the certificate is also trusted (leads through a chain of signatures to a *root certificate* or *trust anchor*)
6. the key used for signing is not revoked (invalidated) by the issuing authority

Step 1-2 can be performed on the basis of the signature, the public key of the signer, and the message content (step 1-2 is called *signature verification*). Step 3-5 requires the certificate (or certificate chain) of the signer, and step 6 requires access to an auxiliary *revocation status provider* or *validation service* (step 3-6 is called *certificate validation).* Consequently, the content of the signature has impact on the network traffic generated by the validation process. If the signature contains the signer's certificate (an option often seen), then step 1-4 can be done without any network operations.

On the other hand, a message signature without a certificate does not necessarily cause a certificate retrieval operation (for example, from a directory service) if the receiver already has a copy of the certificate (for example, from a previous operation). The habit of sending certificates with every signature found in many COTS programs, appears to be somewhat excessive.

Step 6 can be solved in several different ways, but two options seem to be prevalent:

a) The CA maintains a list of revoked certificates (a CRL) which is regularly distributed to all clients, in a CA-initiated (”push”) or an client-initiated (”pull”) manner. Client-initiated distribution can be arranged with a simple HTTP server. CA-initiated distribution requires services on the client that can receive connections from the CA and is not often found (due to for example the presence of firewalls and NAT-devices).

b) The CA offers revocation status information through an online service over the OCSP protocol. Clients may approach the service with the question ”is certificate *x* revoked?”, and get the answer ”yes”, ”no”, or ”don’t know”.

In both cases, revocation information is time stamped and signed, and is therefore well suited for caching. The different COTS products under study show quite different approaches to the process of certificate validation, which is briefly described below:

– *Adobe Acrobat* maintains its own non-volatile storage (on disk) for CRLs. When CRLs are missing or have expired, they are retrieved based on the CRLDP certificate extension and a centrally configured URL in any combination of preference.

  Using CRLs for validation causes the first validation operation to fetch a CRL and generate network traffic. Subsequent validations of this certificate, or other certificates represented on the same CRL, use the cached CRL until it expires. CRLs are cached in

  ```
  C:\Documents and Settings\<username>\Application
  Data\Adobe\Acrobat\7.0\Security\CRLCache
  ```

  The use of OCSP by Acrobat seems to rely on the AIA extension of the certificate. If set, the validation process includes a call to the associated OCSP responder. It is not clear whether the response is cached or not.

  The inclusion of validation evidence in the signature (CRL or OCSP response) is based on user preferences and may cause the validation process to succeed without network operations. If the evidence is expired (or the OCSP response lacks the *nextUpdate* field), the validation process requires updated validation evidence.

– *Microsoft Word 2003* appears to only verify signatures, not to validate keys or certificates.

– *Entrust Entelligence* validates a certificate by retrieving the CRLs from the resource pointed to by the URL value of the certificate’s CRLDP extension. It seems to disregard the AIA extension for OCSP based validation. It is not clear whether the CRLs are being cached.

– *Mozilla Thunderbird* validates certificates based on locally stored CRLs. The CRLs must be manually loaded into the store by the user, but new versions of the imported CRLs are automatically loaded, either on the basis of the *nextUpdate* field or with regular intervals. As for the use of OCSP, Thunderbird offers to validate certificates based on the use of the AIA extension or a list of ”certificate issuer” – ”OCSP responder” pairs.

– *Microsoft Outlook 2003*. Most Microsoft programs use the Windows CryptoAPI library for certificate validation, which means that they share certificate store and validation options. The CryptoAPI library does not offer OCSP based validation, but appears to employ the CRLDP extension of certificates to load CRLs on demand. Outlook was observed to load CRL from the CRL distribution point when the message was prepared in the sender's program, and in the receiver's program when the message was presented in the inbox (before the message was opened). CryptoAPI caches CRLs in

```
C:\Documents and Settings\<username>\ApplicationData
\Microsoft\CryptnetUrlCache
```

- *Corestreet Desktop Validation Client* is a CryptoAPI plug-in which enables CryptoAPI clients (for example Outlook and Internet Explorer) to validate certificates based on OCSP protocol. The choice of OCSP responder can be based on the AIA extension of the certificate or a list of issuer-responder pairs. The plug-in also offers to cache OCSP responses for a fixed period, but not based on the *nextUpdate* field of the response.

There is certificate validation taking place also in web browsers and web servers, which happens through the establishment of authenticated web sessions based on the Secure Socket Layer (SSL) protocol. Section 7.7 discusses SSL authentication, so the matter of certificate validation for the software involved in SSL, is deferred until then.

Network traffic has been measured during different certificate validation scenarios. The results rely heavily on the context and the history of the operation (for example, the state of caches) and are not reported. Instead, the generated network traffic may be predicted on the basis of the stochastic processes involved in the certificate validation and the sizes of the basic elements of exchange. These elements are:

- The CRL: The CRL has been observed to have a ”base part” of 500-600 bytes, and an additional 35-38 bytes for each certificate on the list. The CRL is most oftenly fetched through an HTTP protocol transaction, for which the overhead is known.
- The OCSP operation: The OCSP responder under study uses HTTP protocol (on port 3501). The size of the OCSP response has been observed to be 1460 bytes. The entire OCSP transaction consumed 2838 bytes in 12 packets.

Observe that CryptoAPI allegedly supports the use of delta CRL[19], but this has not been tested.

7.6 Message encryption

Although the primary interest of the PKI scalability study has been on authentication[20] mechanisms, some experiments have been done on encryption services. The observations and results from these experiments are briefly presented in this section.

- *Adobe Acrobat* offers encryption based on public key certificates. The certificate in use is not validated during the encryption process, nor during the decryption process. The sample file size (23158 bytes) increased with 6803 bytes during the encryption process.
- *Entrust Entelligence* also offers encryption of files, without certificate validation during

[19] Distribution of only the most recent additions to a CRL

encryption or decryption. File sizes increased with approximately 700 bytes during the encryption process.

- *Microsoft Outlook 2003* did not encrypt messages, due to a bug in that particular build of the program: a bug that required certificates to have both the Signature and Encryption usage extension, which is not met by the certificate profile in use.
- *Mozilla Thunderbird* offers encyption of outgoing messages based on the value of the *subjectAltName* extension of the certificate. This extension is often used to store the subject's RFC-822 e-mail address (name@domain.com), and Thunderbird browses the certificate store to find a certificate with *subjectAltName* value equal to the message recipient. Encrypted messages are sent in S/MIME format and Base64 coded. The increase in message size is large if the encryption involves a Base64 conversion (33% increase). If the plaintext also would require Base64 coding, the increase is modest (less than 10%, but more than 800 bytes).

In theory, certificates should not be validated during decryption of a message. An encrypted message must be readable also after the certificate has expired, so a valid certificate should not be required for the decryption process.

7.7 SSL Authentication

For secure web communication, Secure Socket Layer (SSL) offers a secure and authenticated connection between a web browser and a web server, characterized by the *https* protocol designation in the location URL. SSL offers one-way or two-way authentication as well as privacy. The establishment of the secure channel is not described here, only the fact that the certificates involved, may be validated during the establishment process.

For the investigation of the validation process, the Jetty web server was used on the server side. For certificate validation, the Jetty server (written in Java) is assumed to use the standard Java classes for certificate management found in the *java.security.cert* libraries. On the client side, Internet Explorer v.7 and Mozilla Firefox v.3.0 was used. Internet Explorer uses the CryptoAPI library, and Firefox employs the same Network Security Services (NSS) cryptographic library as Thunderbird, so they were expected to behave like other applications of these libraries.

7.7.1 One-way authentication

For SSL used in a one-way authenticated connection, only a server certificate is necessary. The server's certificate and private key must be stored in a Java key store file (Entrust can export the certificate to a Public Key Cryptography Standards (PKCS) #12 file which can be imported into the key store file) , and the Jetty must be configured to accept this file as its key store as well as trust anchor. The Jetty server must be started with SSL options enabled. The subject DN of the server certificate must have the CN value equal to the DNS name of the server, for example CN=server.ffi.no, DC=ffi, DC=no.

When a web client opens a https connection to the server, the server certificate is sent to the client as a part of the connection establishment. The client may validate the certificate as a part of the authentication process.

Internet Explorer 7.0 accepted the Entrust-generated certificate and successfully established a connection to the Jetty server. Internet Explorer validates the certificate using CryptoAPI and does only CRL validation (unless the Corestreet Validation Client plug-in is installed).

Mozilla Firefox refused to accept the certificate. It turns out that it requires the Extended Key Usage (EKU) certificate extension to contain the value ”SSL Server Authentication” (OID value: 1.3.6.1.5.5.7.3.1).

Certificates and keys generated with OpenSSL[21] and applied the necessary EKU value were tested and worked fine also with Firefox. Firefox validates certificates with the same library as Thunderbird, and employs manually loaded CRLs and OCSP validation, see section 7.5.

The Jetty server often threw Java exceptions during the connection establishment process, claiming ”Unknown Extension”. Discussion forums on the Internet suggest that this may be due to a bug in Sun’s SSL library.

7.7.2 Two-way authentication

SSL/HTTPS connections can also provide authentication of both the server and the client. The Jetty configuration files need editing to accomplish this *(NeedClientAuth=true)*. Only Mozilla Firefox was tested for this purpose, not Internet Explorer.

Firefox can be configured to hand out one specific certificate on the request from the server, or that the user is prompted to choose one. The server was observed to send a list of trust anchors to the client, and that Firefox only lists the certificates descending from one of these. If the trust anchor of the client is not installed in the Jetty server, the connection fails already at this stage, long before the certificate validation takes place.

The certificate that Firefox sends to the Jetty server is now validated according to the settings in the Java runtime on the server. The Java runtime offers revocation check from CRL sources and OCSP responders depending on the configuration. The configuration options will now be explained, but one reservation must be made: The Java security framework is a highly modular structure of providers and factory classes, and a different Jetty installation may therefore display a different behaviour.

Two system properties control the use of CRLs[22]:

```
    com.sun.net.ssl.checkRevocation=true
    com.sun.security.enableCRLDP=true
```

If these two are set (for example as ”-D” command line options in the startup command) the validation of client certificates will download the CRLs referred to by the CRLDP extension value. The first time this happens, the HTTP protocol retrieves it unconditionally. Subsequent

[21] OpenSSL is an Open Source SSL library and utility program. See http://www.openssl.org/

[22] For a fuller discussion of PKI and Java: http://java.sun.com/j2se/1.5.0/docs/guide/security/pki-tiger.html

validations (of certificates with the same CRLDP value) will download the CRL on the condition that a newer version is available. This is accomplished with the use of the ”If-Modified-Since” element in the HTTP request header. Downloaded CRLs do not survive a server restart. Without the ”enableCRLDP=true” property the validation process checks static CRLs, but it is not clear where they are stored.

For the use of OCSP responders, this security property must be set

```
      ocsp.enable=true
```

It cannot be set through command line options, but is likely to be kept in the file `$java_home/jre/lib/security/security.properties`. The use of OCSP now takes precedence over the CRL checking, either on the basis of the AIA certificate extension value or a fixed value from the `ocsp.responderURL` security property. CRL checking is still the fallback option, and the OCSP check requires the ”checkRevocation=true” property to be set.

The calls to the OCSP responder were observed to happen for every validation operation, and no caching of OCSP responses appeared to take place.

After the successful establishment of a connection, the programs running on the server have now access to the validated client certificate. For a Java Servlet, it is accessible as one of the member variables in the HttpServletRequest object given as a parameter to the doGet/doPost methods.

7.8 Certificate revocation

The process of certificate revocation is initiated by a decision to exclude a key pair from further use. A while after this decision has been made, everyone ”knows” that the key is revoked, and refuses to validate signatures created after the revocation time, and refuse to encrypt data with this key. The latency (time instant between the decision and its effect) involved may be called *revocation latency* and serves as a measure of the security risk of having revoked certificates in operation.

When using Entrust, the revocation mechanism consists of the following steps:

1. The security officer using the ”Entrust Security Manager Admin” (which serves as a Registration Authority) browses the user catalog and chooses the user for which to revoke certificates. S/he can choose to revoke all or some of this user’s certificates, and choose whether a new CRL should be issued at once.
2. The CA receives the request from the security officer and marks the users’ certificates as revoked, removes them from the directory system (LDAP) and possibly issues a new CRL.
3. The new CRL (if made) is not distributed as a result of this operation, but is simply made available for clients to download.

The RA communicates with the directory server (LDAP protocol) and the CA (with the Entrust ”admin shell” protocol). The entire operation involved 37355 bytes of traffic on the RA side, in 88 packets and over 3.3 seconds. The possible increase in network traffic due to the newly issued CRL is not considered.

As an alternative method, the revocation may take place on the CA user interface, in which case the network traffic will be negligible (if the directory server is co-located with the CA).

7.9 Properties of administrative scalability

This chapter has so far considered the generated network load as a means for predicting the scalability properties of a PKI, including the clients that employ the PKI services. On the other hand, highly scalable systems require that the system administration and configuration tasks associated with the operation of the system remain feasible also when the scale grows with several orders of magnitude.

A PKI has the potential of enormous growth. The ultimate scale is that every person, every device and every service on the planet has a certificate which may be validated using the same PKI service instance, that all applications use the same set of certificates, and that all objects with an identified origin contain digital signatures.

It is common knowledge in the field of distributed systems that high complexity should be kept on a small scale. In a client-server environment, this means that the clients (since they are many) should be kept as simple as possible, and the components of the system that require complex management and configuration are kept centralized inside a few servers. The great success of Internet computing lies with the fact that the clients (web browsers) are zero-footprint and zero-management and represent no scaling limits.

In the opinion of the authors, the client configuration of PKI applications introduces scaling limitations. The clients need particular (and complex) client software installation, for example Enstrust Entelligence and Corestreet Desktop Validation Client, and the existing client applications like the ones studied in this chapter, need specific configuration of certificate store and validation options.

It is of particular concern that the different applications keep their certificate stores separate from each other. This means that each application must install their trust anchor, client and server certificates, and revocation lists. Windows-specific software likely use the CryptoAPI store, but even Thunderbird and Firefox, coming from closely related projects, choose to keep the stores separate. Adobe keeps its own certificate store, possibly for reasons of portability between operating system platforms. Java programs may use the ”standard” key store file, but are often found to prefer key store files separate for each application.

Another concern is the portability of certificates. Since we have found that different application have different requirement to the certificate content (for example the required content of EKU extensions in Thunderbird and IE), it is a likely situation to occur that a certificate issued for one application cannot be used in other applications.

Based on these observations, it is reasonable to conclude that the administrative effort associated with the PKI applications scale with both the number of clients and the number of applications.

Also, the number of certificates is likely to scale in the same manner. This is far from an ideal situation, and is likely to limit the deployment of PKI applications.

7.10 Summary

This chapter has reported from a practical experiment with the purpose to assess the scalability properties of a selection of PKI-related COTS software products. The main focus of the investigation has been on network traffic volumes, but also administrative properties of the products have been considered.

It has been shown that different client programs employ different strategies for protocol choice, caching etc., which is the reason why no numbers related to for example ”certificate validation” is reported. Rather, the numbers related to the basic operations are presented in the summary table below, which can be fed into an analysis of simulation engine together with estimations of traffic, client and certificate distribution.

Operation / property	*Number of bytes (approx.)*	*Number of packets*
Issue two certificate pairs	25000	48
Certificate revocation	37000	88
OCSP service invocation	2800	12
Size of CRL	700 + 36 * #entries	
Size of certificate	1200	
Size of signature structure – Adobe Acrobat	12000 alternatively 26000	
Size of signature structure – S/MIME	4400	
Size of signature structure – MS Word	3600	
Size of signature structure – Entrust Entelligence	2000	

Table 7.3 Summary tables for file size increments and network traffic during signature operations

8 Conclusive remarks

Our goal is to provide knowledge of the communication capacities required to operate a Public Key Infrastructure (PKI). For the tactical domain, the deployment of PKI and PKI-dependant applications should not be planned without this knowledge. Also, further research in key management schemes call for a deeper knowledge about scalability issues related to today's PKIs. As far as we know, neither academic nor military research has published studies on this topic.

Based on a high-level description of the NATO PKI (NPKI), we have modeled and specified a generic PKI. We review main operational requirements, but only functionality supposed to have clear impact on communication resource consumption, is modeled and specified. This functionality includes a subset of messages from standard PKI protocols. Such protocols deal with the management of PKI as well as the operation. To ensure realistic size estimations of signature structures, certificates and PKI protocol messages, we have implemented and studied different commercial PKI products. PKI variables include different certificate validation schemes and the number of PKI domains involved.

User scenarios in the tactical domain are modeled and specified. Traffic imposed by a set of user applications, suitable for the tactical domain, is modeled. User behavior may vary according to different traffic matrixes, like uniform and scale free schemes.

Previous publications model and specify the underlying communications network in detail. We assume tactical radios forming one or more ad hoc networks linked to a wired infrastructure. The network handles traffic priority. Network variables include available bandwidth, the number of networks involved and the number of nodes per network.

The model and specification found in this report form the basis for future scalability analyzes. Further work is to study the impact of PKI usage under varying conditions. As a main rule, we describe the impact as a function of the amount of offered traffic.

Abbreviations

AA	Attribute Authority
AIA	Authority Information Access
API	Application Programming Interface
CA	Certification Authority
CC	Certificate Policy
CIS	Communication and Information Systems
CIS	Communications and Information System
CMP	Certificate Management Protocol
COTS	Commercial Off The Shelf
CP	Certificate Policy
CPS	Certification Practice Statement
CPS	Certificate Practice Statement
CRL	Certificate Revocation List
CRLDP	CRL Distribution Point
CRMF	Certificate Request Message Format
DACAN	Military Committee Distribution and Accounting Agency
DEKMS	DACAN Electronic Key Management System
DER	Distinguished Encoding Rules
DN	Distinguished Name
DoS	Denial-of-Service
DVCS	Data Validation and Certification Server Protocol
EKMS	Electronic Key Management System
EKU	Extended Key Usage
EUDAC	Military Committee European Distribution and Accounting Agency
FTP	File Transfer Protocol
HTTP	Hyper-Text Transfer Protocol
ICT	Information and Communication Technology
IESG	Internet Engineering Steering Group
IETF	Internet Engineering Task Force
IKEv2	Internet Key Exchange version 2
IO	International organizations
IP	Internet Protocol
IPSec	Security Architecture for the Internet Protocol
ITU	International Telecommunication Union
ITU-T	ITU-Telecommunication Standardization Sector
KMI	Key Management Infrastructure
LDAP	Lightweight Directory Access Protocol
MAC	Medium Access Control
MAC	Message Authentication Code
NATO	North Atlantic Treaty Organization
NC3B	NATO Consultation, Command and Control (C3) Board
NECEMS	NATO Electronic Key Management System
NNN	non-NATO nations
NPKI	NATO PKI
NPMA	NATO PKI Management Authority
NR	NATO RESTRICTED
NS	NATO SECRET
NSA	National Security Agency
NSCA	NATO CIS Services Agency
NSM	National Security Authority
NSS	Network Security Services
OCSP	Online Certificate Status Policy

OID	Object IDentifier
PAC	NATO PKI Adversary Cell
PAD	Peer Authorization Database
PKC	Public Key Certificate
PKCS	Public Key Cryptography Standards
PKI	Public Key Infrastructure
PKI4IPSEC	Profiling Use of PKI in IPSEC (working group in IETF)
PKIX	Public Key Infrastructure (X.509) (working group in IETF)
PKIX	Public Key Infrastructure (X.509)
PMI	Privilege Management Infrastructure
QoS	Quality of Service
RA	Registration Authority
RFC	Request for Comments
SA	Security Association
SMI	Security Management Infrastructure
SMTP	Simple Mail Transfer Protocol
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
URL	Uniform Resource Locator
X.509	ITU-T standard

References

[1] C. Adams, S. Farrel, T. Kause and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)", IETF rfc 4210, 2005.

[2] A. Arsenault and S. Turner, "Internet-Draft, Internet X.509 Public Key Infrastructure: Roadmap, draft-ietf-pkix-roadmap-09.txt, IETF, 2002 (work in progress).

[3] T. J. Berg, "Modelling and Simulation of MRR networks", FFI Report 2008/00061, 2008.

[4] T. J. Berg, "oTWLAN – a simulator modelling tactical ad hoc networks", FFI Report 2009/00911, 2009.

[5] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley and W. Polk, "Internet X.509 Public Key Infrastructure – Certificate and Certificate Revocation List (CRL) Profile", IETF rfc 5280, 2008.

[6] D. W. Chadwick, "Internet-Draft, Internet X.509 Public Key Infrastructure – Operational Protocols – LDAPv3, draft-ietf-pkix-ldap-v3-05.txt, IETF, 2002 (work in progress).

[7] S. Chokani, W. Ford, R. Sabett, C. Merrill and S. Wu , " Internet X.509 Public Key Infrastructure - Certificate and Certification Practices Framework", IETF rfc 3647, 2003.

[8] A. Fongen, "Scalability analysis of selected certificate validation scenarios", FFI-notat 2008/01016, 2008.

[9] A. Fongen, "XML Based Certificate Management", FFI-rapport 2008/00278, ISBN 978-82-464-1346-4, 2008.

[10] T. Freeman, R. Housley, A. Malpani, D. Cooper and W. Polk, "Server-based Certificate Validation Protocol (SCVP), draft-ietf-pkix-scvp-33.txt, IETF, 2007 (work in progress).

[11] A. M. Hegland, E. Winjum, S. F. Mjølsnes, C. Rong, Ø. Kure, and P. Spilling, "Survey of Key Management in Ad Hoc Networks", IEEE Communications Surveys & Tutorials, 3rd Quarter, 2006.

[12] International Telecommunication Union – Telecommunication Standardization Sector (ITU-T), http://www.itu.int/ITU-T/.

[13] International Telecommunication Union – Telecommunication Standardization Sector (ITU-T), "Recommendation X.509, Information Technology - Open Systems Interconnection - The Directory: Authentication," 1997.

[14] Internet Engineering Task Force (IETF), http://www.ietf.org/.

[15] Internet Engineering Task Force (IETF), "Internet-Draft, Internet X.509 Public Key Infrastructure: Roadmap, draft-ietf-pkix-roadmap-09.txt," 2002.

[16] M. Myers, R. Ankney, A. Malpani, S. Galperin and C. Adams, "X.509 Internet Public Key Infrastructure – Online Certificate Status Protocol – OCSP", IETF rfc 2560, 1999.

[17] M. Myers and H. Tschofenig, "Online Certificate Status Protocol (OCSP) Extensions to IKEv2", IETF rfc 4806, 2007.

[18] National Security Agency (NSA), http://www.nsa.gov/

[19] National Institute of Standards and Technology (NIST),

[20] NATO Consultation, Command and Control (C3) Board, "NATO Public Key Infrastructure (NPKI) Certificate Policy", AC/322-D(2004)0024-REV2, 2008.

[21] NATO Consultation, Command and Control (C3) Board, "Statement of Technical Characteristics for the NATO Public Key Infrastructure, AC/322-N(2008)0004, 2008.

[22] NATO SHAPE/SACT, Statement of Operational Requirements (SOR) for NATO Public Key Infrastructure NPKI for Use in the NATO Alliance, version 1.3, dated 24 March 2009.

[23] D. Pinkas and R. Housley, "Delegated Path Validation and Delegated Path Discovery Protocol Requirements", IETF rfc 3379, 2002.

[24] Public Key Infrastructure (X.509), http://www.ietf.org/html.charters/pkix-charter.html

[25] J. Schaad, "Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", IETF rfc 4211, 2005.

[26] J. Sermersheim (Ed),"Lightweight Directory Access Protocol (LDAP): The Protocol", IETF rfc 4511, 2006.