

## Case study of XML based PKI management protocols.

Tomas Gustavsson  
PrimeKey Solutions AB  
[www.primekey.se](http://www.primekey.se)  
[www.ejbca.org](http://www.ejbca.org)

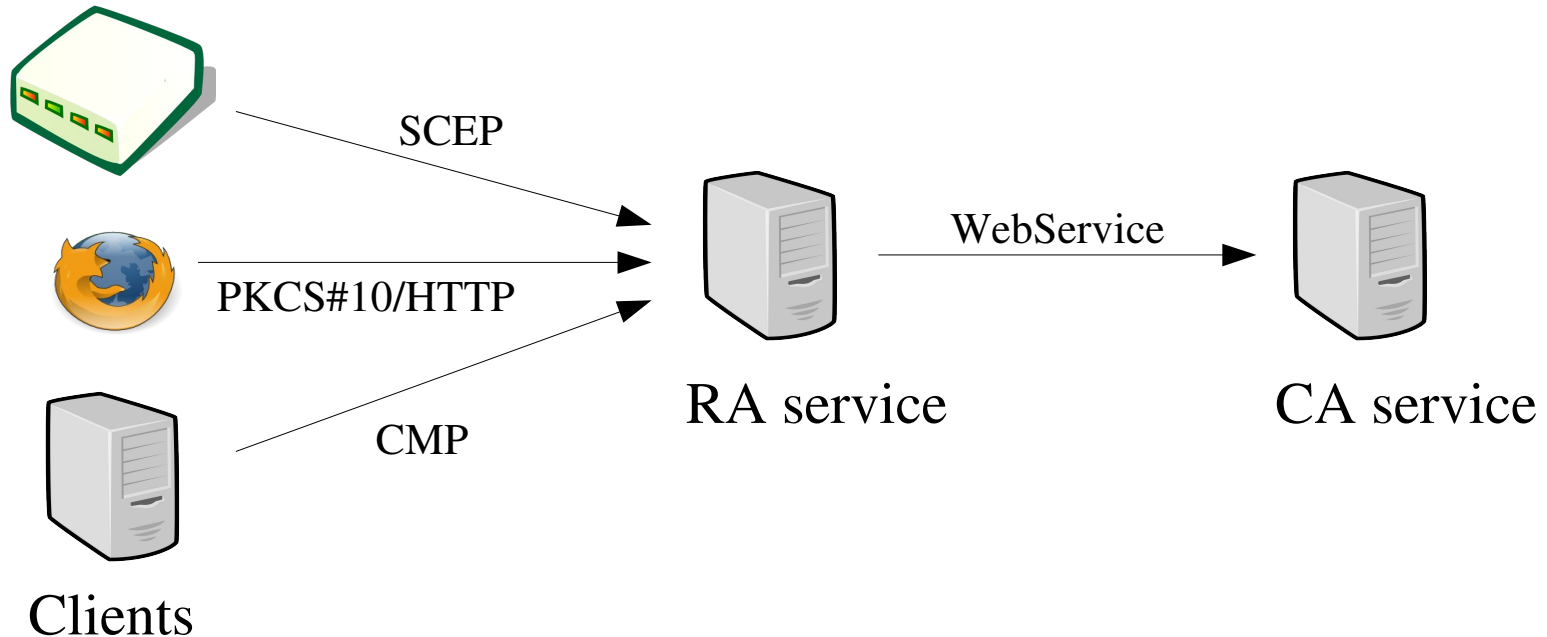


# Background

## **Data loss and key management issues**

- Requirement: encrypt data to prevent unauthorized disclosure
- Data loss: a lost encryption key means data is lost
- Secure key management: generate, store, renew and recover keys

# Common PKI scenario



The WebService API is typically used in these scenarios:

Tolima – Token Lifecycle Management, smart card issuing system with user friendly front-end.

EU ePassport PKI – front end to inter country communication and border control system with passport readers.


# ToLiMa case study

Hard Token Management Framework in Java used to manage the complete lifecycle of an organizations Smartcard and/or USB dongles.

ToLiMa is an application build using the framework, used to manage smart cards in the Swedish Police.


- Issue tokens, regular, temporary and project
- Unlock PIN of a token without exposing the PUK code for the users or administrators
- Revoke lost cards
- Renew expiring cards
- Activate cards in the organizations systems
- It is also possible to issue and unlock tokens on an approval basis, used in scenarios where no token administrator is available.

# ToLiMa


Address  <https://localhost:8443/ejbca/hardtkenmgmt/>

## Hard Token Management

[www.hardtkenmgmt.org](http://www.hardtkenmgmt.org)



### Issuing Card, step 2/4

 Pick type of card.

User Serial Number: 19770611


Name:


Enter name manually. This will be logged

Regular

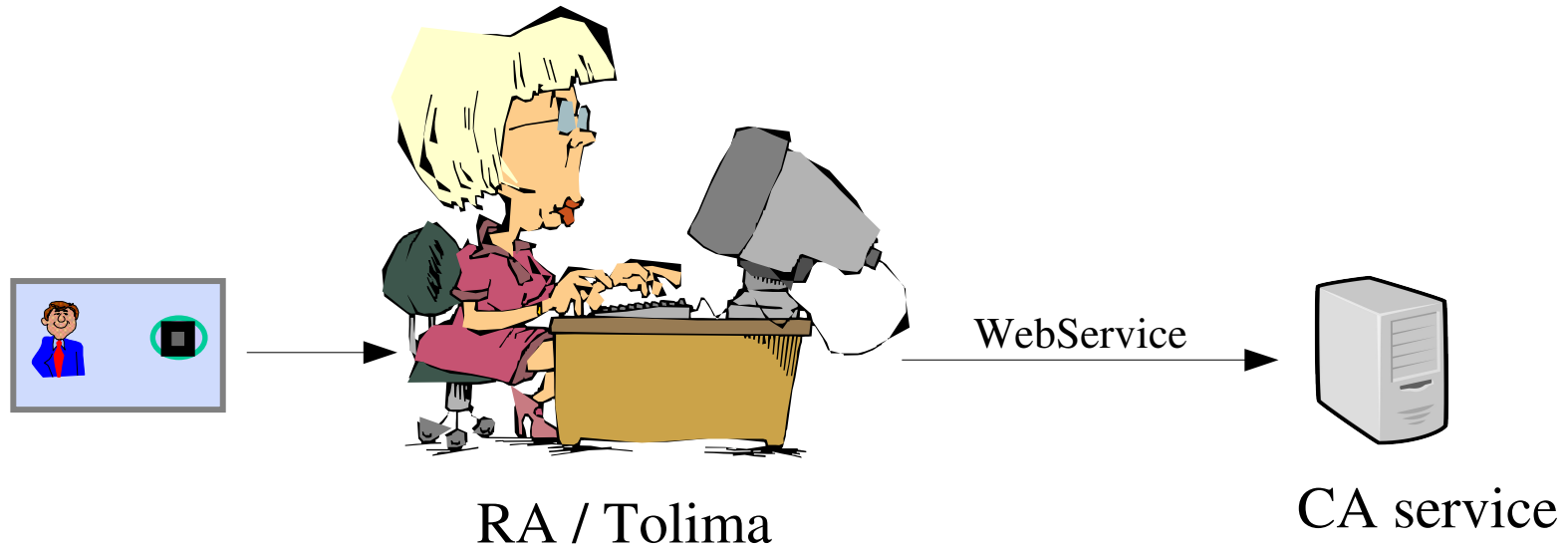
Temporary

Consultant





# ToLiMa - WS

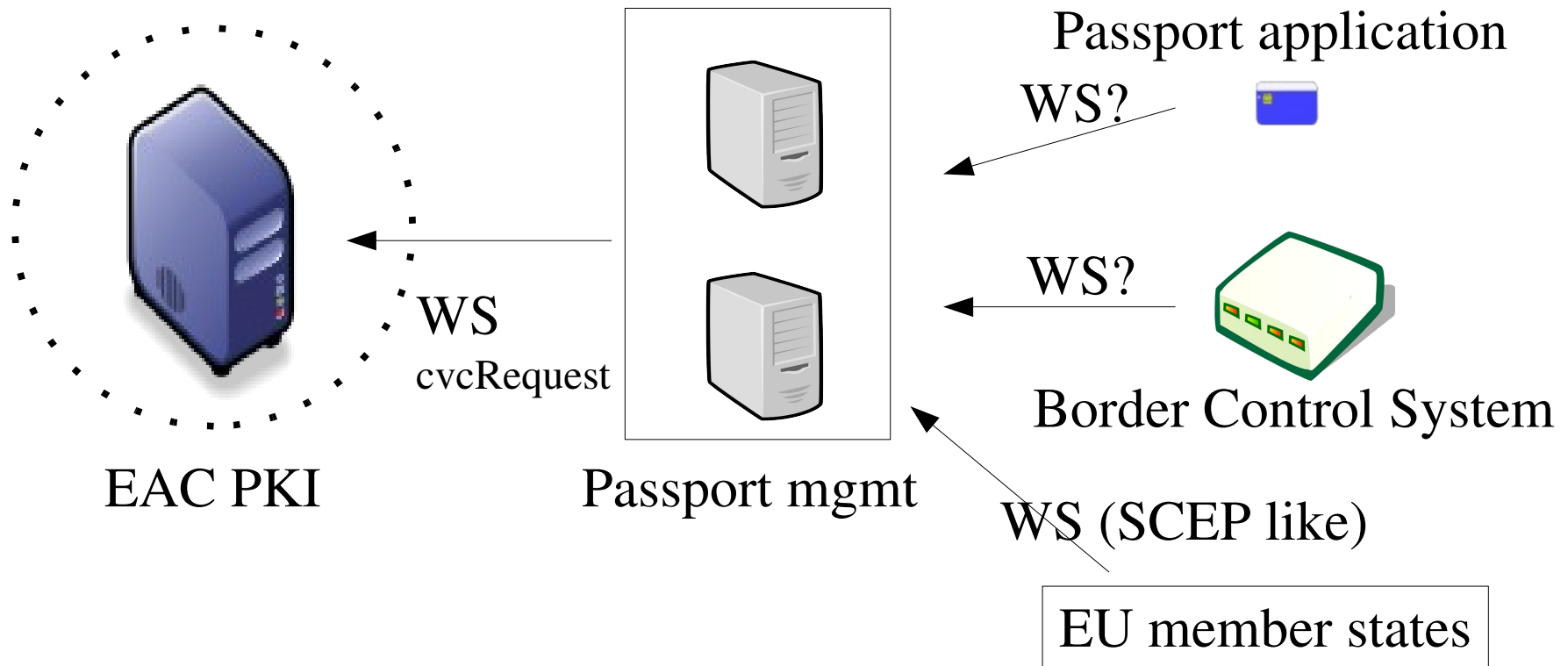


WS-calls used:

- |                        |                        |            |
|------------------------|------------------------|------------|
| -getTokenData          | - getUserData          | -findUser  |
| -deleteUserData        |                        |            |
| -genCertificates       | -republishCertificate  |            |
| -checkRevokationStatus | -revokeCert/Token/User |            |
| -isApproved            | -isAuthorized          | -customLog |

# ePassport PKI case study

Passport management system manages inspection systems. Receives requests from IS, forwards to PKI and returns certificate back to IS. Also requests from other EU member states are handled through this interface.



# ePassport WS standards

First attempt – email based protocol. Will mostly be used for manual fallback.

Second attempt – standardize WS-API (soap) for CVCA communication between member states (CVCA communication protocol).

Future – build on the CVCA communication protocol to include everything needed to inspection system enrollment.

CVCA communication protocol modeled after the SCEP protocol (Simple Certificate Enrollment Protocol). No standard alternative exists.

# Available XML protocols

Current standard:

XKMS - <http://www.w3.org/TR/xkms2/>

Provisioning, distributing and managing asymmetric keys

Proposed Oasis standard:

EKMI – Arshad Noor

Provisioning, distributing and managing symmetric keys

Other proposals:

Keygen2 – Anders Rundgren

Provisioning asymmetric keys for web browsers

# XKMS

X-KISS - XML Key Information Service Specification

X-KRSS - XML Key Registration Service Specification

Early protocol, defined before WebService gained real wide spread use.

Interoperability between stacks was not an easy task.

Many competing, industry standard non-XML protocols.

- SCEP, CMP, LDAP, OCSP, ...

Security is still handled much on local scale, not using services as anticipated for X-KISS.

Complex and some parts of standard is unclear.

Hypothetical examples...

No open source referens implementation.

# EKMI

Enterprise Key Management Infrastructure (EKMI) is the term given to "a collection of technology, policies and procedures for managing all cryptographic keys - symmetric and asymmetric - in the enterprise".

## OASIS EKMI-TC:

To standardize a protocol - the Symmetric Key Services Markup Language (SKSML) - for applications and/or computerized devices to acquire symmetric key management services, securely, over a network.

## EKMI and PKI:

EKMI relies on PKI for strong authentication for management of symmetric keys.

## EKMI



- Generate
- Encrypt
- Decrypt
- Protect
- Escrow
- Authorize
- Recover
- Destroy



- Generate
- Encrypt
- Decrypt
- Protect
- Escrow
- Authorize
- Recover
- Destroy



- Generate
- Encrypt
- Decrypt
- Protect
- Escrow
- Authorize
- Recover
- Destroy



- Generate
- Encrypt
- Decrypt
- Protect
- Escrow
- Authorize
- Recover
- Destroy

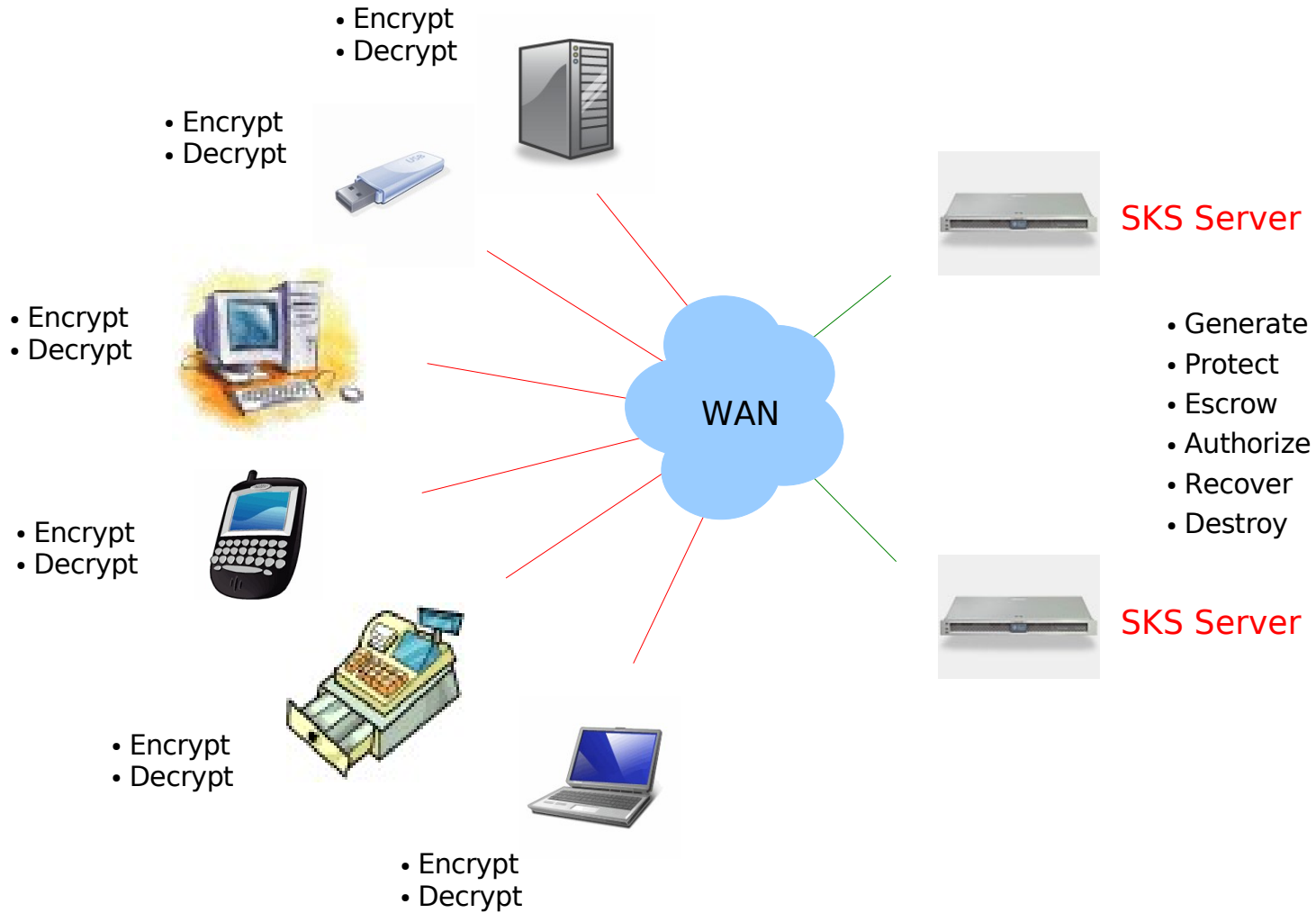


- Generate
- Encrypt
- Decrypt
- Protect
- Escrow
- Authorize
- Recover
- Destroy



- Generate
- Encrypt
- Decrypt
- Protect
- Escrow
- Authorize
- Recover
- Destroy

## EKMI



# Keygen2

KeyGen2 is an effort creating a standard for browser-based on-line provisioning of PKI user-certificates and keys.

In addition, KeyGen2 supports an option for “piggybacking” symmetric keys like OTP (One Time Password) seeds on PKI.

Using a generic credential extension mechanism, KeyGen2 can support things like Microsoft InfoCards and downloadable code associated with a specific key.

One of the core targets for KeyGen2 are mobile phones equipped with TPMs (Trusted Platform Modules), which properly applied, can securely emulate any number of smart-cards. Although TPMs definitely is not a standard utility today, it is anticipated in the future.

More info at <http://webpki.org/>

# Different WS APIs

End entities



Enrollment/key  
mgmt



RAs

Administration



PKI service

EKMI/XKMS/Keygen2

EJBCA WS-API

Different APIs needed for end entities and RAs.

- Enrollment and key management
- Administration

# Different WS APIs

End entities



WS  
provisioning/mgmt



RAs

WS Admin



PKI service

WS exchange



PKI service

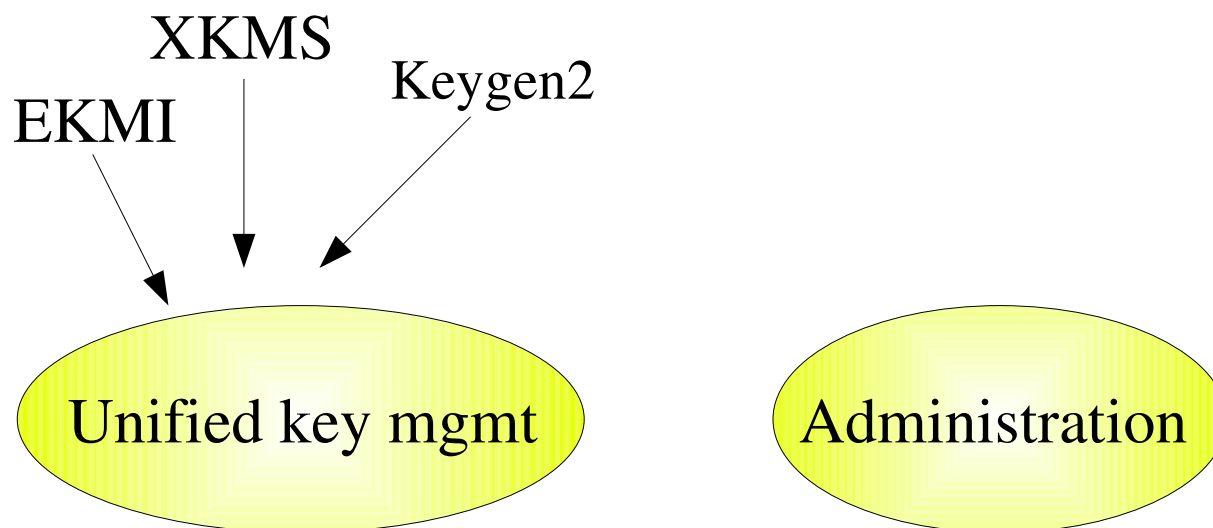
Different APIs needed for end entities and RAs.

- Enrollment and key management
- Administration
- Certificate exchange and cross certification

# Standardization dilemma

- Need for completeness and applicability for different use cases.
- Complexity and ability to deploy.
- WS-APIs relatively easy to invent and deploy your own. Need careful consideration which areas really benefit from standardization.
- "Competition" from howgrown WS-APIs for various use cases, e.g. epassport CVCA – DV communications protocol.
- Open source reference implementations.

# Key management future



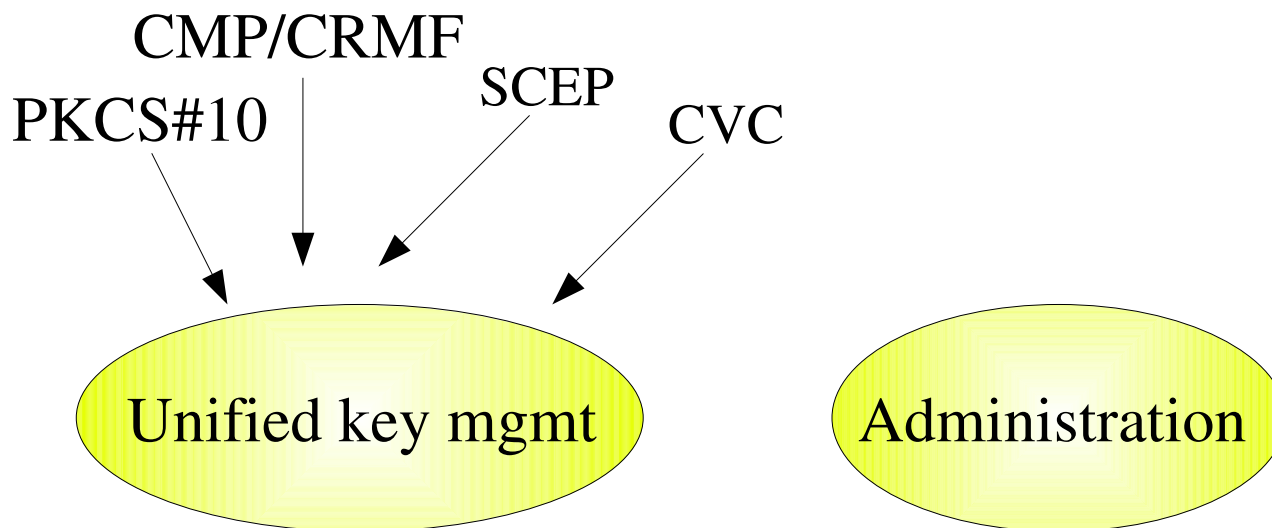
Unified key management protocol handling all aspect of enterprise key management.

XKMS – first try

EKMI – Symmetric key management

Keygen2 – Asymmetric provisioning

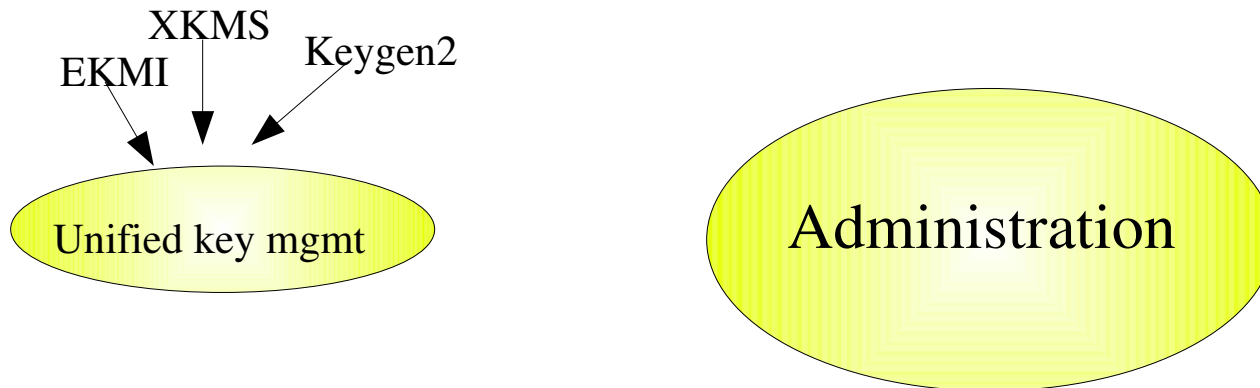
# Key management future



Leverage existing standards. Otherwise WS will be forever left on server-server level.

Example: Most clients generate PKCS#10 request messages. If WS protocol allows that a simple adapter can be used to WS-enable the client.

# Key management future



Administration is where many commercial offerings try to differentiate.