Key Management Interoperability Protocol (KMIP)

Storage Developer's Introduction

SNIA Fall 2009

Gordon Arnold, garnold@us.ibm.com
Chair, Storage Security Industry Forum

Abstract

Data in networked storage environments is significantly more vulnerable to unauthorized access, theft, or misuse than data stored in direct-attached storage. Encrypting data-at-rest can mitigate threats and allow data security, but widespread use of encryption is complicated by inconsistencies and duplication in key management systems supporting encryption environments. An emerging standard, Key Management Interoperability Protocol (KMIP), promises to greatly reduce OPEX and improve compliance visibility, and this session provides a developer's perspective on its implementation.

Learning Objectives:

- How KMIP addresses standardizing communication between encryption systems that need to consume keys and the key management systems that create and manage those keys.
- How KMIP is used with encrypted storage, low-end devices, and certificate distribution
- How enterprises will be able to deploy a single enterprise key management infrastructure to manage keys for all encryption systems in the enterprise that require symmetric keys, asymmetric keys pairs, certificates and other security objects.
- An interactive discussion on KMIP implementation suggestions for storage developers

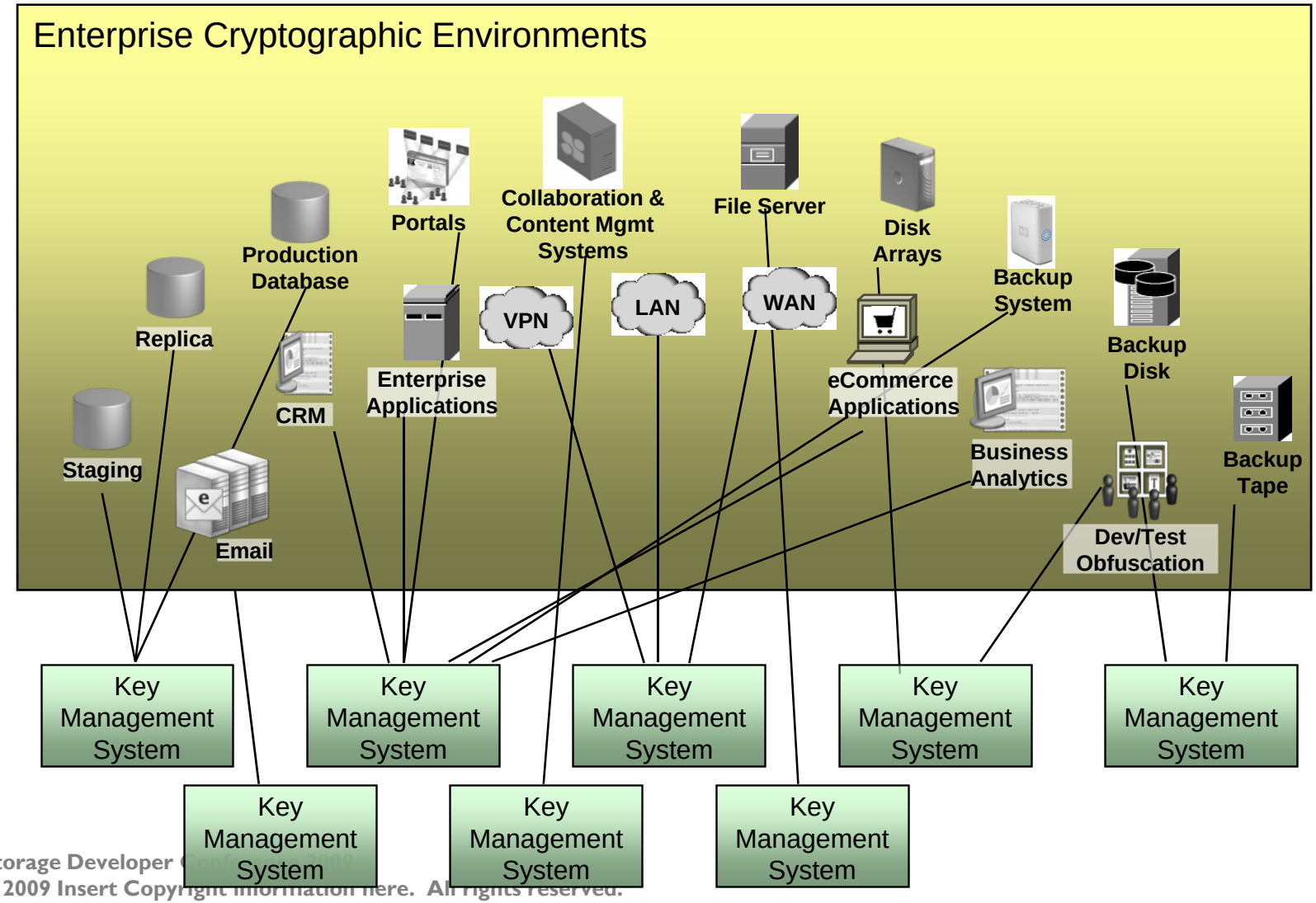
Agenda

- The Need for Interoperable Key Management
- KMIP Overview
- KMIP Specification
- KMIP Use Cases
- Storage specific guidance

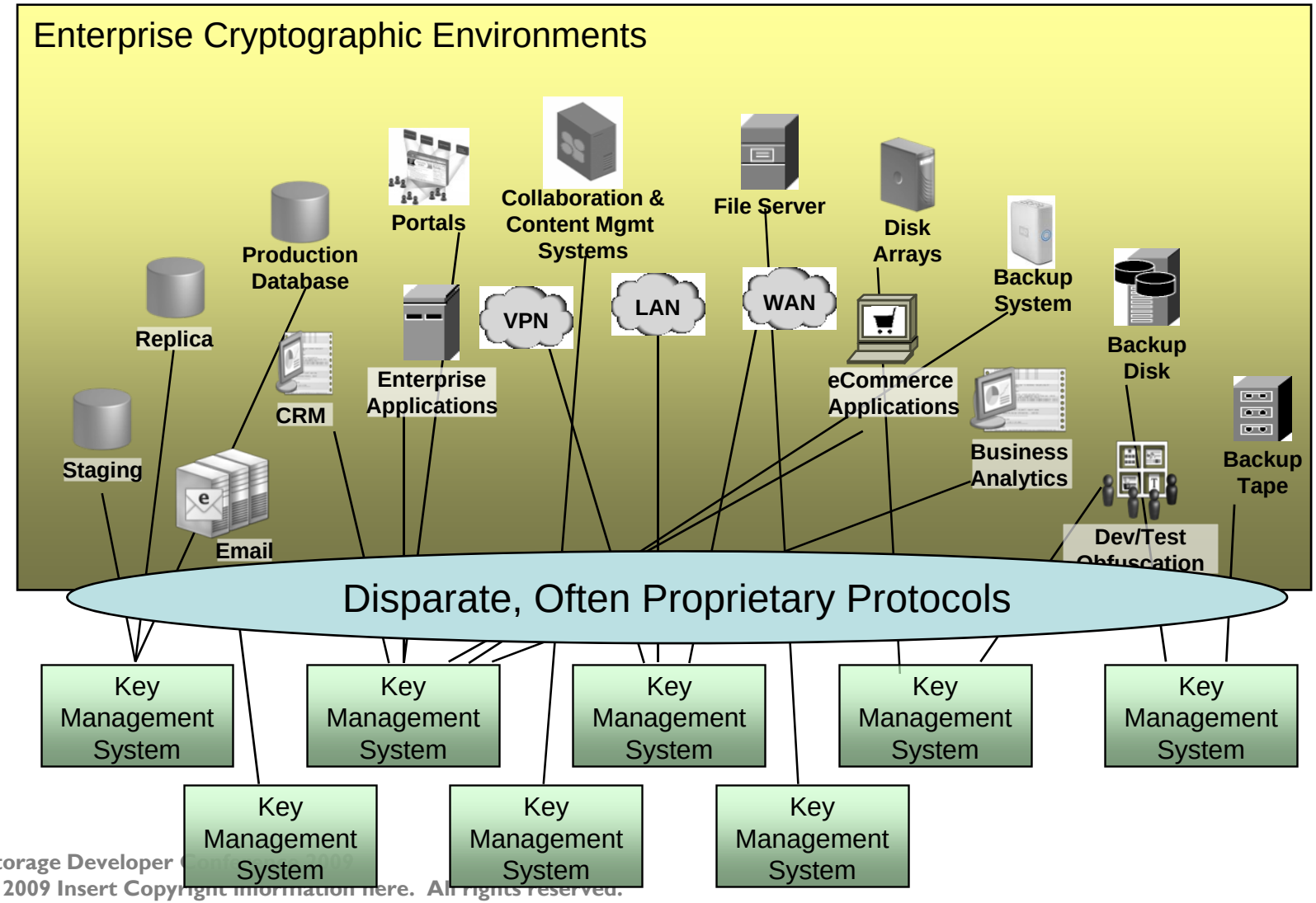
The Need for Interoperable Key Management

- Today's enterprises operate in increasingly complex, multi-vendor environments.
- Enterprises need to deploy better encryption across the enterprise.
- A key hurdle in IT managers deploying encryption is their ability to recover the encrypted data.
- Today, many companies deploy separate encryption systems for different business uses – laptops, storage, databases and applications – resulting in:
  - Cumbersome, often manual efforts to manage encryption keys
  - Increased costs for IT
  - Challenges meeting audit and compliance requirements
  - Lost data

Often, Each Cryptographic Environment Has Its Own Key Management System



Often, Each Cryptographic Environment Has Its Own Protocol



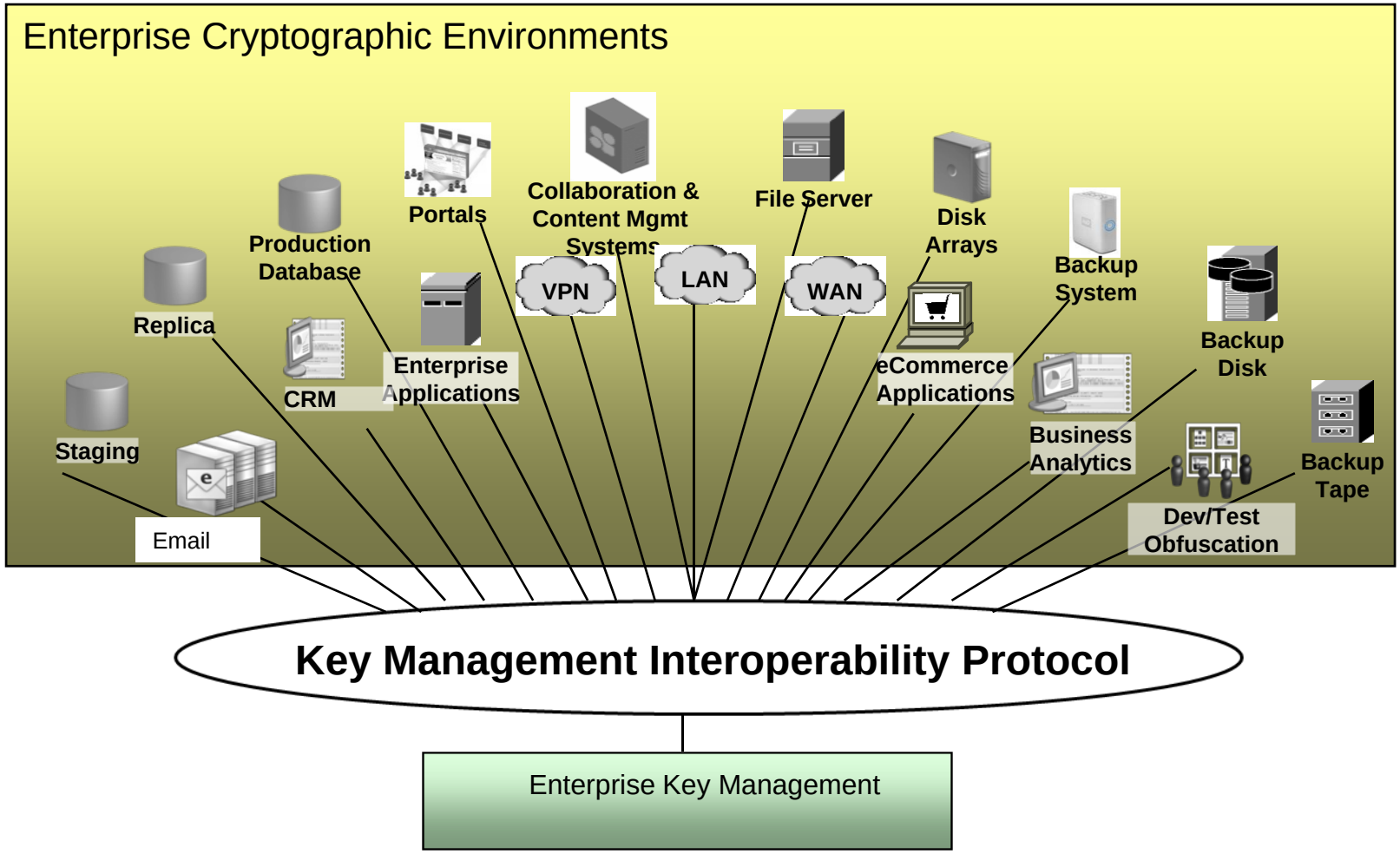
KMIP Overview

What is KMIP

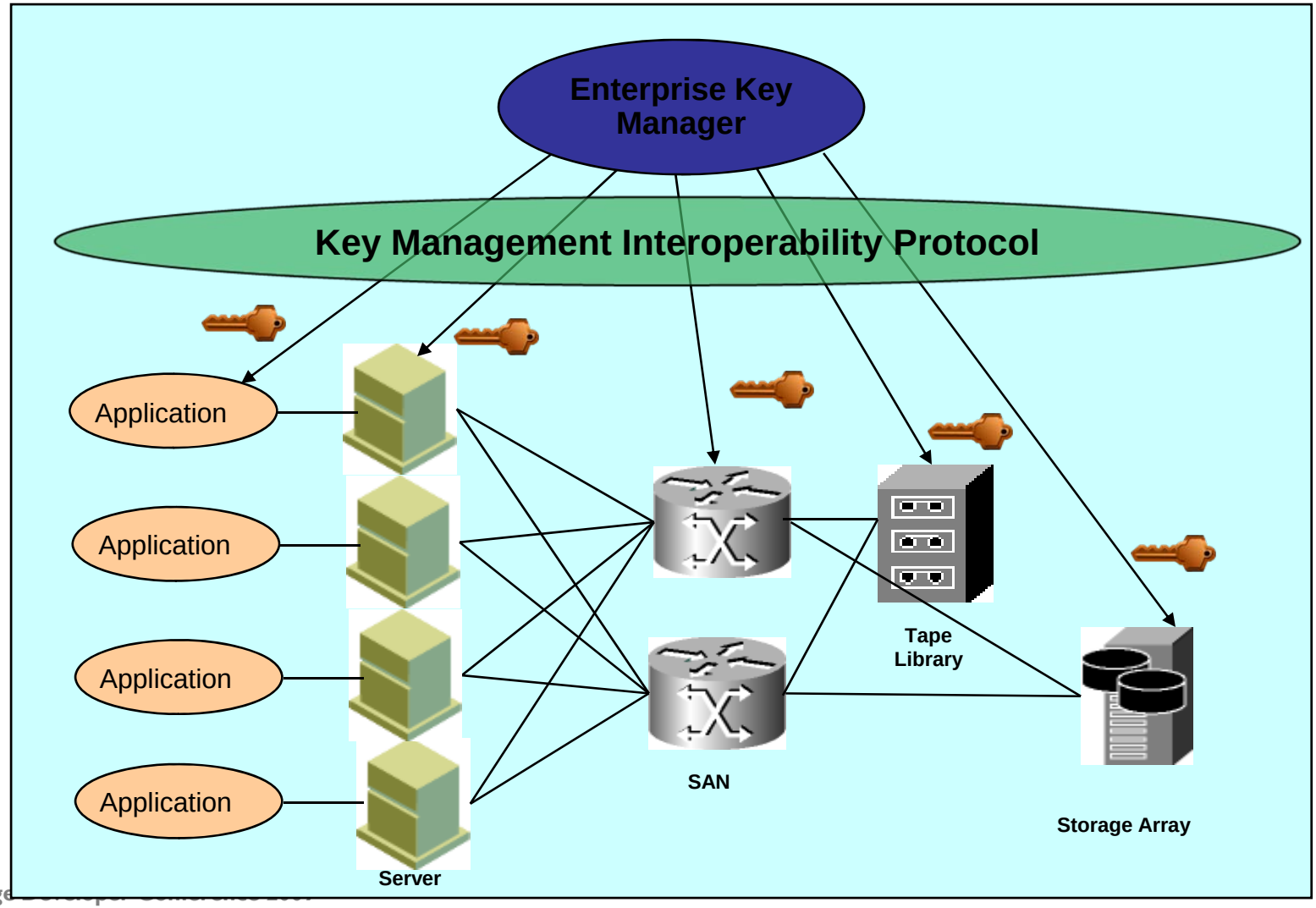
- The Key Management Interoperability Protocol (KMIP) enables key lifecycle management. KMIP supports legacy and new encryption applications, supporting symmetric keys, asymmetric keys, digital certificates, and other "shared secrets." KMIP offers developers templates to simplify the development and use of KMIP-enabled applications.

- KMIP defines the protocol for encryption client and key-management server communication. Supported key-lifecycle operations include generation, submission, retrieval, and deletion of cryptographic keys. Vendors will deliver KMIP-enabled encryption applications that support communication with compatible KMIP key-management servers.

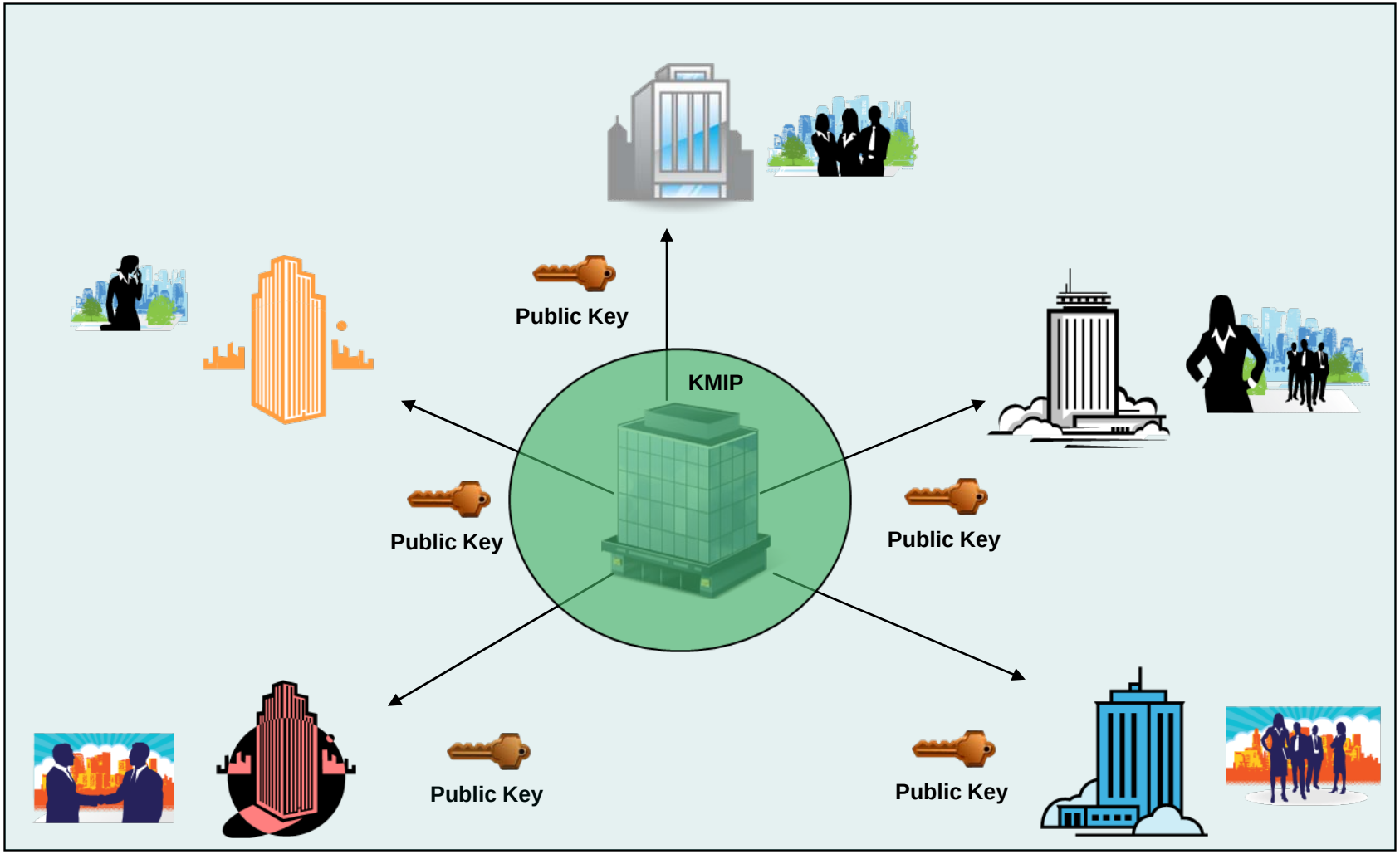
KMIP: Single Protocol Supporting Enterprise Cryptographic Environments



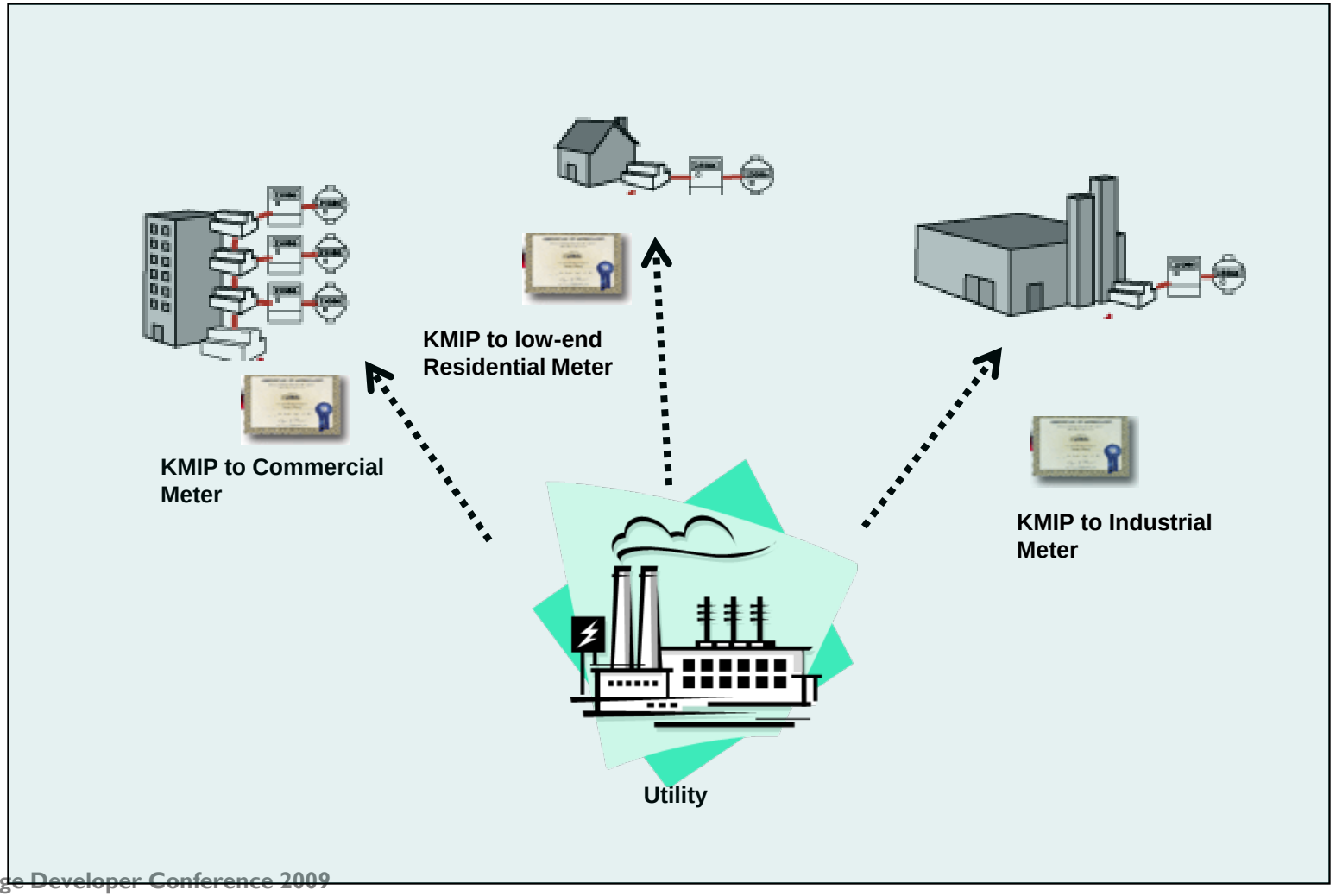
KMIP: Symmetric Encryption Keys



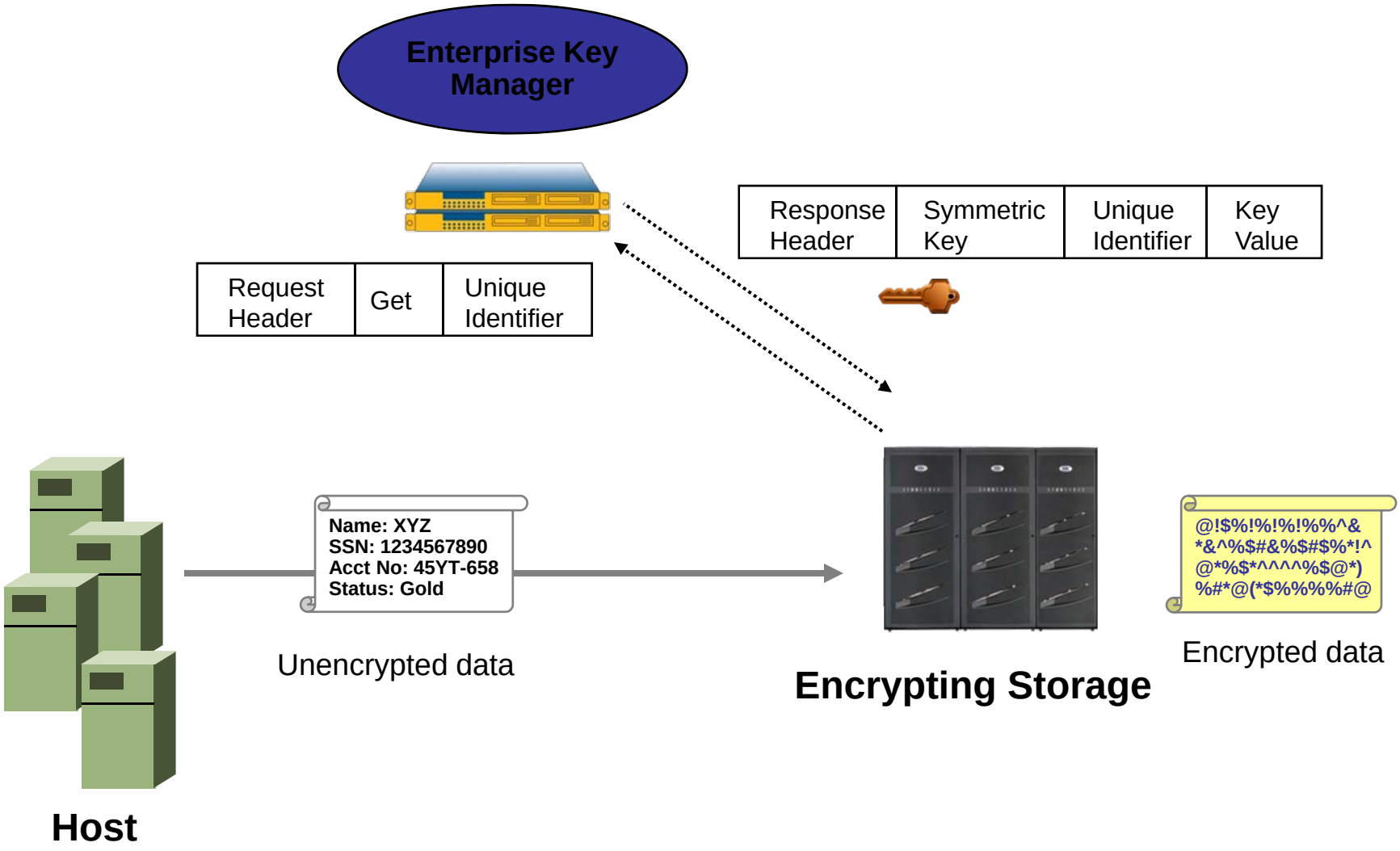
KMIP: Asymmetric Keys



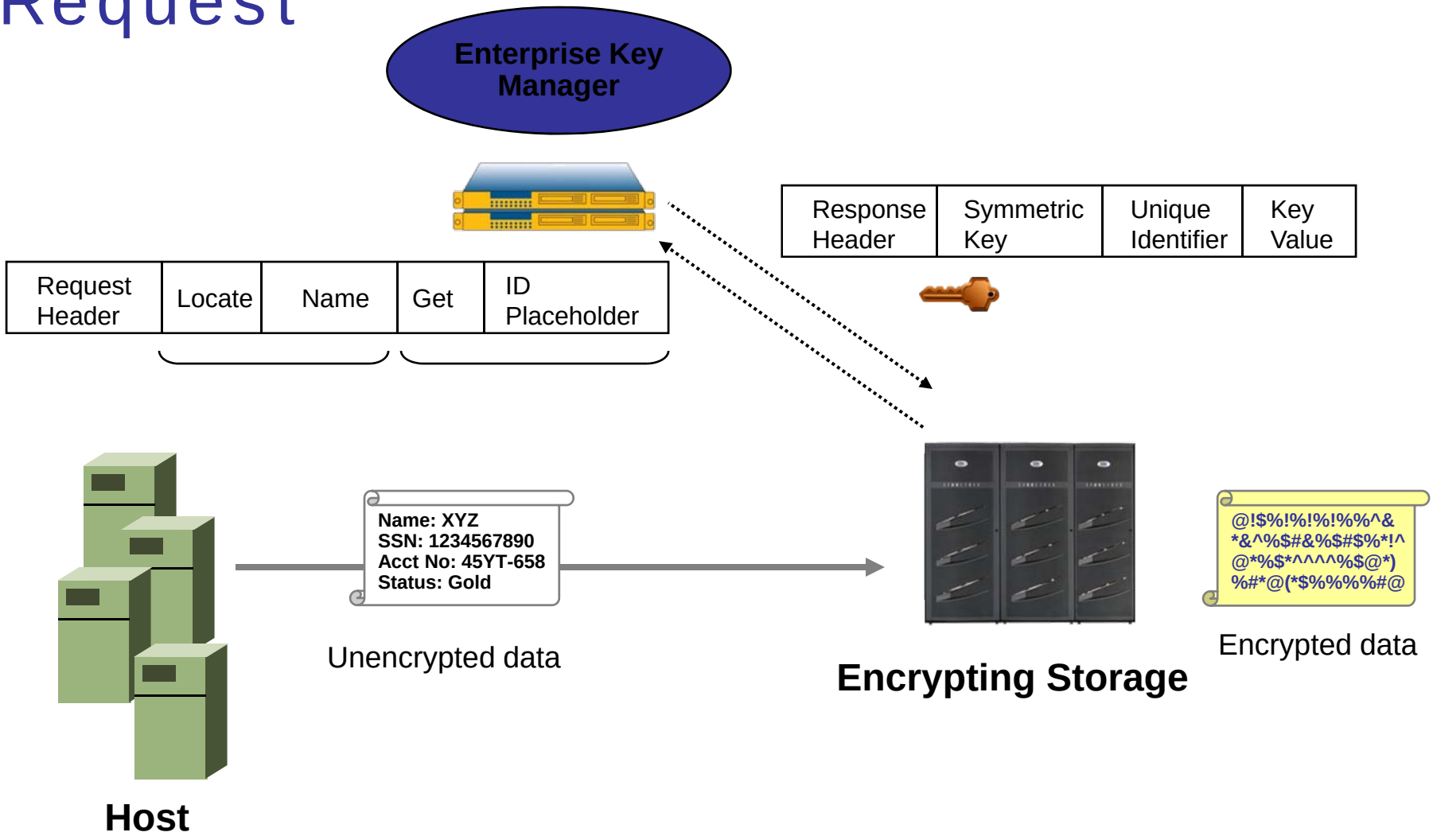
KMIP: Digital Certificates



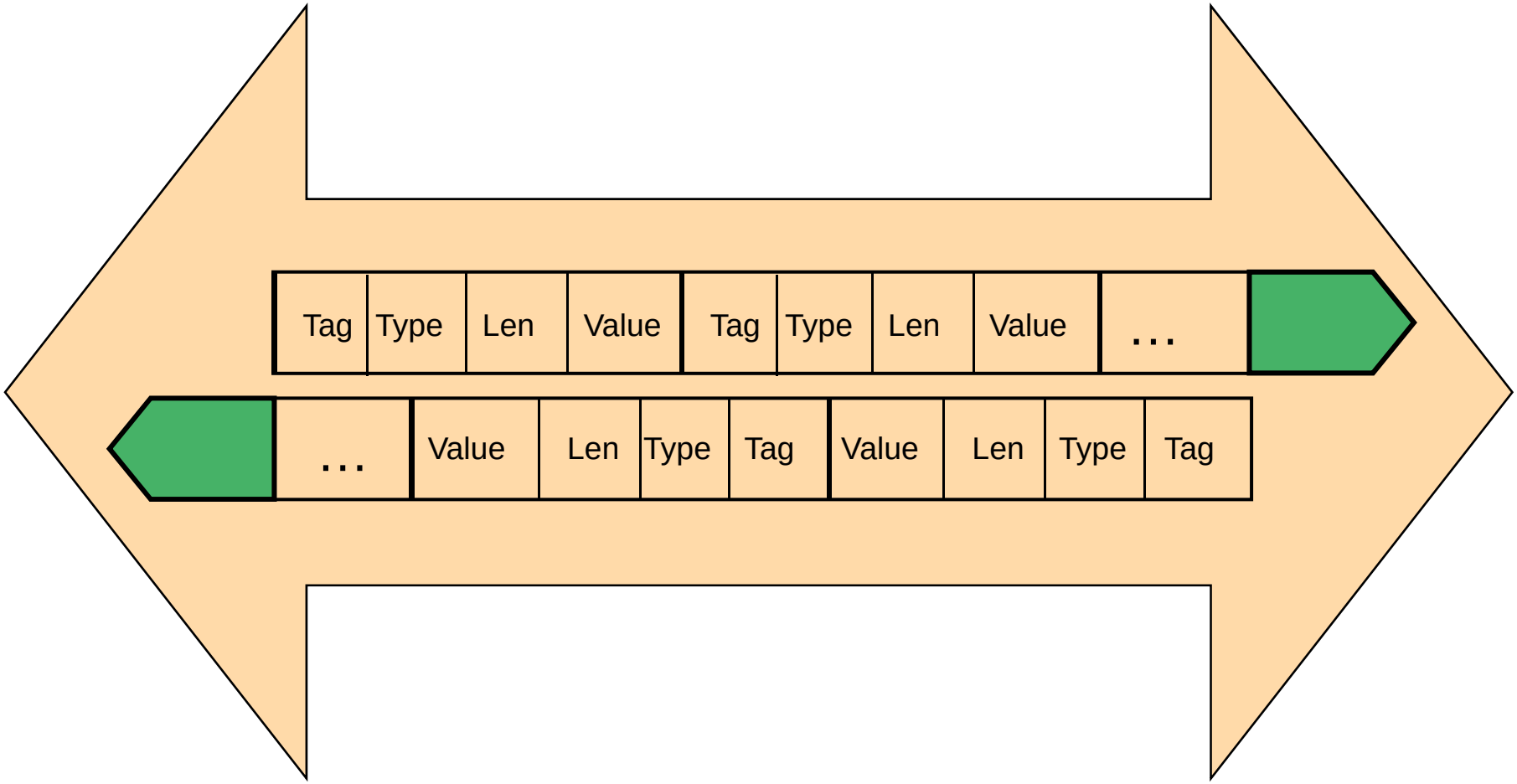
KMIP Request / Response Model



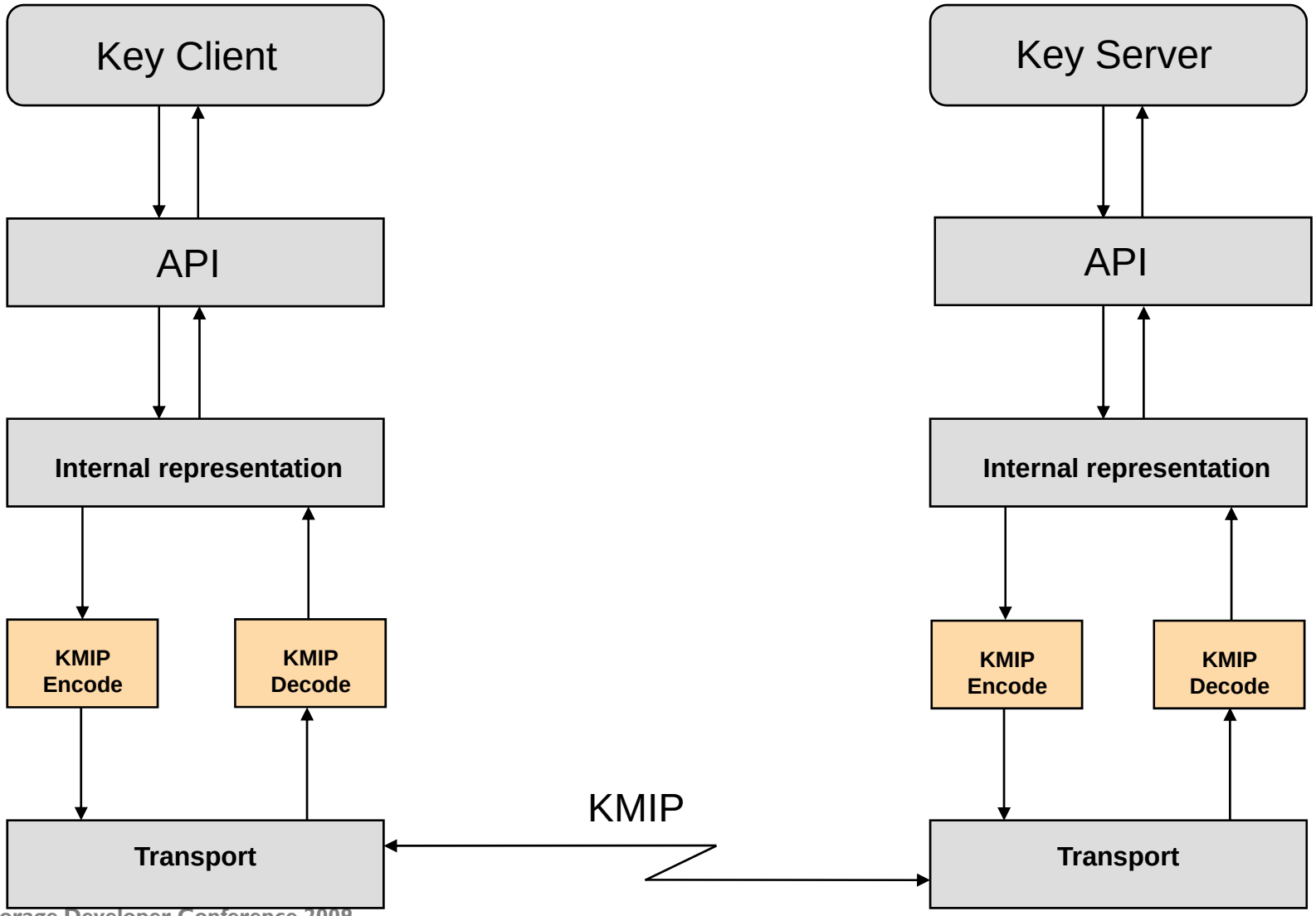
Supporting Multiple Operations per Request



Messages in TTLV Format



Transport-Level Encoding



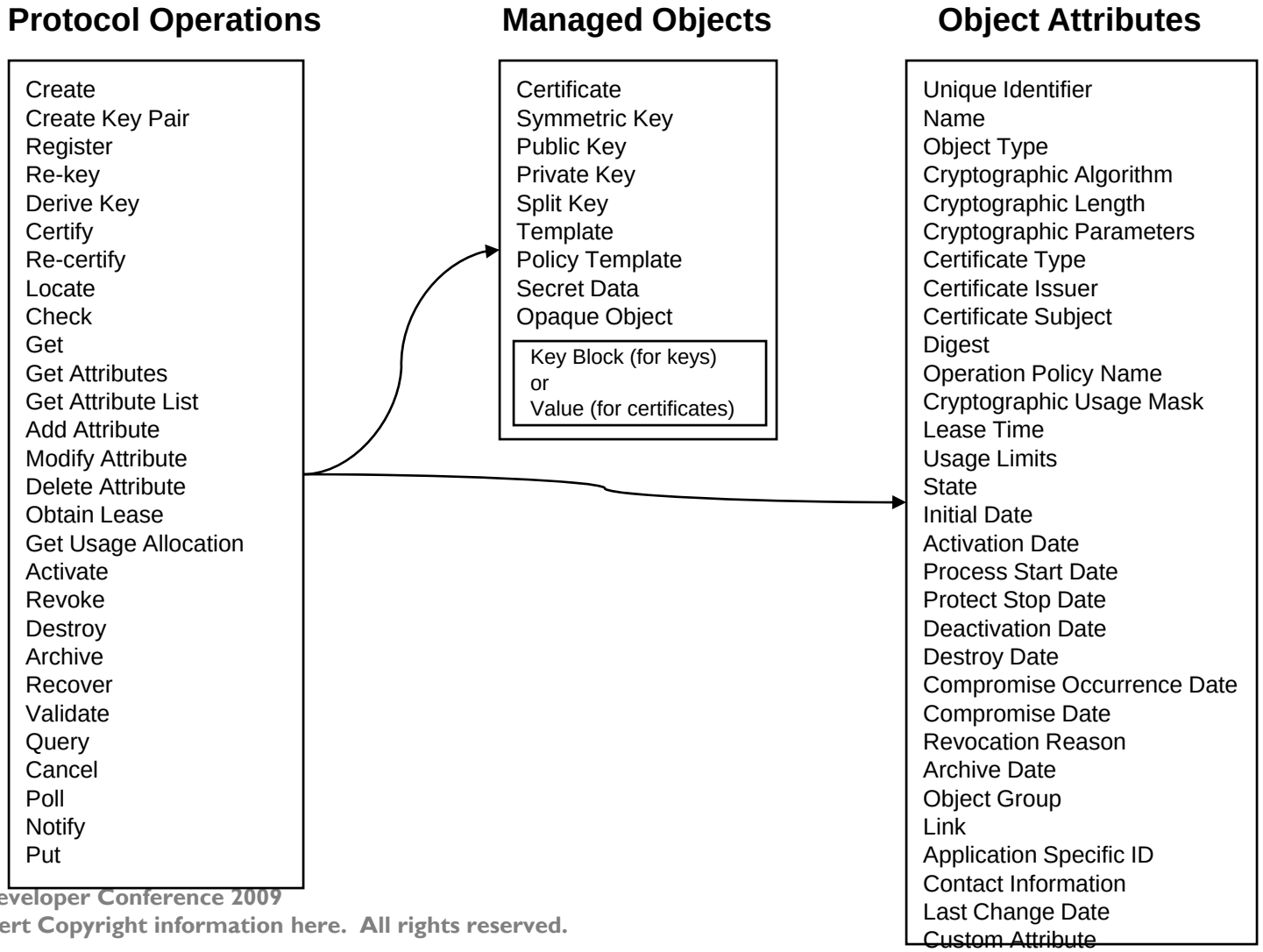
OASIS KMIP Technical Committee

- OASIS (Organization for the Advancement of Structured Information Standards) is a not-for-profit consortium that drives the development, convergence and adoption of open standards for the global information society.
- KMIP Technical Committee chartered in March 2009
  - "The KMIP TC will develop specification(s) for the interoperability of Enterprise Key Management (EKM) services with EKM clients. The specifications will address anticipated customer requirements for key lifecycle management (generation, refresh, distribution, tracking of use, life-cycle policies including states, archive, and destruction), key sharing, and long-term availability of cryptographic objects of all types (public/private keys and certificates, symmetric keys, and other forms of "shared secrets") and related areas."
- KMIP TC IPR mode is Royalty Free on RAND

KMIP Specification

http://xml.coverpages.org/KMIP/KMIP-v0.98-final.pdf

KMIP defines a set of Operations that apply to Managed Objects that consist of Attributes and possibly cryptographic material



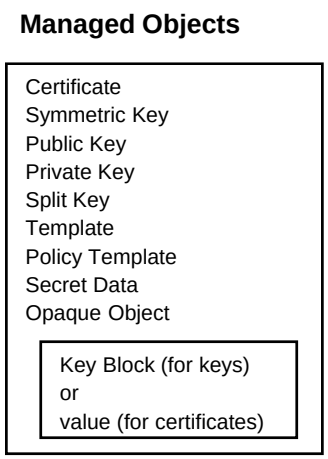
KMIP Base Objects

- Base Objects are:
  - Components of Managed Objects:
    - Attribute, identified by its Attribute Name
    - Key Block, containing the Key Value, either
      - in the clear, either in raw format, or as a transparent structure
      - or "wrapped" using Encrypt, MAC/Sign, or combinations thereof
      - possibly together with some attribute values
  - Elements of protocol messages:
    - Credential, used in protocol messages
  - Parameters of operations:
    - Template-Attribute, containing template names and/or attribute values, used in operations

KMIP Managed Objects

- Managed Cryptographic Objects
  - Certificate, with type and value
  - Symmetric Key, with Key Block
  - Public Key, with Key Block
  - Private Key, with Key Block
  - Split Key, with parts and Key Block
  - Secret Data, with type and Key Block

- Managed Objects
  - Template and Policy Template:
    - Template has a subset of Attributes that indicate **what an object** created from such a template **is**
    - Policy Template has a subset of Attributes that indicate **how an object** created from such a template **can be used**
    - Note that (Policy) Templates have nothing except Attributes: for convenience these Attributes are included in the (Policy) Template structure too.
  - Opaque Object, without Key Block

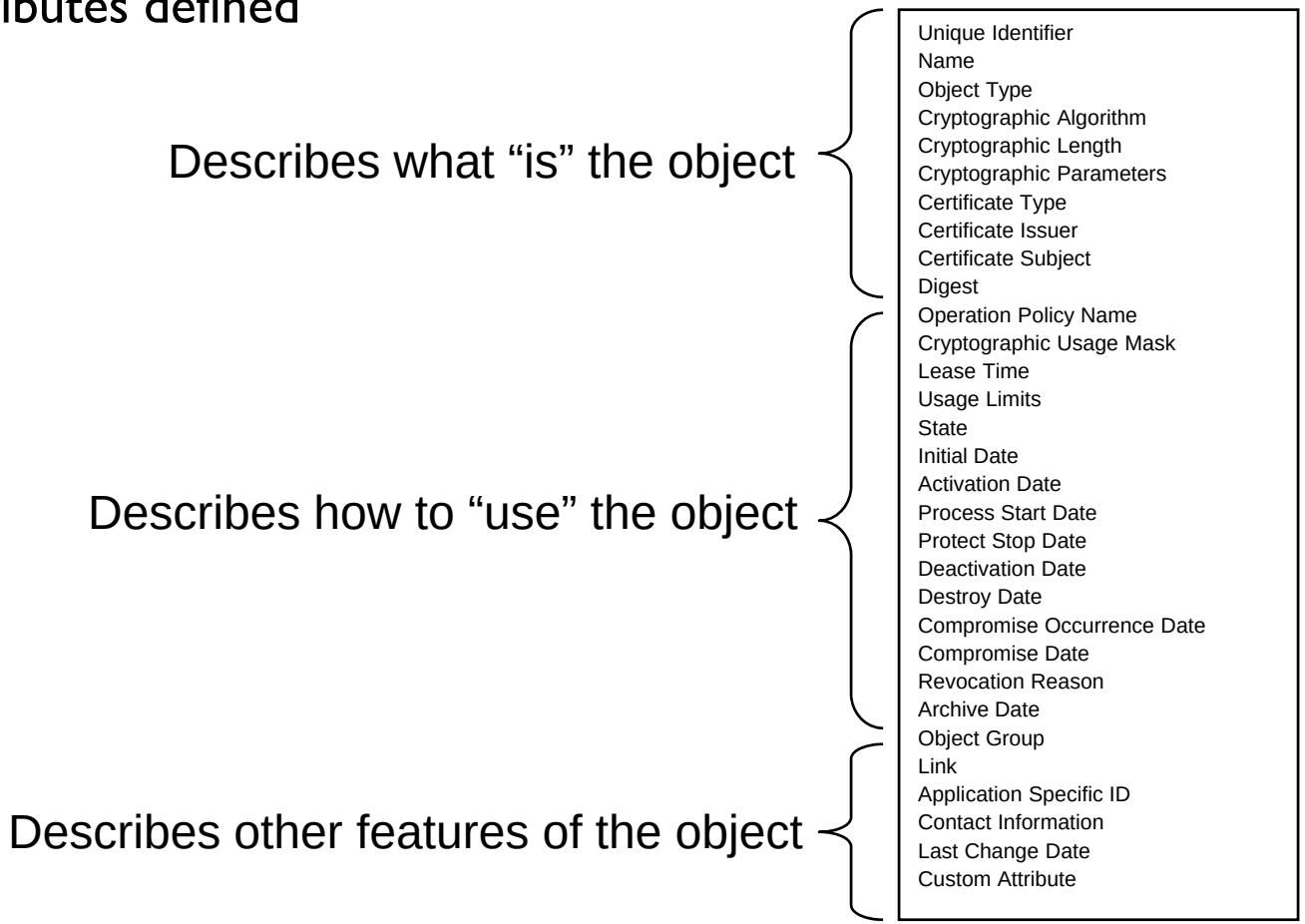


KMIP Attributes

- Attributes contain the "meta data" of a Managed Object
  - Its Unique Identifier, State, etc
  - Attributes can be searched with the Locate operation, as opposed to the content of the Managed Object
- Setting/modifying/deleting Attributes
  - Only some of the Attributes are set with specific values at object creation, depending on the object type
    - For instance, the *Certificate Type* Attribute only exists for Certificate objects
  - Some Attributes are implicitly set by certain operations
    - Certificate Type is implicitly set by Register, Certify, and Re-certify
  - Client can set explicitly some of the Attributes
    - Certificate Type cannot be set by the client
  - Not all Attributes can be added, or subsequently modified or deleted once set
    - Certificate Type cannot added, modified or deleted
  - Some Attributes can have multiple values (or instances) organized with indices
    - For instance, a Symmetric Key object may belong to multiple groups, hence its *Object Group* Attribute will have multiple values

KMIP Attributes cont'd

- 31 Attributes defined



Key Lifecycle States and Transitions

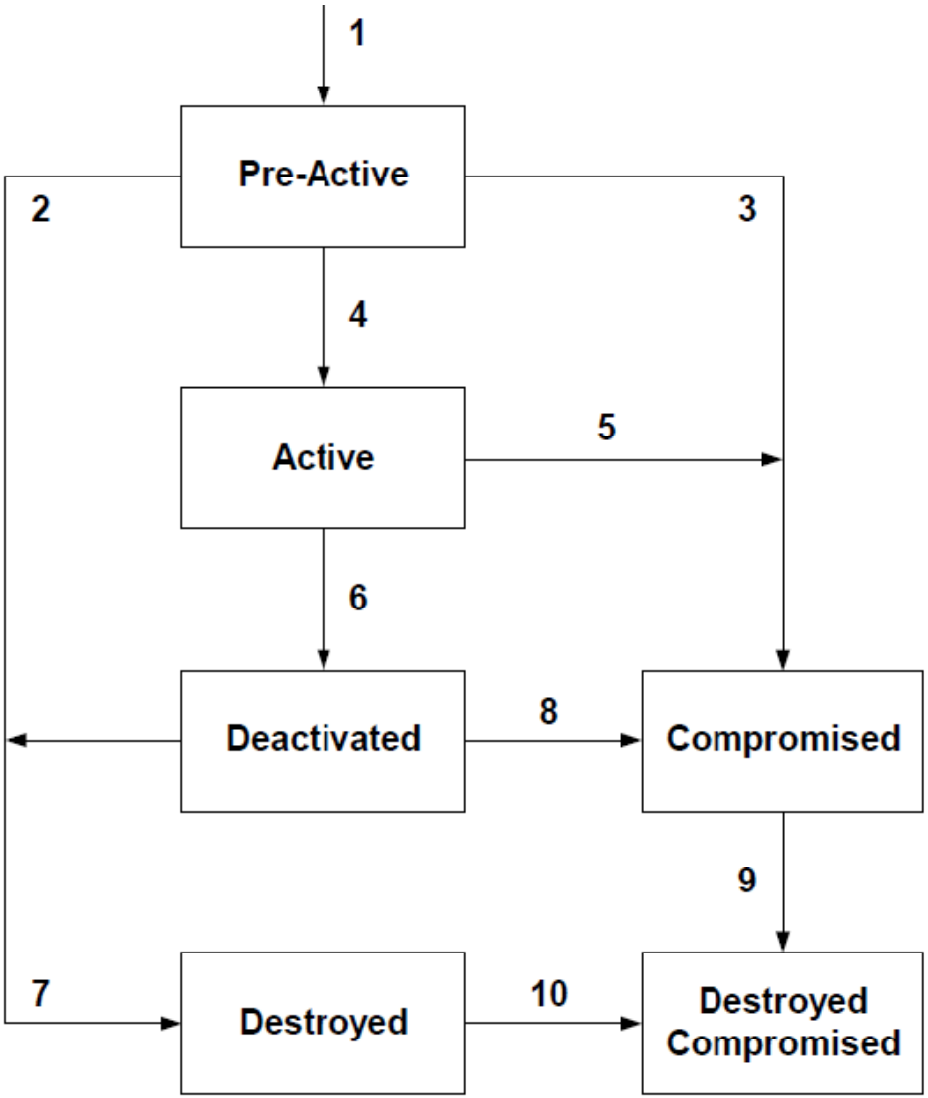
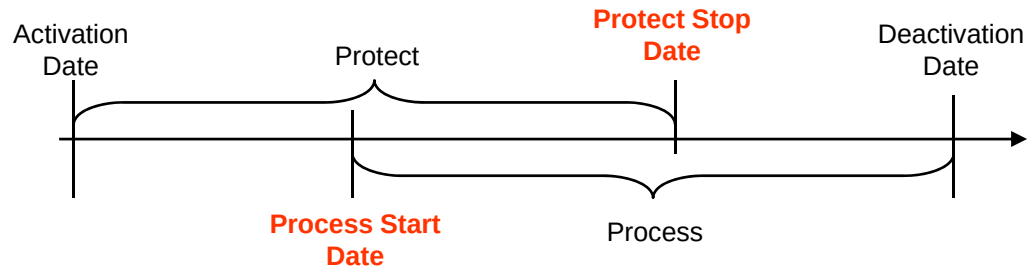
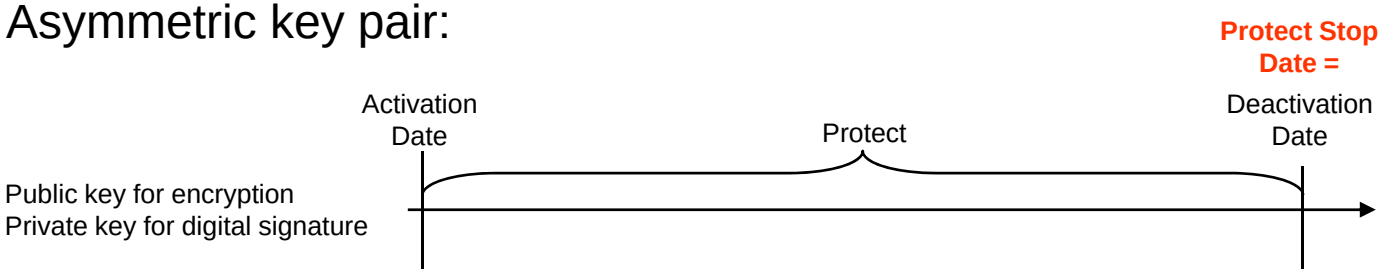


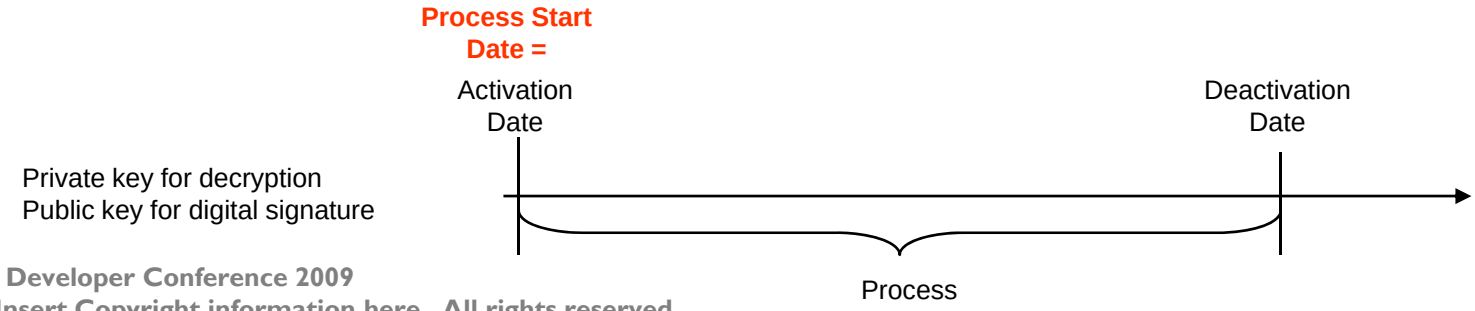
Illustration of the Lifecycle Dates

Symmetric key:



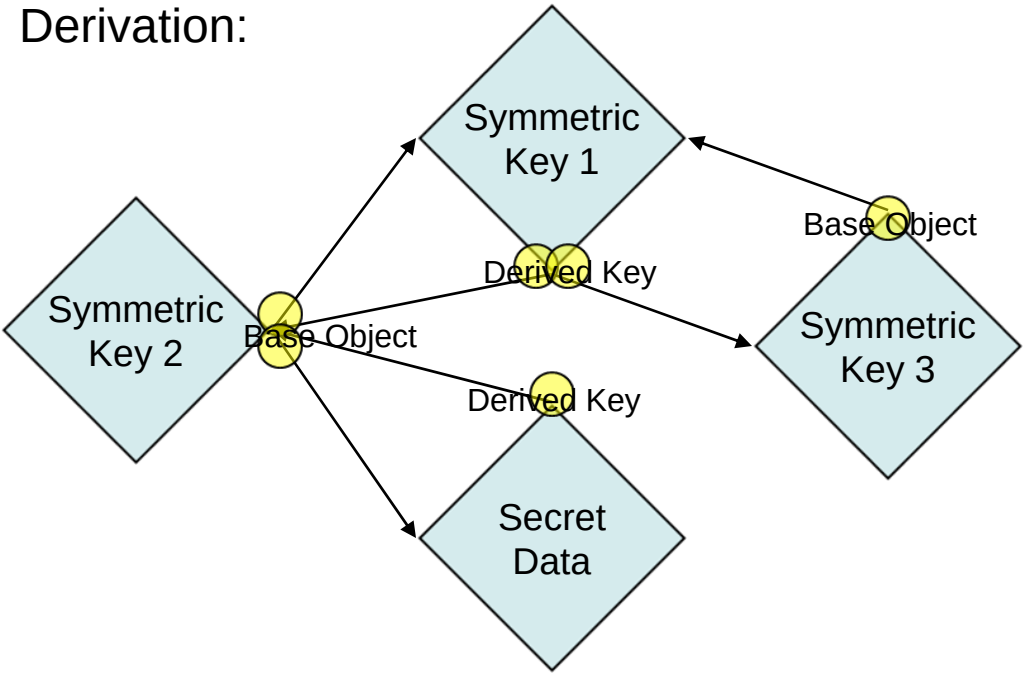
Asymmetric key pair:



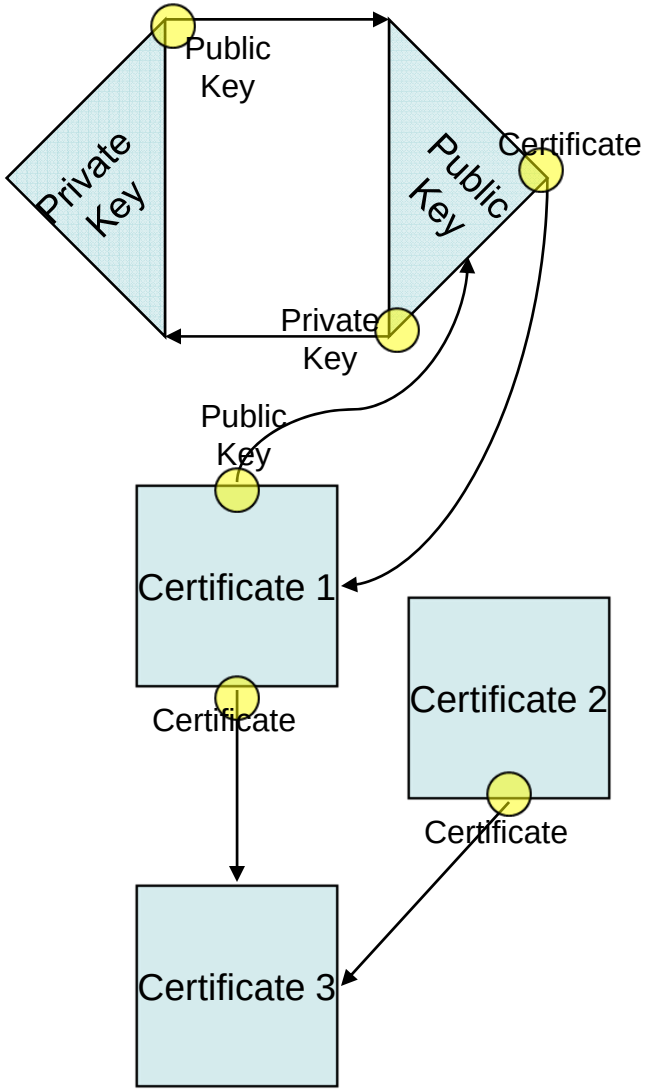


Illustrations of the Link Attribute

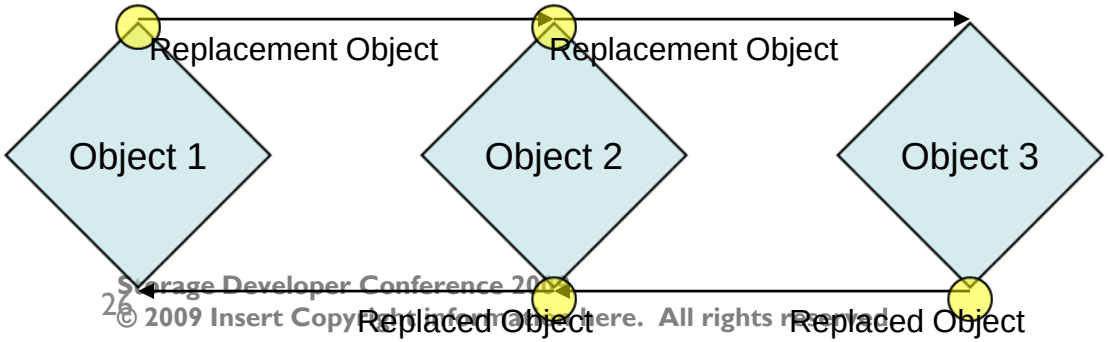
Derivation:



Public/Private/Cert:



Re-key or re-certify:

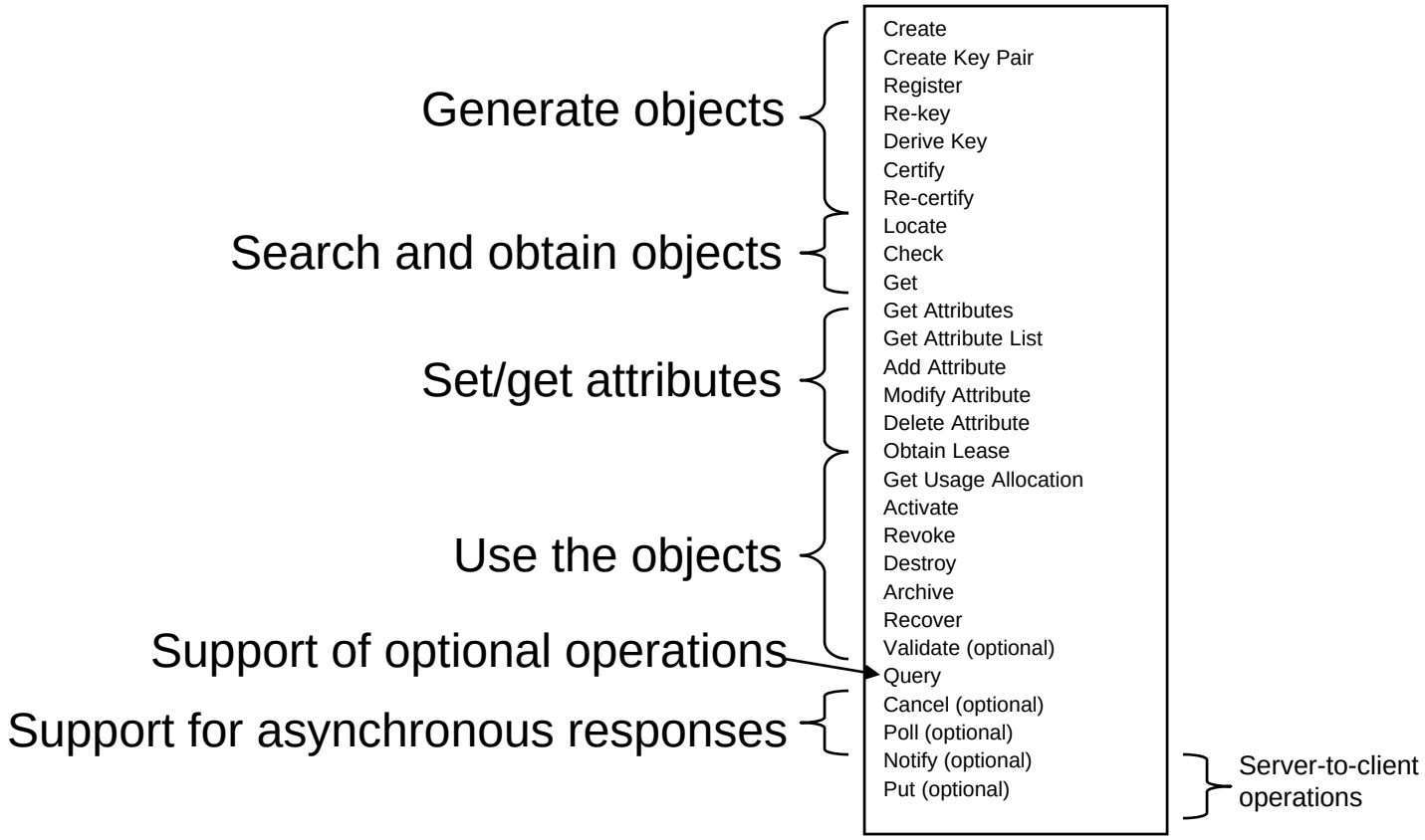


Client-to-server Operations

- Operation consists of a request from client followed by server response
- Multiple operations can be batched in a single request-response pair
  - ID Placeholder can be used to propagate the value of the object's Unique Identifier among operations in the same batch
- Requests may contain Template-Attribute structures with the desired values of certain attributes
- Responses contain the attribute values that have been set differently than as requested by the client

Client-to-server Operations cont'd

- 26 client-to-server operations defined



Server-to-client Operations

- Unsolicited messages from the server to the client with the following operations:
  - Notify operation, used by server to inform client about attribute-value changes
  - Push operation, used by server to provide an object and attributes to client, indicating whether the new object is replacing an existing object or not
  - Batching can be used
  - Support is optional

Message Contents and Format

- Protocol messages consist of requests and responses, each with a header and one or more batch items with operation payloads and message extensions
- Header:
  - Protocol version
  - Maximum response size (optional, in request)
  - Time Stamp (optional in request, required in response)
  - Authentication (optional)
  - Asynchronous Indicator (optional, in request, no support for asynchronous response is default)
  - Asynchronous Correlation Value (optional, in response). Used later on for asynchronous polling
  - Result Status: Success, Pending, Undone, Failure (required, in response)
  - Result Reason (required in response if Failure, optional otherwise)
  - Result Message (optional, in response)
  - Batch Order Option (optional, in request, in-order processing is default). Support at server is optional
  - Batch Error Continuation Option: Undo, Stop, Continue. Stop (optional, in request, Stop is default). Support at server is optional
  - Batch Count
- Batch Item:
  - Operation (enumeration)
  - Unique Message ID (required if more than one batch item in message)
  - Payload (the actual operation request or response)
  - Message Extension (optional, for vendor-specific extensions)

Message Encoding

- Example of TTLV encoding of the *Application Specific ID* Attribute
  - Attribute identified by its name "Application Specific ID"
  - Shows value at index 2

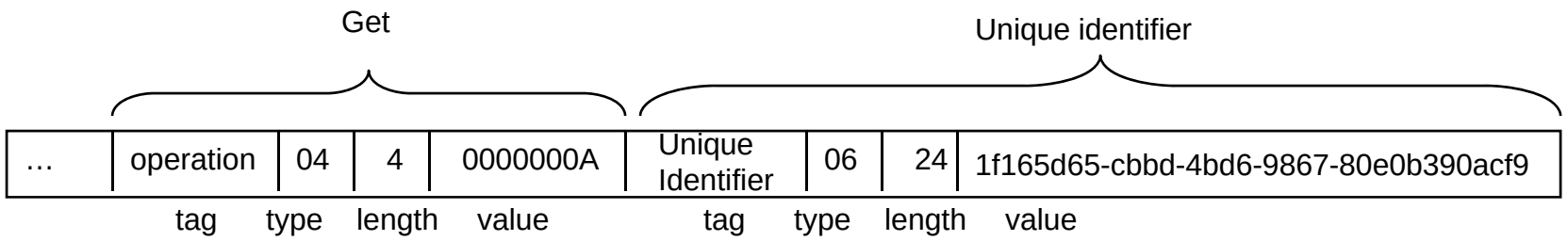
Tag	Type	Length	Value
Attribute	Structure	<varies>	

Tag	Type	Length	Value
Attribute Name	String	<varies>	"Application Specific ID"
Attribute Index	Integer	4	2
Attribute Value	Structure	<varies>	

Tag	Type	Length	Value
App. Name	String	<varies>	"ssl"
App. ID	String	<varies>	"www.example.com"

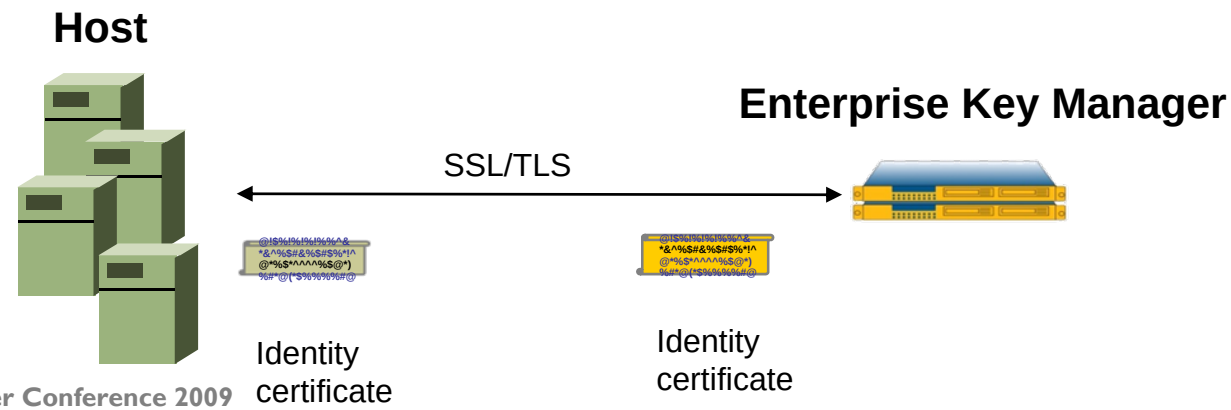
Message Encoding cont'd

- In a TTLV-encoded message, Attributes are identified either by tag value or by their name (see previous slide), depending on the context:
  - When the operation lists the attribute name among the objects part of the request/response (such as Unique Identifier), its tag is used in the encoded message
  - When the operation does not list the attribute name explicitly, but instead includes Template-Attribute (such as in the Create operation) or Attribute (such as in Add Attribute) objects as part of the request/response, its name is used in the encoded message



Authentication

- Authentication is external to the protocol
- All servers should support at least
  - SSL/TLS
  - https
- Authentication message field contains the Credential Base Object
  - Client or server certificate in the case of SSL/TLS or https



KMIP
Use Cases

http://xml.coverpages.org/KMIP/KMIP-UseCases-v0.98-final.pdf

KMIP Use Cases

- Purpose: provide examples of message exchanges for common use cases
- Categories
  - basic functionality (create, get, register, delete of sym. keys and templates)
  - life-cycle support (key states)
  - auditing and reporting
  - key exchange
  - asymmetric keys
  - key roll-over
  - archival
  - vendor-specific message extensions
- Details of the message composition and TTLV encoding (encoded bytes included)

Storage Specific guidance

- Establishment of trust
  - Between the storage device and the key management server
- How and when are keys generated
  - Client, server
- How are keys associated with storage units
  - Implied or explicit and what requirements does that put on the storage device

Establishment of Trust

- Digital certificates used for SSL/TLS session
  - Authentication for session establishment
- How are certificates distributed and installed?
  - During manufacturing
    - Certificate to use to verify device added key management server prior to ship
  - Client could generate and register with server
  - Server can generate and client can request

How and when are keys generated

3 possible scenarios

1. Client generates key when required – sends to the server for backup/recovery
2. Server generates the key when requested by the client
3. Server pre-generates groups of keys, when client requests a key it gets assigned a key from the group

Comparison of the approaches

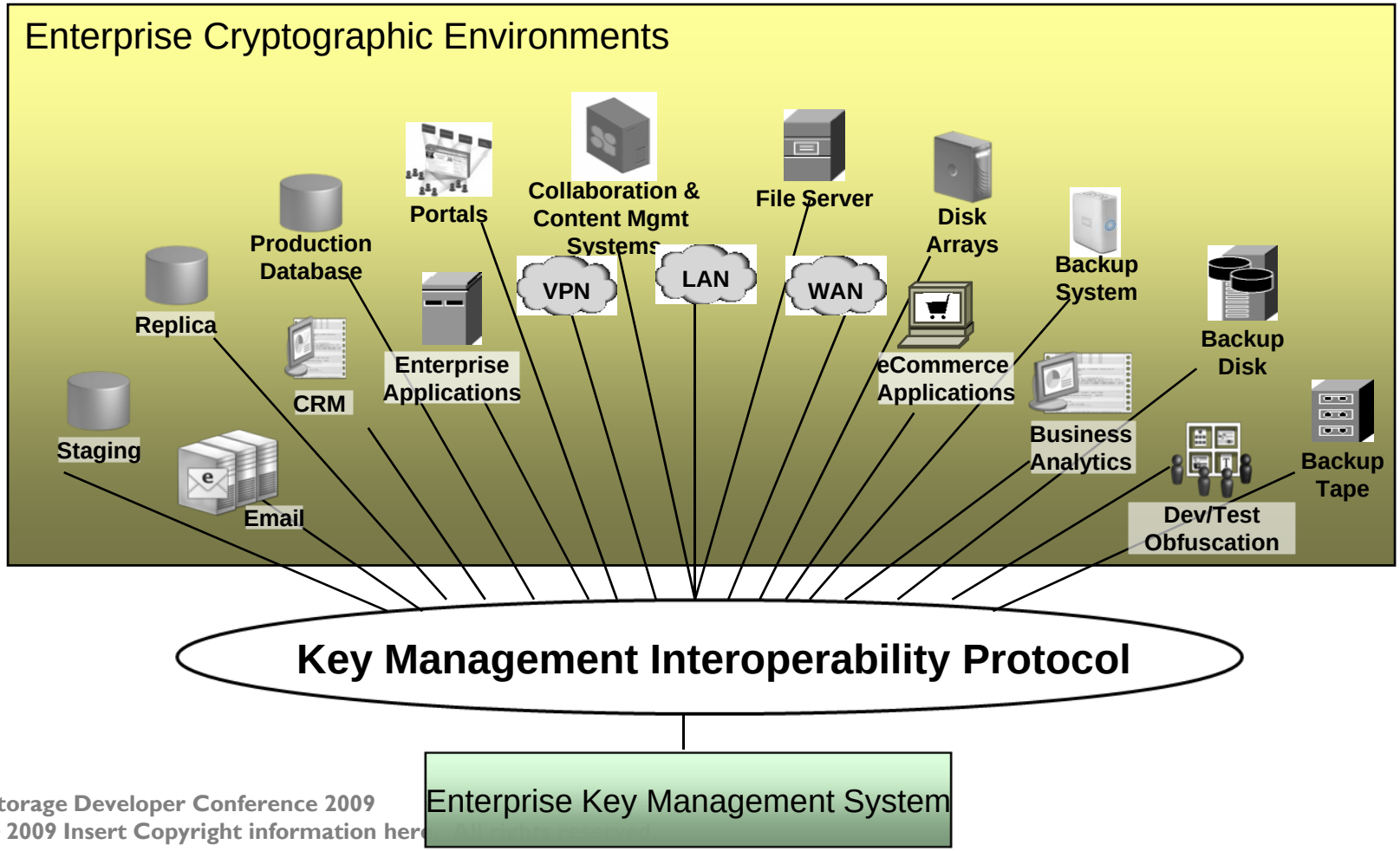
1. Storage device may have special functions for key generation
2. Storage device can take advantage of special functions for key generation in the server
3. Reduces the need for backup and synchronization of keystores

How are keys associated with storage units

- 3 major approaches
  1. Generate a key identifier or label which is stored with the storage unit – used to read previously used key – sent to server and a database lookup occurs
  2. Have a method of identifying the storage unit and associating the key with the storage unit's identifier
  3. Storage an encrypted wrapped copy of the key with the storage unit – have to interact with authentication or the key management server to unwrap the key

Conclusion

KMIP: enabling enterprise key management through standard protocol



Q&A