StrongKey®
Securing the Core

Enterprise Key Management Infrastructure (EKMI)

Arshad Noor

OASIS Adoption Forum, London

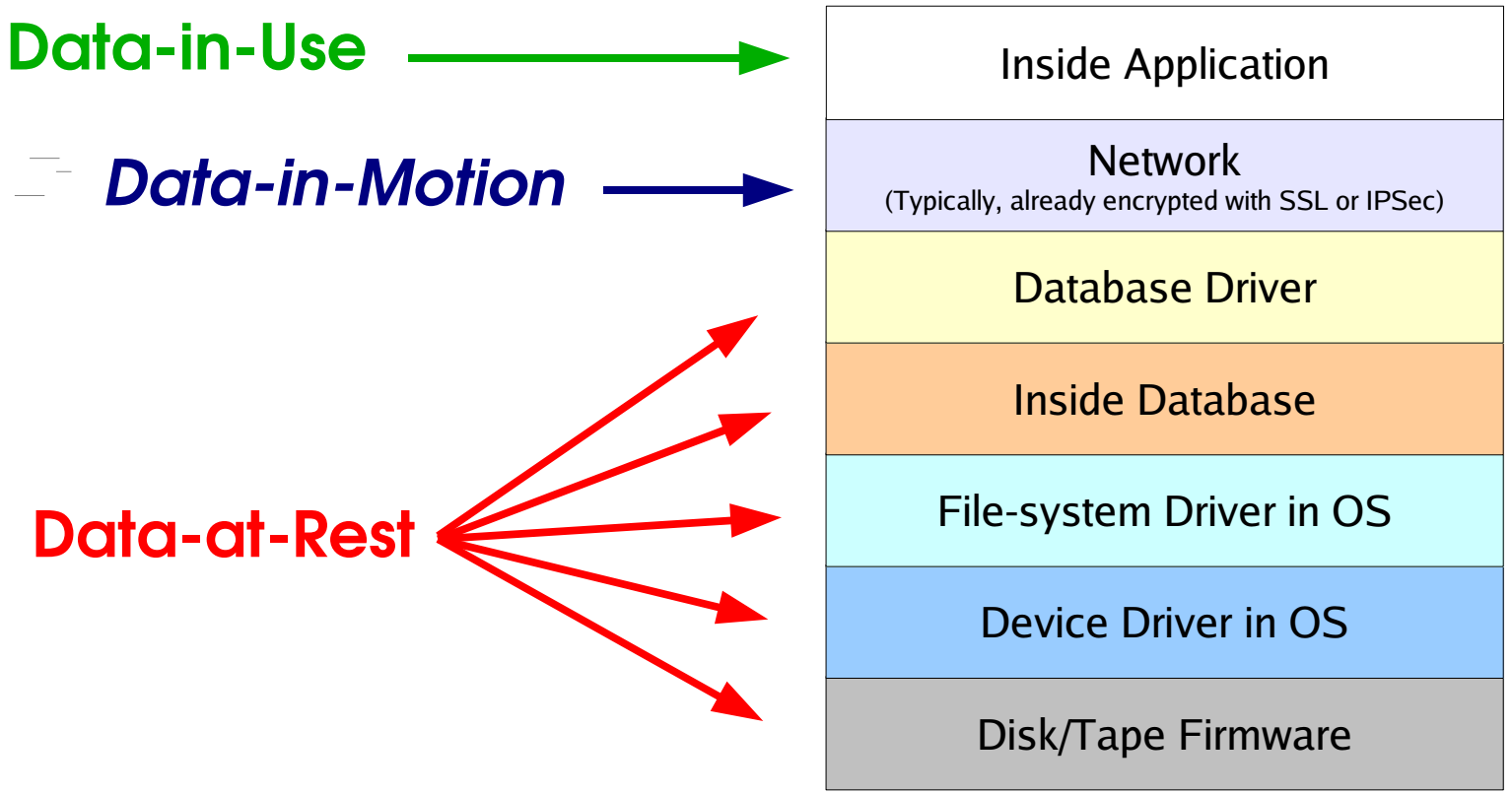
November 27-29, 2006

StrongKey® Securing the Core

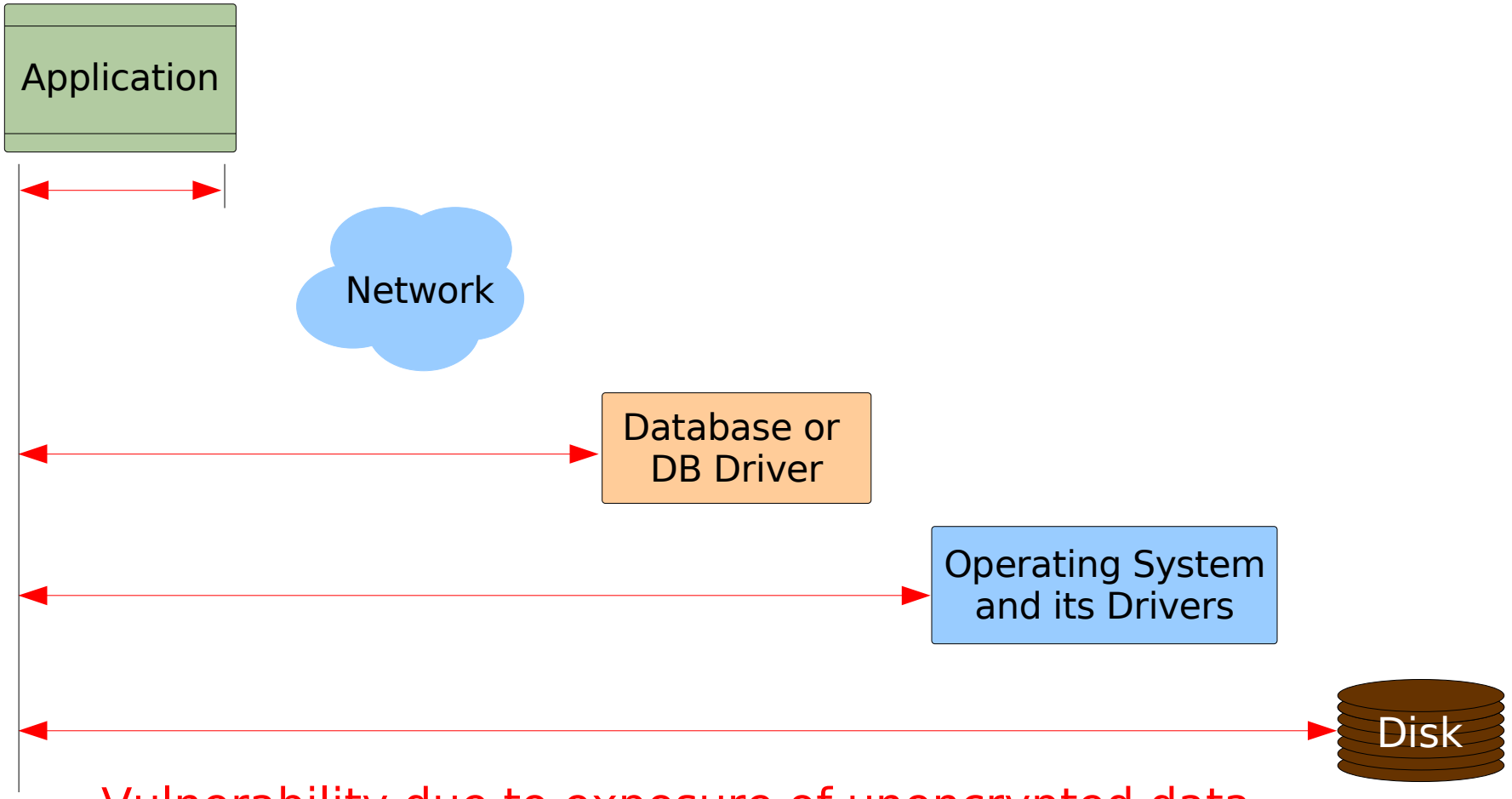
Business Drivers

- Payment Card Industry Data Security Standard
- HIPAA, GLBA, CA (+ 33 US States) SB-1386
- EU Data Protection Directive (Article 16)
- Staying in business - ChoicePoint, Cardsystems
- Avoiding fines - ChoicePoint $15M
- Avoiding negative publicity
  - Intuit, BofA, Wells Fargo, HSBC, Lexis-Nexis, Ralph Lauren, DSW, University of California (LA, SD, Berkeley, Davis), US Veterans Administration, etc.

Encryption Layers



Exposure-Spread



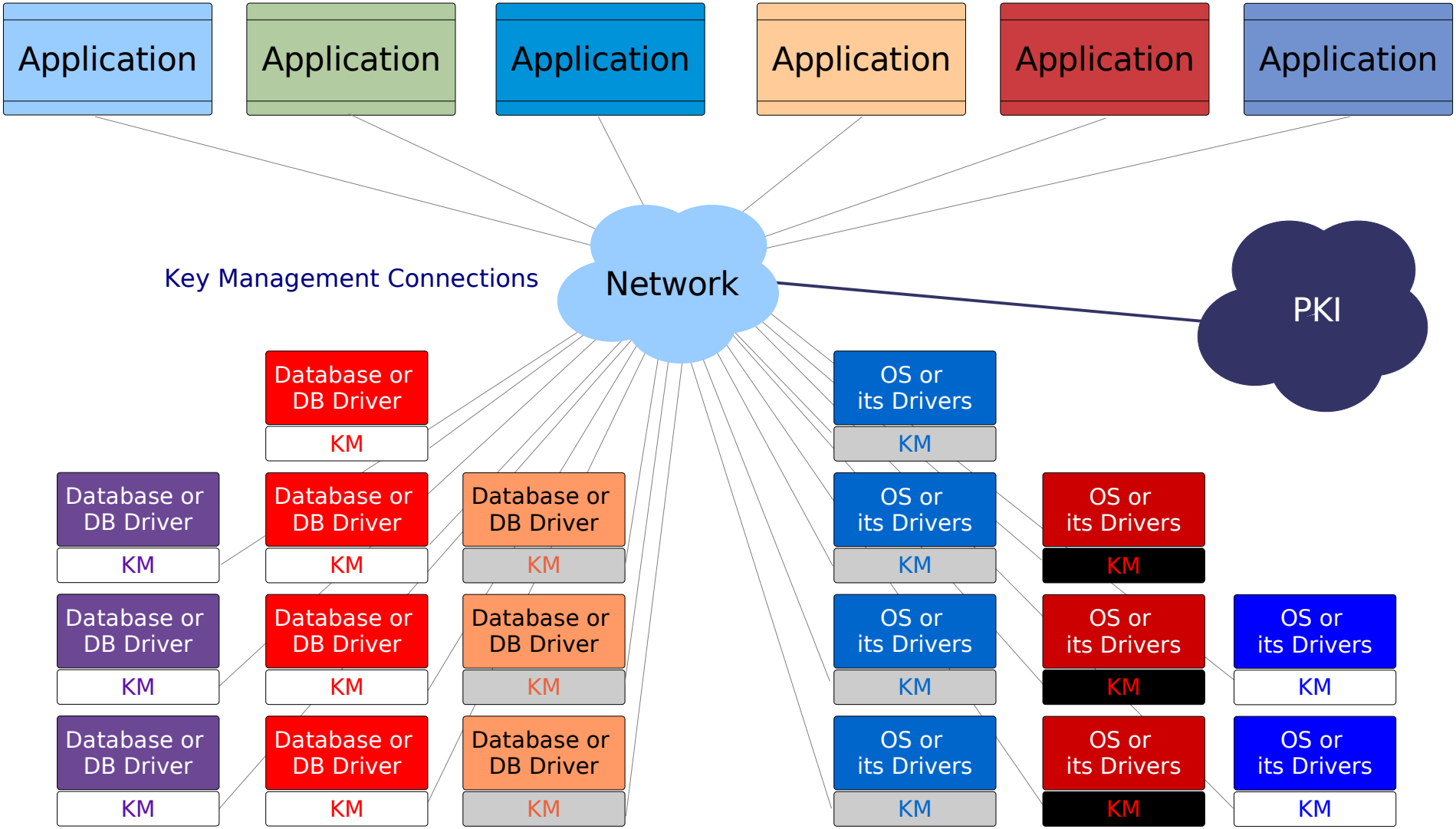
Vulnerability due to exposure of unencrypted data

StrongKey® Securing the Core

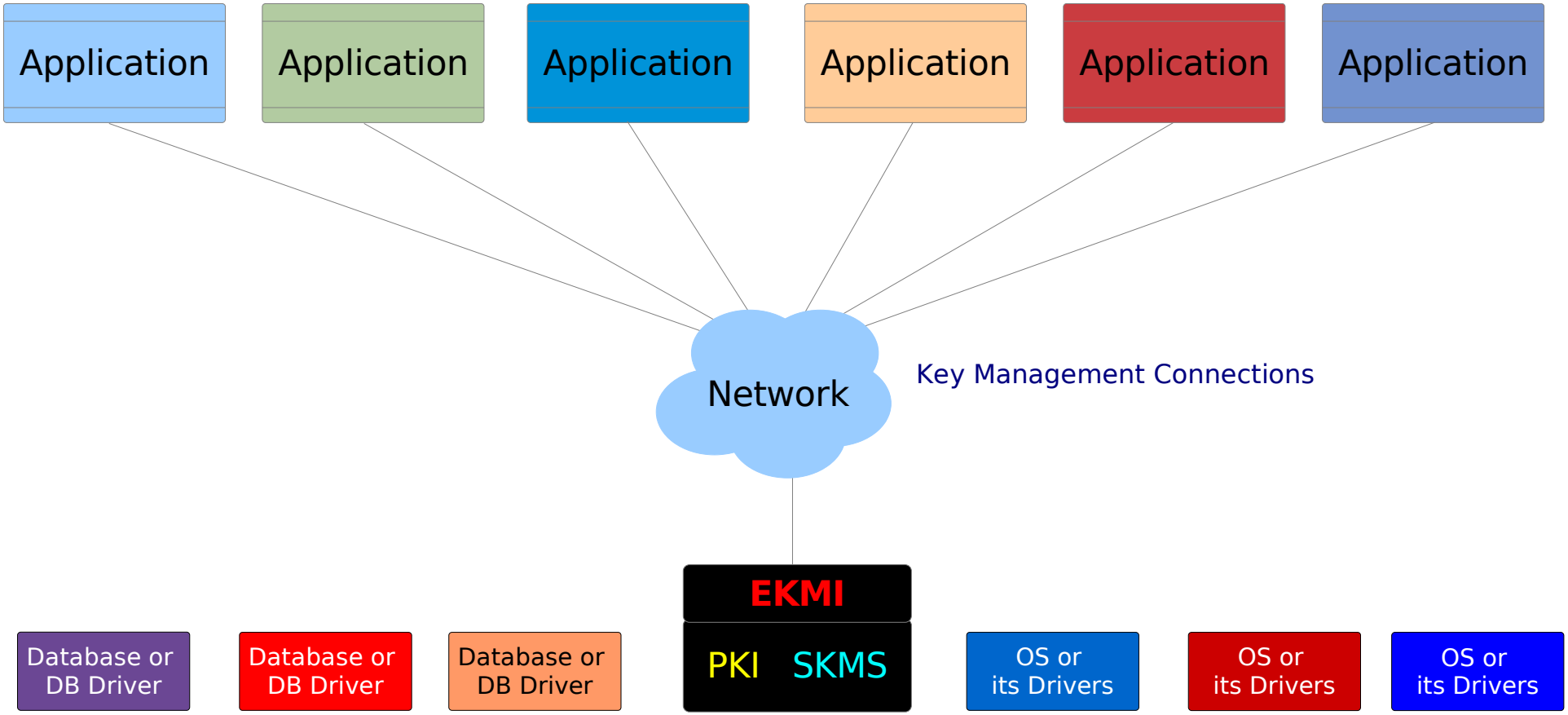
Choices for encryption

Encryption Location	Pros	Cons	Notes
Application	1) Secure everywhere 2) Encrypt only once 3) Database independent 4) OS independent	a) Must modify application b) Must modify DB schema	Needs key management for each application
Database Driver (ODBC, JDBC, etc.)	1) Transparent to application 2) OS independent	a) Secure only past DB driver b) Needs network protection c) Must modify DB schema	Needs key management for each type of DB driver
Inside Database	1) Transparent to application 2) OS independent	a) Secure only inside database b) Needs network protection c) Must modify DB schema	Needs key management for each type of database
Driver for files in OS	Transparent to database and application	a) Secure only inside OS driver b) Needs network protection	Needs key management for each type of OS
Driver for disks in OS	Transparent to database and application	a) Secure only inside OS driver b) Needs network protection c) Needs protection outside disk	Needs key management for each type of OS
Firmware in disks and tape drives	Transparent to database, application and OS	a) Secure only on disk or tape b) Needs protection outside the disk or tape	Needs key management for each type of disk or tape

Key Management Silos...



....or KM Harmony?



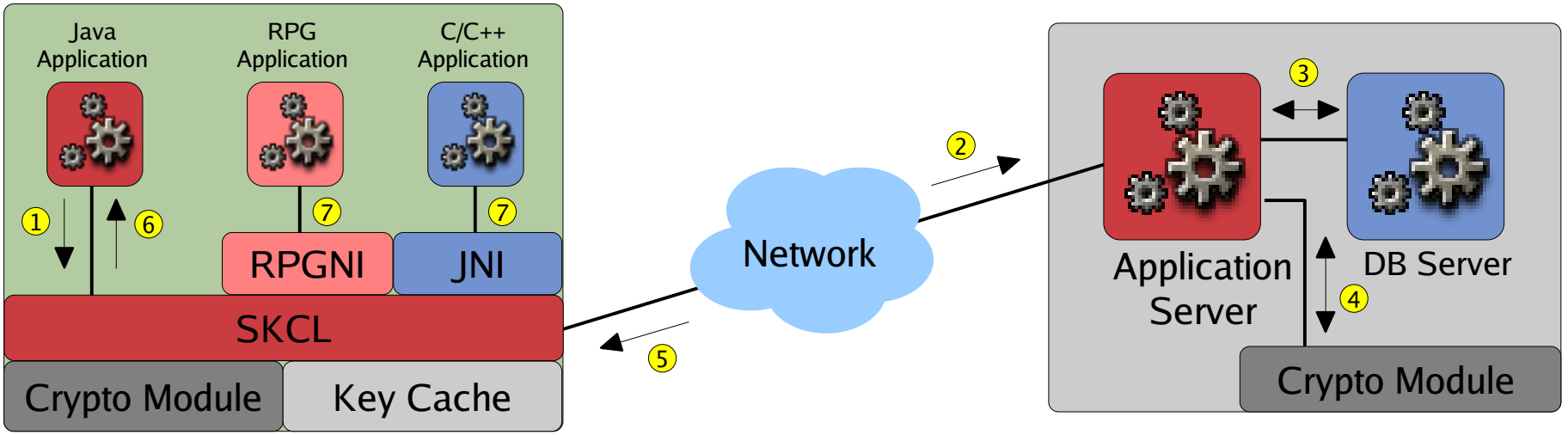
What is an EKMI?

- An **Enterprise Key Management Infrastructure** is "A collection of technology, policies and procedures for managing ***all*** cryptographic keys in the enterprise"
  - A single place to define key-management policy for symmetric and asymmetric (PK) keys
  - Standard protocols for key-management services
  - Operating System, Database & Application independent
  - Scalable for any size enterprise
  - Highly-available (works even during network failures)
  - Extremely secure

EKMI Architecture

- **Public Key Infrastructure**
  - Traditional asymmetric key-management and X.509 certificates
  - For strong-authentication and secure key-transport in symmetric key-management
- **Symmetric Key Management System** (2 parts)
  - **Symmetric Key Services (SKS) server**
    - For key-generation, escrow and recovery
  - **Symmetric Key Client Library (SKCL)**
    - For integrating applications to use the SKS

SKMS – The big picture



Client

Server

1. Client Application makes a request for a symmetric key
2. SKCL makes a digitally signed request to the SKS
3. SKS verifies SKCL request, generates, encrypts, digitally signs & escrows key in DB
4. Cryptographic HSM provides security for RSA Signing & Encryption keys of SKS
5. SKS responds to SKCL with signed and encrypted symmetric key
6. SKCL verifies response, decrypts key and hands it to the Client Application
7. Native (non-Java) applications make requests through Java Native Interface

StrongKey® Securing the Core

Symmetric Key Server

- **SKS** contains all symmetric encryption keys
- Generates, escrows and retrieves keys
- ACLs authorizing access to encryption keys
- Central policy for symmetric keys:
  - Key-size, key-type, key-lifetime, etc.
- Accepts SKSML protocol requests
- Functions like a DNS-server

SKCL

- **Symmetric Key Client Library** communicates with Symmetric Key Server
- Requests (new or existing) symmetric keys
- Caches keys locally, per key-cache policy
- Encrypts & Decrypts data, per key-use policy
  - Currently supports 3DES, AES-128, AES-192 & AES-256
- Makes SKSML requests
- Functions like DNS-client library

SKSML Protocol

- XML-based protocol for
  - Requesting new symmetric key(s) from SKS server, when
    - Encrypting new information, or
    - Rotating symmetric keys for existing ciphertext
  - Requesting existing symmetric key(s) from SKS server for decrypting previously encrypted ciphertext
  - Requesting key-cache-policy information for client
- Why XML and not ASN.1?
- Being submitted to OASIS EKMI-TC for potential standardization on royalty-free basis

Request for a new key

```
<symkey:SymkeyRequest
     xmlns:symkey="http://www.strongauth.com/2006/01/symkey">
    <gkid>0-0</gkid>
</symkey:SymkeyRequest>
```

- Global Key ID
  - Concatenation of "Server ID" - "Key ID"
  - 0-0 is a request for a new symmetric key
- No need for
  - Requester ID or authentication; request is digitally signed inside SOAP header
  - Key information; policy is embedded in the symmetric key

Request for existing key

```
<symkey:SymkeyRequest
    xmlns:symkey="http://www.strongauth.com/2006/01/symkey">
    <gkid>1-234</gkid>
</symkey:SymkeyRequest>
```

- Requester must have authorization for 1-234
- Authorization can be granted based on keys generated based on requests by
  - A single client
  - A group of clients
  - All clients

Symmetric Key Response

```
<xenc:EncryptedKey xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
     xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
     xmlns:symkey="http://www.strongauth.com/2006/01/symkey#">
     <ds:KeyInfo>
          <ds:KeyName>2-2</ds:KeyName>
     </ds:KeyInfo>
     <xenc:CipherData>
          <xenc:CipherValue>CKd4hXZkFGXagTaSPXfOzGgmRVQDik377GZ8hbXfL/
          XxyzynxGRCS1QUusbgSBqXqjq8goRLcb6lrDtyM+q3MeWIv0/BAoZyUJrGGf
          lSJ7OqVwH1vClmhrMfqPmPTWlvBznsPJeG9ICb/kPNFQEFyn8Y8pRnbgc38
          XkMl7uPWAo=</xenc:CipherValue>
     </xenc:CipherData>
     <xenc:EncryptionProperties>
          <xenc:EncryptionProperty>
          <symkey:KeyUsePolicy>
               <symkey:pid>4</symkey:pid>
               <symkey:name>DES-EDE KeyUsePolicy</symkey:name>
               <symkey:start_date>1969-12-31 16:00:00.0</symkey:start_date>
               <symkey:end_date>1969-12-31 16:00:00.0</symkey:end_date>
               <symkey:duration>0</symkey:duration>
               <symkey:tx_allowed>10</symkey:tx_allowed>
               <symkey:policy_type>Tx</symkey:policy_type>
               <symkey:algorithm>
                    http://www.w3.org/2001/04/xmlenc#tripledes-cbc</symkey:algorithm>
               <symkey:keysize>192</symkey:keysize>
               <symkey:status>Active</symkey:status>
          </symkey:KeyUsePolicy>
          </xenc:EncryptionProperty>
     </xenc:EncryptionProperties>
</xenc:EncryptedKey>
```

Symmetric Key Fault

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header>
     ERROR: Other error reported; please review logs for details
     Server error message is: No authorization to request this key: 2-2;
    if you believe this response is an error, please contact your Security Officer
</SOAP-ENV:Header>
<SOAP-ENV:Body xmlns:wsu=
     "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
     wsu:Id="XWSSGID-1154644495295194261602 4">
     <SOAP-ENV:Fault>
     <faultcode xmlns:skf="http://www.strongauth.com/2006/01/symkey#SymkeyFault">
          skf:SymkeyFault
     </faultcode>
     <faultstring>symkey.sks.msg.severe.0085</faultstring>
     <detail>
          <EndEntity>
               <EEID>2</EEID>
               <DN>O=StrongAuth Inc,OU=For DEMO Use Only,CN=POS Register 222,UID=2</DN>
               <Status>Active</Status>
          </EndEntity>
          <Request>
               <RID>3</RID>
               <GKID>2-2</GKID>
               <Timestamp>2006-08-03 15:34:55.0</Timestamp>
               <Disposition>Failed</Disposition>
          </Request>
     </detail>
     </SOAP-ENV:Fault>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

KeyCachePolicy Request

```
<kcpr:KCPRequest xmlns:kcpr=
    "http://www.strongauth.com/2006/01/symkey#KCPRequest"/>
```

- No need for authentication of requester, since request is digitally signed inside SOAP header

KCP Response

```
<kcp:KeyCachePolicy
    xmlns:kcp="http://www.strongauth.com/2006/01/symkey#KeyCachePolicy">
    <kcp:kcpid>2</kcp:kcpid>
    <kcp:name>Active KeyCachePolicy for all devices</kcp:name>
    <kcp:description>A description for KCPID 2</kcp:description>
    <kcp:start_date>2006-08-02 11:49:13.0</kcp:start_date>
    <kcp:end_date>2010-10-12 12:31:35.0</kcp:end_date>
    <kcp:maxnewkeys>0</kcp:maxnewkeys>
    <kcp:maxnewdays>0</kcp:maxnewdays>
    <kcp:maxusedkeys>5</kcp:maxusedkeys>
    <kcp:maxuseddays>60</kcp:maxuseddays>
    <kcp:usefirst>Cache</kcp:usefirst>
    <kcp:status>Active</kcp:status>
</kcp:KeyCachePolicy>
```

- Like everything else, the response is digitally signed by the server inside the SOAP response

Security

- Symmetric keys are encrypted with SKS server's RSA public-key for secure storage
- Client requests are digitally signed (RSA)
- Server responses are digitally signed (RSA) and encrypted (RSA)
- **All** database records are digitally signed (RSA) when stored, and verified when accessed – including history logs – for message integrity

Open

- SKS server and SKCL are open-source (LGPL)
- Client-Server protocol will go through standards process
- Java-based J2EE application; currently runs on Windows, Linux, Solaris, OS/400
- Relational DB for storage; MySQL, Oracle, DB2, SQL Server

Steps to building an EKMI

1. Establish a PKI (or procure managed service)
2. Build SKMS (or procure managed service)
3. Train developers for SKCL integration
4. Integrate application(s) with SKCL
5. Deploy modified application and SKCL
6. Issue digital certificates to clients and servers
7. Configure encryption policies
8. Turn service on

Future of EKMI

- PKI + SKMS = EKMI
- Managing two infrastructures distinctly until convergence
- Reconciling ASN and XML
  - Will the complexity of ASN slow down convergence of the two key management infrastructures?
- Simplifying key-management
  - Encryption is here to stay and will become pervasive
  - Future of information management depends on maintaining information integrity and the infrastructure on which it works

Conclusion

- Questions?
- Contact Information
  - www.strongauth.com
  - arshad.noor@strongauth.com
  - (408) 331-2001 Voice
  - (408) 515-8557 Mobile