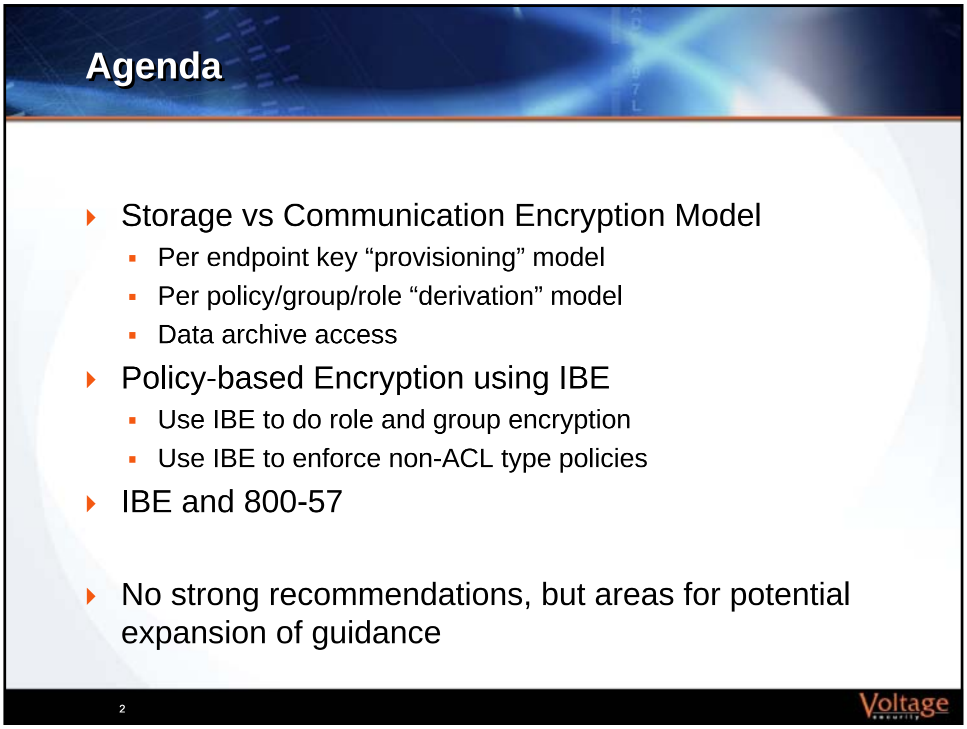
Key Management for Stored Data Requirements and Architectures

Terence Spies
CTO
Voltage Security, Inc.



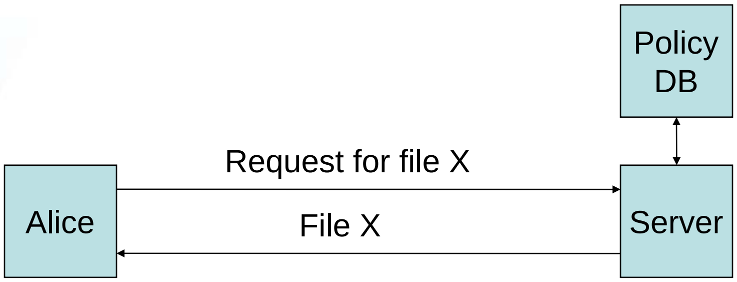
Agenda

- Storage vs Communication Encryption Model
  - Per endpoint key “provisioning” model
  - Per policy/group/role “derivation” model
  - Data archive access
- Policy-based Encryption using IBE
  - Use IBE to do role and group encryption
  - Use IBE to enforce non-ACL type policies
- IBE and 800-57
- No strong recommendations, but areas for potential expansion of guidance

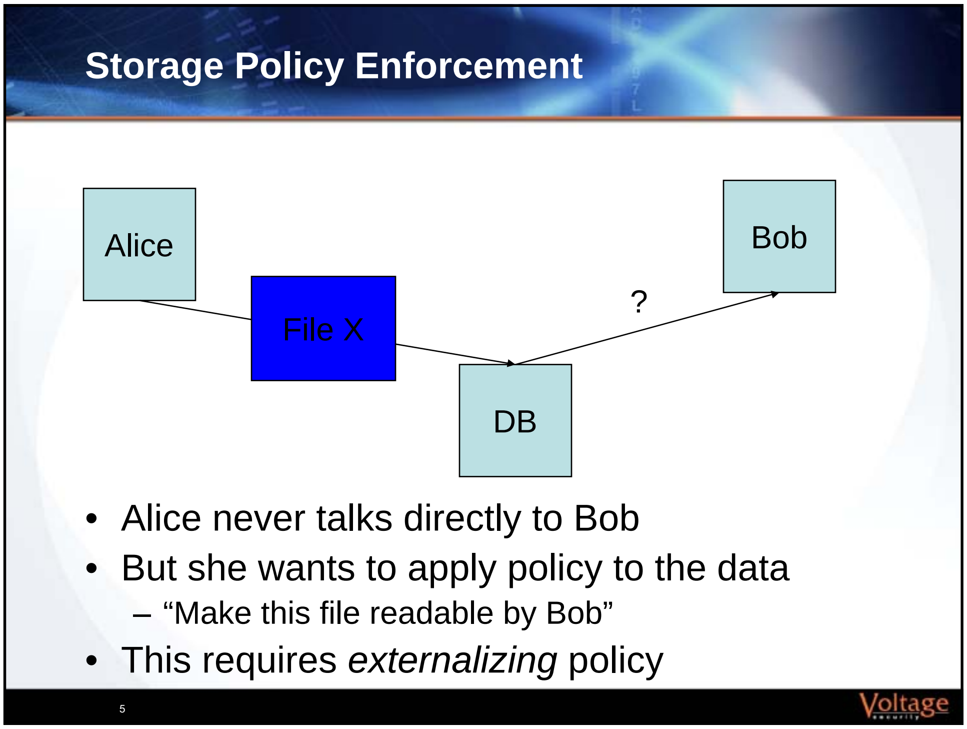
Communication vs Storage

- Communication Mode
  - Protocols independent of application
  - Policy and authentication enforced at connect time
    - Under TLS, check server name immediately
- Storage Mode
  - Protocols and applications interdependent
  - Gap in time between policy creation and enforcement
    - Encrypt to user X, authentication happens before or after
    - Groups and roles complicate this

Communications Policy Enforcement



- Alice connects to server
  - Establishes secure authenticated channel
- Server checks policies
  - Decides whether Alice is authorized



Storage Policy Enforcement

Alice

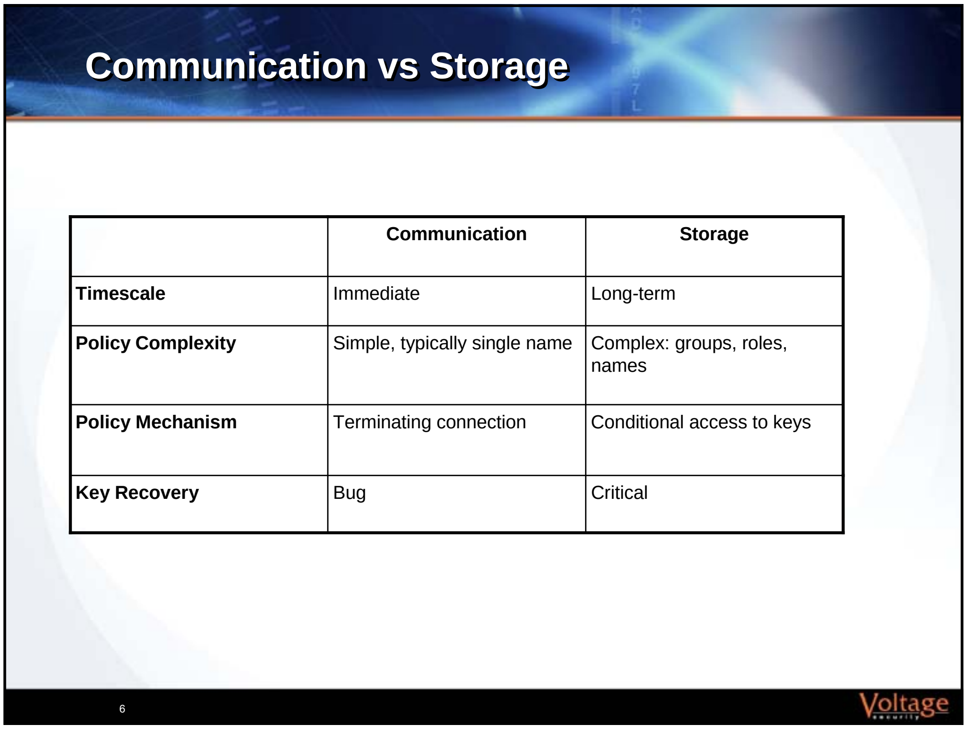
File X

DB

?

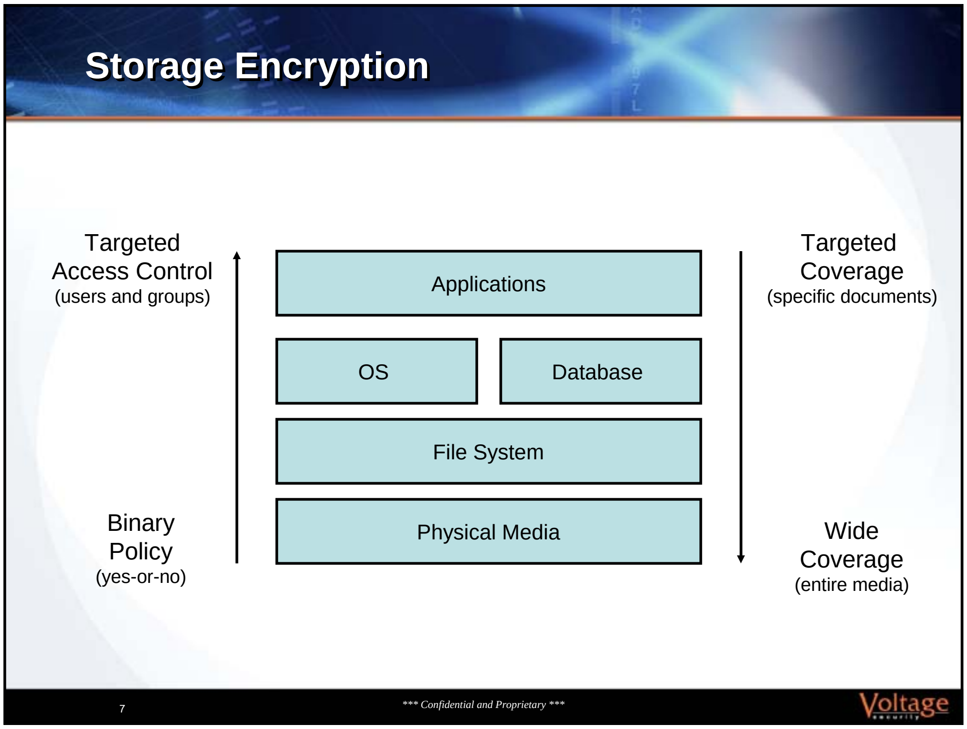
Bob

- Alice never talks directly to Bob
- But she wants to apply policy to the data
  - “Make this file readable by Bob”
- This requires *externalizing* policy



Communication vs Storage

	Communication	**Storage**
Timescale	Immediate	Long-term
Policy Complexity	Simple, typically single name	Complex: groups, roles, names
Policy Mechanism	Terminating connection	Conditional access to keys
Key Recovery	Bug	Critical



Storage Encryption

Targeted
Access Control
(users and groups)

Binary
Policy
(yes-or-no)

Applications

OS

Database

File System

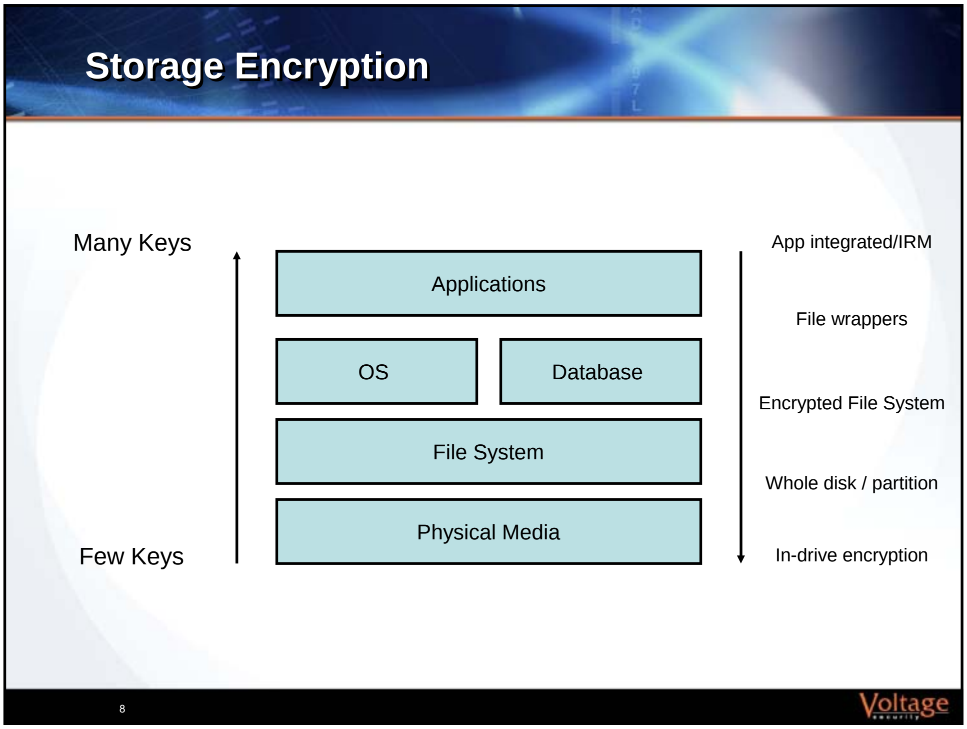
Physical Media

Targeted
Coverage
(specific documents)

Wide
Coverage
(entire media)

7

*** Confidential and Proprietary ***



Storage Encryption

Many Keys

Few Keys

Applications

OS

Database

File System

Physical Media

App integrated/IRM

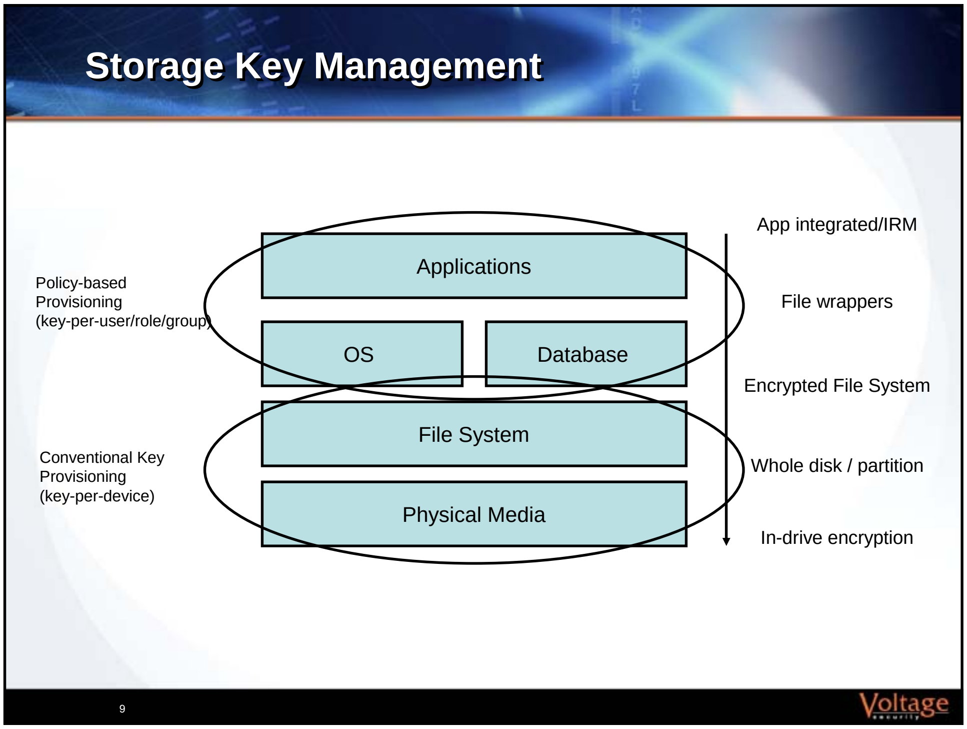
File wrappers

Encrypted File System

Whole disk / partition

In-drive encryption

8



Storage Key Management

Policy-based Provisioning (key-per-user/role/group)

Conventional Key Provisioning (key-per-device)

Applications

OS

Database

File System

Physical Media

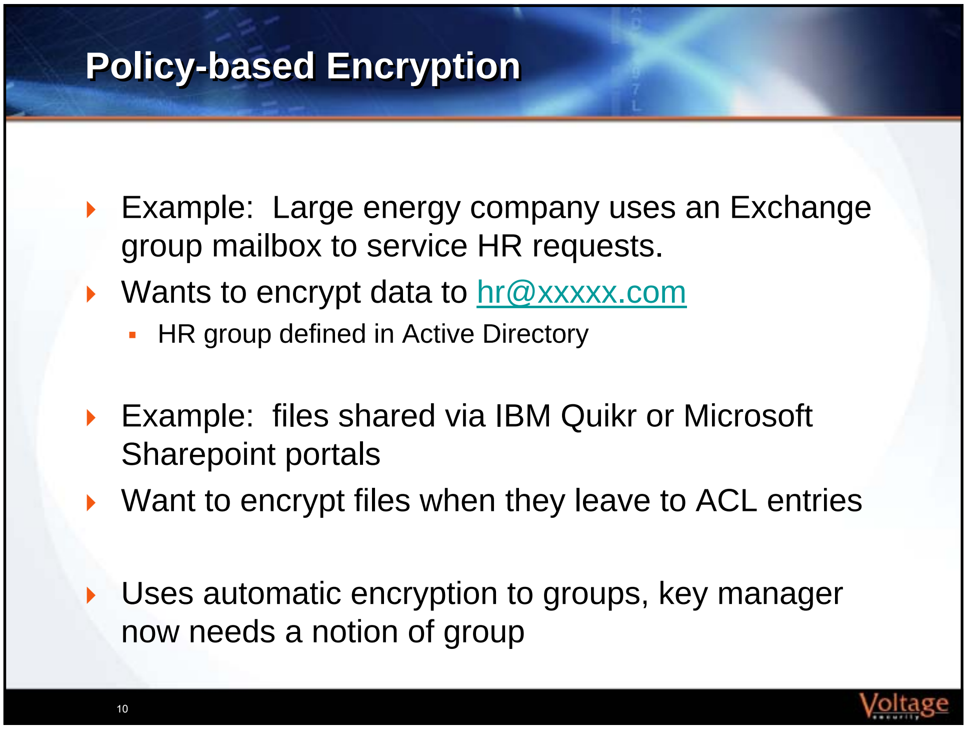
App integrated/IRM

File wrappers

Encrypted File System

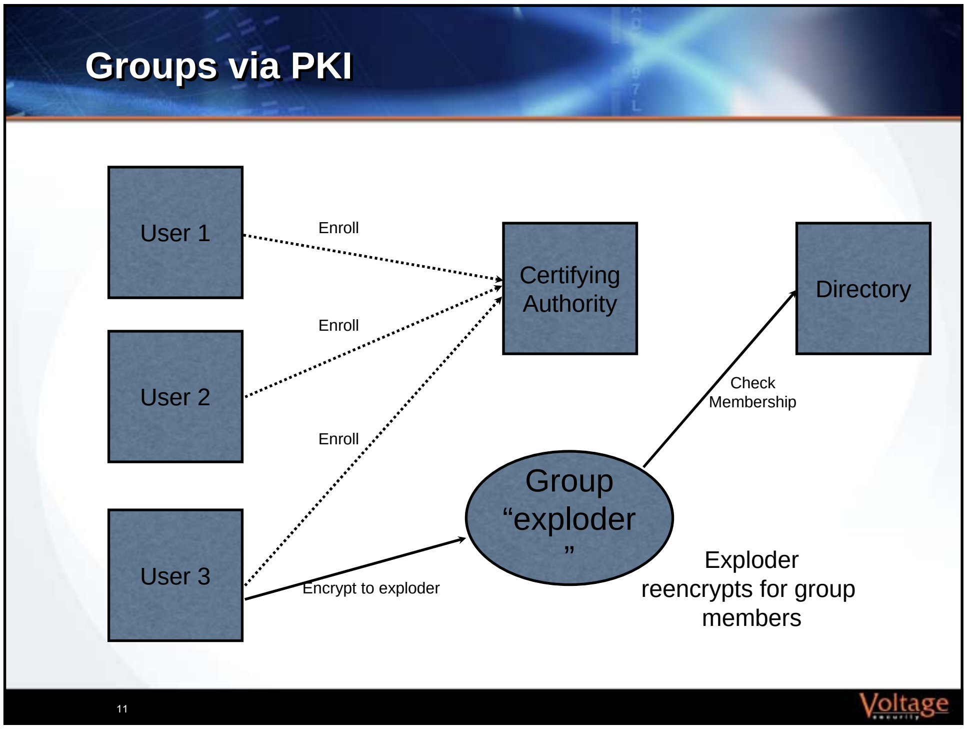
Whole disk / partition

In-drive encryption



Policy-based Encryption

- Example: Large energy company uses an Exchange group mailbox to service HR requests.
- Wants to encrypt data to hr@xxxxx.com
  - HR group defined in Active Directory
- Example: files shared via IBM Quikr or Microsoft Sharepoint portals
- Want to encrypt files when they leave to ACL entries
- Uses automatic encryption to groups, key manager now needs a notion of group



Groups via PKI

User 1

User 2

User 3

Enroll

Enroll

Enroll

Certifying Authority

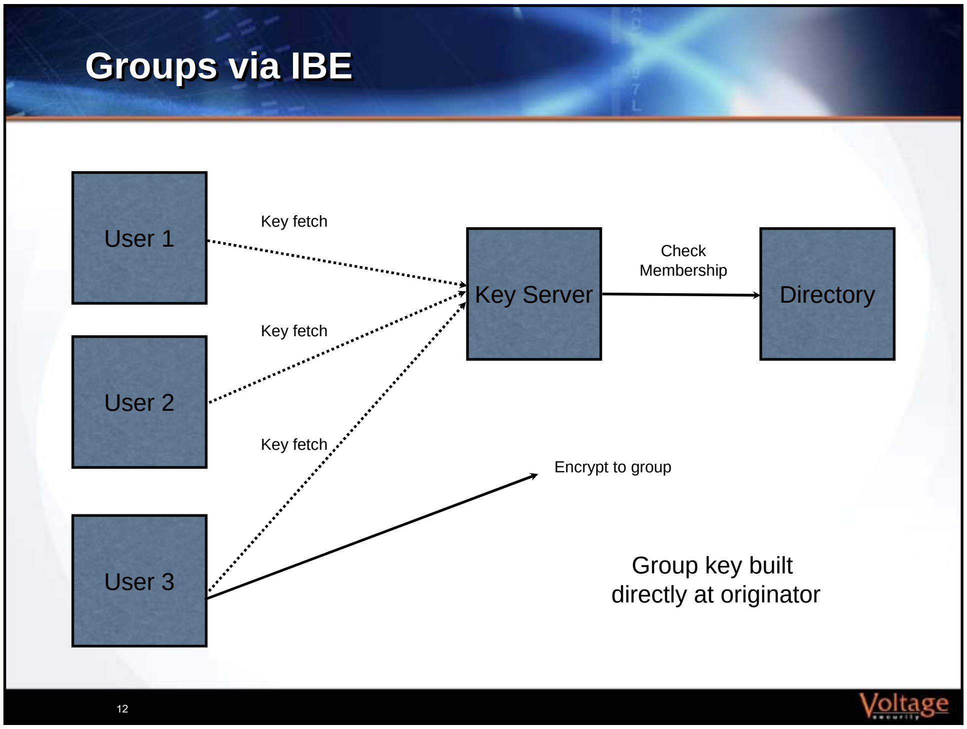
Directory

Check Membership

Group “exploder”

Encrypt to exploder

Exploder reencrypts for group members



Groups via IBE

User 1

User 2

User 3

Key fetch

Key fetch

Key fetch

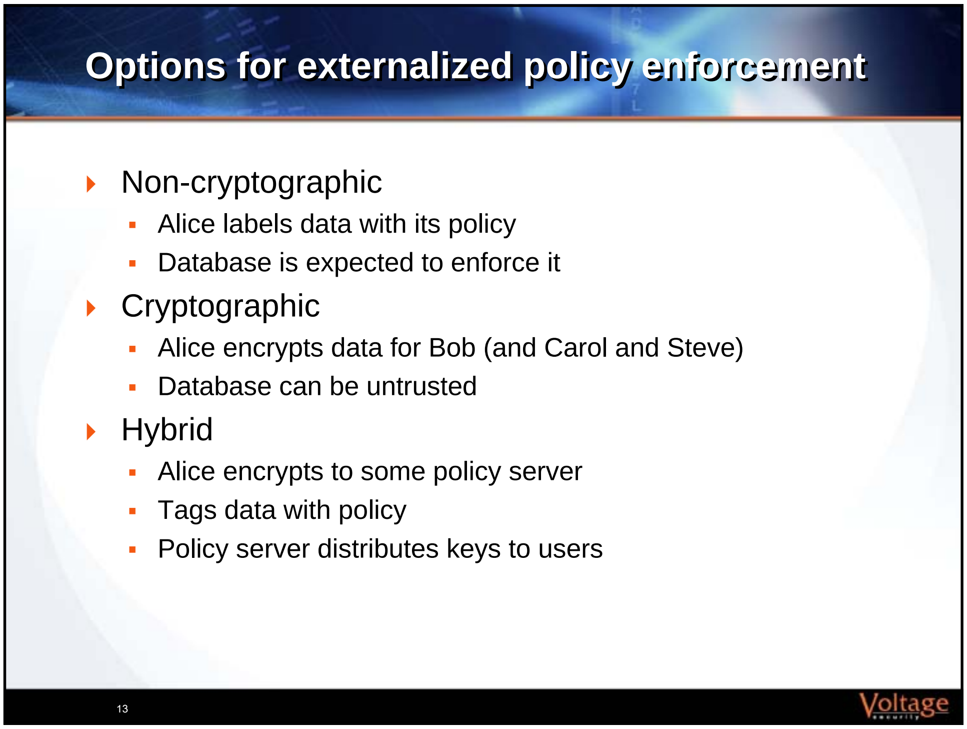
Key Server

Check Membership

Directory

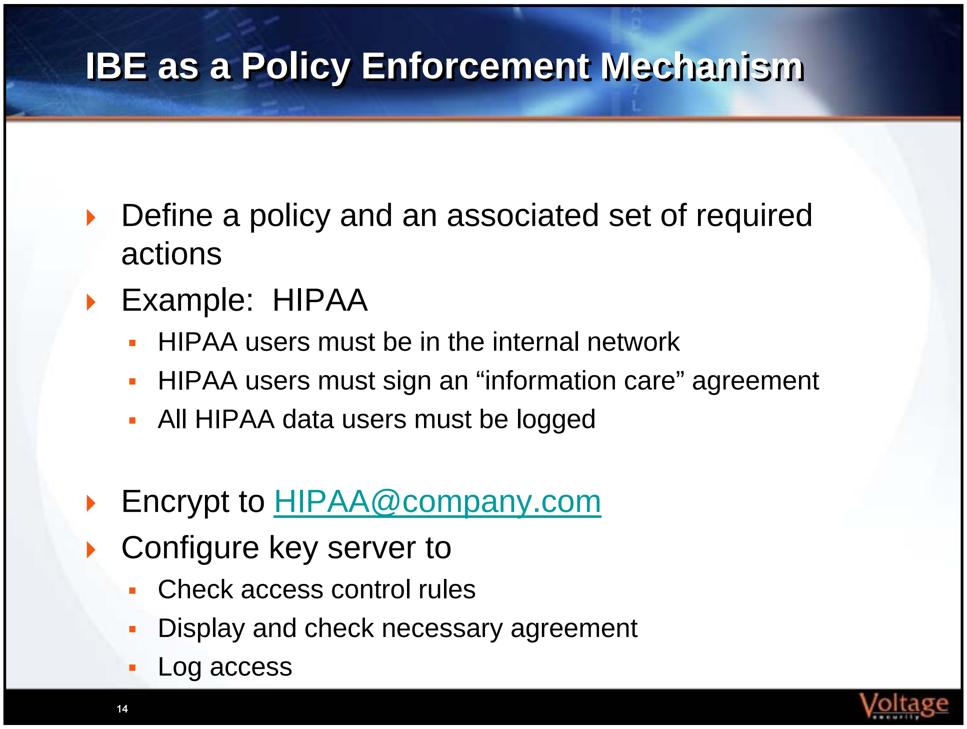
Encrypt to group

Group key built directly at originator



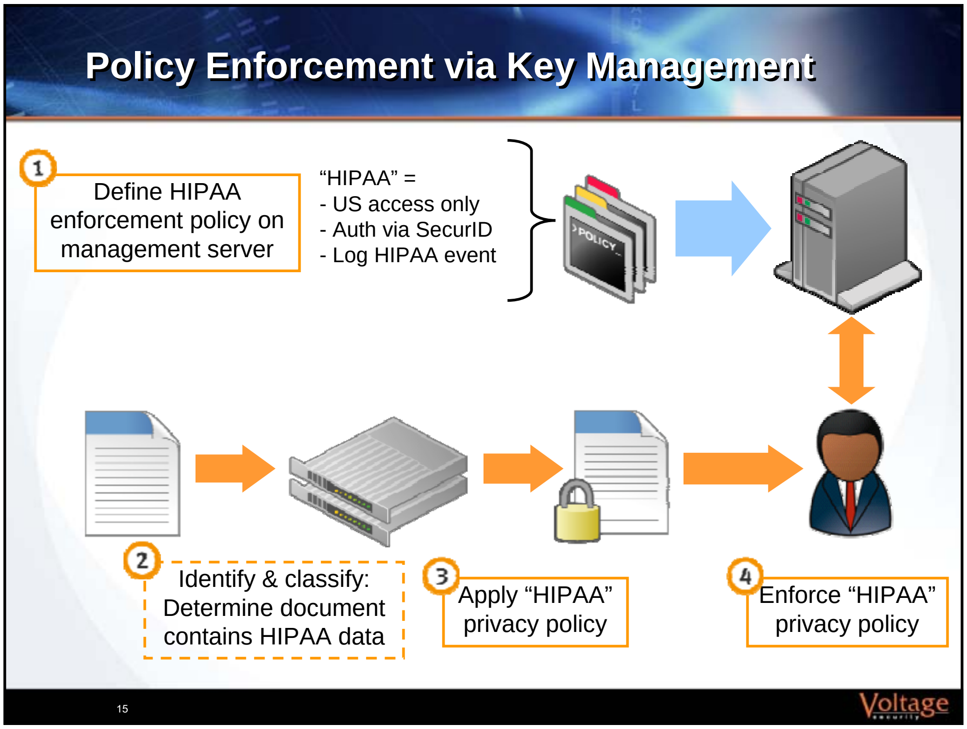
Options for externalized policy enforcement

- Non-cryptographic
  - Alice labels data with its policy
  - Database is expected to enforce it
- Cryptographic
  - Alice encrypts data for Bob (and Carol and Steve)
  - Database can be untrusted
- Hybrid
  - Alice encrypts to some policy server
  - Tags data with policy
  - Policy server distributes keys to users



IBE as a Policy Enforcement Mechanism

- Define a policy and an associated set of required actions
- Example: HIPAA
  - HIPAA users must be in the internal network
  - HIPAA users must sign an “information care” agreement
  - All HIPAA data users must be logged
- Encrypt to HIPAA@company.com
- Configure key server to
  - Check access control rules
  - Display and check necessary agreement
  - Log access



Policy Enforcement via Key Management

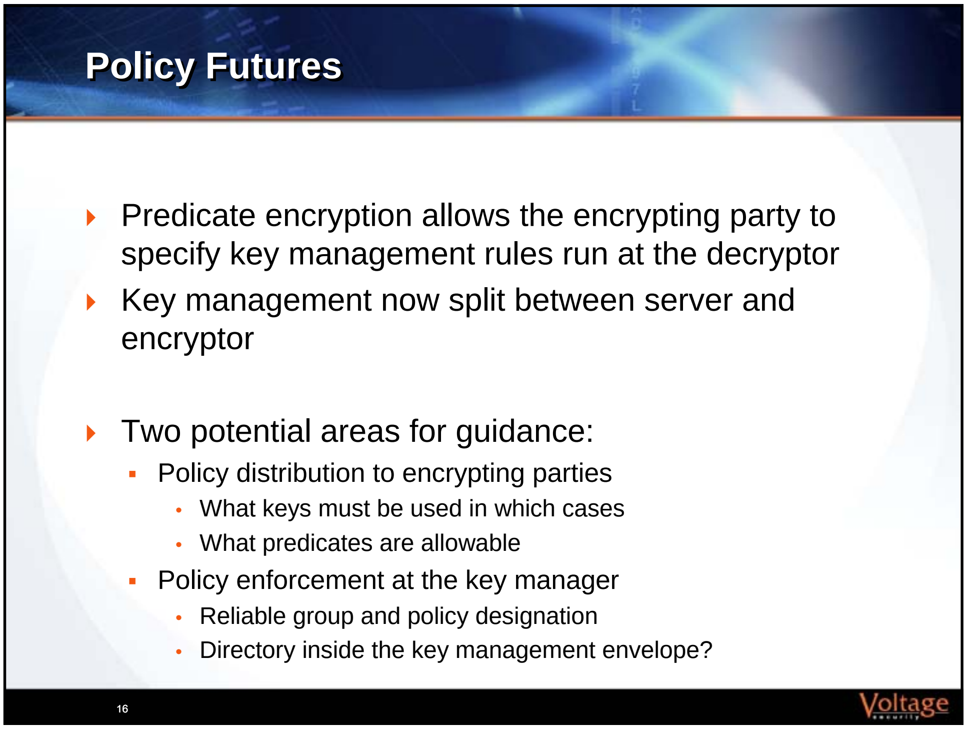
1 Define HIPAA enforcement policy on management server

"HIPAA" =
- US access only
- Auth via SecurID
- Log HIPAA event

2 Identify & classify: Determine document contains HIPAA data

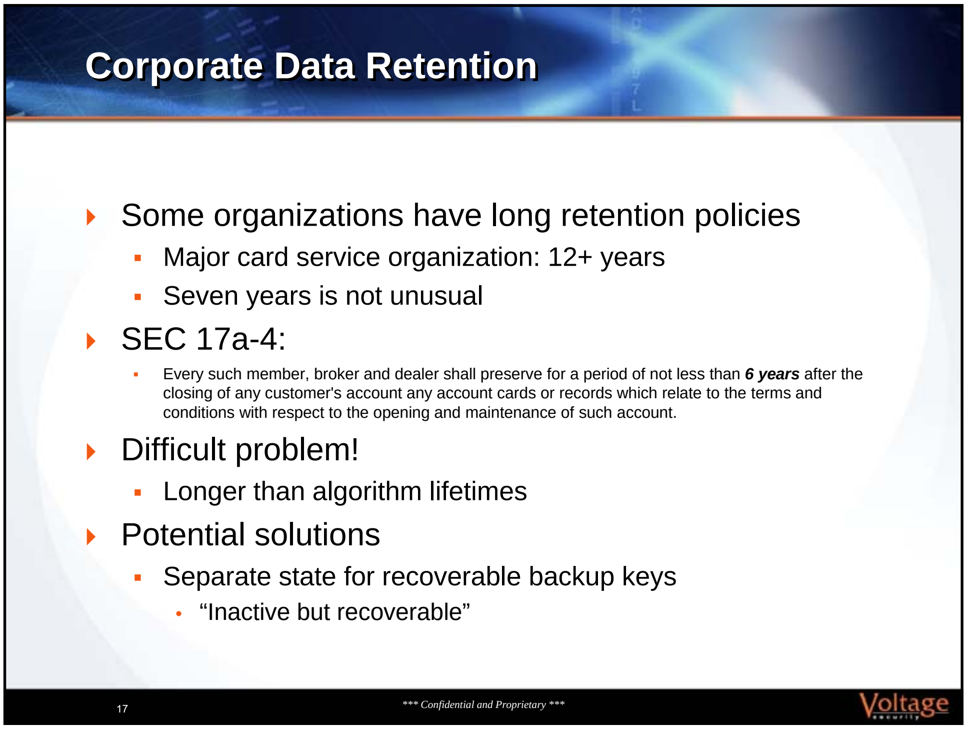
3 Apply "HIPAA" privacy policy

4 Enforce "HIPAA" privacy policy



Policy Futures

- Predicate encryption allows the encrypting party to specify key management rules run at the decryptor
- Key management now split between server and encryptor
- Two potential areas for guidance:
  - Policy distribution to encrypting parties
    - What keys must be used in which cases
    - What predicates are allowable
  - Policy enforcement at the key manager
    - Reliable group and policy designation
    - Directory inside the key management envelope?

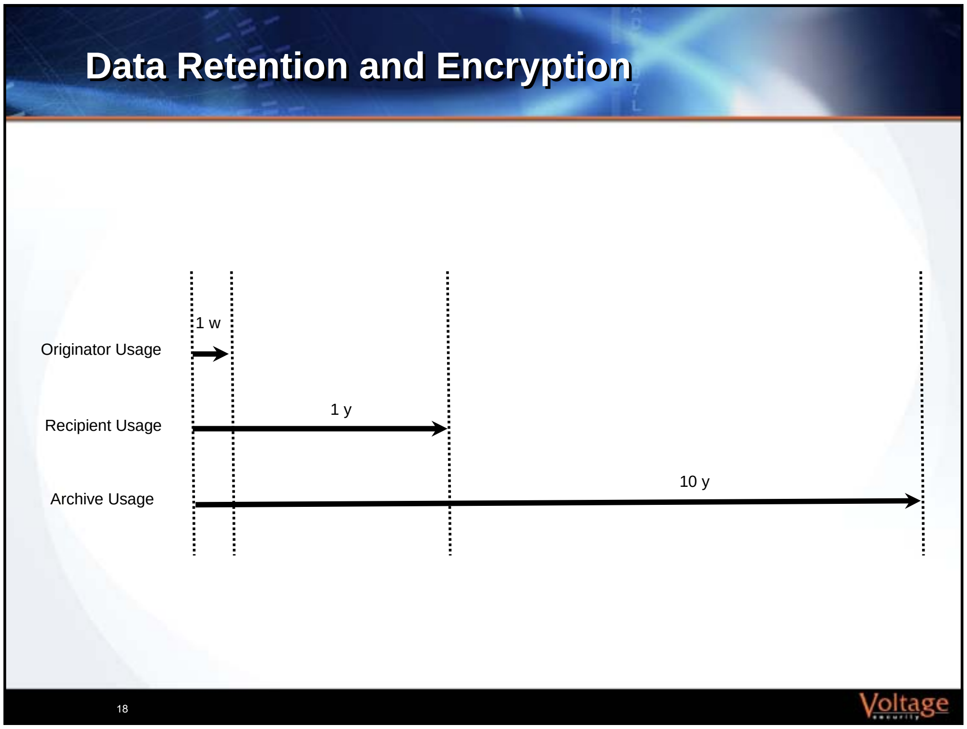


Corporate Data Retention

- Some organizations have long retention policies
  - Major card service organization: 12+ years
  - Seven years is not unusual
- SEC 17a-4:
  - Every such member, broker and dealer shall preserve for a period of not less than ***6 years*** after the closing of any customer's account any account cards or records which relate to the terms and conditions with respect to the opening and maintenance of such account.
- Difficult problem!
  - Longer than algorithm lifetimes
- Potential solutions
  - Separate state for recoverable backup keys
    - “Inactive but recoverable”

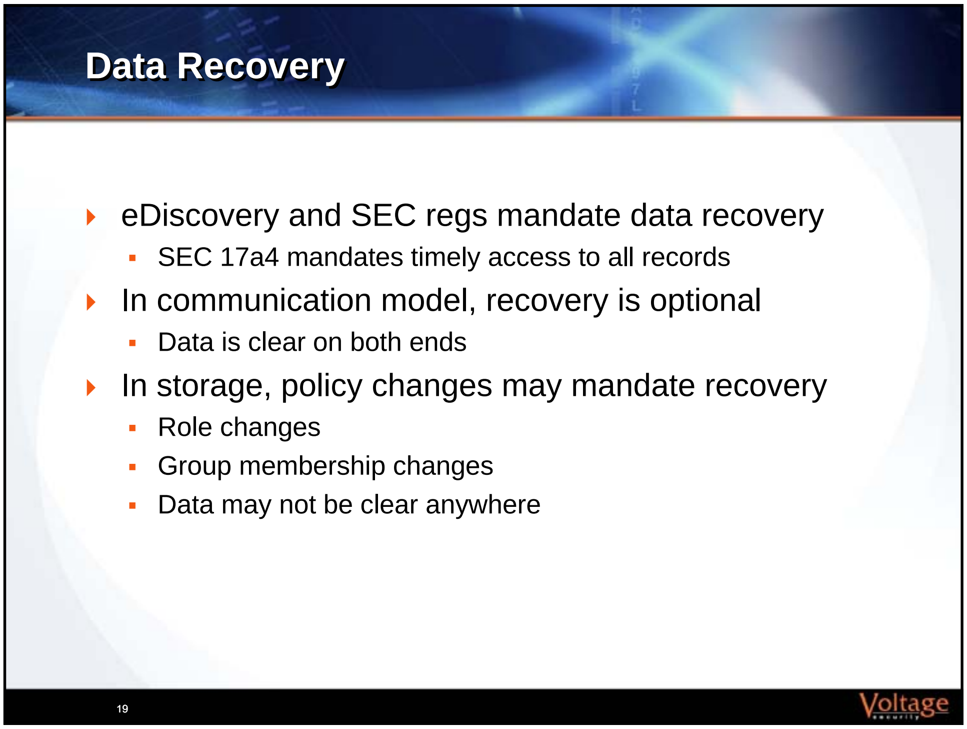
17

**** Confidential and Proprietary ****



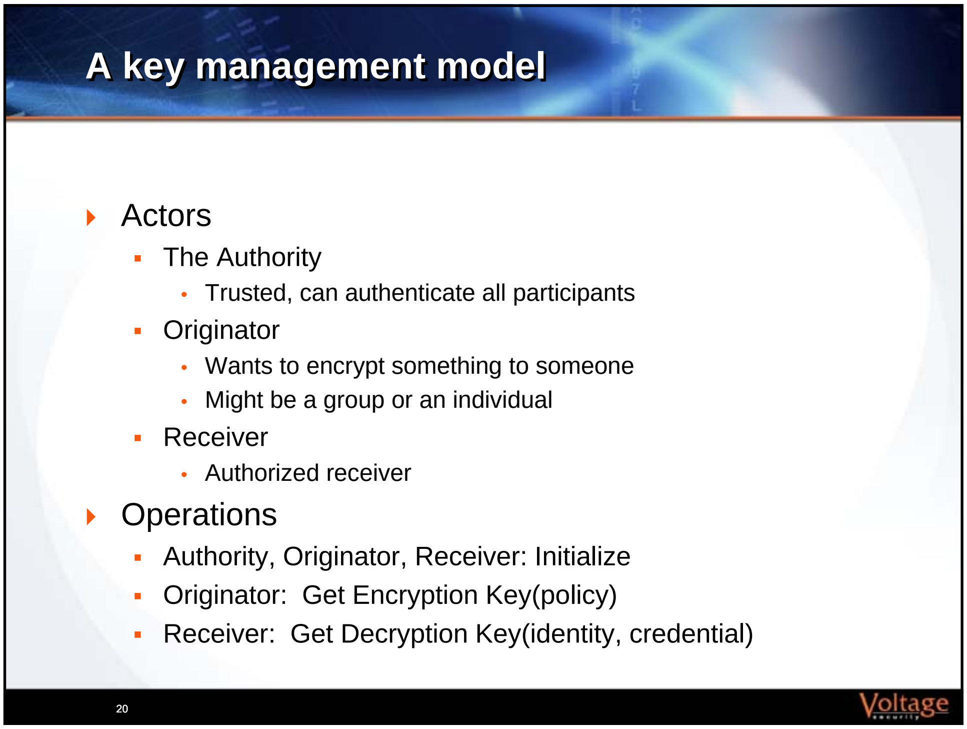
Data Retention and Encryption

18



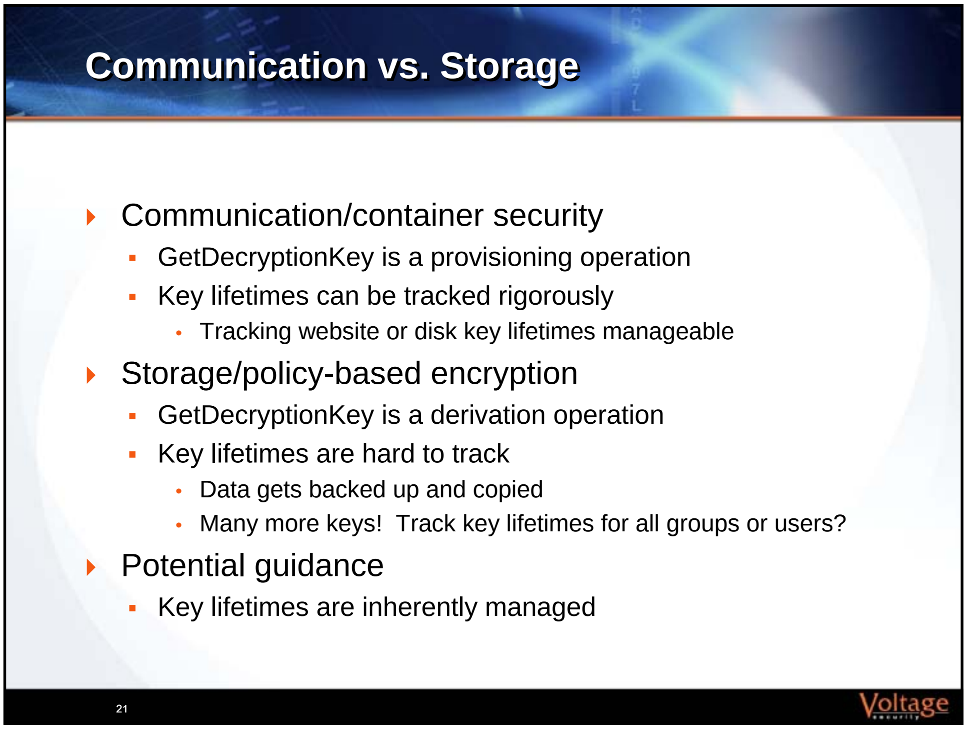
Data Recovery

- eDiscovery and SEC regs mandate data recovery
  - SEC 17a4 mandates timely access to all records
- In communication model, recovery is optional
  - Data is clear on both ends
- In storage, policy changes may mandate recovery
  - Role changes
  - Group membership changes
  - Data may not be clear anywhere



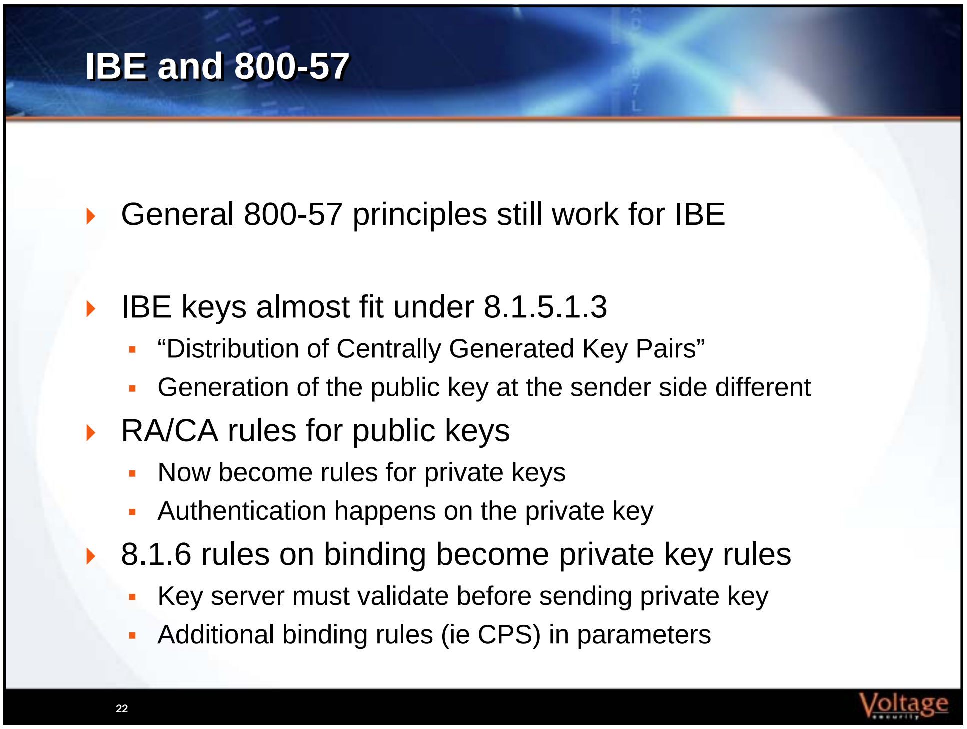
A key management model

- Actors
  - The Authority
    - Trusted, can authenticate all participants
  - Originator
    - Wants to encrypt something to someone
    - Might be a group or an individual
  - Receiver
    - Authorized receiver
- Operations
  - Authority, Originator, Receiver: Initialize
  - Originator: Get Encryption Key(policy)
  - Receiver: Get Decryption Key(identity, credential)



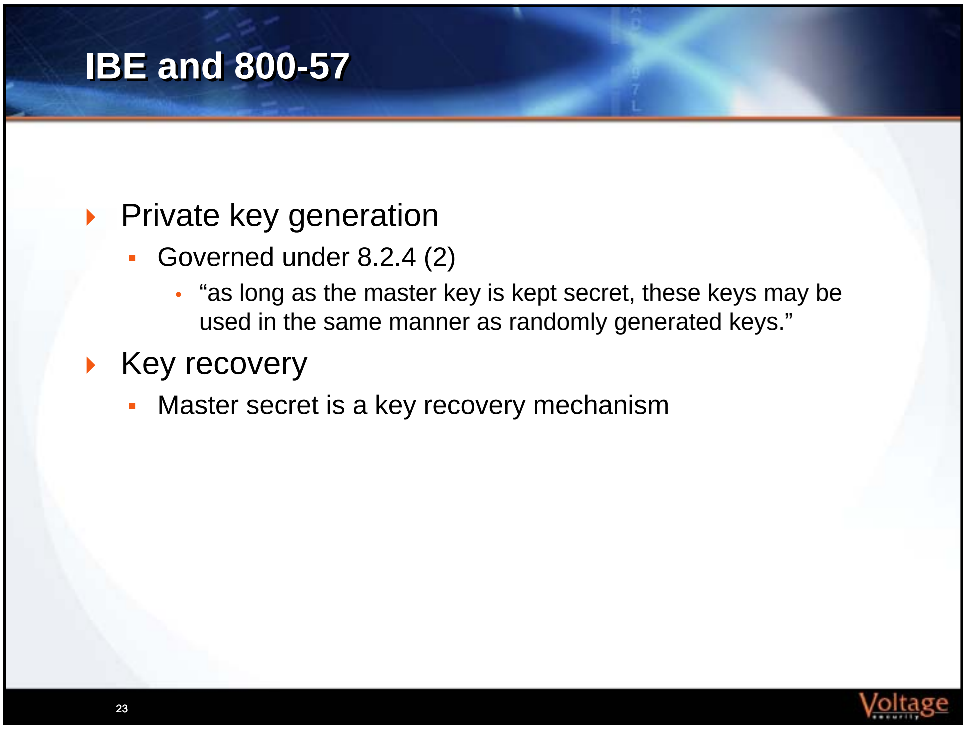
Communication vs. Storage

- Communication/container security
  - GetDecryptionKey is a provisioning operation
  - Key lifetimes can be tracked rigorously
    - Tracking website or disk key lifetimes manageable
- Storage/policy-based encryption
  - GetDecryptionKey is a derivation operation
  - Key lifetimes are hard to track
    - Data gets backed up and copied
    - Many more keys! Track key lifetimes for all groups or users?
- Potential guidance
  - Key lifetimes are inherently managed



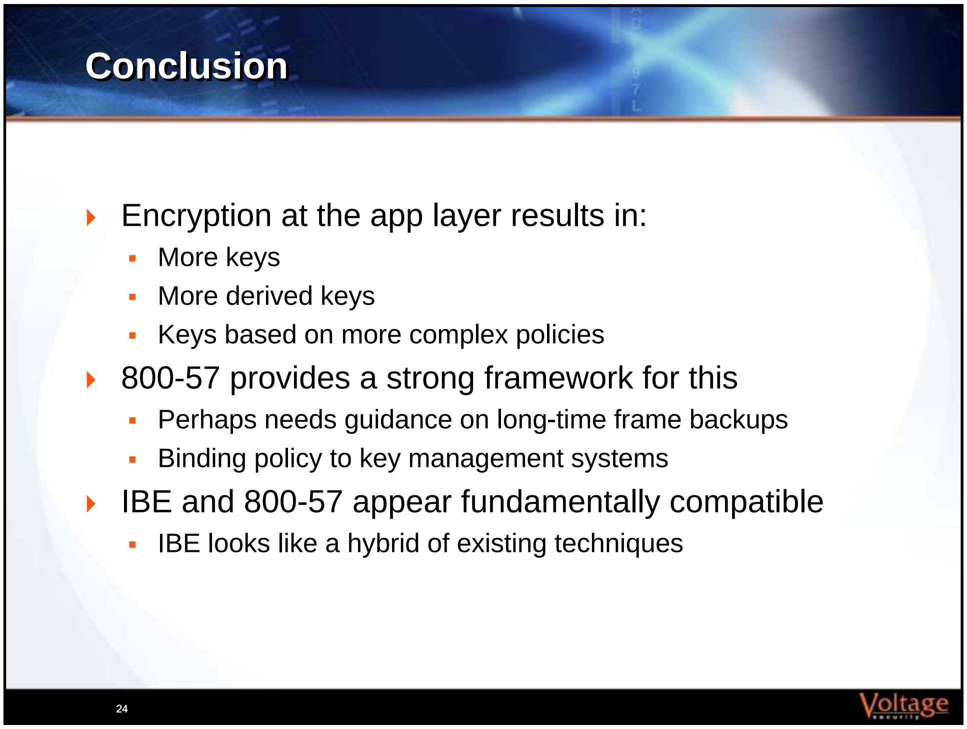
IBE and 800-57

- General 800-57 principles still work for IBE
- IBE keys almost fit under 8.1.5.1.3
  - “Distribution of Centrally Generated Key Pairs”
  - Generation of the public key at the sender side different
- RA/CA rules for public keys
  - Now become rules for private keys
  - Authentication happens on the private key
- 8.1.6 rules on binding become private key rules
  - Key server must validate before sending private key
  - Additional binding rules (ie CPS) in parameters



IBE and 800-57

- Private key generation
  - Governed under 8.2.4 (2)
    - “as long as the master key is kept secret, these keys may be used in the same manner as randomly generated keys.”
- Key recovery
  - Master secret is a key recovery mechanism



Conclusion

- Encryption at the app layer results in:
  - More keys
  - More derived keys
  - Keys based on more complex policies
- 800-57 provides a strong framework for this
  - Perhaps needs guidance on long-time frame backups
  - Binding policy to key management systems
- IBE and 800-57 appear fundamentally compatible
  - IBE looks like a hybrid of existing techniques