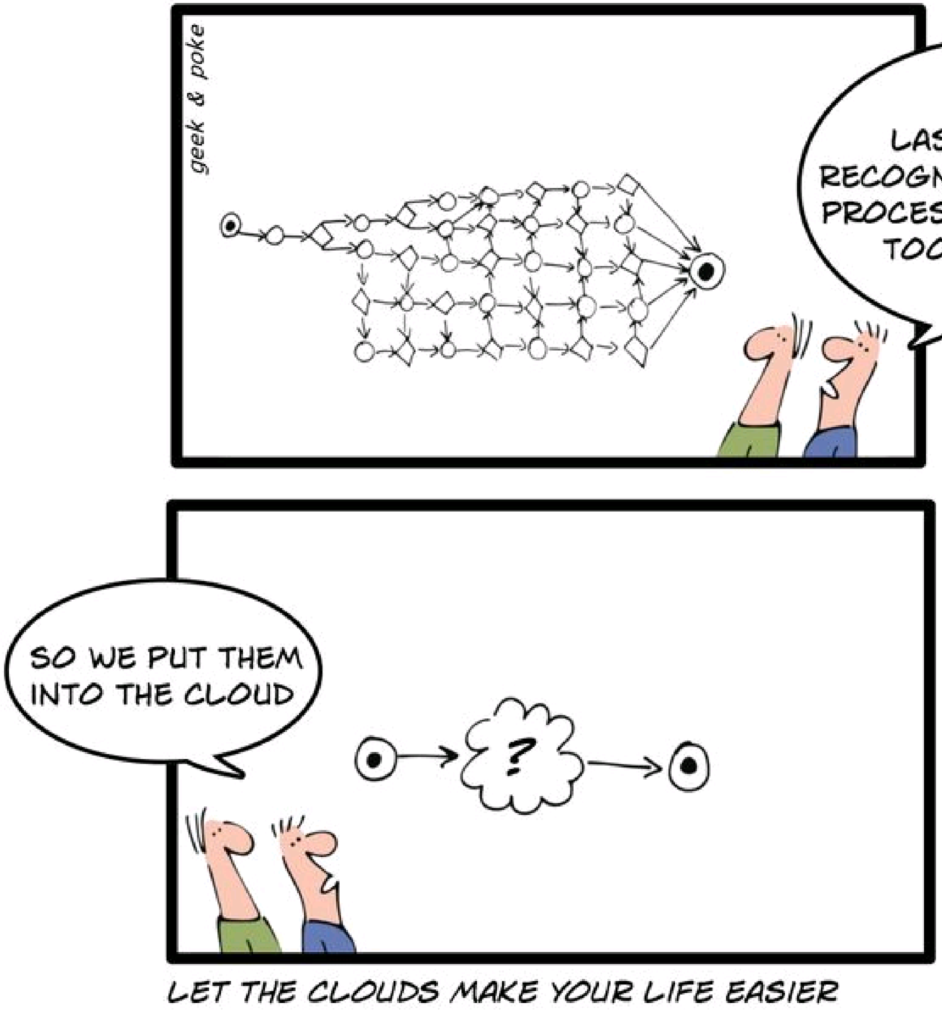
Federated Key Management for Secure Cloud Computing

Click to edit Master subtitle style

Overview

- Key management and why it's important
- Federated key management
- How federated key management can provide the infrastructure needed to protect sensitive data in a cloud environment
- Properties of a future key management service

Complexity and Cloud Computing


geek & poke
LAST YEAR WE RECOGNIZED THAT OUR PROCESSES WERE FAR TOO COMPLEX
SO WE PUT THEM INTO THE CLOUD
?
LET THE CLOUDS MAKE YOUR LIFE EASIER

Source: “Geek and Poke” http://geekandpoke.typepad.com

What is key management?

- Key management covers everything that you do with a key *except* encrypt or decrypt
- Creation/generation of keys
- Activation/deactivation of keys
- Transport of keys
- Storage of keys
- Destruction of keys
- Etc.

Key management

- With a secret combination, a vault is safe
  - How do you keep the combination?
- How do you manage access at an airport
  - Mechanisms protect
  - Need a policy for the mechanism
- “Amateurs talk tactics, professionals talk logistics.”

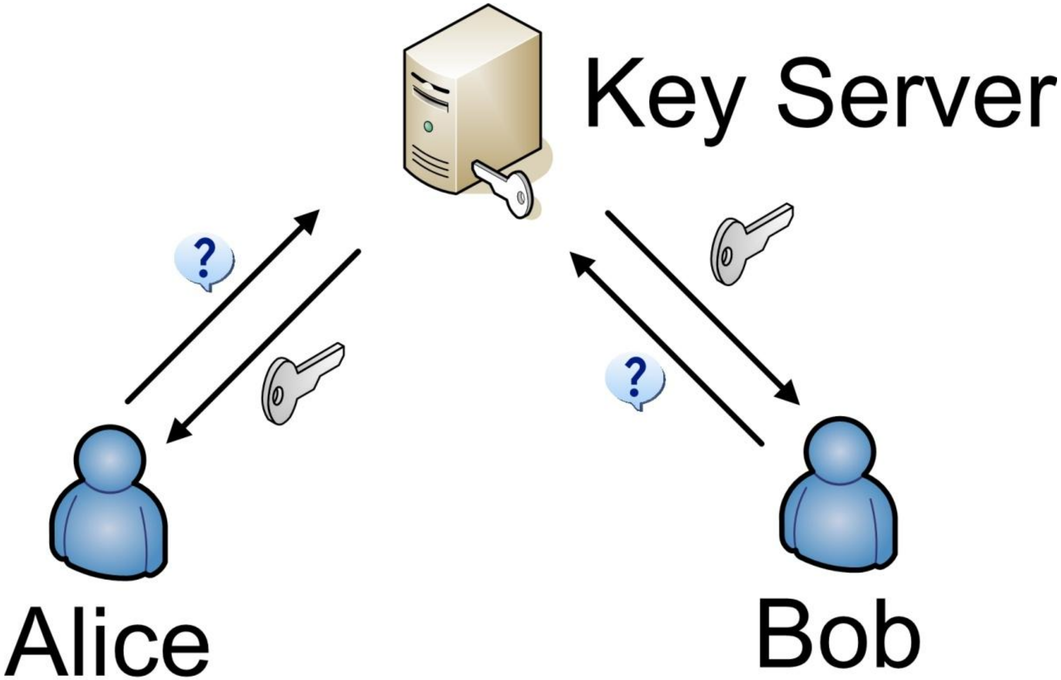
Key management

- Key management is harder than cryptography
- Cryptography boils down to math
- Key management involves
  - Technology
  - People
  - Processes
- Strong encryption is almost always impossible to beat
- Key management isn’t as robust

Example: unauthenticated users

- Consider a key server where a user needs to authenticate to the server to get a key
- Authentication can be expensive to implement and support, so you might (?) want to use no authentication at all
- If you asked for a key you'd get it
- But the encryption algorithm itself was still very strong, wasn't it?

Unauthenticated users

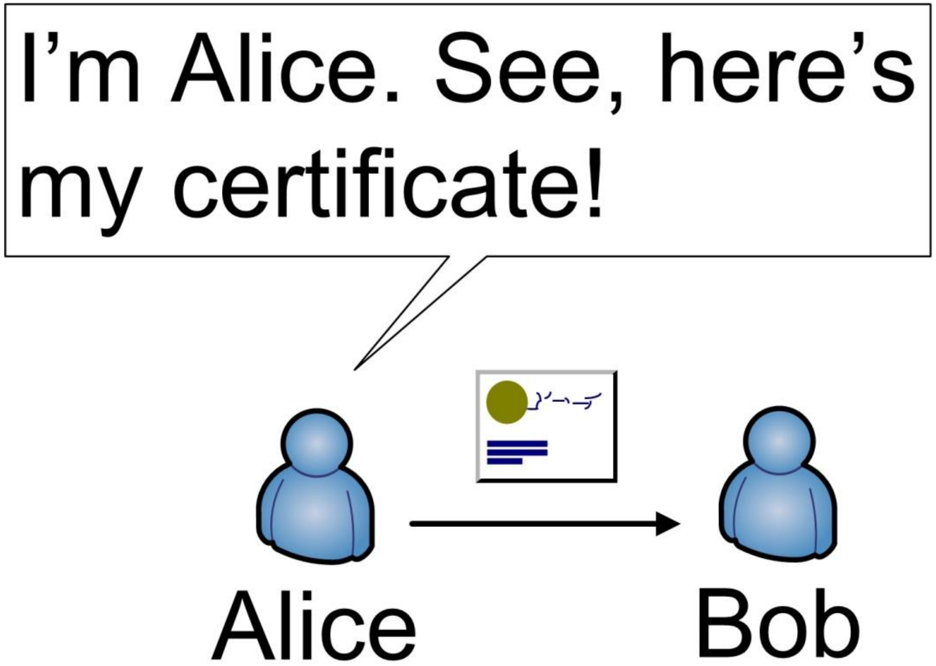


Example: abusing PKI

- A digital certificate carries a user’s public key
- Anyone can get a certificate
- Certificates can be used as part of an authentication protocol, but they’re *not* the equivalent of a password
- Public keys are public

[This is an example of a really bad case – sadly its been seen in the field…]

Abusing PKI

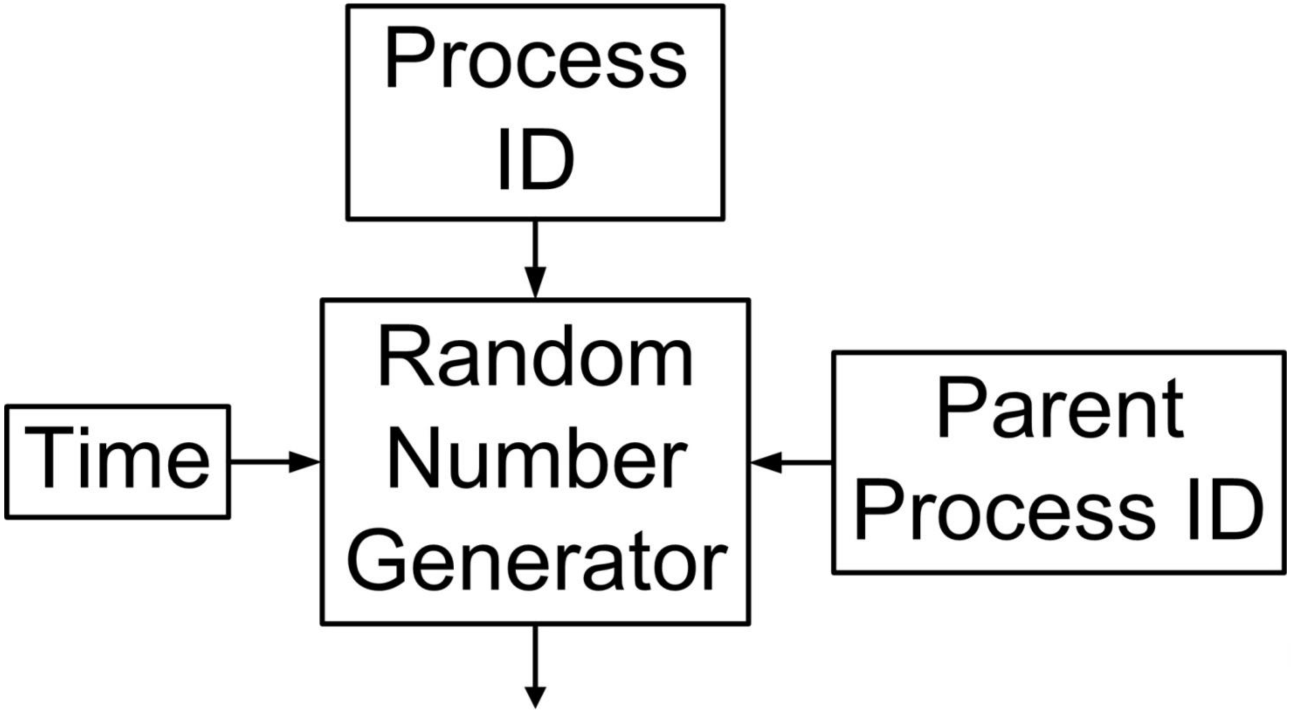


(Of course, anyone can do this, not just Alice....)

Example

- We’re assuming that keys look random, so there’s no reason to think that a particular key was or was not used
- An early version of the Netscape browser generated keys for use in SSL in a way that made them fairly easy to guess
- 47 bits vs. 128 bits
- Feasible vs. infeasible

Netscape random number generator



Process ID

Time

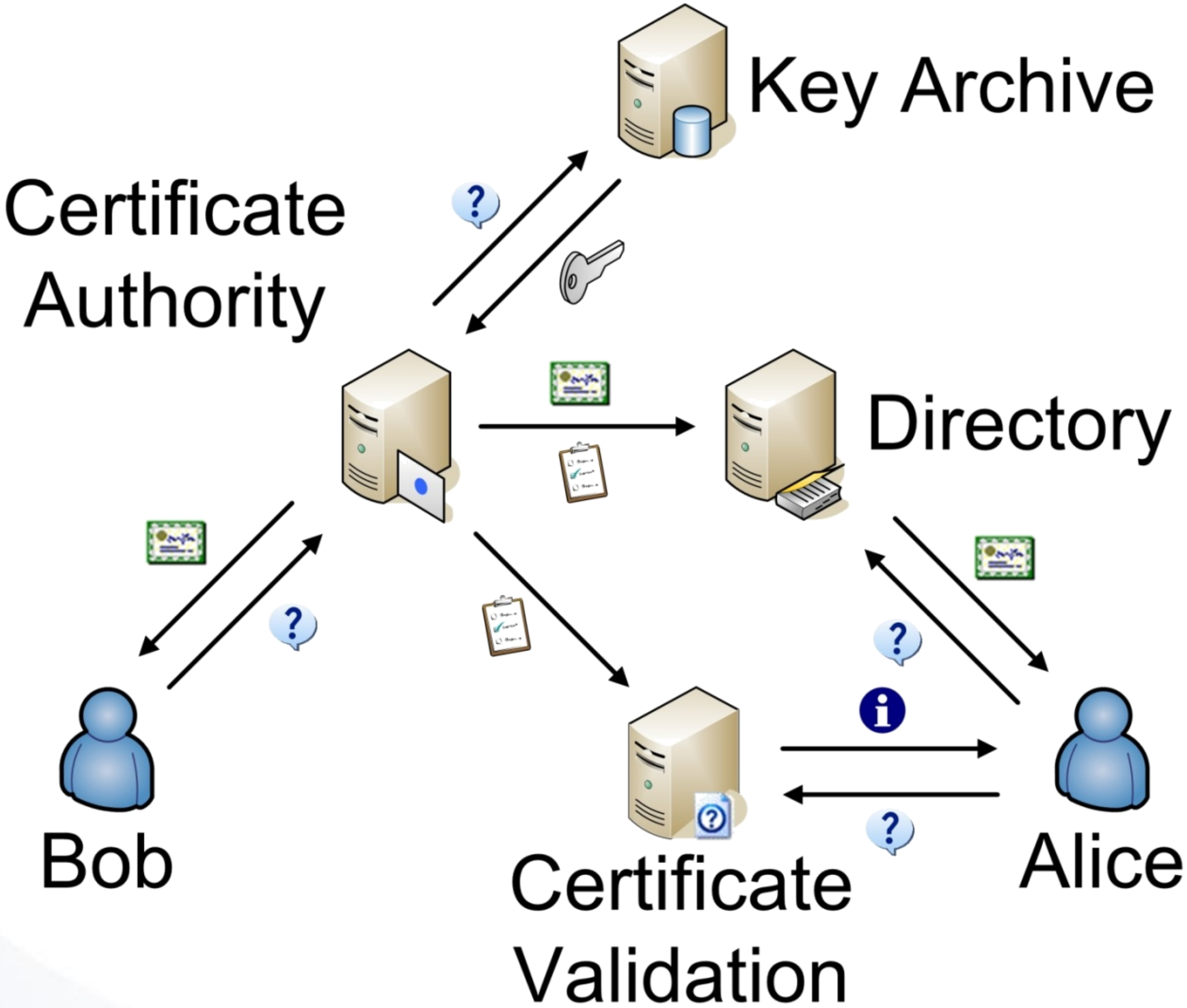
Random Number Generator

Parent Process ID

Example

- *Everything* that a PKI system does is key management
- There are lots of components to a PKI system
- The failure or compromise of any one of these components results in the failure or compromise of the system

PKI


Key Archive
Certificate Authority
Directory
Bob
Certificate Validation
Alice

What is federated key management?

- Federated identity management
  - Authentication across domains
- Federated key management
  - Access control across domains
- Authentication is needed to get keys and keys can be used for authentication, so the two are somewhat similar
- SAML exists for one, what about the other?

Key management standards

- Existing key management standards just tell you what to do, not how to do it
  - NIST’s SP 800-57, ISO/IEC 11770, etc.
- They’re not interoperability standards
- This will be changing soon
  - OASIS Key Management Interoperability Protocol
  - IEEE P1619.3 Standard for Key Management Infrastructure for Cryptographic Protection of Stored Data

In a cloud environment

- In a cloud environment, data can potentially be anywhere
  - Same data, different application
  - Same data, different server
- To encrypt/decrypt it, you need to get the right key
- Federated key management solves this very problem

Federated Key Management Requirements

- Applications should be able to specify:
  - Who or what should have access to data
    - Namespace should be universal
  - What key server authenticates access
- Enterprises should have recovery ability
  - E-discovery
  - Internal controls

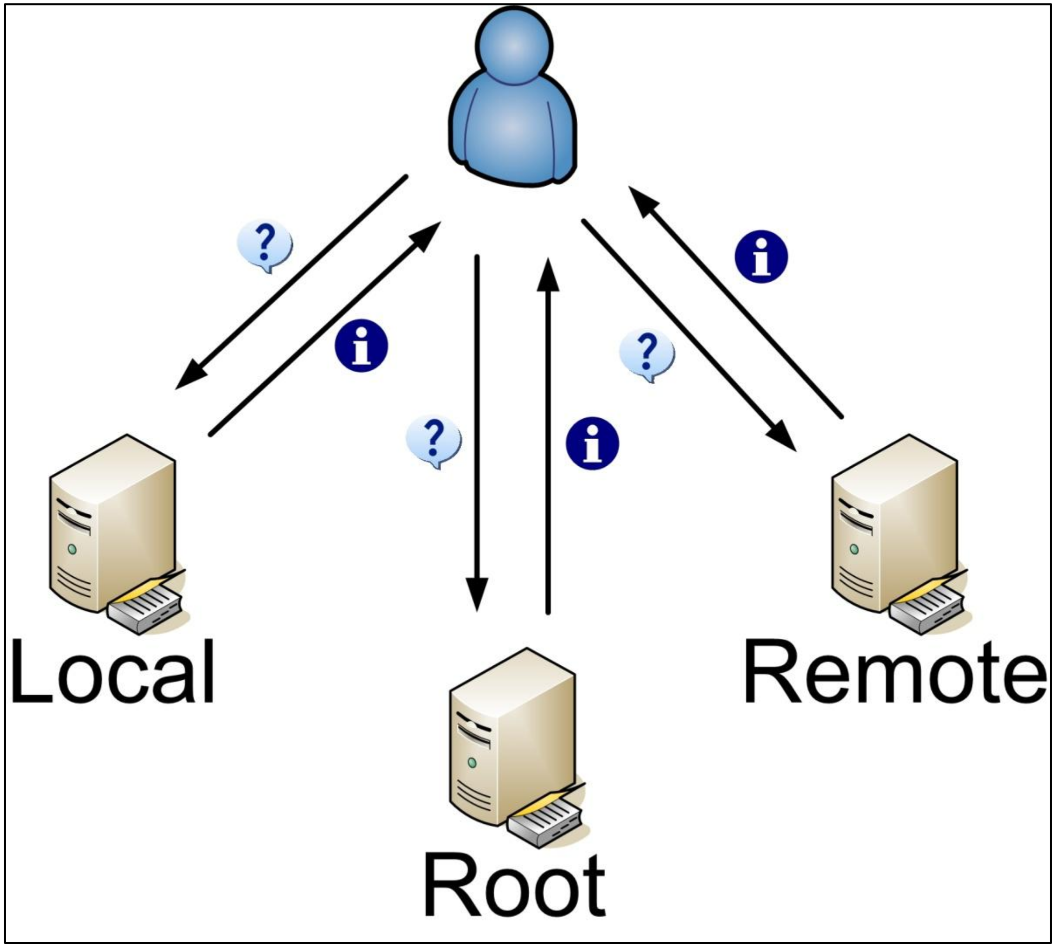
Federated Key Management Examples

- Bank transferring records through a service
  - Accessors: customer, bank auditors
  - Key server: bank authenticates access
- Design partners storing CAD drawings
  - Accessors: project group at A & B
  - Key server: A authenticates group A, B authenticates group B

Federated Key Management Examples

- Card data at a point-of-sale
  - Payment systems: the first “cloud”
  - Accessor: Issuing bank and brand only
    - Note: encryptor cannot decrypt!
  - Key server: Bank and brand authenticate

A hypothetical key management service



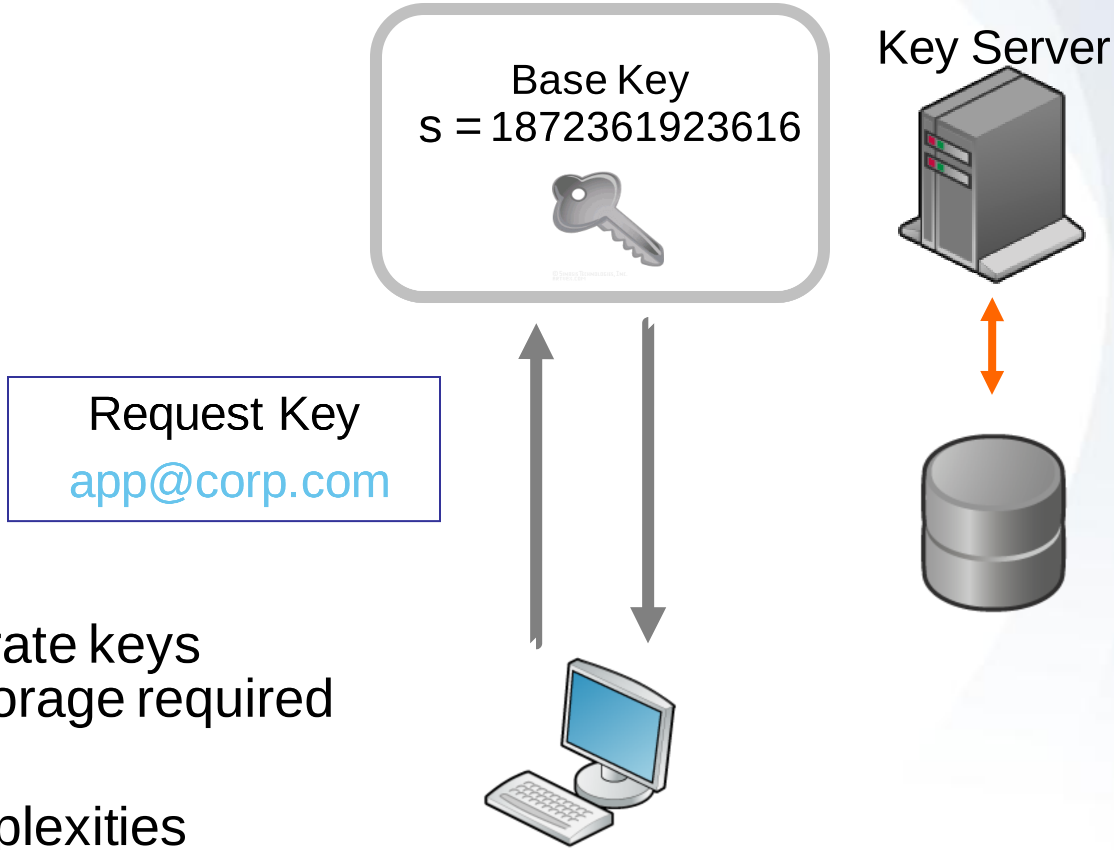
Federated Key Management Components

- Client API
  - Encrypt(accessor, key server, data)
  - Decrypt(name, credential, data)
- Key Management Protocol
  - RequestKey, DestroyKey, CheckStatus
- Policy Description Language
  - Specify who has access to what keys
  - Deal with recovery situations

Technical Hurdles

- Client
  - Given a policy, how to map this to a key?
- Key Manager
  - How to name keys
  - How to store keys
- Policy Description Language
  - How to establish legitimate recoveries
    - ie. Bank to bank

Strategy One: Key Derivation



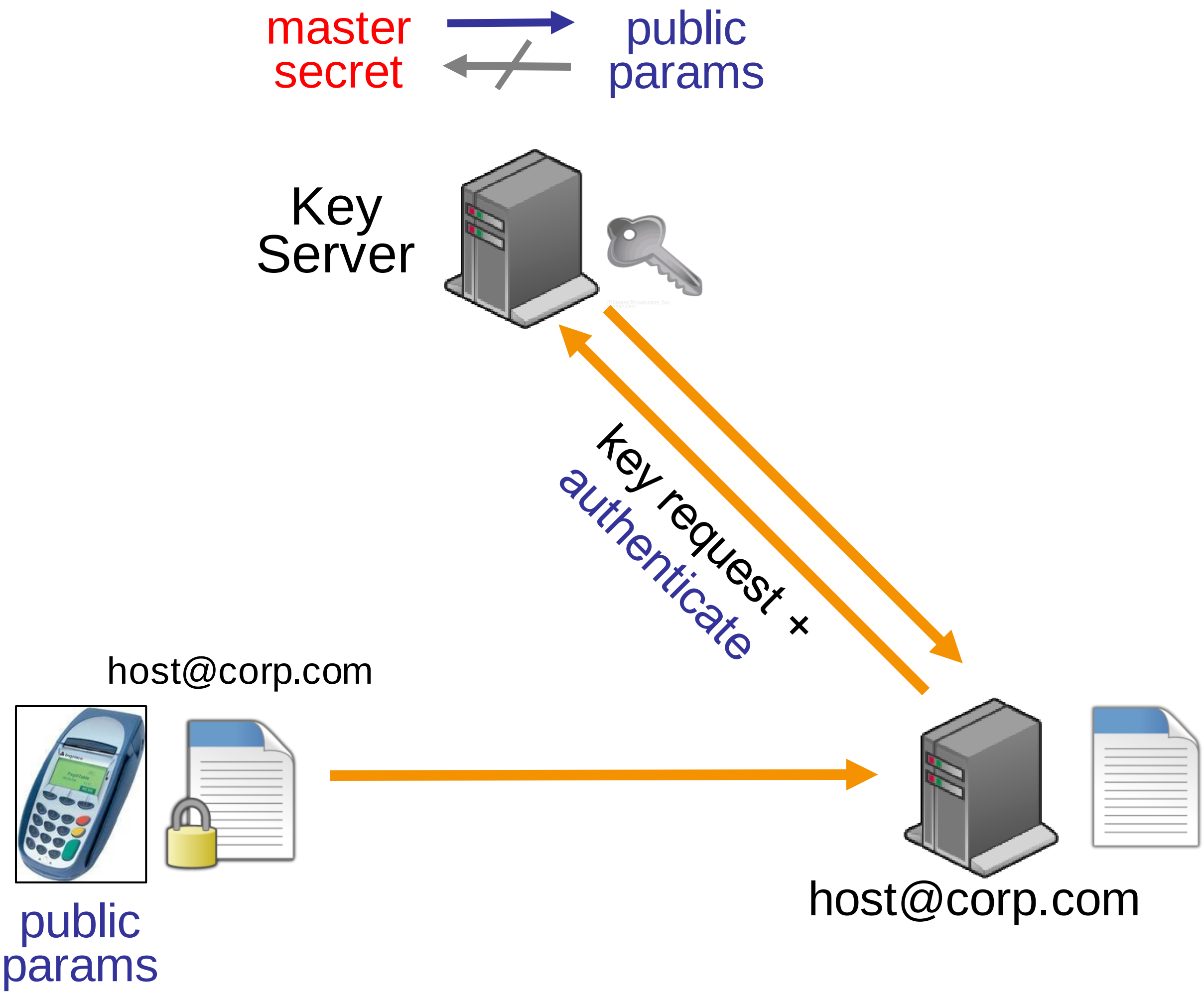
Key Server

Base Key
s = 1872361923616

Request Key
app@corp.com

- Base Key is used to generate keys on-demand – no server storage required
- Eliminates traditional complexities
  - Simplified high availability, disaster recovery
  - Highly scalable

Strategy One : Public Key Derivation via IBE



master secret → public params

Key Server

key request + authenticate

host@corp.com

public params

host@corp.com

Strategy Two: Key Naming

name@domain is *extremely* useful

- Direct mapping to LDAP and other standards
- Nearly human readable
- Not subject to email attacks
  - name@domain Is a lookup tag
  - Authentication method is independent

Strategy Two: Key Naming

name@domain is *extremely* useful

- Direct mapping to LDAP and other standards
- Nearly human readable
- Not subject to email attacks
  - name@domain Is a lookup tag
  - Authentication method is independent

Conclusion

- The cloud requires encryption to maintain access control
- Key management is crucial to make this work in practice
- Careful design strategies can make the burden of key management lighter