Virtual Hierarchies - An Architecture for Building and Maintaining Efficient and Resilient Trust Chains

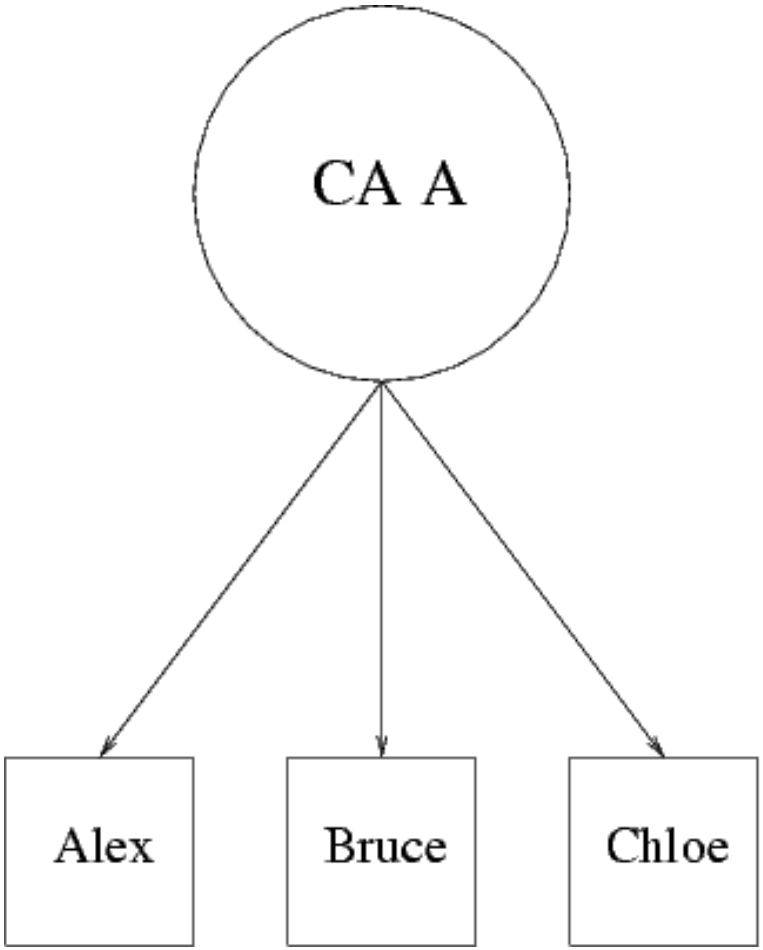
John Marchesini and Sean Smith
Department of Computer Science - Dartmouth College

7 November 2002

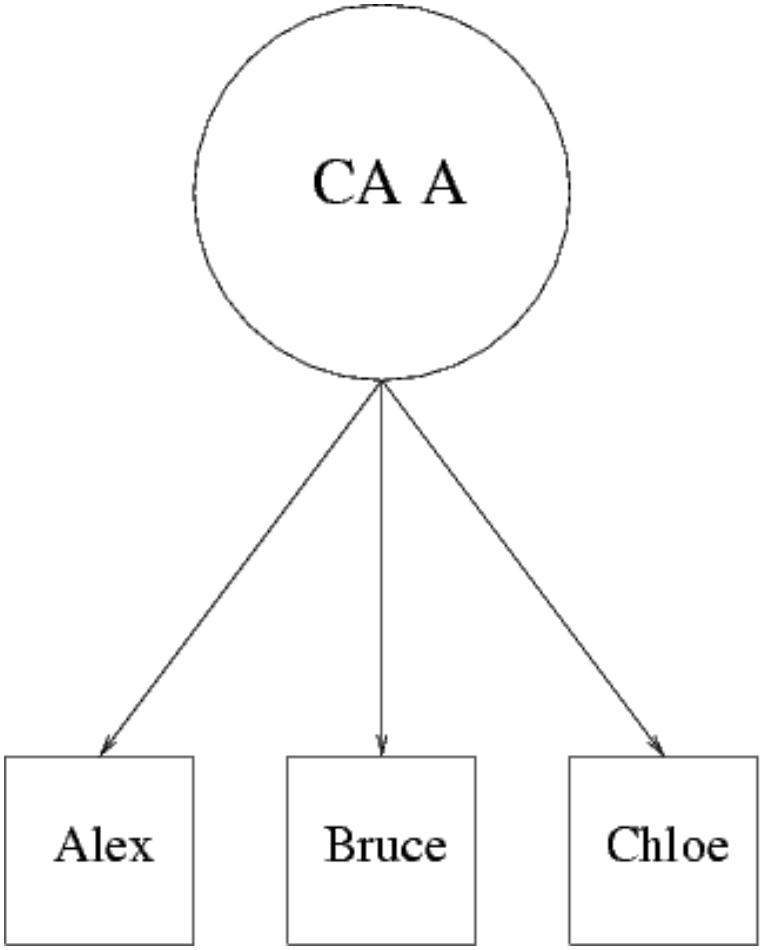
How Do We Find Public Keys?

- Keeping everyone's public key is a nightmare
- Need some infrastructure - i.e. Public Key Infrastructure (PKI)
- Just need the public key of the *Certificate Authority* (CA)
  - ⋆ Issues *signed* statements about the population
  - ⋆ *Certificates* bind a name to a public key
- The *Registration Authority* (RA) identifies keyholders
- *Enterprise PKI* - the CA and RA are run together
- We define CA as the CA and RA together

A Simple Enterprise PKI



What's Wrong With This Picture?



- It's good to be king
- Bad performance
- Which is the right Chloe
- Single point of failure

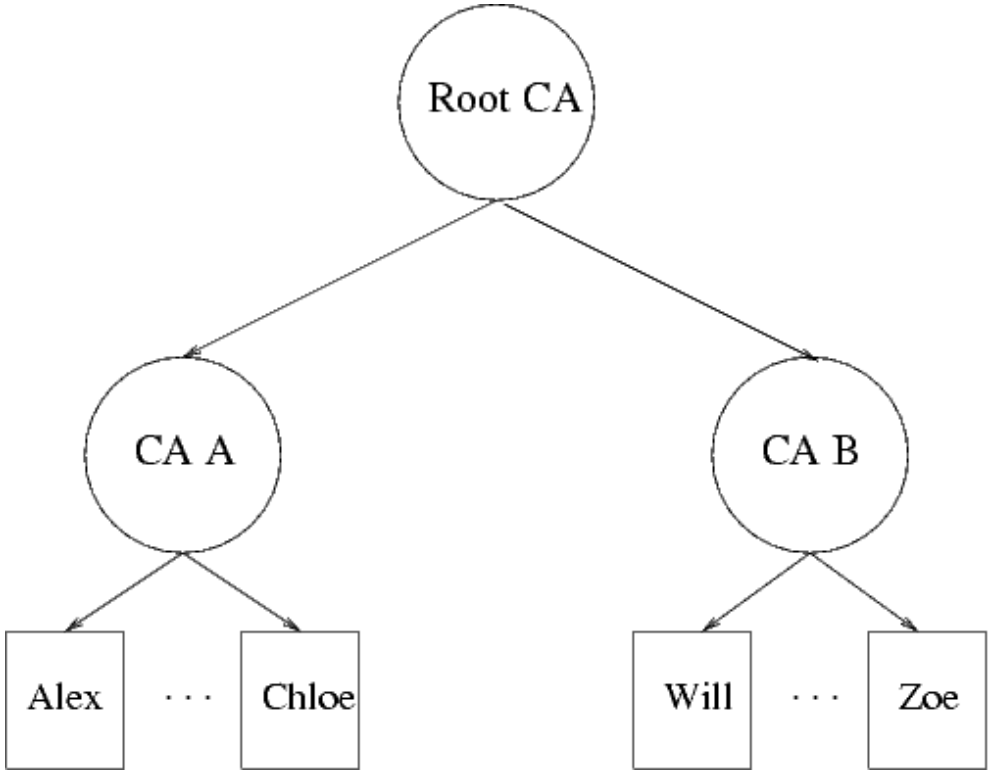
Multiple CAs

- Having more than one CA:
  - ⋆ Eases power struggle
  - ⋆ Cuts namespace $=>$ better performance & fewer collisions
  - ⋆ Isolates failures
  - ⋆ Brings CAs closer to their user populations
- Need a *trust path* (certificate chain) from a trusted CA to target

Evaluation Techniques

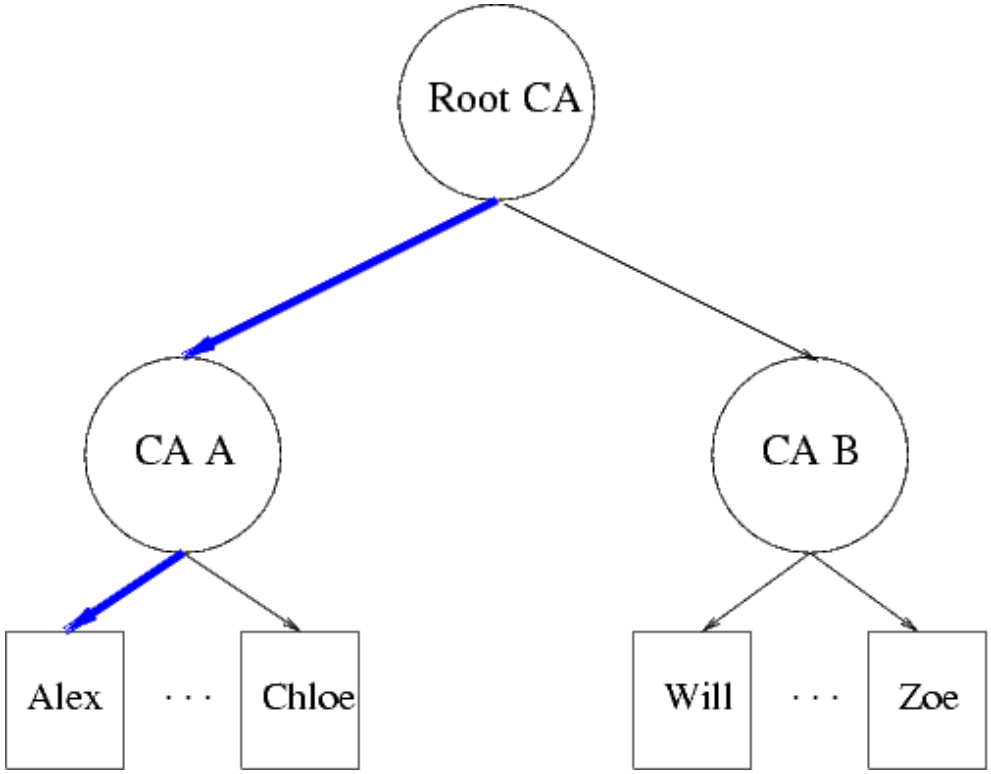
- Our metrics for evaluating PKI architectures:
  1. *Efficiency* - running time of the algorithm which finds a trust path
  2. *Resilience* - ability of the architecture to resist and tolerate key disclosure
- We would like to find an architecture which has both properties

Architecture 1: The Hierarchy



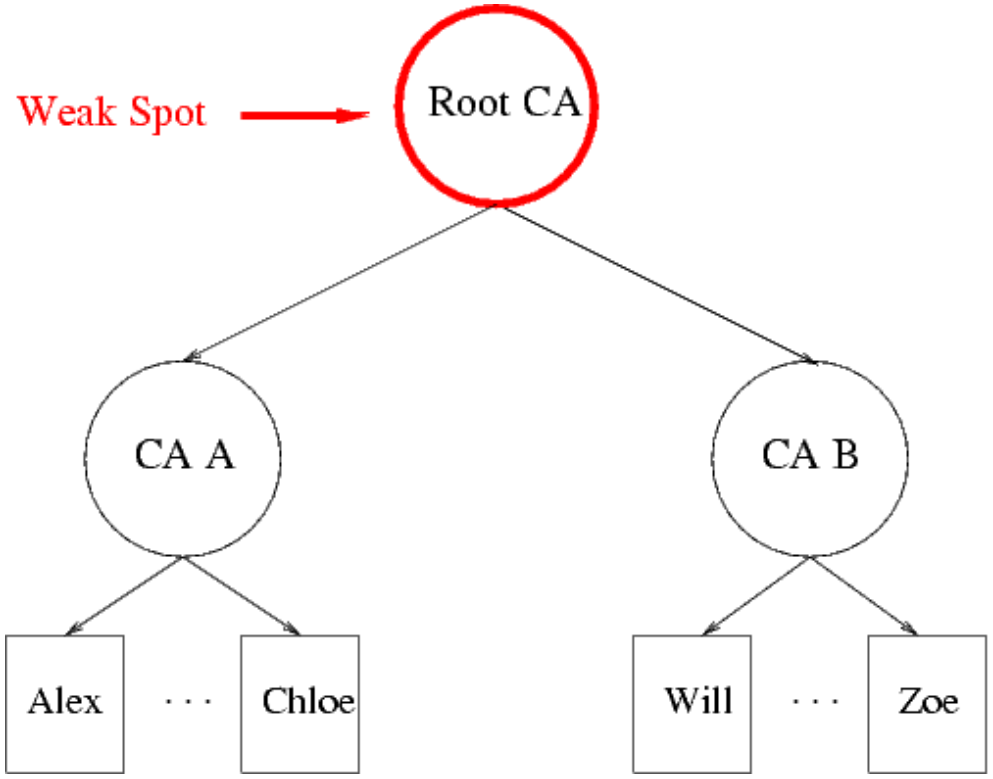
- Pick a Root CA to issue certificates to CA A and CA B
- All the users have a common trust anchor - the Root CA

Is a Hierarchy Efficient?



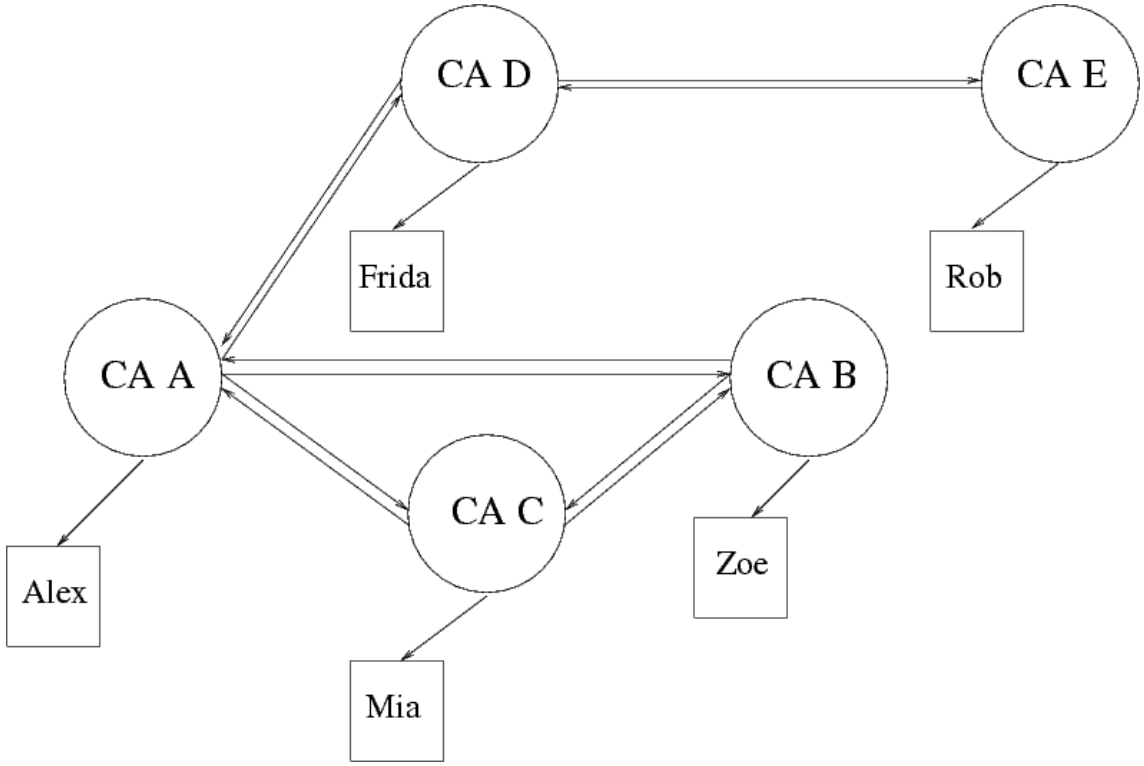
- Efficient - takes $O(logV)$ to find a trust path, where V is the number of CAs in the architecture

Is a Hierarchy Resilient?



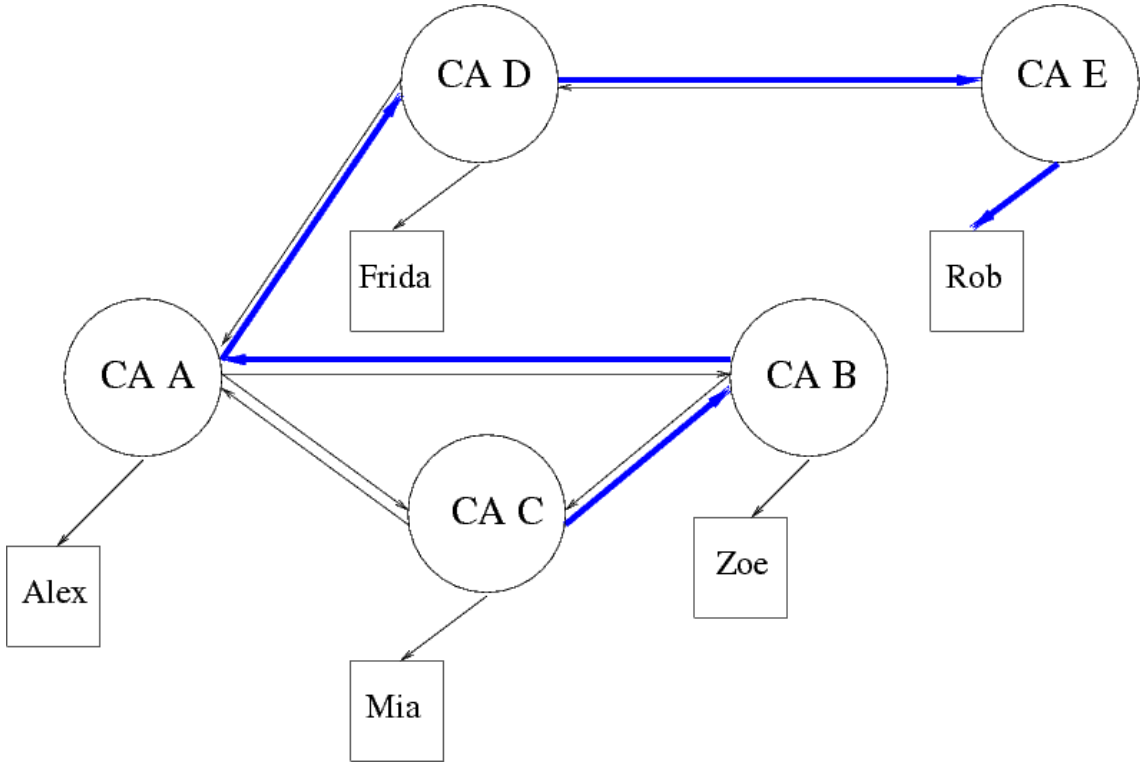
- *Not* Resilient - Root CA's key compromise $=>$ Game Over
- Hierarchies get efficiency at the cost of resilience

Architecture 2: The Mesh



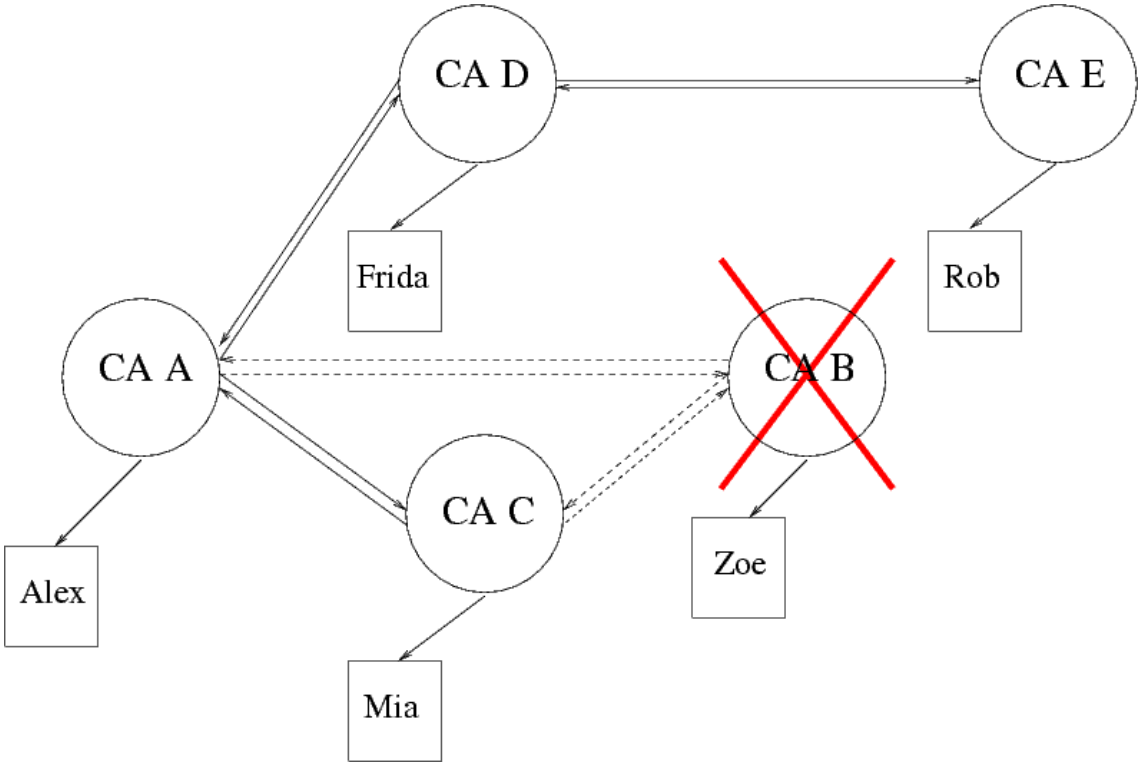
- CAs cross certify (to) one another
- Peer-to-peer $=>$ no common root

Is a Mesh Efficient?



- *Not* efficient - takes $O(V)$ to find a trust path
- Some paths never terminate - topology may have loops

Is a Mesh Resilient?

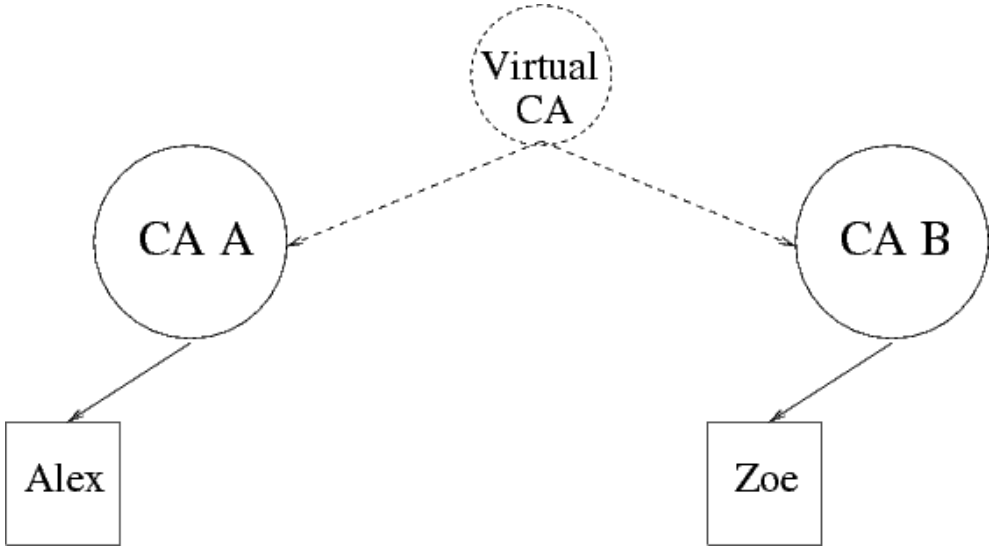


- Resilient - CA key compromise $=>$ it goes offline
- Rest of the architecture still functions fine
- Meshes get resilience at the cost of efficiency

The Problem

- The previous PKI architectures are either efficient OR resilient
- Competing goals
- We want an architecture which is efficient AND resilient

Virtual Hierarchies

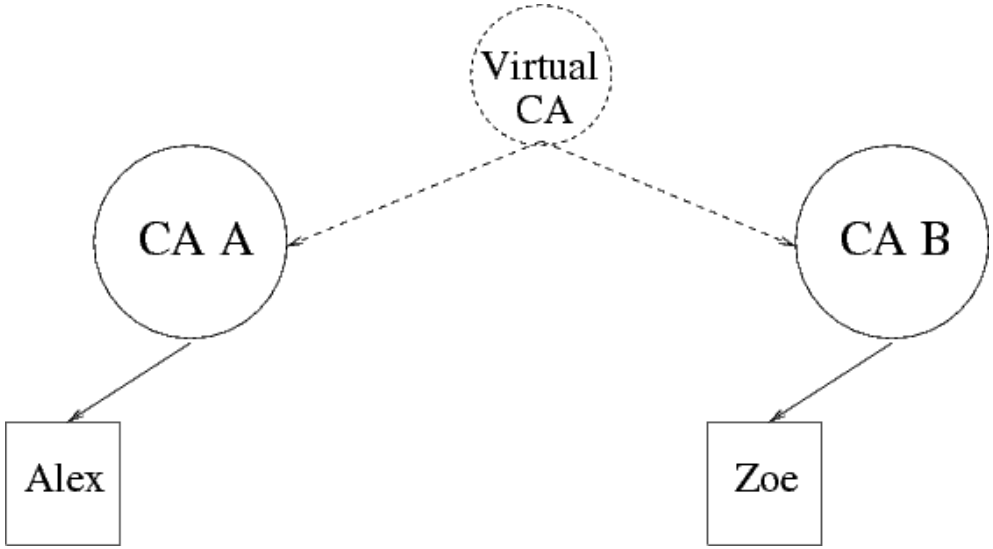


- *Virtual Hierarchy* - logical hierarchy in a peer-to-peer network
- The Virtual CA is a fictional construct
- n peers share the Virtual CA's private key responsibility
- More precisely, a Virtual Hierarchy is a hierarchy of Virtual CAs

Threshold Cryptography

- Require collaboration to use the private key
- n parties share the key responsibility, some threshold k (where $k \leq n$) must act to invoke the private key
- e.g. Multi-Party RSA [e.g. Bel & San], cooperative signature schemes [e.g. Mac & Rei], etc.
- Virtual CA key compromise now requires k successful compromises

Collectives

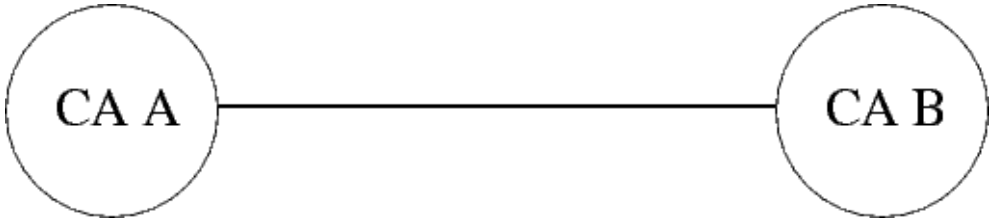


- A *collective* is a set of CAs which have the same Virtual CA as their root

Collectives

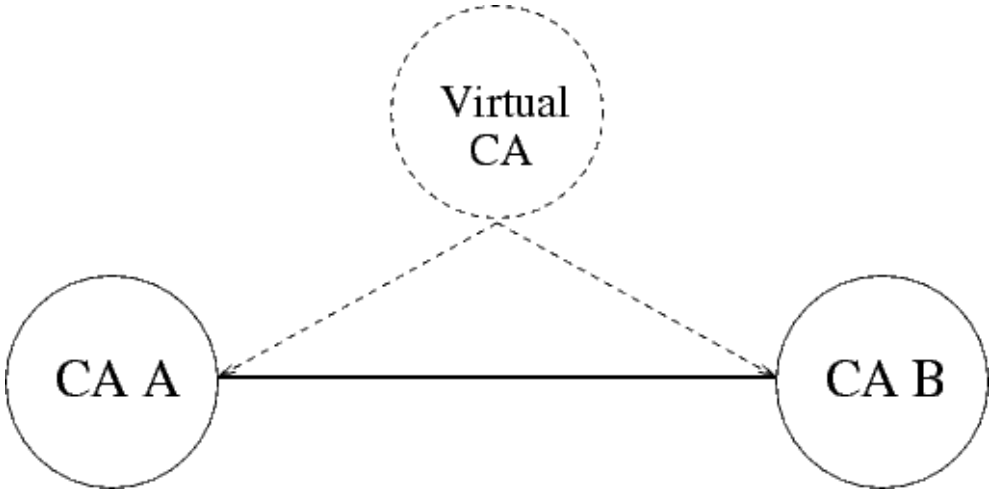
- The story begins with two lonely CAs

Collectives



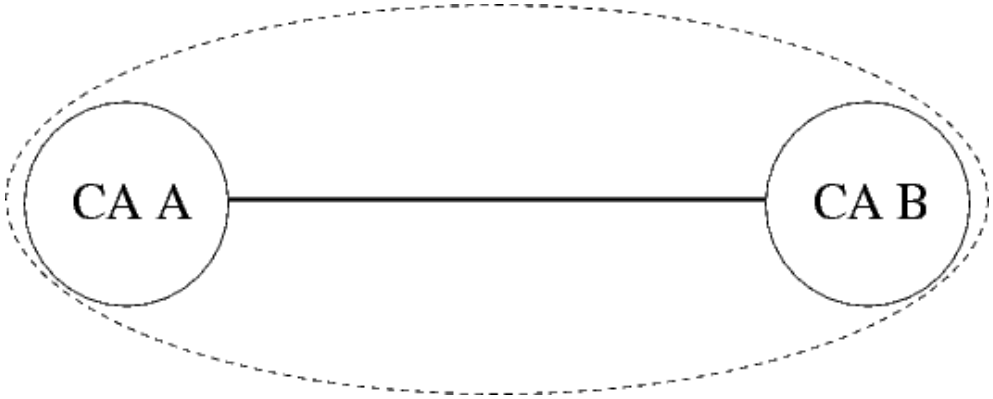
- The CAs connect in the peer-to-peer layer *(hardened Gnutella)*
- All CAs have a cryptographic module
- IBM 4758s - general purpose programming environment
- Gnutella packet logic, routing tables, and secrets inside
- A successful connection $=>$ mutual authentication and an encrypted channel between 4758s

Collectives



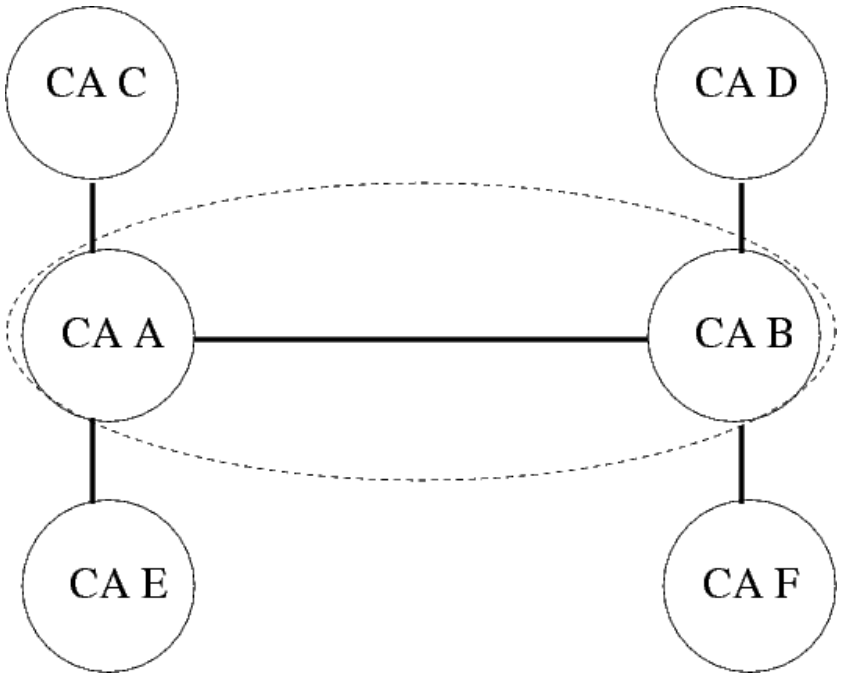
- The CAs establish a key for the Virtual CA and store a share
- At this point, the collective is formed

Collectives



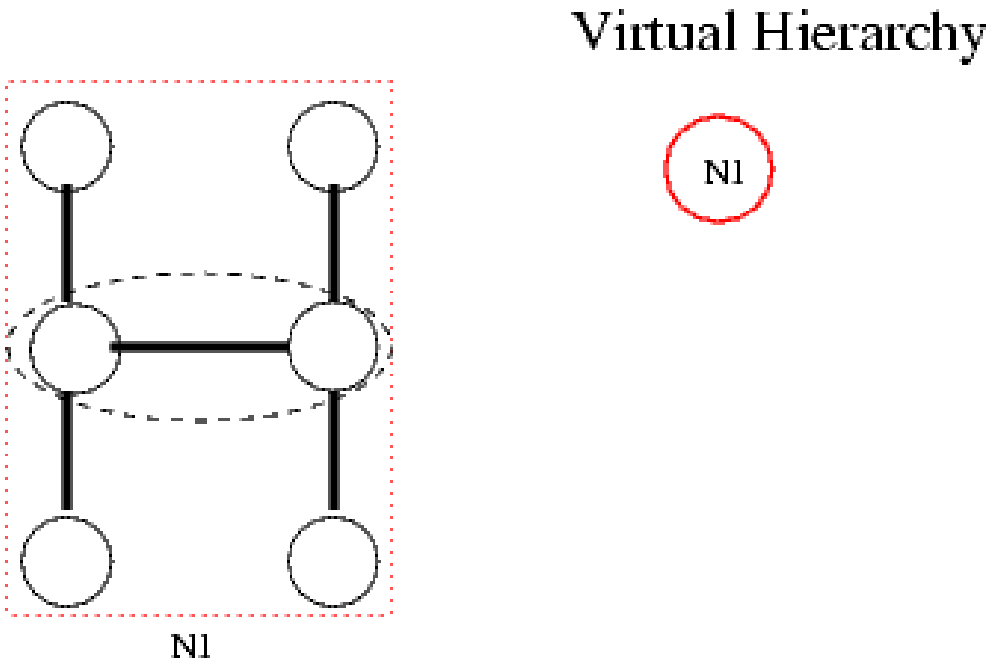
- The CAs in the oval are participating in the key privilege of the Virtual CA

Collective Expansion



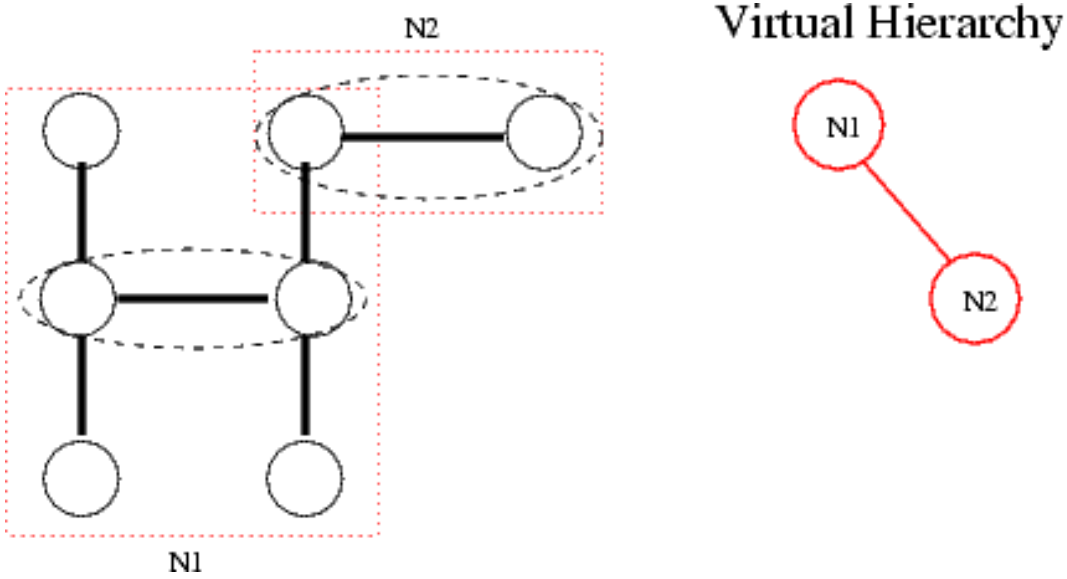
- Other CAs may “use” the Virtual CA without holding a share of the Virtual CA’s key
- They make requests to the Virtual CA by broadcasting
- Small collective size & no loops $=>$ efficient broadcast

The Virtual Hierarchy



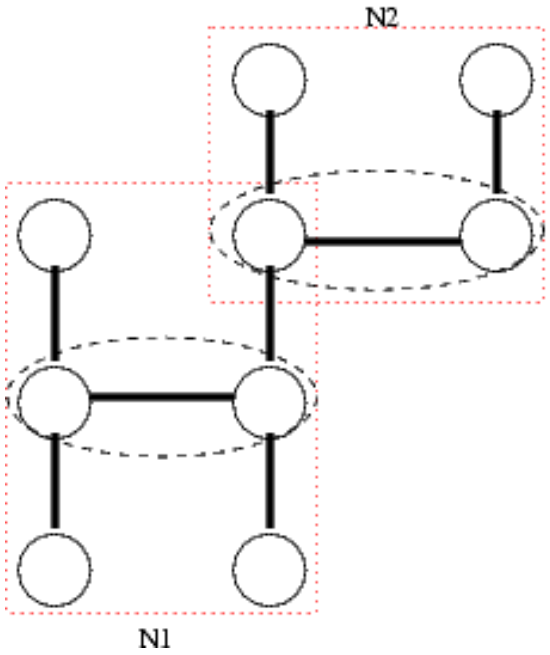
- The Virtual Hierarchy contains one node for this collective $N1$

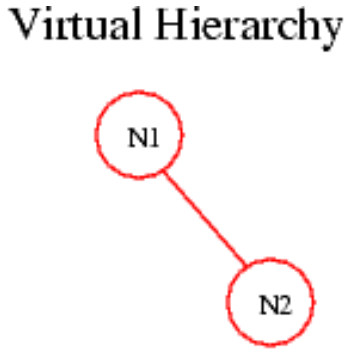
The Virtual Hierarchy



- A new CA connects and begins a new collective $N2$

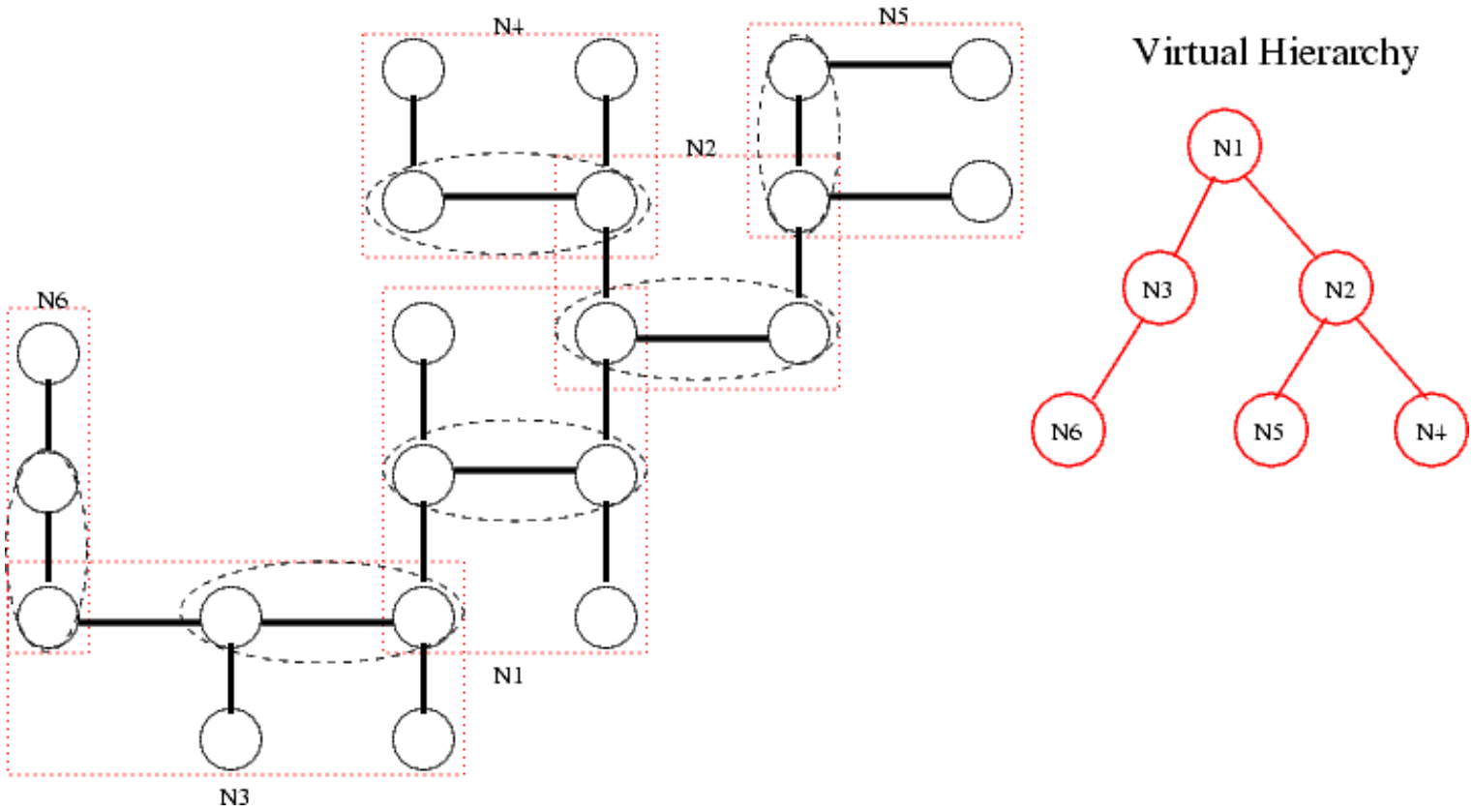
The Virtual Hierarchy





- More CAs join $N2$ and “use” the Virtual CA

The Virtual Hierarchy



- The tree continues to grow downward
- If Virtual Hierarchies join, one collective must be a root

Evaluation

- *Efficient* - We maintain a hierarchical structure $=> O(logV)$ trust path construction (where V is the number of CAs)
- *Resilient* - Threshold cryptography $=> k$ successful compromises are needed to get the Virtual CA key

Summary and Future Work

- We are in the process of prototyping
  - ⋆ Hardened Gnutella in 4758 work is complete
  - ⋆ We have the algorithms which build the hierarchy designed, need to code them
- We are planning to use some of the ideas in other projects (e.g. AXIS (Mellon Foundation), Marianas (NSF))
- What about a more dynamic approach
  - ⋆ Better routing algorithms
  - ⋆ Balance