Sensitive Information in a Wired World

CPSC 457/557, Fall 2013

Lecture 10, October 1, 2013

1:00-2:15 pm; AKW 400

http://zoo.cs.yale.edu/classes/cs457/fall13/

What is a certificate?

- A certificate is a digitally signed statement that binds a public key to some identifying information.
  - The signer of the certificate is called its issuer.
  - The entity talked about in the certificate is the subject of the certificate.
- That's all a certificate is, at the 30,000-foot level.

Certs in the "real world"

- A driver's license is *like* a certificate
  - It is a "signed" document (sealed, tamper-resistant)
  - It is created and signed by an "issuing authority" (the state DMV)
  - It binds together various pieces of identifying information
    - Name
    - License number
    - Driving restrictions (must wear glasses, etc.)

More certs in the real world

- Many physical objects are like certificates:
  - Any type of license (vehicle tabs, restaurant liquor license, amateur radio license, etc.)
  - Government-issued IDs (passports, green cards)
  - Membership cards (Costco, discount cards, etc.)
- All of these examples bind an identity and certain rights, privileges, or other identifiers
  - "BAL ==N1TJT" signed FCC

Why do we believe what certs say?

- In the physical world, why do we trust the statements contained on a physical cert?
  - We believe it's hard to forge the cert
  - We trust the entity that "signed" the cert
- In the digital world, we need analogous properties
  - We need to believe it's hard to forge the digital signature on a signed document
  - We need to trust the issuer/signer not to lie to us

Getting a certificate

- How does Bob get a certificate for his key?
- He goes to a Certificate Authority (CA) that issues certificates and asks for one...
- The CA *issues* Bob a certificate for his public key.
  - CA is the *issuer*
  - Bob is the *subject*

Using Certificates

- Now that Bob has a certificate, is it useful?
- Alice will believe Bob's key belongs to Bob if Alice believes the certificate Bob gives her for his key.
- Alice will believe Bob's key belongs to Bob if Alice trusts the issuer of Bob's certificate to make key-name binding statements
- Have we made the situation any better?

Does Alice Trust Bob's CA?

How can we convince Alice to trust Bob's CA?

- Alice and Bob's CA could have met previously and exchanged keys directly.
  - *Bob's CA isn't going to shake hands with everyone he has certified, let alone everyone whom Bob wants to talk to.*

Does Alice Trust Bob's CA?

How can we convince Alice to trust Bob's CA?

- Alice and Bob's CA could have met previously & exchanged keys directly.
  - *Bob's CA isn't going to shake hands with everyone he's certified, let alone everyone whom Bob wants to talk to.*
- Someone Alice trusts could vouch to her for Bob's CA and Bob's CA's key
  - *Infinite Loop: See Loop, Infinite.*
  - Actually, it's just a bounded recursion...

What is Alice's Trust Model?

- Alice has to implicitly trust *some* set of keys
  - Once she does that, those keys can introduce others to her.
- In the model used by SSL/TLS, CAs are arranged in a hierarchy
  - Alice, like everyone else, trusts one or more "root CA" that live at the top of the hierarchy
- Other models work differently

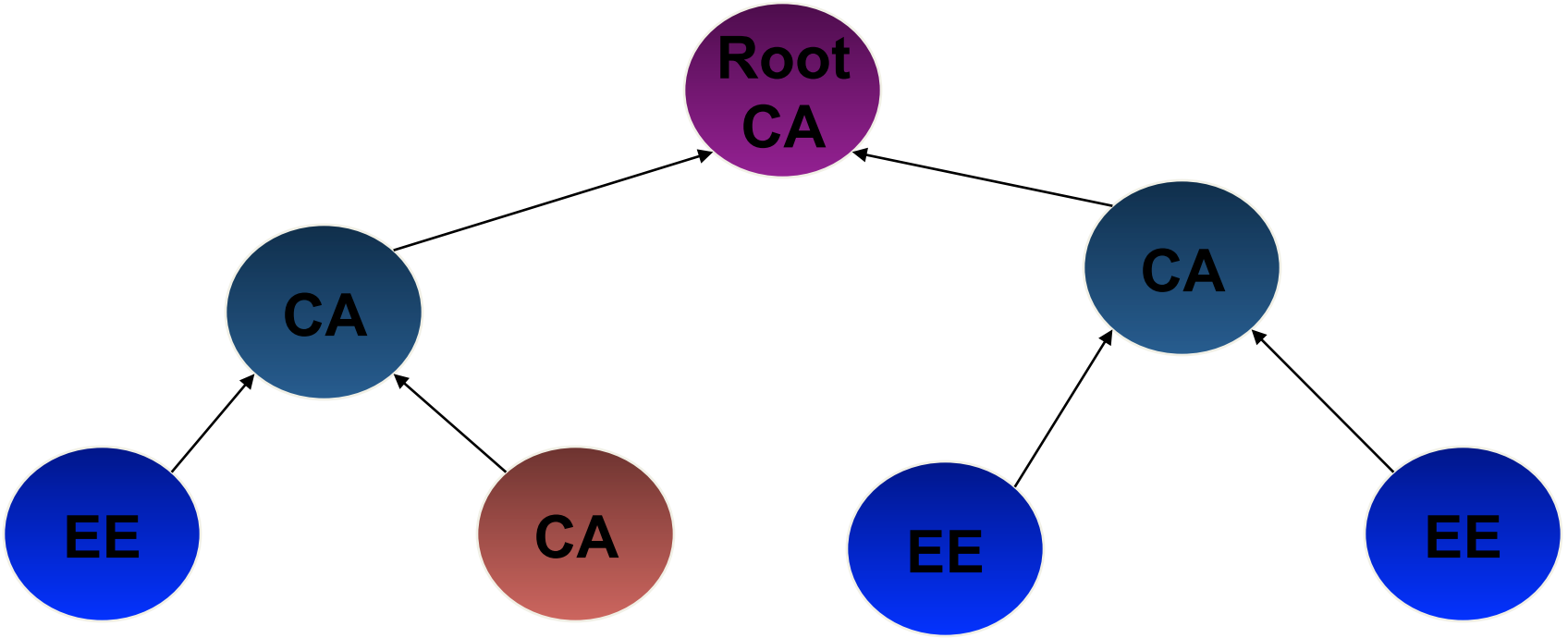
Public Key Infrastructure

Certificate Authorities

- A certificate authority (CA) guarantees the connection between a key and another CA or an "end entity."
- An end entity is:
  - A person
  - A role ("VP of sales")
  - An organization
  - A pseudonym
  - A piece of hardware or software
  - An account
- Some CAs only allow a subset of these types.

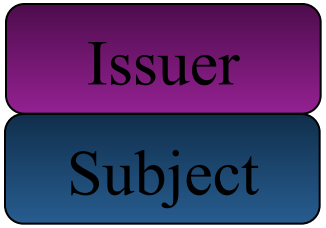
CA Hierarchies

- CAs can certify other CAs or "end entities"
- Certificates are links in a tree of EEs & CAs



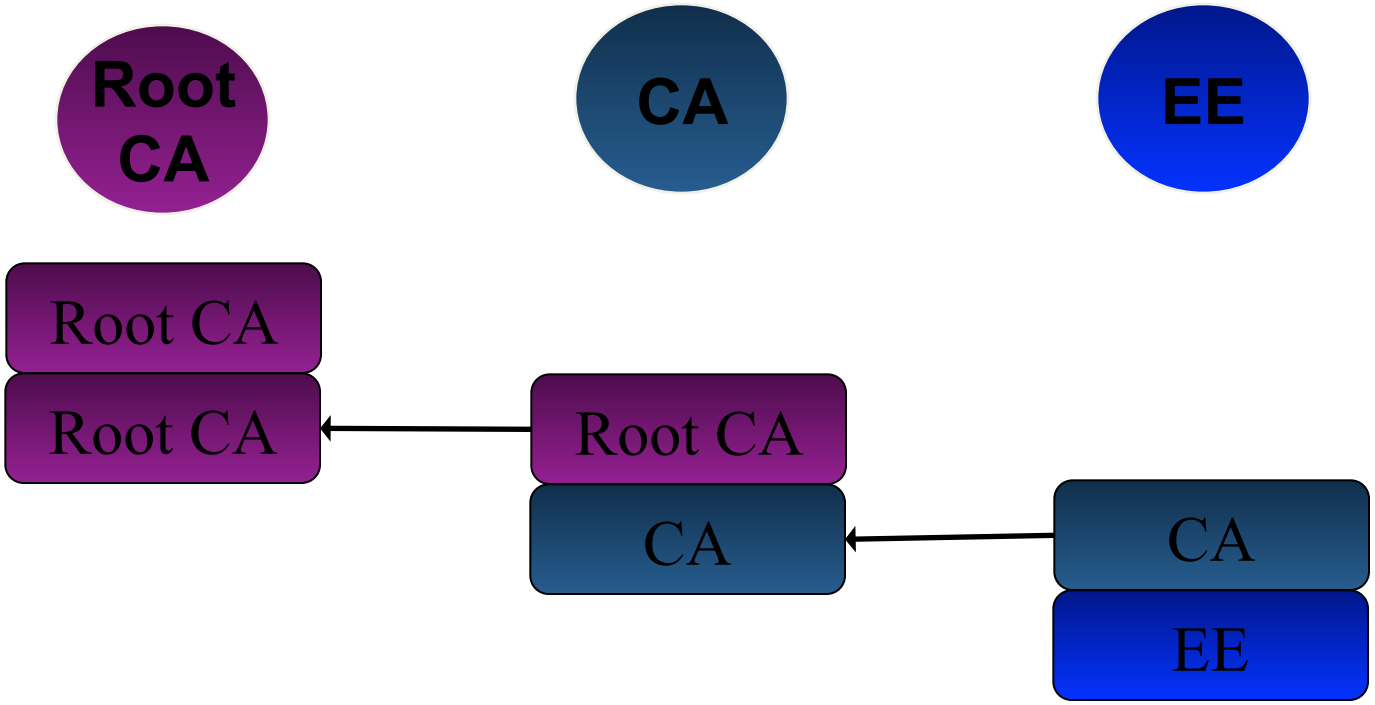
No-Frills Certs

- Certificates can contain all sorts of information inside them
  - We'll talk about the details in a little bit
- In the abstract, though, they're just statements by an issuer about a subject:



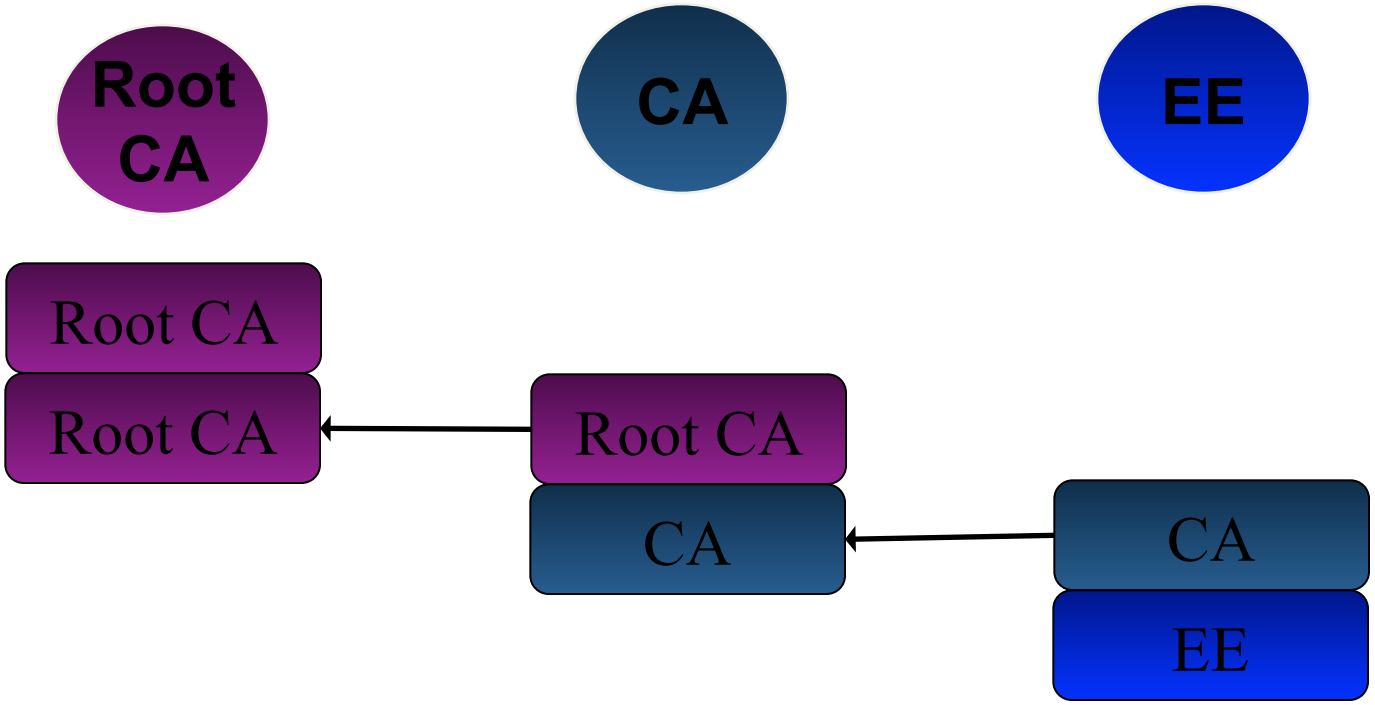
Does Alice trust Bob's Key?

- Alice trusts Bob's key if there is a chain of certificates from Bob's key to a root CA that Alice implicitly trusts

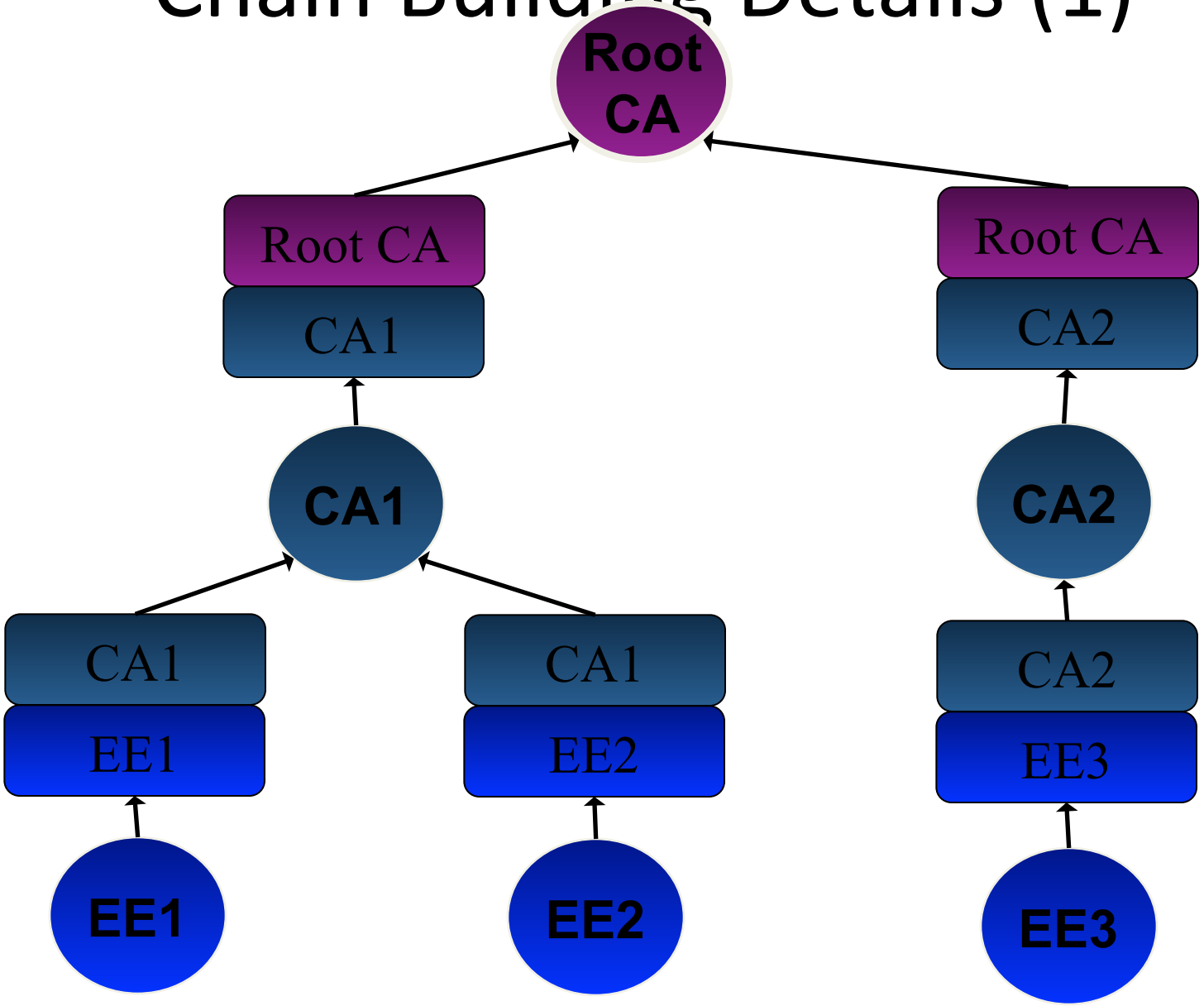


Chain Building & Validation

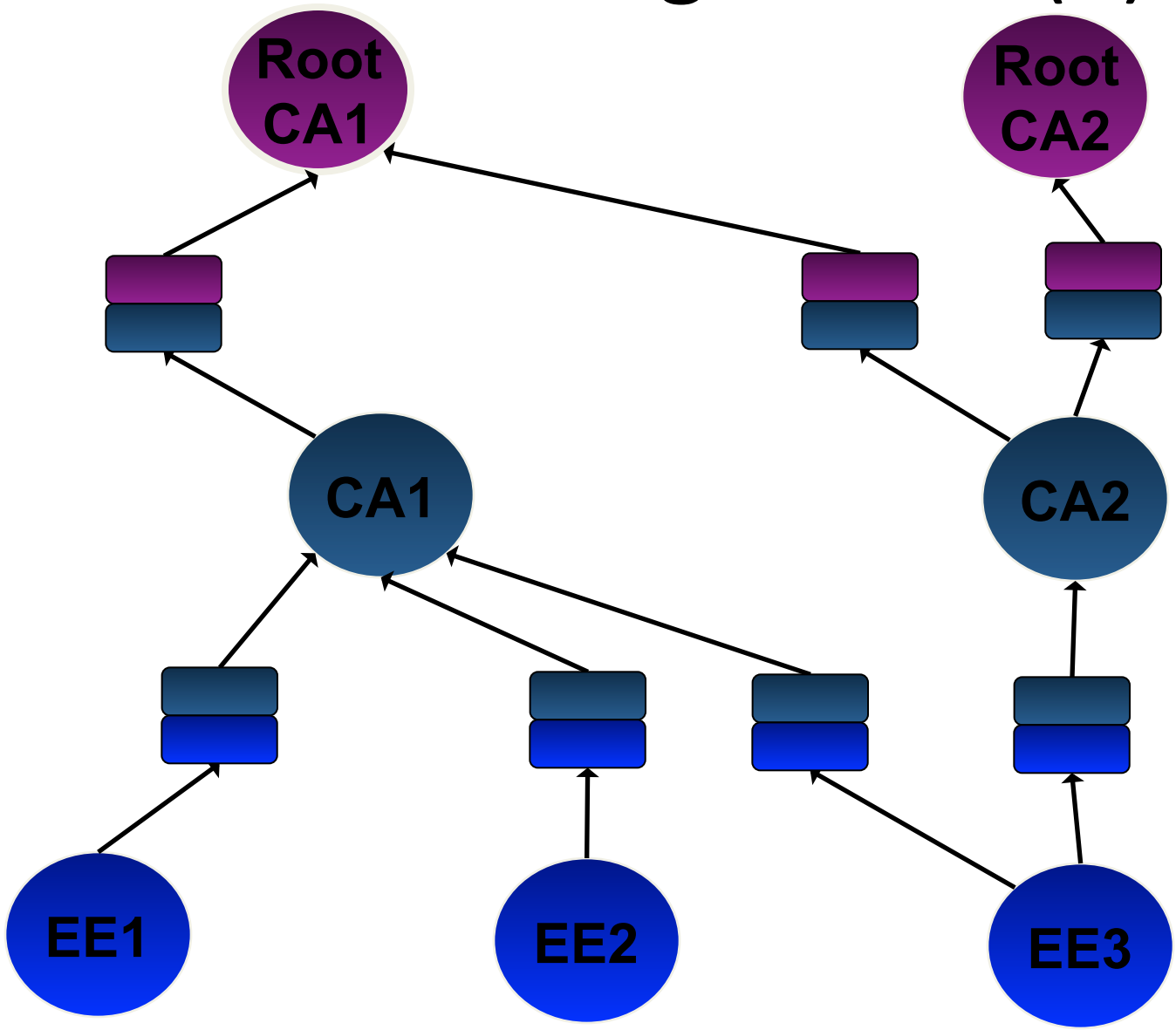
- "Given an end-entity certificate, does there exist a cryptographically valid chain of certificates linking it to a trusted root certificate?"



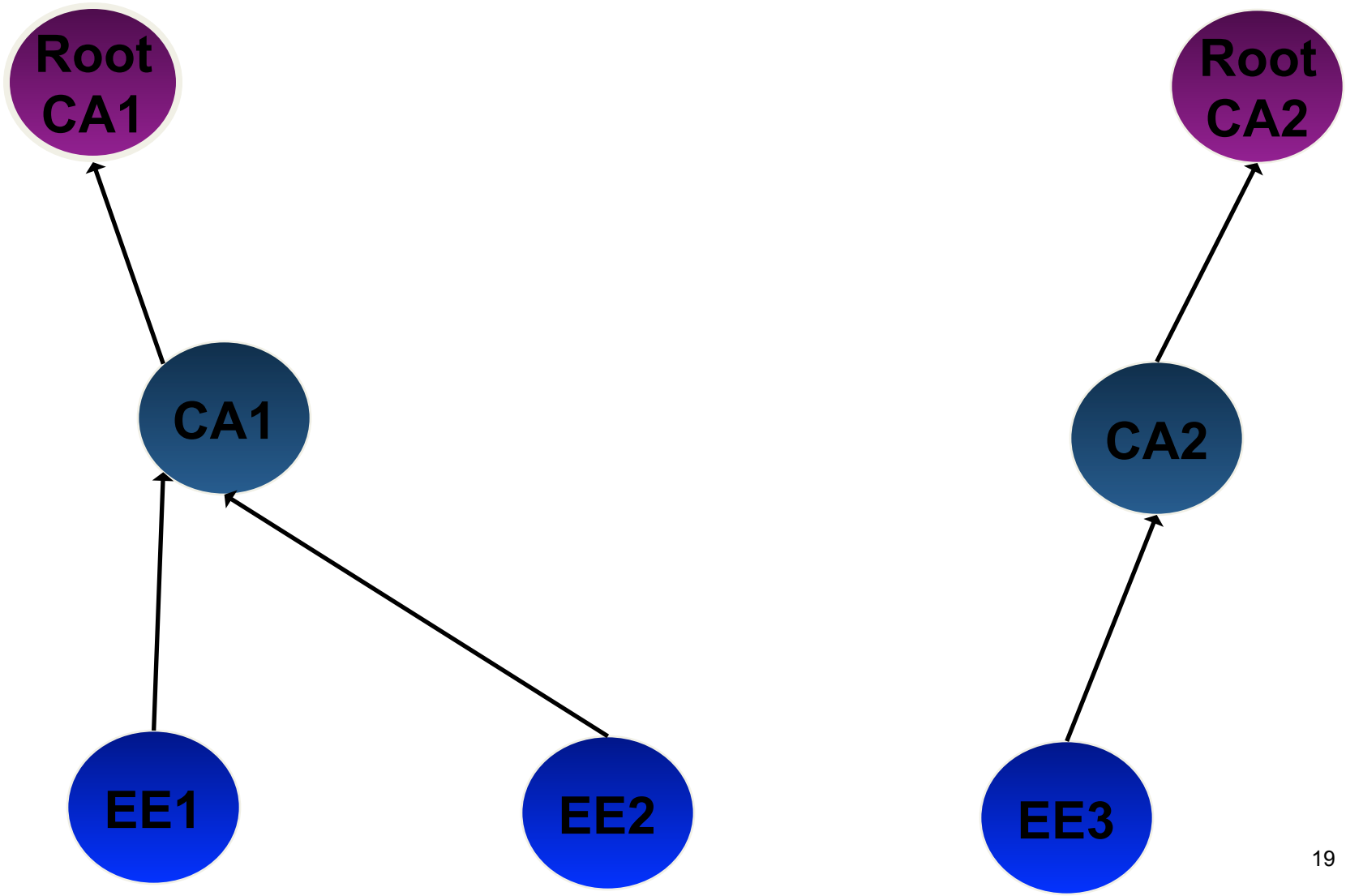
Chain Building Details (1)



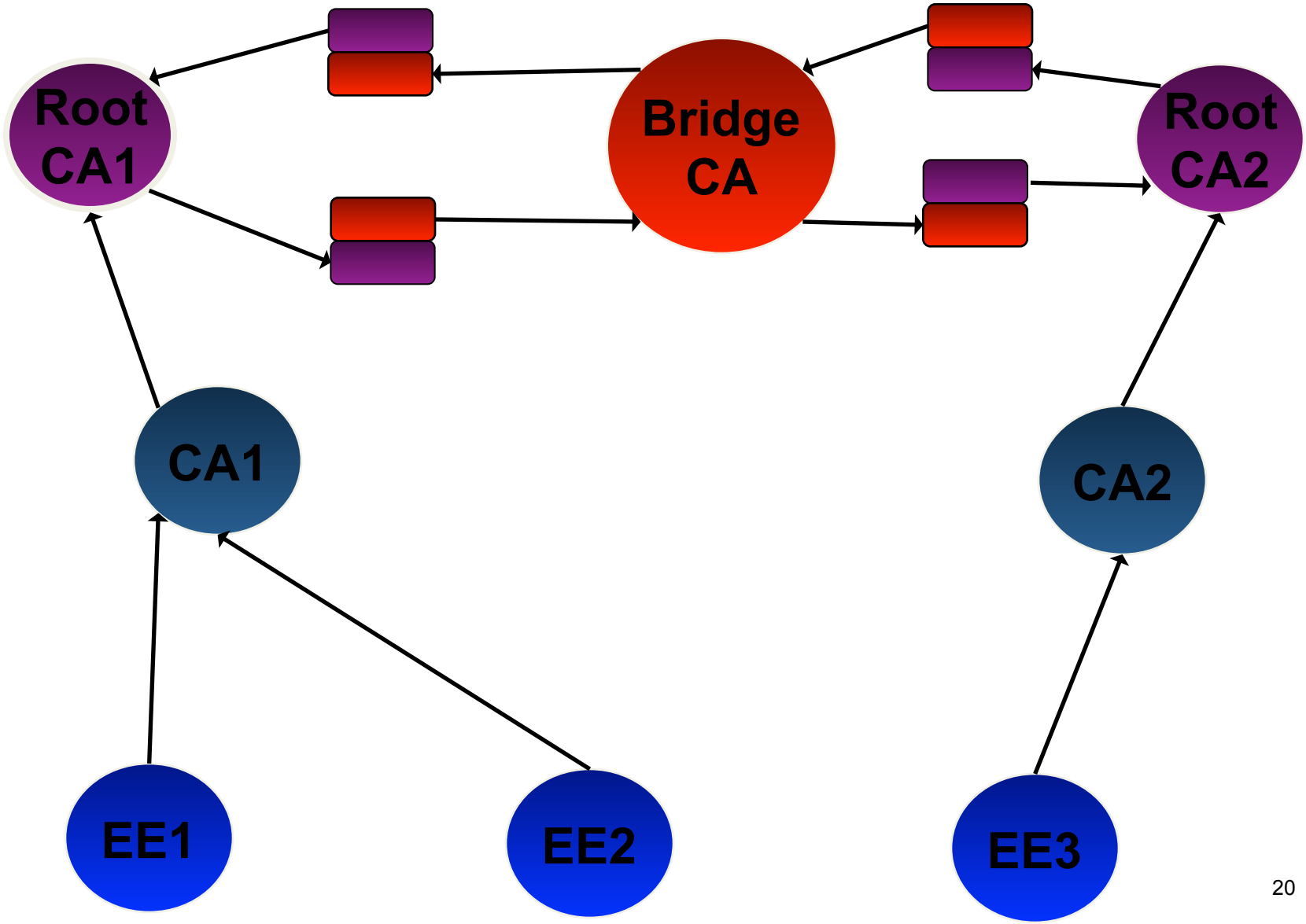
Chain Building Details (2)



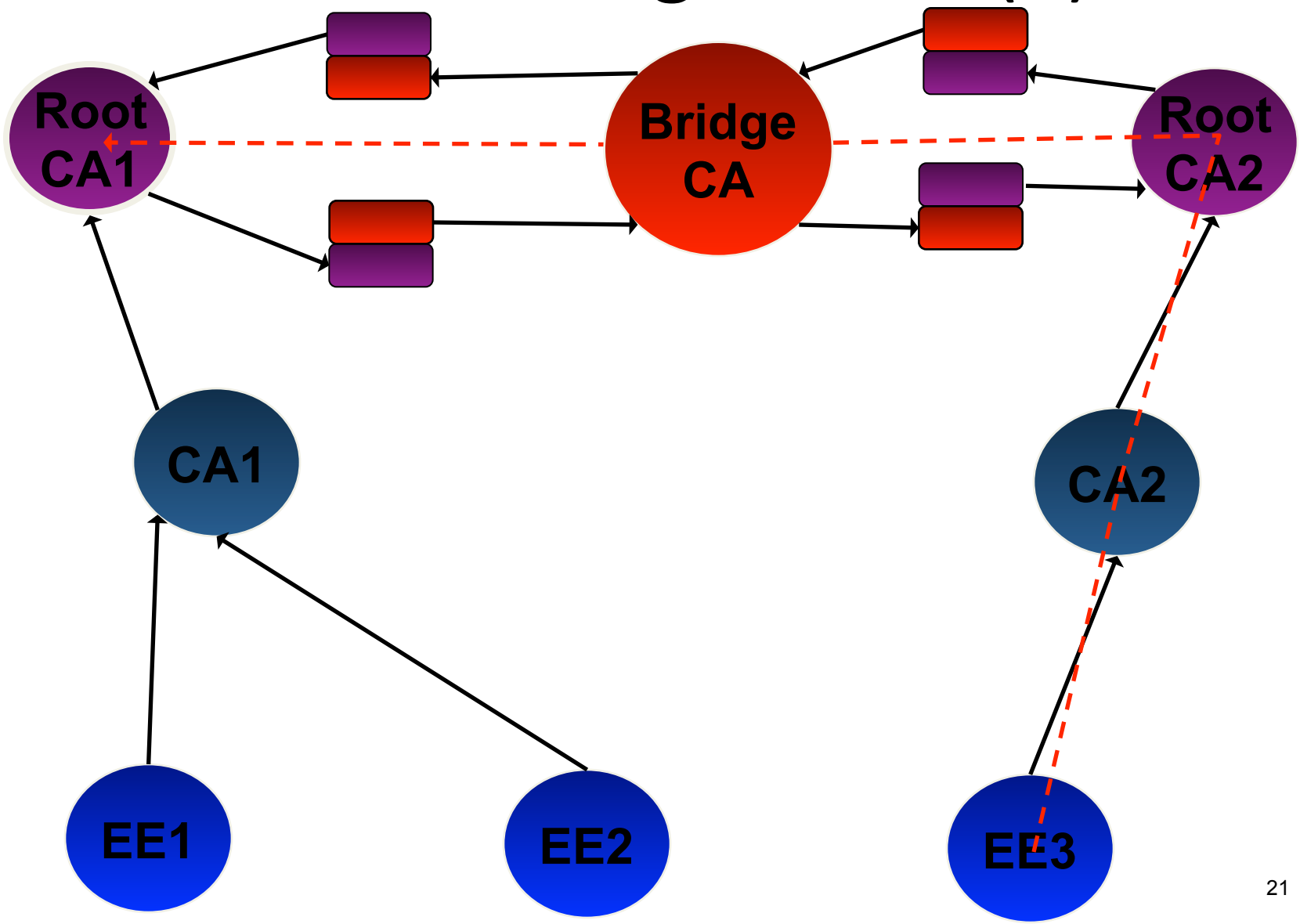
Chain Building Details (3)



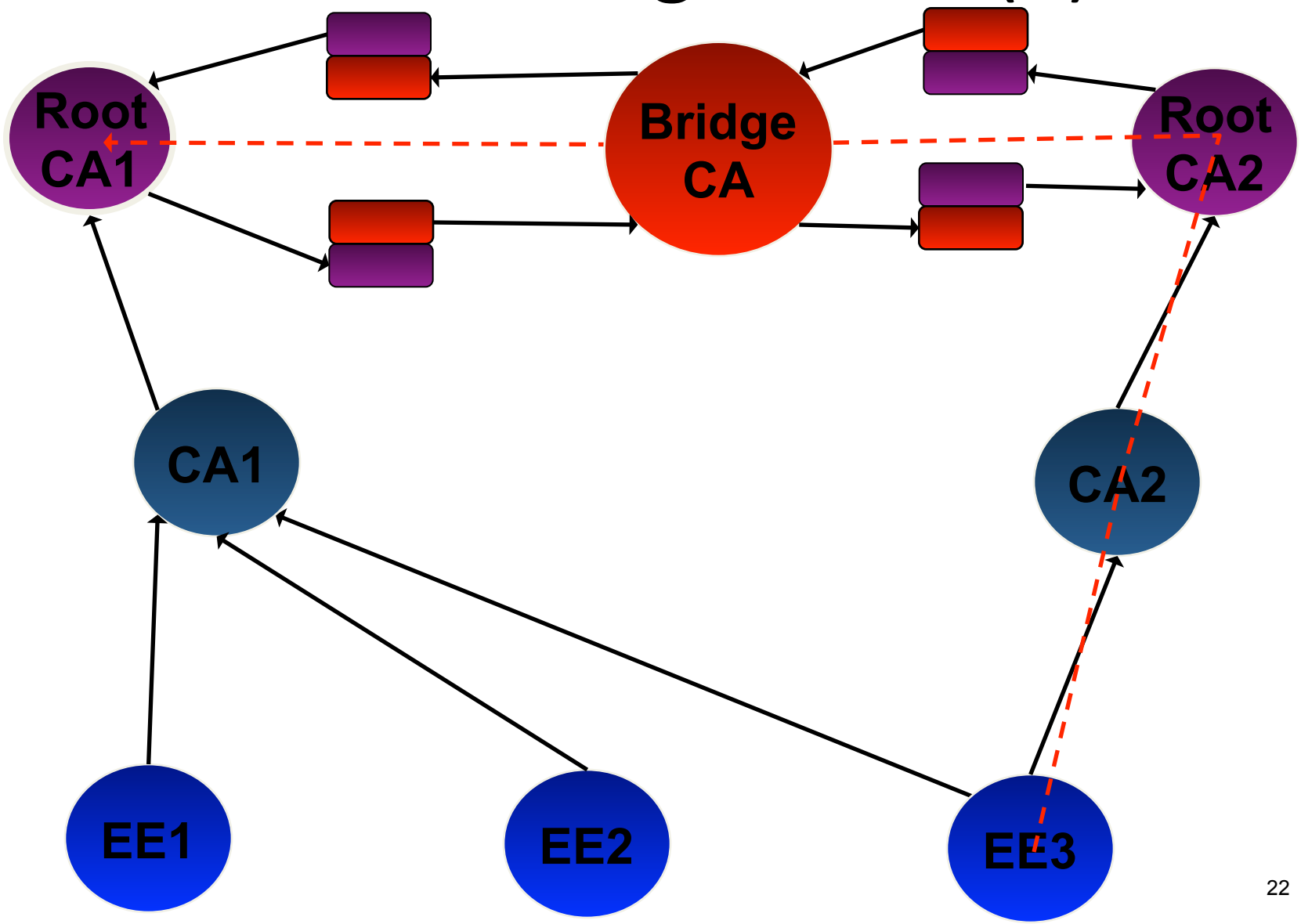
Chain Building Details (3)



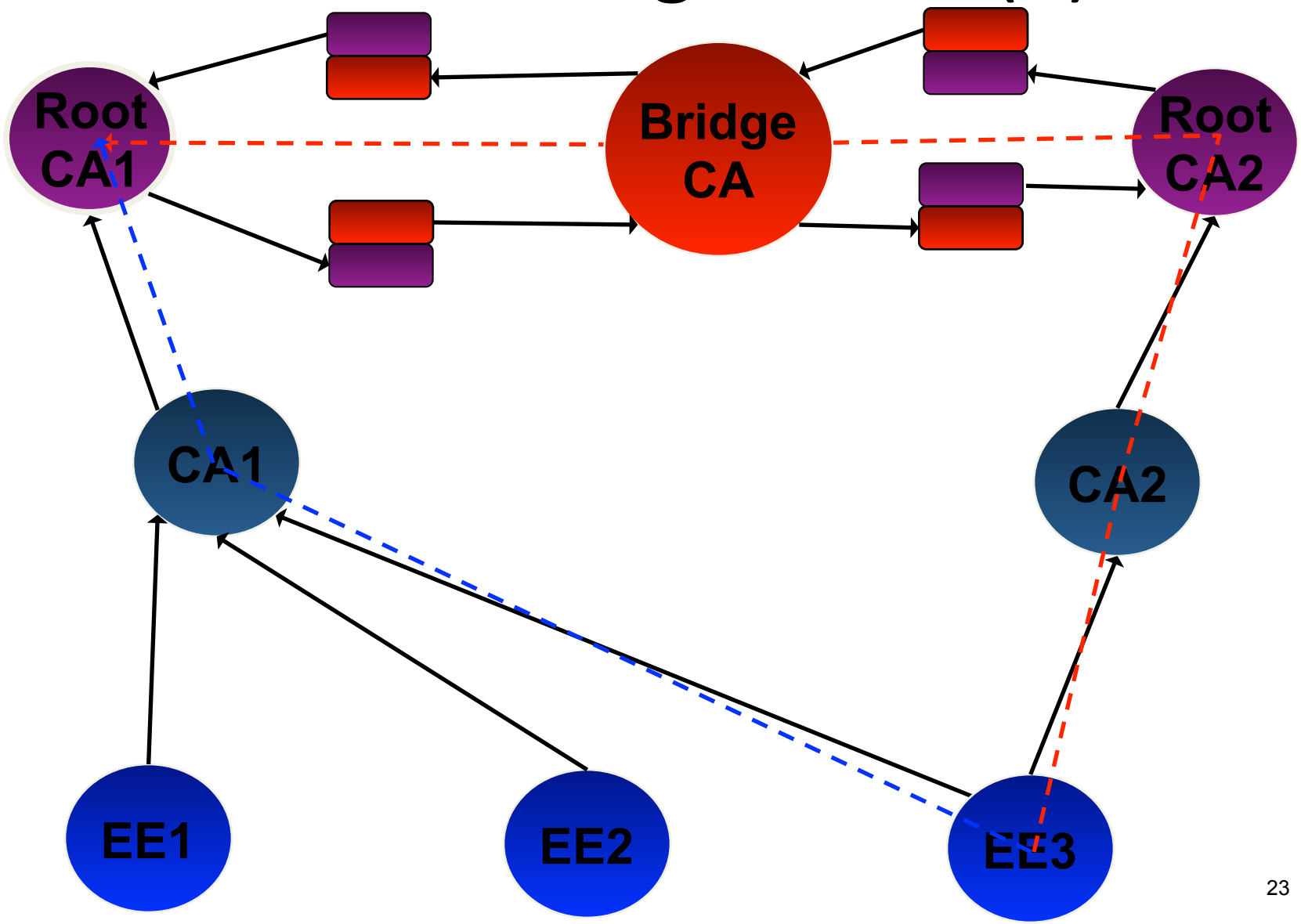
Chain Building Details (3)



Chain Building Details (3)



Chain Building Details (3)



Chaining Certificates

- How do we determine whether two certificates chain together?
  - *You'd think this was an easy problem...*
  - *But it's actually a question with religious significance in the security community*
  - *"Are you a believer in names, or in keys?"*
- In order to understand the schism, we need to digress for a bit and talk about names and some history

PKI Alphabet Soup

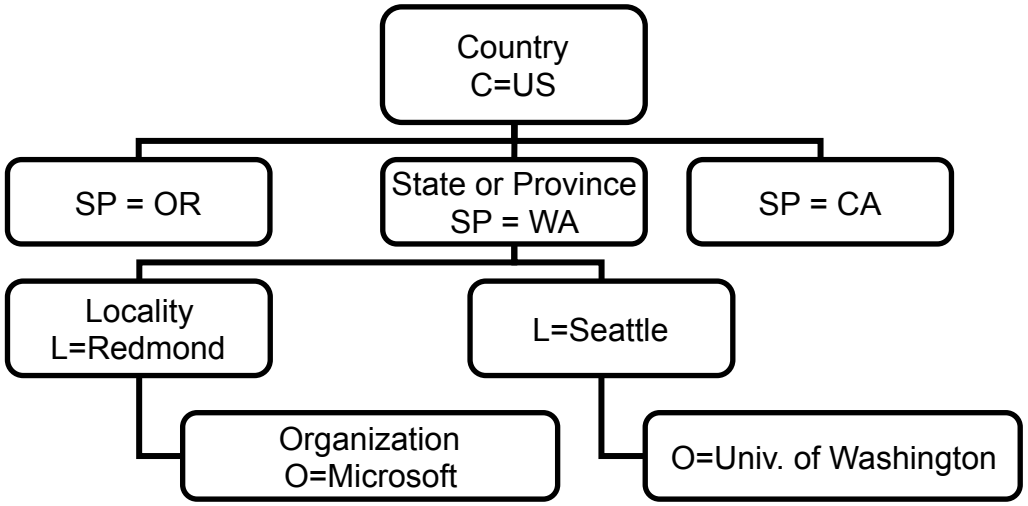
- X.509v3 - standard content of a certificate
- PKIX – IETF Working Group on PKI interoperability
  - PKIX == Public Key Infrastructure using X.509v3 certificates
- ASN.1 - Abstract Syntax Notation, exact description of a certificate format
- DER - Distinguished Encoding Rules, how to physically package a certificate

The X.500 Directory Model

- The model SSL/TLS uses, the X.509 certificate model, is based on names
  - *Names as principles*
- Specifically, X.509 is based on the X.500 directory model
- X.500 defined a global, all-encompassing directory, to be run by the telcos
  - *One directory to rule them all, one directory to define them...*

X.500 Distinguished Names

- In the X.500 model, everything has a single, unique, global, assigned name
  - There is a worldwide hierarchy, and you're in it!



DNs in Practice

- Name is unique within the scope of the CA's name
- Public CAs (e.g., Verisign) typically set
  - C = CA Country
  - O = CA Name
  - OU = Certificate type/class
  - CN = User name
  - E= email address

Private-label DNs

- If you own the CA, you get to decide what fields go in the DN
  - Really varies on what the software supports
- Can get really strange as people try to guess values for fields that are required by software
  - Software requires an OU, we don't have OUs, so I better make something up!

DNs in X.509 Certificates

- The X.509 certificate standard began as a way to associate a certificate with a node in the directory.
- How is the subject of a cert identified?
  - By its DN.
- How is the issuer of a cert identified?
  - By its DN.
- How are certificates linked together?
  - By DNs.

Key fields in a certificate

- The core fields of an X.509 certificate are
  - The subject public key
  - The subject Distinguished Name
  - The issuer Distinguished Name
- What's missing here?

Key fields in a certificate

- The core fields of an X.509 certificate are
  - The subject public key
  - The subject Distinguished Name
  - The issuer Distinguished Name
- What's missing here?
  - The issuer's public key is not present in the certificate.
  - You can't verify the signature on the cert without finding a parent cert!

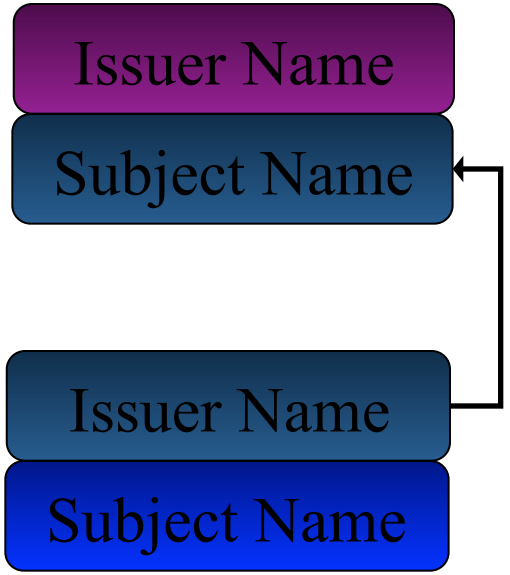
Back to Chain Building

- OK, assume we're a "relying party application" -- something that received an end-entity certificate and wants to verify it.
  - Our task is to build a cert chain from that end-entity cert to one of our trusted roots
- How do we do that?
  - We start with our EE cert, and using the information contained within we look for possible parent certificates.

Parent certs

- What's a valid parent certificate?
  - In the raw X.509 model, parent-child relationships are determined solely by matching Issuer DN in the child to Subject DN in the parent
  - Recall that there's an assumption that you have a big directory handy to find certs.
- If you don't have a directory handy, you need to do the matching yourself
  - This is not as easy as you might think…

Name matching



Matching Names

- How do we determine if two DNs match?
  - "Use directory name matching rules!"
  - Try to be mildly smart about it
    - Remove spaces, case-fold, etc.
    - Disaster…
  - Try to be really dumb about it
    - Exact binary match
    - Less of a disaster, but there are still problems we can't work around…

Unicode Names

- Are these two character equal?

  é é

- They look equal…

Unicode Names

- Are these two character equal?

  é é

- They look equal…
- …but may not be
- In Unicode, you can compose characters, so:
  - “é” as one character
  - “é” as two characters – “e” followed by non-spacing accent
  - “é” as two characters – non-spacing accent followed by “e”
- Ick!

Even More Chain Building

- Name matching is just the beginning of the chain-building process
  - It is necessary that subject and issuer DNs exactly match for two certs to chain, but not always sufficient
- The chain building process is also influenced dynamically by other information contained within the certs themselves
  - *What other information is there in certs?*

Trusted Root Certificates

- Who do I trust to be roots at the top of the cert chain?
- In theory, "anyone you want"
- In practice, trusted roots come from two sources
  - They're baked into your web browser or operating system
  - They're pushed onto your "enterprise managed desktop"

More Aspects of PKI

- X.509 Cert "extensions"
  - Usage extensions
  - Constraint extensions
- Cert life-cycle management
  - Validity / expiration
  - Revocation!
  - Online status checking

❖ Certs not based on names

- "Trust management" need not be cert management