CNT4406/5412 Network Security

PKI (Public Key Infrastructure)

Zhi Wang

Florida State University

Fall 2014

Introduction

Public key infrastructure is a system to securely distribute public keys.

Introduction

Public key infrastructure is a system to securely distribute public keys.

It consists of:

Introduction

Public key infrastructure is a system to securely distribute public keys.

It consists of:

- certificates
  - ➠ X.509 is the specification for a certificate

Introduction

Public key infrastructure is a system to securely distribute public keys.

It consists of:

- certificates
  ➠ X.509 is the specification for a certificate
- a repository for retrieving certificates

Introduction

Public key infrastructure is a system to securely distribute public keys.

It consists of:

- certificates
  ➠ X.509 is the specification for a certificate
- a repository for retrieving certificates
- a method of revoking certificates

Introduction

Public key infrastructure is a system to securely distribute public keys.

It consists of:

- certificates
  ➠ X.509 is the specification for a certificate
- a repository for retrieving certificates
- a method of revoking certificates
- a method of evaluating a chain of certificates from **trust anchors** to the **target** name

Terminology

- A **certificate** is a signed message vouching a name with a public key
  - [Bob's public key is 829248]$_{Alice}$
  - Alice is the **issuer**, and Bob is the **subject**

Terminology

- A **certificate** is a signed message vouching a name with a public key
  - ➠ [Bob's public key is 829248]$_{Alice}$
  - ➠ Alice is the **issuer**, and Bob is the **subject**
- A **principal** is anything that has a public key

Terminology

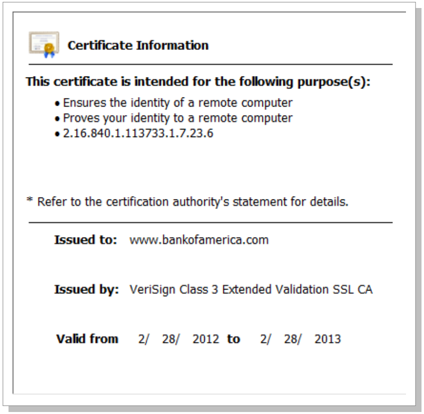
- A **certificate** is a signed message vouching a name with a public key
  - ➠ [Bob's public key is 829248]$_{Alice}$
  - ➠ Alice is the **issuer**, and Bob is the **subject**
- A **principal** is anything that has a public key
- If Alice wants to find a path to Bob's key
  - ➠ Bob is the **target**, Alice is the **verifier**

Terminology

- A **certificate** is a signed message vouching a name with a public key
  - [Bob's public key is 829248]$_{Alice}$
  - Alice is the **issuer**, and Bob is the **subject**
- A **principal** is anything that has a public key
- If Alice wants to find a path to Bob's key
  - Bob is the **target**, Alice is the **verifier**
- A **trust anchor** is a public key that is trusted to sign a certificate

X.509 Certificate: Example

A certificate for Bank of America signed by VeriSign



Certificate Information

This certificate is intended for the following purpose(s):

- Ensures the identity of a remote computer
- Proves your identity to a remote computer
- 2.16.840.1.113733.1.7.23.6

* Refer to the certification authority's statement for details.

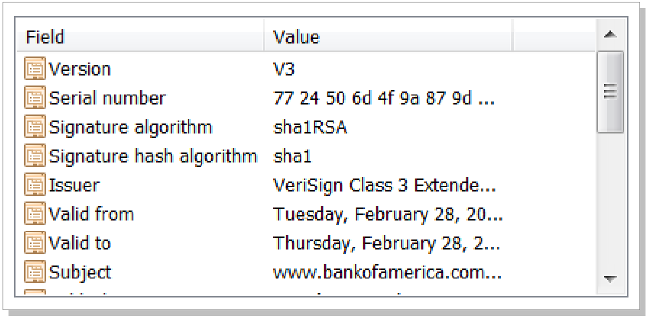
Issued to: www.bankofamerica.com

Issued by: VeriSign Class 3 Extended Validation SSL CA

Valid from 2/ 28/ 2012 **to** 2/ 28/ 2013

X.509 Certificate: Example

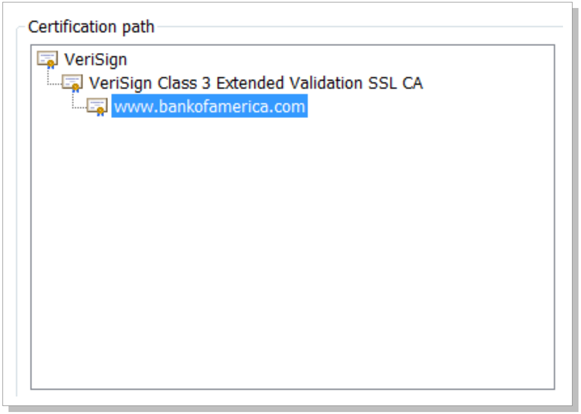
Details for the BOA certificate



Field	Value
Version	V3
Serial number	77 24 50 6d 4f 9a 87 9d ...
Signature algorithm	sha1RSA
Signature hash algorithm	sha1
Issuer	VeriSign Class 3 Extende...
Valid from	Tuesday, February 28, 20...
Valid to	Thursday, February 28, 2...
Subject	www.bankofamerica.com...

X.509 Certificate: Example

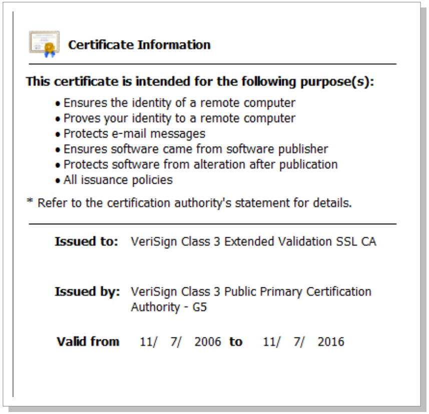
Verification path for the BOA certificate


Certification path

VeriSign
VeriSign Class 3 Extended Validation SSL CA
www.bankofamerica.com

X.509 Certificate: Example

A certificate from VeriSign signed by **VeriSign** (why?)



Certificate Information

This certificate is intended for the following purpose(s):

- Ensures the identity of a remote computer
- Proves your identity to a remote computer
- Protects e-mail messages
- Ensures software came from software publisher
- Protects software from alteration after publication
- All issuance policies

* Refer to the certification authority's statement for details.

Issued to: VeriSign Class 3 Extended Validation SSL CA

Issued by: VeriSign Class 3 Public Primary Certification Authority - G5

Valid from 11/ 7/ 2006 **to** 11/ 7/ 2016

X.509 Certificate Format

- Basic fields:

Field	Description	Example
version	X.509 certificate version	V3
serial number	unique id for the certificate in the CA	`77 24 50 6d...`
signature	algorithm used to signed the certificate	sha1 RSA
issuer	X.500 name of the issuer	VeriSign Class 3 ...
validity from	starting date of the certificate	02/28/2012 8:00:00 PM
validity to	end date of the certificate	02/28/2013 8:00:00 PM
subject	X.500 name of the subject	www.bankofamerica.com
public key	public key of the subject	RSA `30 82 01 0a ...`

*CA may have multiple keys

X.509 Certificate Format

- Basic fields:

Field	Description	Example
version	X.509 certificate version	V3
serial number	unique id for the certificate in the CA	`77 24 50 6d...`
signature	algorithm used to signed the certificate	sha1 RSA
issuer	X.500 name of the issuer	VeriSign Class 3 ...
validity from	starting date of the certificate	02/28/2012 8:00:00 PM
validity to	end date of the certificate	02/28/2013 8:00:00 PM
subject	X.500 name of the subject	www.bankofamerica.com
public key	public key of the subject	RSA `30 82 01 0a ...`

- Optional fields: basic constrains, key usage, CRL distribution points, authority key identifier*, subject key identifier...

*CA may have multiple keys

PKI Trust Models

PKI trust models defines where to get trust anchors, and how to evaluate a chain of certificates from trust anchors to the target

PKI Trust Models

PKI trust models defines where to get trust anchors, and how to evaluate a chain of certificates from trust anchors to the target

1. Monopoly model
2. Monopoly + RAs
3. Delegated CAs
4. Oligarchy model
5. Anarchy model
6. Top-down with name constraints
7. Bottom-up with name constraints

Monopoly Model

- One CA is universally trusted by the world
  ➠ everyone must get certificates from the (only) CA

Monopoly Model

- One CA is universally trusted by the world
  ➠ everyone must get certificates from the (only) CA
- The CA's public key is embedded in all software and hardware
  ➠ the only trust anchor

Problems of Monopoly Model

Problems of Monopoly Model

- There is NO universally trusted organization

Problems of Monopoly Model

- There is NO universally trusted organization
- Monopoly control $\rightarrow$CA can charge whatever it wants for a certificate

Problems of Monopoly Model

- There is NO universally trusted organization
- Monopoly control $\rightarrow$CA can charge whatever it wants for a certificate
- Entire security of the world rests on the CA
  ➠ infeasible to change the CA's key everywhere if compromised

Problems of Monopoly Model

- There is NO universally trusted organization
- Monopoly control $\rightarrow$CA can charge whatever it wants for a certificate
- Entire security of the world rests on the CA
  ➠ infeasible to change the CA's key everywhere if compromised
- It is expensive and insecure for the CA to remotely certify a public key

Monopoly + Registration Authorities (RAs)

- The CA chooses and trusts other organizations as RAs

Monopoly + Registration Authorities (RAs)

- The CA chooses and trusts other organizations as RAs
- RAs check identities and obtain and vouch for public keys

Monopoly + Registration Authorities (RAs)

- The CA chooses and trusts other organizations as RAs
- RAs check identities and obtain and vouch for public keys
- The CA certifies public keys according to information provided by RAs

Monopoly + Registration Authorities (RAs)

- The CA chooses and trusts other organizations as RAs
- RAs check identities and obtain and vouch for public keys
- The CA certifies public keys according to information provided by RAs

- It is more convenient and secure to obtain certificates

Monopoly + Registration Authorities (RAs)

- The CA chooses and trusts other organizations as RAs
- RAs check identities and obtain and vouch for public keys
- The CA certifies public keys according to information provided by RAs

- It is more convenient and secure to obtain certificates
- It is still a monopoly model and has all of its problems

Monopoly + Registration Authorities (RAs)

- The CA chooses and trusts other organizations as RAs
- RAs check identities and obtain and vouch for public keys
- The CA certifies public keys according to information provided by RAs

- It is more convenient and secure to obtain certificates
- It is still a monopoly model and has all of its problems
  ➠ RA can be incorporated into any of the PKI models

Delegated CAs

- **Trust anchor** issues certificates to other CAs (**delegated CAs**) vouching for their keys and their trustworthiness as CAs

Delegated CAs

- **Trust anchor** issues certificates to other CAs (**delegated CAs**) vouching for their keys and their trustworthiness as CAs
- User can obtain certificates from a delegate CA

Delegated CAs

- **Trust anchor** issues certificates to other CAs (**delegated CAs**) vouching for their keys and their trustworthiness as CAs
- User can obtain certificates from a delegate CA
- Alice follows a chain of certificates from a trust anchor to the target to verify a certificate

Delegated CAs

- **Trust anchor** issues certificates to other CAs (**delegated CAs**) vouching for their keys and their trustworthiness as CAs
- User can obtain certificates from a delegate CA
- Alice follows a chain of certificates from a trust anchor to the target to verify a certificate
- Delegated CA has security and operational properties similar to RAs

Delegated CAs

- **Trust anchor** issues certificates to other CAs (**delegated CAs**) vouching for their keys and their trustworthiness as CAs
- User can obtain certificates from a delegate CA
- Alice follows a chain of certificates from a trust anchor to the target to verify a certificate

- Delegated CA has security and operational properties similar to RAs
- Delegated CAs can be incorporated into any of the PKI models

Oligarchy Model

- Oligarchy model is commonly used in browsers

Oligarchy Model

- Oligarchy model is commonly used in browsers
- The products are pre-configured with many trust anchors, a certificated issued by any one of them is accepted

Oligarchy Model

- Oligarchy model is commonly used in browsers
- The products are pre-configured with many trust anchors, a certificated issued by any one of them is accepted
- User may examine and edit the list of trust anchors

Oligarchy Model

- Oligarchy model is commonly used in browsers
- The products are pre-configured with many trust anchors, a certificated issued by any one of them is accepted
- User may examine and edit the list of trust anchors
  ➠ Google Chrome (ver 22.0.1229.79): 27 root certificates, 28 revoked (blacklisted) certificates

Oligarchy Model

- Oligarchy model is commonly used in browsers
- The products are pre-configured with many trust anchors, a certificated issued by any one of them is accepted
- User may examine and edit the list of trust anchors
  ➠ Google Chrome (ver 22.0.1229.79): 27 root certificates, 28 revoked (blacklisted) certificates
  ➠ Firefox (ver 15.0.1): hundreds of root certificates from 86 organizations around the world, none revoked

Problems of Oligarchy Model

- Security of the world is at risk if any trust anchor is compromised

Problems of Oligarchy Model

- Security of the world is at risk if any trust anchor is compromised
- The list of trust anchors is at the discretion of the product vendor

Problems of Oligarchy Model

- Security of the world is at risk if any trust anchor is compromised
- The list of trust anchors is at the discretion of the product vendor
  ➠ there is no practical ways for users to verify their trustworthiness
  ➠ some vendors will include any CAs that pay a fee

Problems of Oligarchy Model

- Security of the world is at risk if any trust anchor is compromised
- The list of trust anchors is at the discretion of the product vendor
  - ➠ there is no practical ways for users to verify their trustworthiness
  - ➠ some vendors will include any CAs that pay a fee
- It might be easy to trick a navie user into adding a bogus trust anchor
  - ➠ encrypted does not mean secure

Problems of Oligarchy Model

- Security of the world is at risk if any trust anchor is compromised
- The list of trust anchors is at the discretion of the product vendor
  ➠ there is no practical ways for users to verify their trustworthiness
  ➠ some vendors will include any CAs that pay a fee
- It might be easy to trick a navie user into adding a bogus trust anchor
  ➠ encrypted does not mean secure
- Some software (browsers) accept expired certificates
  ➠ Google Chrome includes a certificate from Microsoft which has expired since 12/30/1999

Anarchy Model (Web of Trust)

- Anarchy model is used by PGP

Anarchy Model (Web of Trust)

- Anarchy model is used by PGP
- Anarchy model is fully distributed and has no CAs
  ➠ anyone can sign certificates for anyone else

Anarchy Model (Web of Trust)

- Anarchy model is used by PGP
- Anarchy model is fully distributed and has no CAs
  ➠ anyone can sign certificates for anyone else
- A database is maintain (by volunteers) to keep certificates

Anarchy Model (Web of Trust)

- Anarchy model is used by PGP
- Anarchy model is fully distributed and has no CAs
  ➠ anyone can sign certificates for anyone else
- A database is maintain (by volunteers) to keep certificates
- Each user is responsible to certificate and deposit his trust anchors
  ➠ e.g., the public key from someone he met personally

Anarchy Model (Web of Trust)

- Anarchy model is used by PGP
- Anarchy model is fully distributed and has no CAs
  ➠ anyone can sign certificates for anyone else
- A database is maintain (by volunteers) to keep certificates
- Each user is responsible to certificate and deposit his trust anchors
  ➠ e.g., the public key from someone he met personally
- User searches through the database to verify a public key
  ➠ a path from one of his trust anchors to the target $\rightarrow$trusted

Problems of Anarchy Model

Anarchy model is unworkable on a large scale (why?)

Problems of Anarchy Model

Anarchy model is unworkable on a large scale (why?)

- The database would be unworkably large if deployed on Internet scale

Problems of Anarchy Model

Anarchy model is unworkable on a large scale (why?)

- The database would be unworkably large if deployed on Internet scale
- It's hard to verify the trustworthiness of every certificate on the chain
  ➠ no problem for a small community where everyone is trustworthy
  ➠ can we trust a certificate if there are multiple chains?

Models with Name Constraints

- A CA is only trusted for certifying users in his domains
  - ➠ FSU CA certifies FSU students
  - ➠ a certificate for Amazon.com from Nigeria?

Models with Name Constraints

- A CA is only trusted for certifying users in his domains
  - ➠ FSU CA certifies FSU students
  - ➠ a certificate for Amazon.com from Nigeria?
- Name constraints defines they way to verify (search) a target
  - ➠ top-down or bottom-up

Top-down with Name Constraints

- Everyone agrees upon a root CA, root CA delegates other CAs
  ➠ similar to the monopoly model with delegate CAs

Top-down with Name Constraints

- Everyone agrees upon a root CA, root CA delegates other CAs
  ➠ similar to the monopoly model with delegate CAs
- A delegated CA is only trusted to certify keys in its namespace

Top-down with Name Constraints

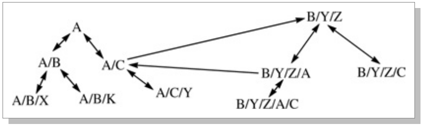
- Everyone agrees upon a root CA, root CA delegates other CAs
  ➠ similar to the monopoly model with delegate CAs
- A delegated CA is only trusted to certify keys in its namespace
- Use can verify a target by following the namespace down from root

Top-down with Name Constraints

- Everyone agrees upon a root CA, root CA delegates other CAs
  ➠ similar to the monopoly model with delegate CAs
- A delegated CA is only trusted to certify keys in its namespace
- Use can verify a target by following the namespace down from root
- It has the other problems of the monopoly model

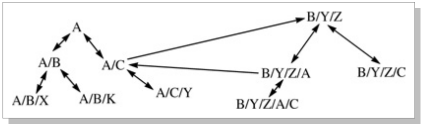
Bottom-up with Name Constraints

- The model assumes a hierarchical namespace (like DNS)



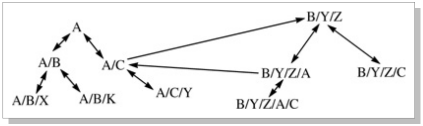
Bottom-up with Name Constraints

- The model assumes a hierarchical namespace (like DNS)
- Each organization can create its own PKI and link to others
  - ➠ a parent certifies its children (**down-link**)
  - ➠ a child also certifies its parent (**up-link**)
  - ➠ a **cross-link** connects two nodes neither is an ancestor of the other



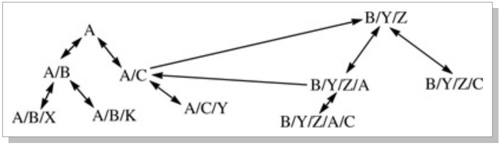
Bottom-up with Name Constraints

- The model assumes a hierarchical namespace (like DNS)
- Each organization can create its own PKI and link to others
  - ➠ a parent certifies its children (**down-link**)
  - ➠ a child also certifies its parent (**up-link**)
  - ➠ a **cross-link** connects two nodes neither is an ancestor of the other
- The namespace can be traversed starting from any node
  - ➠ follow up-links and/or one cross-link to an ancestor of the target
  - ➠ follow donw-links (only) from there



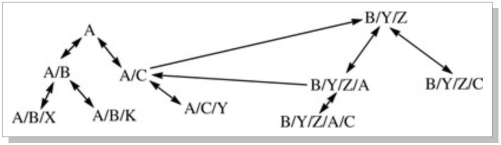
Bottom-up with Name Constraints...

- How can A/C/Y verify the certificate of B/Y/Z/C?



Bottom-up with Name Constraints...

- How can A/C/Y verify the certificate of B/Y/Z/C?
- How can B/Y/Z/C verify the certificate of A/C/Y?



Advantages of Bottom-up with Name Constraints

- The model is easy to understand and configure

Advantages of Bottom-up with Name Constraints

- The model is easy to understand and configure
  - ➠ PKI can be deployed in any organization independent of others
  - ➠ Authentication between users in the same organization stays within

Advantages of Bottom-up with Name Constraints

- The model is easy to understand and configure
  - ➠ PKI can be deployed in any organization independent of others
  - ➠ Authentication between users in the same organization stays within
- It is easy to find out if a path exists

Advantages of Bottom-up with Name Constraints

- The model is easy to understand and configure
  - ➠ PKI can be deployed in any organization independent of others
  - ➠ Authentication between users in the same organization stays within
- It is easy to find out if a path exists
- No monopoly is possible

Advantages of Bottom-up with Name Constraints

- The model is easy to understand and configure
  - ➠ PKI can be deployed in any organization independent of others
  - ➠ Authentication between users in the same organization stays within
- It is easy to find out if a path exists
- No monopoly is possible
- Replacing any key is reasonably easy

Certificate Revocation

- Certificates might need to be revoked
  ➠ employment is terminated, private key is stolen...

Certificate Revocation

- Certificates might need to be revoked
  ➠ employment is terminated, private key is stolen...
- Certificates normally have an expiration time
  ➠ validity time is months, lots of damage can be done in between
  ➠ browsers normally do not enforce expiration time!

Certificate Revocation

- Certificates might need to be revoked
  ➠ employment is terminated, private key is stolen...
- Certificates normally have an expiration time
  ➠ validity time is months, lots of damage can be done in between
  ➠ browsers normally do not enforce expiration time!

Solution: Certificate Revocation List (CRL)

Certificate Revocation

- Certificates might need to be revoked
  ➠ employment is terminated, private key is stolen...
- Certificates normally have an expiration time
  ➠ validity time is months, lots of damage can be done in between
  ➠ browsers normally do not enforce expiration time!

Solution: Certificate Revocation List (CRL)

- CA maintains and periodically releases a CRL

Certificate Revocation

- Certificates might need to be revoked
  ➠ employment is terminated, private key is stolen...
- Certificates normally have an expiration time
  ➠ validity time is months, lots of damage can be done in between
  ➠ browsers normally do not enforce expiration time!

Solution: Certificate Revocation List (CRL)

- CA maintains and periodically releases a CRL
- Each transaction is checked against the CRL

Certificate Revocation List

- Why are CRL issued periodically even if no certificates are revoked?

Certificate Revocation List

- Why are CRL issued periodically even if no certificates are revoked?
- How frequent should CRL be issued?

Certificate Revocation List

- Why are CRL issued periodically even if no certificates are revoked?
- How frequent should CRL be issued?
- If a CRL is maintained, why set an expiration time for certificates?

Delta CRL

- CRL may be very large and needs to be downloaded frequently

Delta CRL

- CRL may be very large and needs to be downloaded frequently
- A delta CRL lists changes from the last complete CRL
  ➠ not from the last delta CRL

Delta CRL

- CRL may be very large and needs to be downloaded frequently
- A delta CRL lists changes from the last complete CRL
  ➠ not from the last delta CRL
- Delta CRLs can be issued very frequently
  ➠ full CRLs are issued less frequently

On-line Revocation Servers (OLRS)

- OLRS is a online system to query the revocation status of certificates
  - ➠ OLRS maints the full CRL list
  - ➠ OLRS is not as security sensitive as a CA (or a KDC)

On-line Revocation Servers (OLRS)

- OLRS is a online system to query the revocation status of certificates
  - ➠ OLRS maints the full CRL list
  - ➠ OLRS is not as security sensitive as a CA (or a KDC)
- OLRS can be a performance bottleneck

On-line Revocation Servers (OLRS)

- OLRS is a online system to query the revocation status of certificates
  - ➠ OLRS maints the full CRL list
  - ➠ OLRS is not as security sensitive as a CA (or a KDC)
- OLRS can be a performance bottleneck
  - ➠ Alice can obtain a (timestamped) certificate from OLRS
    "Alice's certificate was not revoked as of ..."

On-line Revocation Servers (OLRS)

- OLRS is a online system to query the revocation status of certificates
  - ➠ OLRS maints the full CRL list
  - ➠ OLRS is not as security sensitive as a CA (or a KDC)
- OLRS can be a performance bottleneck
  - ➠ Alice can obtain a (timestamped) certificate from OLRS
    "Alice's certificate was not revoked as of ..."
  - ➠ Bob can query OLRS in advance and cache the result

Good-lists vs. Bad-lists

- CRL is a **black-list** of revoked certificates

Good-lists vs. Bad-lists

- CRL is a **black-list** of revoked certificates
- How about maintaining a list of valid certificates?
  ➠ Is it more secure than bad-lists?

Good-lists vs. Bad-lists

- CRL is a **black-list** of revoked certificates
- How about maintaining a list of valid certificates?
  ➠ Is it more secure than bad-lists?

Problems:

Good-lists vs. Bad-lists

- CRL is a **black-list** of revoked certificates
- How about maintaining a list of valid certificates?
  ➠ Is it more secure than bad-lists?

Problems:

- A good-list is likely much larger than the bad list

Good-lists vs. Bad-lists

- CRL is a **black-list** of revoked certificates
- How about maintaining a list of valid certificates?
  ➠ Is it more secure than bad-lists?

Problems:

- A good-list is likely much larger than the bad list
- Organizations might not want its list of valid certificates public

Good-lists vs. Bad-lists

- CRL is a **black-list** of revoked certificates
- How about maintaining a list of valid certificates?
  ➠ Is it more secure than bad-lists?

Problems:

- A good-list is likely much larger than the bad list
- Organizations might not want its list of valid certificates public
  ➠ solution: use hashes of the certificates

Summary

- X.509 certificate
- PKI trust models
- Certificate revocation

- Next lecture: IPSec