X.509 PKI

Trust no-one

@apoikos

Athens CryptoParty #2

TLS

- Provides *endpoint authentication* (X.509 + Key exchange)
- Guarantees *data integrity* (MAC)
- Protects *communication confidentiality* (Symmetric encryption)

Endpoint authentication

- During the TLS handshake, the server provides a *certificate* to the client (and optionally requests a certificate *from* the client).
- These certificates act as a proof of the {server,client}'s identity.
- X.509 is a standard that specifies:
  - *What* these certificates contain
  - *How* this information is encoded
  - *How* identity validation is performed (*trust model*)

Identity validation

Trust?

- O hai!
- O hai! Name is google.com!!!11
- ORLY?

Identity validation

Trust?

- O hai!
- O hai! Name is google.com!!!11
- ORLY?

How do we *trust* what the server claims?

Identity validation

Trust?

- O hai!
- O hai! Name is google.com!!!11
- ORLY?

How do we *trust* what the server claims?

- By having a *trusted third party* attest the identity or
- By utilizing a *web of trust*

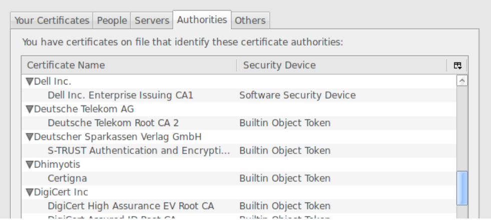
X.509 uses the former: a hierarchy of a priori trusted *Certification Authorities*

X.509 certificates

- Version (e.g. 3)
- Serial Number (unique per issuer)
- Algorithm (e.g. SHA-1 with RSA encryption)
- Issuer
- Validity
  - Not before
  - Not after
- Subject
- Subject public key
- Issuer signature
- Extensions

Certificate verification

- The X.509 PKI model builds a *chain* of certificates.
- We "only" need to have a copy of the top certificate issuer's certificate (*root CA*).
- Where do we get these?



You have certificates on file that identify these certificate authorities:

Certificate Name	Security Device
▼Dell Inc.	
Dell Inc. Enterprise Issuing CA1	Software Security Device
▼Deutsche Telekom AG	
Deutsche Telekom Root CA 2	Builtin Object Token
▼Deutscher Sparkassen Verlag GmbH	
S-TRUST Authentication and Encrypti...	Builtin Object Token
▼Dhimyotis	
Certigna	Builtin Object Token
▼DigiCert Inc	
DigiCert High Assurance EV Root CA	Builtin Object Token
DigiCert Assured ID Root CA	Builtin Object Token

Examining certificates

```
$ openssl s_client -connect google.com:443 -verify 3
$ openssl x509 -noout -text < /etc/ssl/certs/…
```

Revocation

- Certificates *expire*
- What happens if a certificate's key is compromised before it expires?
  - Need a way to check if a certificate is still "good"
- CRLs (Certificate Revocation Lists)
  - Published either directly, or available over OCSP
  - CRLs and OCSP responses are *signed*
  - A client needs to parse the CRL *or* query over OCSP

Problems/limitations

- Centralized trust model, mostly serving a specific business model.
- The trust pool is *flat*. *Any* trusted CA could have signed an accepted certificate.
- A compromised/malicious sub-CA for any trusted CA can be used for SSL Man-In-The-Middle attacks.
- The significance of the trusted CA pool is hidden from users (they "don't need to know")
- Alternative: Web-of-trust (PGP/GPG/OpenPGP)