CSci 4271W Development of Secure Software Systems Day 24: Protocols in practice

Stephen McCamant University of Minnesota, Computer Science & Engineering

Outline

Key distribution and PKI

Announcements intermission

SSH

SSL/TLS

DNSSEC

Public key authenticity Symmetric key servers Public keys don't need to be secret, but they must be right Wrong key → can't stop middleperson So we still have a pretty hard distribution problem So we still have a pretty hard distribution problem So we still have a pretty hard distribution problem Drawback: central point of trust



A name and a public key, signed by someone else
 C_A = Sign_S(A, K_A)
 Basic unit of transitive trust

Commonly use a complex standard "X.509"

Certificate authorities

- CA" for short: entities who sign certificates
- Simplest model: one central CA
- Works for a single organization, not the whole world



CA hierarchies

- 🖲 Organize CAs in a tree
- Distributed, but centralized (like DNS)
- Check by follow a path to the root
- Best practice: sub CAs are limited in what they certify

PKI for authorization

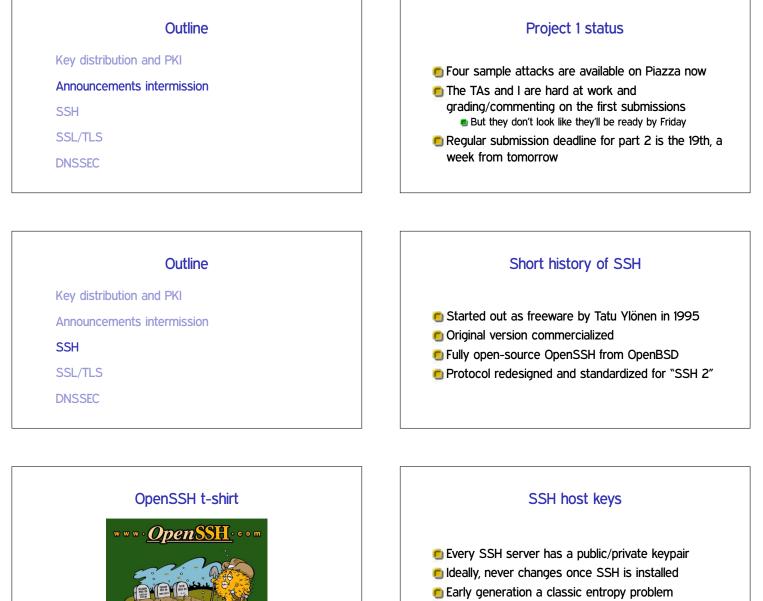
Enterprise PKI can link up with permissions

- One approach: PKI maps key to name, ACL maps name to permissions
- Often better: link key with permissions directly, name is a comment

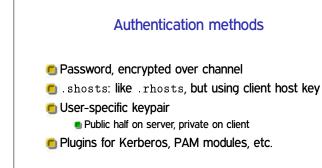
The revocation problem

How can we make certs "go away" when needed?

- Impossible without being online somehow
- 1. Short expiration times
- 2. Certificate revocation lists
- 3. Certificate status checking



Especially embedded systems, VMs



Old crypto vulnerabilities

- 1.x had only CRC for integrity Worst case: when used with RC4
- Injection attacks still possible with CBC CRC compensation attack
- For least-insecure 1.x-compatibility, attack detector
- Alas, detector had integer overflow worse than original attack

Newer crypto vulnerabilities

- IV chaining: IV based on last message ciphertext
 - Allows chosen plaintext attacks
 - Better proposal: separate, random IVs
- Some tricky attacks still left
 - Send byte-by-byte, watch for errors
 - Of arguable exploitability due to abort
- Now migrating to CTR mode

SSH over SSH

- SSH to machine 1, from there to machine 2 Common in these days of NATs
- Better: have machine 1 forward an encrypted connection
- 1. No need to trust 1 for secrecy
- 2. Timing attacks against password typing

SSH (non-)PKI

When you connect to a host freshly, a mild note When the host key has changed, a large warning

WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY! Someone could be eavesdropping on you right now (man-in-the-middle attack)

It is also possible that a host key has just been changed.

Outline

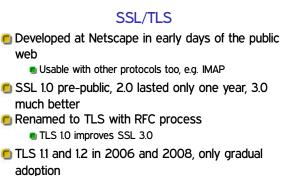
Key distribution and PKI

Announcements intermission

SSH

SSL/TLS

DNSSEC



IV chaining vulnerability

TLS 1.0 uses previous ciphertext for CBC IV But, easier to attack in TLS: More opportunities to control plaintext Can automatically repeat connection "BEAST" automated attack in 2011: TLS 1.1 wakeup

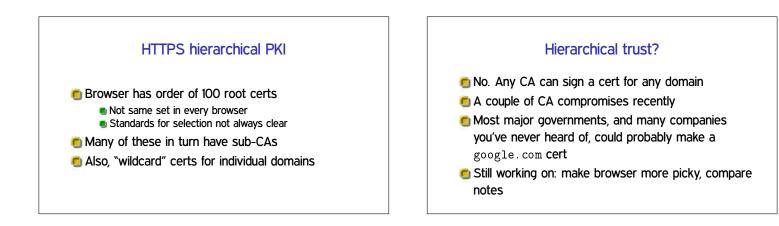
call

Compression oracle vuln.

- O Compr(S \parallel A), where S should be secret and A is attacker-controlled
- Attacker observes ciphertext length
- If A is similar to S, combination compresses better
- Compression exists separately in HTTP and TLS

But wait, there's more!

- Too many vulnerabilities to mention them all in lecture
- Kaloper-Meršinjak et al. have longer list "Lessons learned" are variable, though
- 🖲 Meta-message: don't try this at home



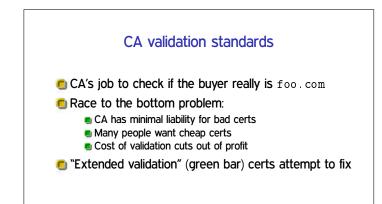
CA vs. leaf checking bug

Certs have a bit that says if they're a CA
All but last entry in chain should have it set
Browser authors repeatedly fail to check this bit
Allows any cert to sign any other cert

MD5 certificate collisions



- Create innocuous cert and CA cert with same hash
 - Requires some guessing what CA will do, like sequential serial numbers
 - Also 200 PS3s
- Oh, should we stop using that hash function?





- Many HTTPS security challenges tied with user decisions
- Is this really my bank?
- Seems to be a quite tricky problem
 - Security warnings often ignored, etc.

Outline	
Key distribution and PKI	
Announcements intermission	
SSH	
SSL/TLS	4
DNSSEC	
	l

DNS: trusted but vulnerable

- Almost every higher-level service interacts with DNS
- UDP protocol with no authentication or crypto
 Lots of attacks possible
- Problems known for a long time, but challenge to fix compatibly

DNSSEC goals and non-goals

- + Authenticity of positive replies
- + Authenticity of negative replies
- + Integrity
- Confidentiality
- Availability

First cut: signatures and certificates

- Each resource record gets an RRSIG signature
 E.g., A record for one name→address mapping
 Observe: signature often larger than data
- Signature validation keys in DNSKEY RRs
- Recursive chain up to the root (or other "anchor")

Add more indirection

- DNS needs to scale to very large flat domains like . com
- Facilitated by having single DS RR in parent indicating delegation
- Chain to root now includes DSes as well

Negative answers

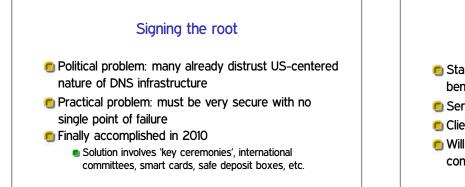
- Also don't want attackers to spoof non-existence Gratuitous denial of service, force fallback, etc.
- 🖲 But don't want to sign "x does not exist" for all x –
- Solution 1, NSEC: "there is no name between acacia and baobab"

Preventing zone enumeration

- Many domains would not like people enumerating all their entries
- DNS is public, but "not that public"
- Unfortunately NSEC makes this trivial
- Compromise: NSEC3 uses password-like salt and repeated hash, allows opt-out

DANE: linking TLS to DNSSEC

- *DNS-based Authentication of Named Entities"
- DNS contains hash of TLS cert, don't need CAs
- How is DNSSEC's tree of certs better than TLS's?



Deployment

- Standard deployment problem: all cost and no benefit to being first mover
- Servers working on it, mostly top-down
- Clients: estimated around 30%
- Will probably be common for a while: insecure connection to secure resolver

What about privacy?

- Users increasingly want privacy for their DNS queries as well
- Older DNSCurve and DNSCrypt protocols were not standardized
- More recent "DNS over TLS" and "DNS over HTTPS" are RFCs
- DNS over HTTPS in major browsers might have serious centralization effects