ARPKI: Attack Resilient Public-Key Infrastructure

David Basin, Cas Cremers, Tiffany Hyun-Jin Kim, Adrian Perrig, Ralf Sasse, Pawel Szalachowski

ETH Zurich, University of Oxford, CMU
Public keys and certificates

- Public key allows anyone to encrypt a message that only the owner of the associated private key can decrypt
- Problem: how do I know I have the right key for service x?
  - Direct exchange scales poorly
  - Unknown which websites you want to access
- Public key infrastructure
  - Certificates bind identities to public keys
  - Browser delivered with keys for trusted Certificate Authorities
  - Root of trust – chained to actual certificate for some domain
- Use case: online banking, shopping, account access
SSL / TLS X.509 PKI

(1a) Key_D Sign Request

(1b) CA-Signed Cert_D

(2a) Client Hello

(2b) Key_D, Cert_D

Domain D

CA
CA BREACHES

- 2010: VeriSign hacked, successfully and repeatedly
  - Revealed in U.S. SEC filing in October 2011
- Mar 2011: attack on Comodo reseller
  - Fraudulent certificates for: Google, Yahoo, Microsoft domains
- Aug 2011: DigiNotar – issued fraudulent certificates for Google
  - Used for spying on Iran’s citizens by its government in August 2011
- Oct 2011: Stuxnet – certificates from 2 Taiwanese CAs
- Dec 2012: EGO receives signing certificate from TurkTrust
- Possibly a large number of CA breaches remain undetected
**Man-in-the-Middle Attack**

Normal case

- Adversary obtains fraudulent certificate

Adversary obtains fraudulent certificate

- Man-in-the-Middle attack
CERTIFICATE LOGGING

- CAs are *vulnerable* and represent a single point of failure
- Unauthorized certificates become *visible*
  - Public logs of all valid certificates are kept
  - Certificate *must be in log* to be usable
  - Deterrence of misbehavior

- Logs struggle with:
  - *Increased system complexity*
  - Certificate update and revocation
  - Key loss – Domains and Certification Authorities

- Google plans Certificate Transparency *rollout* for EV certs in 2015
CONTRIBUTIONS

- Logging-based PKI system design
- Validation: Verification of core security property
- Formal model
- Co-design
- Proof-of-concept implementation
- Security guarantees with high assurance
CONTRIBUTIONS

- **New** logging-based PKI system
  - Mitigates the problem of fraudulent certificates
  - First co-designed PKI

- **Validation** through formal verification of core security property in model

- Proof-of-concept implementation

- Substantially stronger security guarantees with high assurance
APPROACH: ATTACK RESILIENT PKI

- Co-design of formal model and design
  - Makes all possible requirements **precise**
  - Tight link between design, model and implementation

- Incremental verification
  - Provides quick feedback on issues with design

- High-level **prototype**
  - Message-flow and all checks visible
  - Ensures no re-engineering of implementation is needed
ATTACK RESILIENT PKI – CERTIFICATE FORMAT

- Combines 2 standard X.509 certificates

- Client requires **proof** that certificate is in the **log**
  - Signed by the log server – **non-repudiable**
  - Verified and signed by 2 CAs

- Contains domain’s **policy**
  - Trusted entities
  - Update/revocation parameters

- All communication signed – **attributable** to entities
POLICIES – whom to trust

- ARPKI certificates include policy
  - Trusted log/CA servers
  - Update requirements, etc.

- Domain must have unique policy, so:
  - domain can only have one single certificate

- Separate out policy:
  - PoliCert paper at CCS 2014
**ARPKI Certificate Registration**

1. Domain → CA1
2. CA1 → Log Server1
3-6. Log Server1 ↔ Log Servers
7. Log Server1 ↔ CA2
8. CA2 → 1
9. 1 → Domain

10* & 11* repeat often; 1-9 setup only
OUR GOALS

- Reduce trust in any single component
  - CA private key compromise tolerable
  - Resilience against even two compromised entities

- Adversarial event protection
  - Make attacks visible
  - Prevent attacks where possible

- High assurance guarantees
  - Formal model of specification
  - Analysis with tool-support
Manual verification is complicated by system complexity
- Results in low confidence

Ad hoc design will likely result in vulnerable system

Accountable Key Infrastructure [WWW’13] analysis shows:
- Proposed off-line validators insufficient
- Unspecified min/max parameters

Formal verification is necessary
PKI – CRITICAL INFRASTRUCTURE

- Tool-supported analysis required
  - We use the Tamarin prover
- Manual analysis infeasible – low confidence
  - For systems of this scale, with many interactions, manual analysis and reasoning generally fails as state space is too large
- Discovered issues in analysis of AKI:
  - Proposed off-line validators insufficient
  - Missing synchronization requirements on log servers
  - Observation of integrity must be mutual
  - Unspecified min/max parameters
**Desired Security Properties**

- Connection integrity
  - Client connecting based on certificate – must be communicating with legitimate domain owner

- Legitimate initial certificate registration

- Legitimate certificate updates

- Visibility of attacks
Attack Possibilities

- Attack requires \textit{at least} \( n \) compromised entities (default:3)

- Security parameter \( n \) \textit{can be increased}
  - Resilient to \( n-1 \) compromised entities
  - More overhead and latency
  - Must be done for the whole system, not possible on a per-domain basis
FORMAL VERIFICATION

- Core security property
  - Prevents impersonation attack
  - Property formally specified and
  - Proven in 80 minutes on 32GB + 16 Cores

- Verified in the $n=3$ setting
  - Tool-supported proof with Tamarin prover
  - Full model is 23 rules, 1k lines of code
  - Verified 5 lemmas

- Tamarin extended – largest verification by Tamarin, by far.
theorem core_security_property:

\[ \forall a \ b \ \text{reason oldkey key} \ t_1 \ t_2 \ t_3 \ t_4 \ . \]
\[
( \text{Gen_ltk(a,oldkey,'trusted')@t1} \ \& \ \text{AskedForARCert(a, oldkey) @t2} \ \& \ \text{ReceivedARCert(a, oldkey) @t3} \ \& \ \text{ConnAcc(b, a, reason, key) @t4} \ \& \ t_3 < t_4)
\]
\[ \Rightarrow ( \neg (\exists t. K(\text{key})@t)) \]

FORMAL VERIFICATION

Proof scripts

Visualization display
ABSTRACTIONS IN FORMAL MODEL

- Abstracted logs from Merkle hash trees
  - Tamper-proof, represented as lists
- Abstracted ILS quorum finding
  - Set of ILSs represented by single ILS – no quorum modeling

- Formal model very close to design
  - Differences are nevertheless possible – not verifiable
  - Implementation may differ from design
ARPKI IMPLEMENTATION

1. Domain to CA1
2. CA1 to C++
3-6. C++ to Log Server1
7. Log Server1 to C++
8. C++ to CA2
9. CA2 to Log Servers
10. Log Servers to Client
11. Client to Domain

Client

Domain

CA1

Log Server1

Log Servers

CA2
**ARPKI IMPLEMENTATION**

- Small overhead

- Browser side validation averages 2.2ms
  - Standard validation: 0.7ms
  - Confirmations: 1.5ms

- No additional TLS level roundtrip
  - Possibly additional TCP roundtrip for large certificates (> 4kB)

- Incrementally deployable
**RELATED WORK**

- **CA-centric**
  - Certificate Revocation List (CRL)
  - Online Certificate Status Protocol (OCSP)
  - Short-lived certificates
  - Must trust single CA, no attack visibility or prevention

- **Client-centric**
  - Perspectives
  - Convergence
  - Must trust single CA, additional latency, privacy issues

- **Log-based**
  - EFF: Sovereign Keys
  - Google: Certificate Transparency (CT)
  - Accountable Key Infrastructure (AKI)
### Comparison to Log-Based Approaches

<table>
<thead>
<tr>
<th>Property</th>
<th>CT</th>
<th>AKI</th>
<th>ARPKI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resilient against</td>
<td>0</td>
<td>1</td>
<td>2+</td>
</tr>
<tr>
<td>Update/Revocation</td>
<td>Restricted</td>
<td>Restricted</td>
<td>✓</td>
</tr>
<tr>
<td>Formal validation</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
</tbody>
</table>
CONCLUSIONS

- New PKI proposal
  - Resilient against \( n-1 \) compromised entities
  - Formally verified co-designed model’s main security property using the Tamarin prover
- Proof-of-concept implementation
  - Small overhead, incremental deployment possible
- Improvements over existing approaches
- Open questions:
  - CA certificate management
  - Policies and business models
- http://www.netsec.ethz.ch/research/arpki