Fault-Based Attack of RSA Authentication

Andrea Pellegrini,
Valeria Bertacco and Todd Austin
University of Michigan

Cryptography

• Public key cryptography is pervasive in our digital world

TPM Coprocessor
Online Banking
Entertainment Systems
Are these algorithms secure? (i.e., cryptanalysis)

**Attack the algorithm**
by guessing key

```
13506641086599522334960
3216788059699388814756
05667027524485143851526
510604859538394026715
05719094417982072821644
71551373680419703964191
74304649658927425623934
1020864382021103729587
25762358509643110564073
5015081751067659462920
556368555294....
```

time consuming:
> age of Universe

**Attack the implementation**

**Side-channel**
by monitoring side effects
- Timing [Brumley03]
- Power [Kocher99]

**Fault-Based**

- a faulty processor may leak secrets
  - Theoretical [Joye99, Boneh01, Wagner04]
  - Demonstrated on simple components [Bar-El06]

---

Our Contribution

Fault-based attack on a complete system

Discovered vulnerability in OpenSSL

1024-bit secret key extracted in 100 hours

Faults manifest on the critical path (multiplier)
RSA Authentication

Correct behavior:
- Server challenge: \[ s = m^d \mod n \]
- Client verifies: \[ m = s^e \mod n \]

Faulty Server:
\[ s \neq m^d \mod n \]

Computing: \[ s = m^d \mod n \]

Fall-back algorithm in OpenSSL

The algorithm partitions the exponent into windows:
\[ d = [110110110001...110110010101] \]

\[ s = 1 \]
for each window:
  for each bit in window: \(/4\)times
    \[ s = (s * s) \mod n \]
    \[ s = (s * m^d[\text{window}]) \mod n \]
return s
Computing: \( s = m^d \mod n \)

\[
d = 214 = 1101 \ 0110
\]

window 1 window 2

\[
s = 1
g\text{for each window:}
\]

\[
\text{for each bit in window: //4 times}
\]

\[
s = (s \ast s) \mod n
\]

\[
s = (s \ast m^d[\text{window}]) \mod n
\]

\[
\text{return } s
\]

\[
s = (\cdots (m^{1101})^2)^2)^2)^2)^2)m^{0110}
\]

Faulty Signature: \( \hat{s} \neq m^d \mod n \)

\[
d = 214 = 1101 \ 0110
\]

window 1 window 2

\[
\hat{s} = 1
\text{for each window:}
\]

\[
\text{for each bit in window: //4 times}
\]

\[
\hat{s} = (\cdots (m^{1101})^2)^2)^2)^2)^2)^2\pm 2f)^2)^2)^2)^2)m^{0110}
\]
Retrieving the Private Key

- The attacker collects the faulty signatures

- The private key is recovered one window at the time

- The attacker checks its guess against the collected signatures

\[ d = d_3 X X \]

Andrea Pellegrini
University of Michigan
Physical Attack

Fault Injection

- A corrupted signature leaks data only if one multiplication was corrupted by a single bit flip
Physical Attack

- 8,800 corrupted signatures collected in 10 hours
- RSA 1024-bit private key
- Distributed application with 81 machines for offline analysis
- Private key recovered in 100 hours

Conclusions

- Faults can leak vital private key data
- Fault-based attack devised for OpenSSL Fixed Window Exponentiation algorithm
- Attack demonstrated on a complete physical Leon3 SPARC system
- Future work: general software fix using “blind”ing (patch for OpenSSL sent to developers)