Overview

➲ What is RSA-PSS?
➲ Why RSA-PSS?
➲ Comparing original and standardized PSS
➲ Status of Protocols, Standards and Implementations
➲ RSA-PSS in X.509
➲ Algorithmenkatalog
RSA

- Public key cryptosystem
- Invented 1977 by Ron Rivest, Adi Shamir, Leonard Adleman
- Public Key \((e, N)\), private key \((d, N)\) with \((X^{de}) \mod N = X\)
- Encrypt: \(E = (M^e) \mod N\), Decrypt: \(M = (E^d) \mod N\)
- Sign: \(S = (M^d) \mod N\), Verify: \(M = (S^e) \mod N\)
- What is \(M\)?
Hash-then-sign, PKCS #1 v1.5
Probabilistic Signature Scheme
Probabilistic Signature Scheme

- Developed 1996 by Mihir Bellare and Phil-ipp Rogaway
- "Provable Secure" in the random oracle model
- That means: Secure if hash function is ideal, factoring is hard and RSA itself is as hard as factoring
- Uses a salt (randomization) and uses full size of RSA input
Status of PSS in standards

- RSASSA-PSS primitives are part of IEEE P1363a and PKCS #1 v2.1 / RFC 3447
- RSASSA-PSS supported by standards for X.509 (RFC 4055), CMS (RFC 4055)
- Not supported in OpenPGP, DNSSEC, XM-LDsig, TLS
X.509 Implementations

- Latest OpenSSL 1.0.0d: bare PSS signatures supported, no support for X.509
- X.509 Support in OpenSSL 1.1 CVS (not yet released)
- Latest Mozilla nss / Firefox: Not supported
- I created patches for nss in the Google Summer of Code 2010, not yet merged
- Microsoft Windows (since Vista) supports X.509 with RSASSA-PSS
- Microsoft was faster than any other browser vendor in implementing an open standard!!
X.509 online test

🔗 http://ssl.hboeck.de/
Hashing

- A lot has happened in hash function research in recent years
- MD5 collision in 2004
- SHA-1: Collision attacks with a complexity of $2^{63}$
- Successful fake of a CA certificate in 2008 (25C3, calculated on a PS3 cluster)
- SHA-3 competition running
PSS96 and PKCS #1 v2.1
Direct input randomization secures against possible collision flaws in the hash

- eTCR (enhanced Target Collision resistance)
- PSS96 provides eTCR, PSS from standards PKCS #1 v2.1 / IEEE 1363a does not
- Randomized hashing: brings back eTCR
Randomized Hashing

- Generate random value $rv$
- Repeat $rv$ and XOR it with input message (XOR vigenere)
- Use $rv \parallel (M \oplus rv) \parallel rv_{length}$ as hash function input
- Problem: $rv$ has to be shipped separately
- Randomized hashing and PSS: salt can be used as $rv$
PSS with randomized hashing

1. Input: \( M \)
2. Add salt: \( M \) -> \( \text{salt} \) -> \( \text{RMX} \)
3. Compute \( \text{RM} \) from \( \text{RMX} \)
4. Compute \( \text{Hash} \) from \( \text{RM} \)
5. Compute \( \text{Padding1} \), \( m\text{Hash} \), \( \text{salt} \)
6. Compute \( \text{Padding2} \), \( \text{salt} \)
7. Compute \( \text{Hash} \) from \( \text{Padding2} \), \( \text{salt} \)
8. Compute \( \text{xor} \) and \( \text{MGF} \)
9. Compute \( \text{maskedDB} \), \( H \), \( bc \)
Good: Pushing for better security
Bad: Not pushing for better standards and implementations
Technische Richtlinie 03125 (long time archiving) requires algorithms from “Algorithmenkatalog”
TR 03125 is based on XMLDsig
XMLDsig does not support PSS!
Is it possible to provide really provable security for public key cryptography?
Not today: We don't know enough about complexity theory.
Our whole trust in cryptography relies on assumptions – we believe that if nobody was able to break something in a long time, it must be secure.
Is factoring hard? Is RSA as hard as factoring? Anyone with a Quantum computer out there?
But if we could:

- Prove $P \neq NP$
- Create trapdoor function out of FNP problem
- Create cryptosystem and prove that we only hit the hard problems in our FNP problem
- Create a provable secure scheme that is not based on a hypothetical ideal hash function, but a real one
- Prove that the whole thing is also resistant to Quantum computers

$P \neq NP$ is considered to be one of the hardest problems in mathematics and theoretical computer science – and that's only the first step.
RSA-PSS

Questions? Discussion?

Diploma thesis on RSA-PSS will be available at
http://rsapss.hboeck.de/