Applied Cryptography
CPEG 472/672
Lecture 9B

Instructor: Nektarios Tsoutsos

RSA (1977)

Rivest-Shamir-Adleman

- First public-key cryptosystem
  - Asymmetric encryption
  - One key to encrypt, different key to decrypt
- Trapdoor permutation
  - Transform X to Y in the same range
  - One-way unless you know a "trapdoor" key
- Can create digital signatures
  - Separate key to sign, another key to verify

RSA KeyGen

- Select 2 large primes p, q of similar size
  - p must never be the same as q (very bad)
  - Compute $N = p \cdot q$ and $\varphi(N) = (p-1) \cdot (q-1)$
    - $\varphi(N)$ must be secret
- Select public exponent e
  - Must select e so that $GCD(e, \varphi(N)\,) \;=\; 1$
  - Usually $e = 2^{16} + 1$ (Fermat prime)
    - Never use $e = 3$
- Find secret exponent $d = e^{-1} \bmod \varphi(N)$
  - Check $e \cdot d = 1 \bmod \varphi(N)$
  - Use Extended Euclidean Algorithm

Textbook RSA Enc and Dec

https://people.csail.mit.edu/rivest/Rsapaper.pdf

- Enc: $\mathrm{C} = M^e \bmod N$, Dec: $\mathrm{M} = C^d \bmod N$
  - Message $M > 0$ smaller than N
  - $C^d \bmod N = M^{ed} \bmod N = M^1 \bmod N$

How to decrypt without d?

- Attacker may factor N to p, q
  - Attacker recovers $\varphi(N) = (p-1) \cdot (q-1)$
  - Attacker uses EEA to find $e^{-1} \bmod \varphi(N)$
    - $GCD(e, \varphi(N)) = 1 = e \cdot d + Y \cdot \varphi(N) \bmod \varphi(N)$
- If e is small, attacker can compute root
  - Consider $e = 3$: Attacker can find cube root
- Security depends on:
  - Size of N
  - Choice of p, q (must never be the same)

RSA Multiplicative Homomorphism

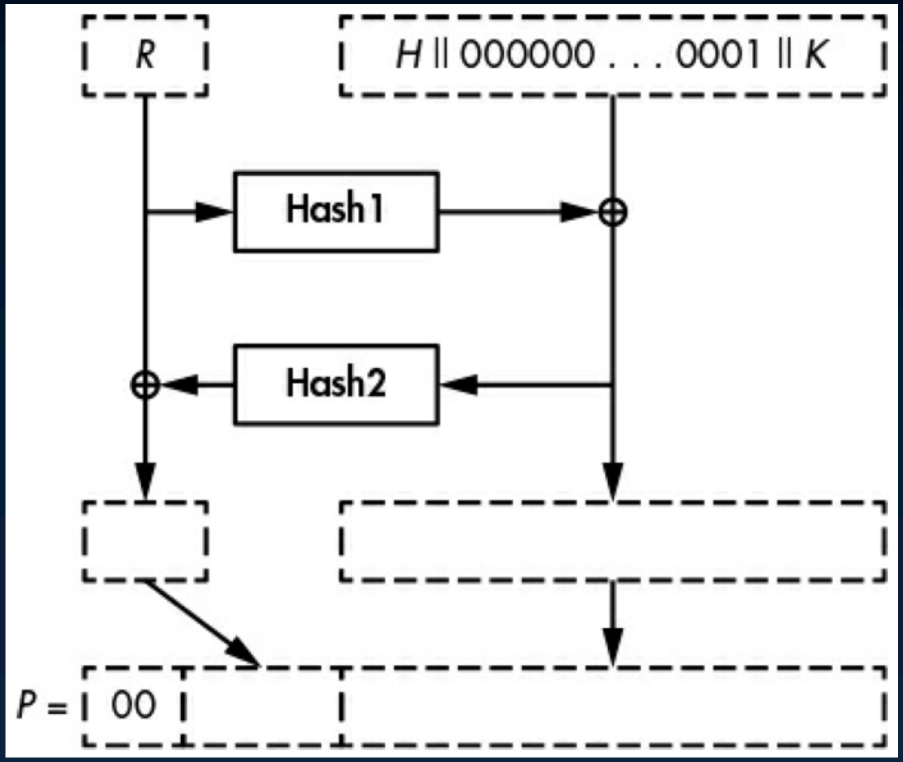
- Let $E(M_1) = M_1^e \bmod N$, $E(M_2) = M_2^e \bmod N$
  - It is: $E(M_1) \cdot E(M_2) = E(M_1 \cdot M_2) \bmod N$

RSA in real life

- RSA is slower than AES
- We can use the benefits of asymmetric encryption and the speed of symmetric
  - Select RSA input M to be an AES key K
  - Encrypt Ctxt=AES(K,Ptxt)
  - Compute C=RSAenc((e,N),K)
  - Send C,Ctxt to recipient
  - Get K=RSAdec((d,phi(N)), C)
  - Get Ptxt=AES(K,Ctxt)

RSA-OAEP

- To prevent malleability we need padding
  - Extend the message M to the size of N
  - Need a PRNG and two hash functions $\mathcal{H}_1, \mathcal{H}_2$
- $M = H \,||\, 0000..0001 \,||\, K$
  - H = OAEP constant
  - K is the AES key
- $M = M \oplus \mathcal{H}_1(R)$
  - R is a PRNG output
- $R = R \oplus \mathcal{H}_2(M)$
- $P = 00 \,||\, M \,||\, R$
  - Encrypt P with RSA



RSA Signatures

- Prove the holder of d signed msg M
  - d is tied to a digital signature of M
  - The digital signature is authentic
    - The holder of d cannot deny signing M
    - Non-Repudiation property
- RSA sign is not the converse of RSA enc
  - It is NOT the same as encrypting with the private key
  - It is OK for a signature to leak parts of M
  - Signing does not protect confidentiality
- We typically sign the Hash of M

Blinding attack on textbook RSA

- Create a forged signature for M
  - M is some incriminating message the user should not sign
- Ask a user to sign $R^e M \bmod N$
  - This message looks innocent
  - Obtain signature $S = (R^e M)^d \bmod N$
    - Observe $(R^e)^d \bmod N = R \bmod N$
  - Compute $S/R = RM^d/R = M^d \bmod N$
- We need to prevent malleability

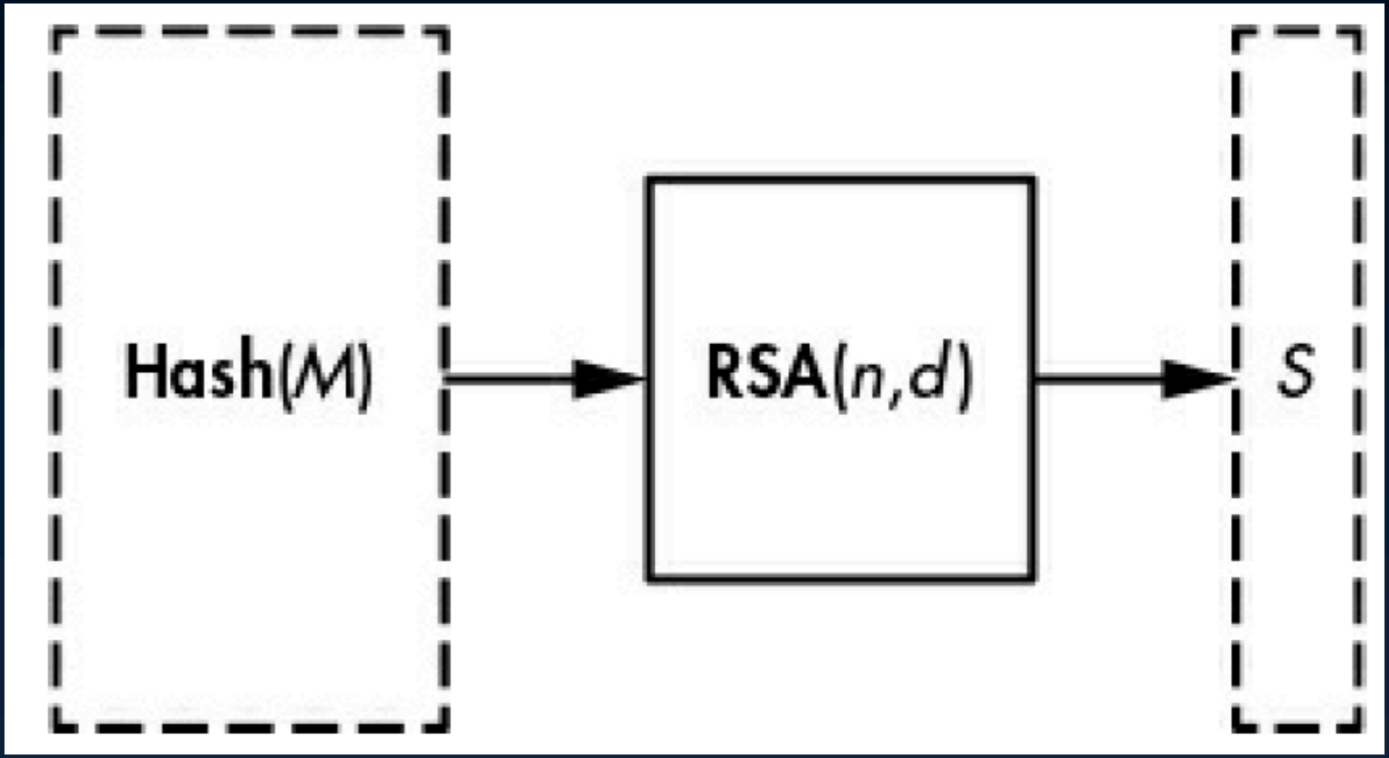
RSA-PSS

- We will securely sign the Hash of M
  - Need a PRNG and two hash functions $\mathcal{H}_1, \mathcal{H}_2$
- Compute $H = \mathcal{H}_1(\, 0000 \ldots 00 \,||\, \mathcal{H}_1(M) \,||\, R\,)$
  - R is a PRNG output
- Set $L = 0000 \ldots 00 \,||\, 01 \,||\, R$
- Update $L = L \oplus \mathcal{H}_2(H)$
- Compute $P = L \,||\, H \,||\, BC$
  - BC is a fixed value
- Compute the RSA signature $P^d \bmod N$

Full Domain Hash (FDH)

PSS offers stronger cryptographic guarantees than FDH

- $H = Hash(M)$
- Sign H as $H^d \bmod N$



Hands-on exercises

- RSA Key Generation
- RSA-OAEP Encryption/Decryption
- RSA-PSS Signing/Verification

Reading for next lecture

- Aumasson: Chapter 10 until the end
  - We will have a short quiz on the material
- 20 years of attacks on RSA
  - https://crypto.stanford.edu/~dabo/pubs/papers/RSA-survey.pdf