FINE 434: FinTech

Lecture 8

Professor Fahad Saleh

McGill University - Desautels

"If you don't understand what is being asked of you, get clarification. It is always good to ask all the questions you need to have answered to accomplish your task. Don't nod and act like you know what is needed when you don't. You will save an incredible amount of time for yourself and have a quality work product if you ask all the questions up front."

- Rich Handler (Chairman and CEO, Jefferies Group)
and
Brian Friedman (President, Jefferies Group)

Motivation

We need a scheme that allows you to spend your funds without allowing others to spend your funds. We want to implement this scheme without a trusted third party serving as an intermediary. (Why?)

Motivation

We need a scheme that allows you to spend your funds without allowing others to spend your funds. We want to implement this scheme without a trusted third party serving as an intermediary. (Why?)

Solution: Digital Signature Scheme

Digital Signature Scheme

A Digital Signature Scheme is a triple of algorithms (G, S, V) such that:

- G, the probabilistic "key generation algorithm", takes no inputs and outputs a pair (pk, sk) with $pk \in \mathcal{K}_p$ being the "public key" and $sk \in \mathcal{K}_s$ being the "secret key"
- $S(sk, m)$, the probabilistic "signing algorithm", takes sk and a message $m \in \mathcal{M}$ and outputs a signature $\sigma \in \Sigma$
- $V(pk, m, \sigma)$, the deterministic "verification algorithm", takes pk, m and σ and outputs *True* or *False*

Notes

- *pk* is public information; *sk* is private information
- $\mathbb{P}\{V(pk, m, S(sk, m)) = True\} = 1$
- It must be hard to forge a signature

Disclaimer

We will discuss a (deterministic) digital signature verification algorithm known as RSA. The algorithm, as discussed, is insecure, so I discourage use of the algorithm for any purpose. The ensuing discussion, however, should help you understand the underlying thinking behind digital signature verification algorithms.

Bitcoin and Ethereum do not use any version of RSA. Moreover, we discuss the initial implementation of RSA put forth in Rivest, Shamir, and Adleman (1977) rather than subsequent versions.

G: Generating pk and sk

- Choose two large random primes, p and q
- Set $N = p \times q$ and $\phi = (p-1)(q-1)$
- Choose integers $e, d \in [1, \phi]$ such that $e \cdot d \equiv 1 \ (mod\ \phi)$
- Output $pk = (N, e)$ and $sk = (N, d)$

G: Generating pk and sk

- Choose two large random primes, p and q
- Set $N = p \times q$ and $\phi = (p-1)(q-1)$
- Choose integers $e, d \in [1, \phi]$ such that $e \cdot d \equiv 1 \ (mod\ \phi)$
- Output $pk = (N, e)$ and $sk = (N, d)$

Question: How do we know such an e and d exist?

S and V: Signature and Verification

- Let H be a hash function
- $S(sk, m) = [H(m)]^d \; \% \; N$
- $V(pk, m, \sigma) = [H(m) \; \% \; N == \sigma^e \; \% \; N]$

S and V: Signature and Verification

- Let H be a hash function
- $S(sk, m) = [H(m)]^d \;\%\; N$
- $V(pk, m, \sigma) = [H(m) \;\%\; N == \sigma^e \;\%\; N]$

Question: Does $\mathbb{P}\{V(pk, m, S(sk, m)) = True\} = 1$ hold?

Questions

- How do we know such an *e* and *d* exist?
- How do we know the signature verifies correctly?
- Why is it hard to forge a signature?

Questions

- How do we know such an e and d exist?
- How do we know the signature verifies correctly?
- Why is it hard to forge a signature?

Answer: Modular Math

Detour

The remainder of this lecture assumes familiarity with Lecture 9's content. Lecture 9 presents modular mathematics concepts relevant for this course.

How do we know such an *e* and *d* exist?

We pick *e* relatively prime to ϕ which ensures existence of a multiplicative inverse and thus existence of *d*.

To find an *e* relatively prime to ϕ, we may select *e* first and then select *p* and *q* such that *e* is relatively prime to $p - 1$ and $q - 1$ which in turn ensures that *e* is relatively prime to ϕ

How do we know the signature verifies correctly?

$$\begin{aligned}
&S(sk, m)^e (mod\ p)\\
&\equiv H(m)^{de} (mod\ p)\\
&\equiv H(m)^{k\phi+1} (mod\ p)\\
&\equiv [H(m)][H(m)^{(p-1)}]^{k(q-1)} (mod\ p)\\
&\equiv H(m) (mod\ p)
\end{aligned}$$

The last line follows from Euler's Theorem. The same argument applies if p and q are switched. Thus, the Chinese Remainder Theorem then implies $V(pk, m, S(sk, m)) = True$ pointwise which establishes that signatures verify correctly.

Why is it hard to forge a signature?

N and *e* constitute public information. Further knowledge of *d* would allow signature forging. We know that *d* constitutes the multiplicative inverse of *e* modulo ϕ. Given ϕ and *e*, an attacker may easily compute *d*. ϕ may be deduced from *N* via *N*'s prime factorization, so forging is trivial given infinite time.

However, factoring large primes is practically infeasible. Try it...