Distributed Trust and Blockchains	Date: *14 Aug'18*
Instructor: *Sujit Prakash Gujar*	Scribes: Rajat Agarwal, Vikrant Goyal

Lecture 5: Digital Signatures

1 Recap

Hash Pointers :

- A Pointer to data along with its hash. It is pointer to where some info is stored and cryptography hash of the information.
- If we have hash pointer, we can ask for information back and can verify that it hasn't changed.

Hash Functions:

1. Input - Any string of any size.
2. Output- Fixed size output.
3. Computation - It should be efficiently computable.

Pre-image Resistance:

- It implies that it is computationally infeasible to retrieve a message from its hash value.
- Any change in message should change the hash value extensively.
- For a given hash value h, it is difficult to find any message m such that $H(m) = h$.

Second pre-image resistance:

- Given a message x, it is difficult to find a message y such that $H(x) = H(y)$ where $x! = y$.

Collision resistance:

- It is difficult to find two messages x & y such that $H(x) = H(y)$.

2 Introduction

Definition 1 (Digital Signature)**.** *It is a mathematical technique used to validate the authenticity and integrity of a message, software or digital document.*

2.1 Properties

1. Analogous of Signature on document. Signature is tied to particular document.
2. Only a person can sign, but everyone can verify.
3. It should not be feasible to forge.
4. Signing authority should not be able to claim that I did not sign.

2.2 Working of Digital Signature

- Digital signatures are based on public key cryptography.Using a public key algorithm, one can generate two keys that are mathematically linked: one private and one public.

$$(s_k, p_k) \leftarrow generateKeys(keysize)$$

where s_k is secret key and p_k is public key.

- To create a digital signature, signing software creates a one-way hash of the electronic data to be signed. The private key is then used to encrypt the hash. The encrypted hash along with other information, such as the hashing algorithm is the digital signature.

$$sig \leftarrow sign(s_k, message)$$

- The value of the hash is unique to the hashed data. Any change in the data, even changing or deleting a single character, results in a different value. This attribute enables others to validate the integrity of the data by using the signer's public key p_k to decrypt the hash. Anybody can verify the validity of the document.

$$isValid \leftarrow V(p_k, message, sig)$$

Now, signature is valid if , otherwise invalid

$$V(p_k, message, sig) == true$$

i.e,

$$Probability[V(p_k, message, sign(s_k, message)) == accepted] = 1$$

2.3 Challenger vs Attacker

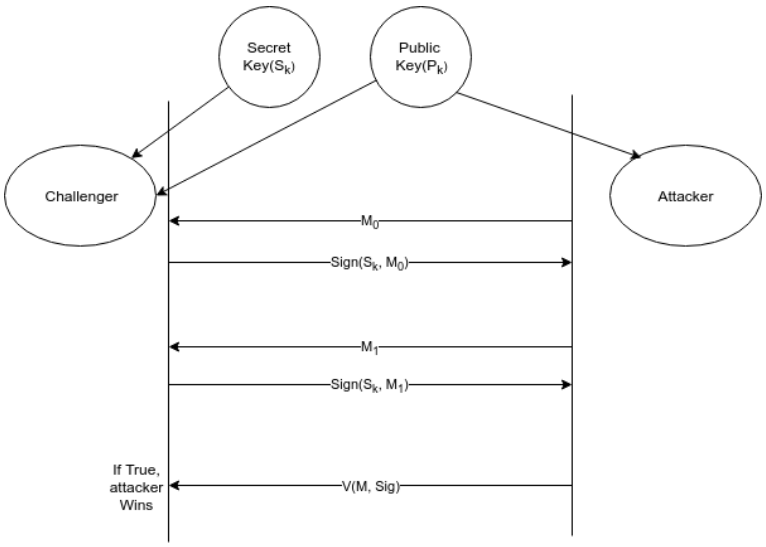

Figure 1

Here secret key(s_k) is shared only to signer and public key(p_k) is shared to both signer and attacker. Message (M) is new message, attacker wins if he is able to validate the message from signer which is probabilistic impossible.

3 RSA Algorithm

It is an asymmetric cryptographic algorithm. Asymmetric means that there are two different keys. This is also called public key cryptography, because one of the keys can be given to anyone. The other key must be kept private. The algorithm is based on the fact that finding the factors of a large composite number is difficult: when the integers are prime numbers.

3.1 Key Generation

- Choose two different large random prime numbers p and q.
- Compute n=p∗q.
- Calculate the Euler's Totient: ϕ(n) = (p - 1) ∗ (q - 1).
- Select random integer e : 1<e< ϕ(n) such that gcd(e, ϕ) = 1.
- Calculate unique number d : 1<d< ϕ(n) such that ed=1 mod ϕ(n).
- Public key (p_k) is (e, n) and Private key (s_k) is (d, n). i.e,

$$x^k = x \mod n \quad if\ k \equiv 1 \mod \phi(n)$$

- Encryption : $C = \mathrm{m}^e \mod \mathrm{n}$
- Decryption : $m = \mathrm{c}^d \mod \mathrm{n} = \mathrm{m}^{ed} \mod \mathrm{n} = \mathrm{m} \mod \mathrm{n}$.

3.2 Signing

$$s = Sign(d, n, m) = m^d \mod n$$

where (d, n) is secret key and m is message.

3.3 Verification

Correctly verified if,

$$V(e, n, s, m) == m$$

where (e, n) is public key and s is Sign(d, n, m) and m is message.

3.4 Example

Given $P = 61$ and $Q = 53$

- $n = P * Q = 61 * 53 = 3233$
- $\phi(n) = (P - 1) * (Q - 1) = 60 * 52 = 3120$
- Choose e : 1<e< ϕ(n) such that gcd(e, ϕ) = 1
  i.e, $e = 17$
- Calculate d : 1<d< ϕ(n) such that ed=1 mod ϕ(n)
  i.e, 17 * d = 1 mod ϕ(n)
  $d = 2753$
- Public key(p_k) is (17, 3233) and Secret Key(s_k) is (2753, 3233)
- For example, to encrypt message $m = 123$, we calculate
  Encrypted message $C = 123^{17} \mod 3233 = 855$
  Decrypted message $m = 855^{2753} \mod 3233 = 123$

4 SHA-256 Hash function

- It's the hash function that the bitcoins uses primarily.
- It uses Merkle-Damgard transform to turn a fixed-length collision-resistant compression function into a hash function that accepts arbitrarylength inputs.
- It uses a collision-resistant compression function that takes 768-bit input and produces 256-bit output.

4.1 Working of Merkle-Damgard Transform

- Let us say the compression function takes inputs of length m and produces an output of a smaller length n.
- The arbitrary sized input to the hash function is divided into blocks of size $m - n$.
- Each block together is passed along with the output of the previous block into the compression function.
- For the first block, for which we have no previous block output, we instead use an Initialization Vector(IV).

5 GoofyCoin - A Simple Cryptocurrency

The rules of GoofyCoin are:

- Goofy can create new coins by simply signing a statement that hes making a new coin with a unique coin ID.
- Whoever owns a coin can pass it on to someone else by signing a statement that saying, Pass on this coin to X (where X is specified as a public key).
- Anyone can verify the validity of a coin by following the chain of hash pointers back to its creation by Goofy, verifying all of the signatures along the way.

GoofyCoin suffers from a fundamental security problem called double-spending attack. Lets say Alice passed her coin on to Bob by sending her signed statement to Bob but didn't tell anyone else. She could create another signed statement that pays the very same coin to Chuck. To Chuck, it would appear that it is perfectly valid transaction, and now hes the owner of the coin. Bob and Chuck would both have valid-looking claims to be the owner of this coin. This is called a double-spending attack - Alice is spending the same coin twice.

To solve double-spending problem ScroogeCoin is used. ScroogeCoin is built off of GoofyCoin, but its a bit more complicated in terms of data structures.

6 References

- `https://simple.wikipedia.org/wiki/RSA_algorithm`
- `https://www.coursera.org/learn/cryptocurrency`
- `https://lopp.net/pdf/princeton_bitcoin_book.pdf`