CMPSCI 691ST Systems **Fall 2011**

Lecture 14

Lecturer: Emery Berger *Scribe: Hardeep Uppal*

14.1 A Method for Obtaining Digital Signatures and Public-Key Cryptosystems

This paper describes public key cryptography and presents a method for implementing privacy and accountability. More specifically, the authors use the Diffie and Hellman's concepts of public key crypto-systems and use this to implement a new encryption and signing method that can be used for secure transmission of email, and for generating digital signatures that can be authenticated by the receiver. The proposed method avoids the key distribution problem and does not require a pre-established secure communication channel. The paper also proves the correctness behind the math used in the new encryption and signing method, and shows through example how breaking the encryption and decryption key is hard and almost impossible.

14.1.1 Diffie-Hellman Key Exchange

Diffie-Hellman key exchange is one of the earliest example of key exchange in cryptosystems. It allowed two people to jointly establish a shared secret key over an insecure communication channel without any prior knowledge of each other. Diffie-Hellman key exchange algorithm used trap door functions that are easy to compute in one direction, but difficult to compute in the opposite direction without special information.

14.1.2 Encryption and Decryption Methods

The encryption and decryption method described in the paper relies on generating large decimal number n from the multiplication of two large decimal prime numbers. The security of this new method is shown by the fact that factoring n into the two prime numbers is very hard and unfeasible. The fact that existing techniques, and tests conducted by the author, could not factor the number n, is used as a proof that the method described in this paper is secure and ready to be used in a real environment. Many of the algorithms (e.g. SSL, RSA etc.) in the field of Security use similar arguments to show why they are secure, and breaking such algorithms is used as a benchmark for new research.

14.1.2.1 Prime Number Factorization

The time complexity of prime number factorization in unknown. Many attempts have been made to break crypto-system using brute force on a large cluster of machines. It is now feasible to break 512-bit keys with custom hardware. Breaking keys and crypto-system act as a benchmark for the security of the next generation of crypto-systems.

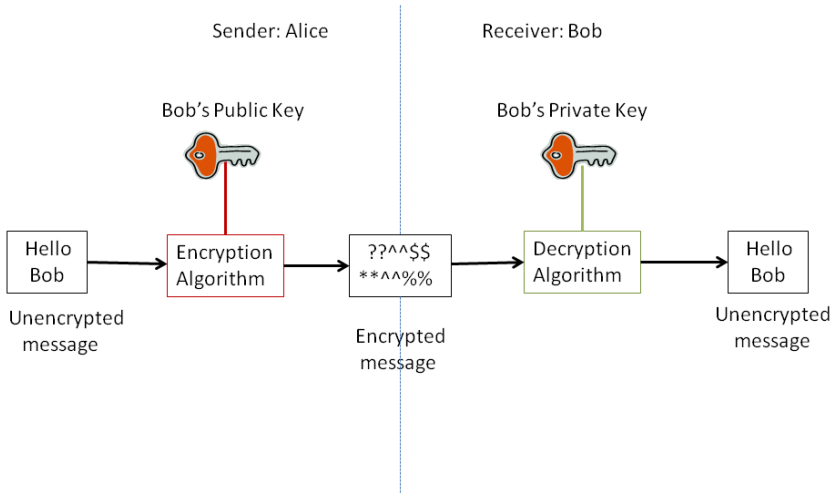

Figure 14.1: Example of Public-Key Cryptography

14.1.3 Public-Key Cryptography

Public-key cryptography requires the use of asymmetric keys, where the key used to encrypt a message is different from the key used to decrypt it. Neither key will do both functions. The keys are categorized as public and private key, where the public key is published for everyone to access and the private key is only known to the owner of the key. Messages are encrypted with a recipient's public key and can only be decrypted with the corresponding private key. Since only the receiver has the private key to decrypt the message, it allows secure communication over insecure channel without any prior exchange of secret keys.

Figure 1 shows an example of secure message exchange between Alice and Bob. Alice uses Bob's public key to encrypt the message "Hello Bob" and transmits the encrypted message over the channel. When Bob receives the encrypted message, it uses the private key to decrypt the message.

Intel provides user level api to encrypt and decrypt messages. The system is build into the chip to speedup the process of generating keys and executing encryption and decryption methods. The system provides opcodes to access these methods.

14.1.4 Digital Signatures

A digital signature is used to implement electronic signatures to demonstrating the authenticity of a message or document. A valid digital signature gives a recipient to authenticate that the message was created by a known sender, and that it was not altered in transit. Digital signatures are commonly used for software distribution, financial transactions, and in other cases where it is important to detect forgery or tampering. To sign a message, the private key is used to encrypt the message and the receiver can use the public key (which is publicly available) to decrypt the cryptic message C. The fact that the receiver was able to decrypt C guarantees that the message was signed by the sender.

Lets consider the case where Alice wants to send a document to Bob, and the documents needs to be signed by Alice. Alice applies a publicly known hash function to a document to create a digest of the document. Alice then uses her private key to encrypt the digest, and sends the document to Bob with the encrypted digest. When Bob receives the document and the encrypted digest, he uses Alice's public key to decrypt the digest. Bob now applies the same hash function used by Alice on the document to directly obtain the digest, and compares its value with the decrypted digest. If the values match, it proves that Alice signed the document and that the document was not altered.

14.2 RSA

RSA is an algorithm for public-key cryptography that is based on the presumed difficulty of factoring large numbers generated from two prime numbers. The algorithm involves encryption, decryption, and generating public and private keys. The public key is know to everyone and is used to encrypt messages and only the private key can decrypt those messages.

The key generation process in RSA involves two distinct prime numbers p and q of similar bit-length, chosen at random. A number n is generated by computing the product of p and q, and is used as the modulus for both the public and private key. The public key consists of the modulus n and the public exponent e. The private key consists of the modulus n and the private exponent d where $d = e^{-1} mod \phi(n)$,

To securely transmit a message, the sender turns the message into an integer m such that $0 < m < n$ using a padding scheme. The sender then computes the ciphertext c such that $c = m^e mod n$, and transmits c to the receiver. The receiver gets m from c using the private key exponent d and the equation $m = c^d n$. Using the padding scheme the receiver can get the original message from m.

14.3 Public Key Infrastructure (PKI)

One big problem with public key cryptography is the storage and authentication of public keys. In cryptography, PKI is an agreement that binds a public key to its owner by the means of a trusted third party called Certificate Authority (CA). The binding is established when the owner registered its identification and gets issued a key from the CA. CA are the trusted component that authenticates public keys and are the root of trust in the chain of public key cryptography. The main role of a CA is to digitally sign and publish the public key bound to a given user. CA uses its private key to digitally sign public keys.

All of web commerce relies on CA to provide authenticated public keys to users accessing a website. The list of all CA are stored in the browser and keys are cached so that the authentication is only required once. The government has outsourced the PKI to companies like VerSign. The security community views PKI as a huge problem.

14.4 Integrity

In information security, integrity of a data means the data has not been altered. One way to check the integrity of the data is to compute the checksum by hashing fixed blocks of data. MD5, SHA-1 etc. are examples of cryptographically secure hash functions that are used to compute the checksum. These function try to create a more complex function that rely of each bit of the data. This greatly reduces the chances of hash collisions. Using cryptographically secure hash function, its hard to deliver a payload where the

hash matches but the payloads are different. These hash functions are also used to generate salt for storing passwords or to verify if any changes where made to some content.

14.5 Security Models for Mobile Device

14.5.1 iPhone Model

iOS provides security by isolating data and communication of every application running on the phone. The isolation is not only for processes executing on the phone, but also for the data generated and used by each application. iOS creates a separate directory for storing each application. The application can only run and access data within its directory. The directory structure and permissions is similar to Unix where the owner (application) of the directory has read, write and execute control and permissions of everyone else is null. All applications written for the iPhone have to be approved by Apple and applications can only be downloaded through iTunes.

14.5.2 Andriod Model

Andriod on the other hand has an open architecture for their application. There is an open market place where people post and download apps written in Java. When installing an application, the user is asked to set permissions to access location, internet, address book etc. If the user declines to grant permissions, then the application is not downloaded on the phone. The user can also change permissions for individual operation or data (e.g. GPS, mail, contacts) access by an application.