CPSC 467: Cryptography and Computer Security

Instructor: Michael Fischer
Lecture by Ewa Syta

Lecture 24b
December 2, 2013

Encryption with Special Properties

Homomorphic Encryption

Encryption with Other Properties

Encryption with Special Properties

Goals of encryption

The main goal of encryption is to provide data confidentiality.

Normally, there is a lot you can do with your unencrypted data: analyze, search, compute, etc.
However, once data is encrypted there is not much you can do with it.

Encrypted data $\rightarrow$ secured and useless
Unencrypted data $\rightarrow$ unsecured and useful[1]

[1] Slight oversimplification

Working with encrypted data

Solution: Encrypt – decrypt – perform operations – re-encrypt

Problems: Can get very expensive very quickly. Privacy issues.

Another solution: Perform at least *some* operations on encrypted data *without* decrypting it.

Problems: How do we do that? What operations should be allowed? Will it affect security properties of the encryption scheme?

Encryption with special properties

The goal is to design an encryption function E so that we can perform meaningful operations on the ciphertexts without decrypting it.

To make it possible, E would have to "give" some special properties to the ciphertext.

Homomorphic Encryption

Homomorphic encryption

Informally, homomorphic encryption is an encryption scheme with a special property that allows operations applied to ciphertext be preserved and carried over to the plaintext.

Types of homomorphism

An encryption scheme can be homomorphic with respect to one or more group operators.

An encryption scheme is *additively* homomorphic if we consider the addition operator, and *multiplicatively* homomorphic if we consider the multiplication operator.

Types of homomorphic encryption

Partially homomorphic encryption – it is possible to perform operations on encrypted data with respect to one group operator. For example, from $E(x)$, $E(y)$ compute $E(x + y)$ but not $E(x * y)$.

Fully homomorphic encryption – it is possible to perform operations on encrypted data with respect to two group operator. For example, from $E(x)$, $E(y)$ compute $E(x + y)$ and $E(x * y)$.

Somewhat homomorphic encryption – it is possible to perform a limited number of operations on encrypted data with respect to two group operators. For example, we can only evaluate low-degree polynomials over encrypted data.

Applications of homomorphic encryption

Applications of homomorphic encryption

- Cloud computing (untrusted third parties can be used)
- E-voting (votes can be counted without revealing what they are)
- Private information retrieval (searching encrypted databases)

Partially homomorphic encryption schemes

There are many encryption schemes which have the desired homomorphic property.

You should be familiar with at least some of them:

- RSA (multiplicatively)
- ElGamal (multiplicatively)
- Goldwasser-Micali (additively)

Fully homomorphic encryption

The first fully homomorphic encryption scheme using lattice–based cryptography was presented by Craig Gentry in 2009.[2]

Later in 2009 a second fully homomorphic encryption scheme which does not require ideal lattices was presented.[3]

[2] C. Gentry, *Fully Homomorphic Encryption Using Ideal Lattices*, STOC 2009

[3] M. van Dijk, C. Gentry, S. Halevi and V. Vaikuntanathan *Fully Homomorphic Encryption over the Integers*, Eurocrypt 2010

FHE performance

Gentry estimated[4] that performing a Google search with encrypted keywords would increase the amount of computing time by about a trillion. Moore's law calculates that it would be 40 years before that homomorphic search would be as efficient as a search today.

At Eurocrypt 2010, Craig Gentry and Shai Halevi presented a working implementation of fully homomorphic encryption. Martin van Dijk about the efficiency:

> "Computation, ciphertext-expansion are polynomial, but a rather large one..."

[4] M. Cooney, *IBM Touts Encryption Innovation. New technology performs calculations on encrypted data without decrypting it* computerworld.com

Security of homomorphic encryption

Let's (informally) rephrase what homomorphic encryption is.

"If you encrypt some plaintext using homomorphic encryption, then by changing ciphertext you can change the corresponding plaintext".

Is it a good or bad property?

Security of homomorphic encryption

Non–malleability is a desirable security goal for encryption schemes so that the attacker cannot tamper with the ciphertext to affect the plaintext and go undetected.

However, homomorphic encryption implies malleability!

To reconcile this situation, we want an encryption scheme to be non-malleable except for some desired operations.

However, it's difficult to capture the notion of "some malleability allowed."[5]

[5] B. Hemenway and R. Ostrovsky, *On Homomorphic Encryption and Chosen-Ciphertext Security*, PKC 2012

Encryption with Other Properties

Encryption schemes

We will have a look at the following schemes:

1. Searchable encryption
2. Deniable encryption
3. Signcryption
4. Identity-based encryption

Searchable encryption

Searchable encryption[6] allows to test whether certain keywords are included in a ciphertext message without decrypting it or learning anything about its content.

[6] D. Boneh, G. Crescenzo, R. Ostrovsky and G. Persiano, *Public Key Encryption with Keyword Search*, EUROCRYPT 2004

SES scenario

Suppose Alice wishes to read her email on a number of devices. Alice's mail server can route emails based on the keywords in the email. For example, when Bob sends email with the keyword "urgent" the email is routed to Alice's pager.

Bob sends encrypted email to Alice using Alice's public key. Both the contents of the email and the keywords are encrypted.

The goal is to enable Alice to give the server the ability to test whether "urgent" is a keyword in the email, but the server should learn nothing else about the email.

SES highlights

Bob encrypts his email using a standard public key system and Alice's public key A_{pub}, and appends to the resulting ciphertext a Searchable Encryption (SES) of some keywords $W_1, \ldots, W_n$.

$$E_{A_{pub}}(M) || SES(A_{pub}, W_1, \ldots, W_n)$$

Alice gives the server a certain trapdoor T_W which depends on A_{priv}. This enables the server to test whether the SES ciphertext equals to some known keyword(s) but nothing else.

Bob generates the searchable encryption just given Alice's public key.

Deniable encryption

Deniable encryption[7] allows an encrypted message to be decrypted to different plausibly looking plaintexts, depending on the input information used.

This feature gives the sender *plausible deniability* if compelled to reveal the encryption information.

[7] R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky, *Deniable Encryption*, CRYPTO 1997

Deniable encryption scenario

Regular encryption schemes provide confidentiality of encrypted data in the presence of a (powerful) adversary who given a ciphertext is trying to learn the corresponding plaintext.

However, assume that custom agents at border crossings have the authority to request decryption keys to encrypted data on one's laptop. Let Chuck be a custom agent.

It is difficult to provide security in such an attack scenario. Amazingly, deniable encryption offers some protection against this very different and more hostile attack.

Types of deniable encryption schemes

Deniable encryption schemes can be categorized according to which parties may be coerced:

- Sender deniable
- Receiver deniable
- Sender–and–receiver deniable

Deniable encryption can be symmetric or asymmetric.

Public-key sender-deniable encryption scheme

Assume that only the receiver possesses the decryption key, m is the correct plaintext and $c = E(m, r)$ is the corresponding ciphertext, where r is the key and possibly other input parameters.

If approached by Chuck, Alice can reveal fake parameters that yield a plaintext m_f instead of the original plaintext m.

The goal is to present r_f and $m_f \neq m$ such that $c = E(m_f, r_f)$ where m_f is a plausibly looking plaintext. The protocol for finding such m_f and r_f is called a *faking algorithm*.

Signcryption

Encryption and signature schemes are the basic tools offered by public key cryptography.

They are normally viewed as important but distinct building blocks for higher level protocols, but there are many settings were both are needed.

Signcryption[8] is a scheme that provides both functionalities simultaneously.

[8] From ECRYPT report D.AZTEC.7, *New Technical Trends in Asymmetric Cryptography*.

Signcryption scenario

Consider a secure email as an example.

Encryption provides confidentiality of the content and digital signature provides authenticity.

Performing both operations at once improves efficiency and usability.

Identity-based encryption

ID-based encryption allows to use some known aspect of the user identity, for example an email address or IP address, to generate a public key.

This means that Alice can send confidential messages to anyone, even people who has not set up their public keys yet! Unfortunately, it will be a bit tricky to retrieve the corresponding private key.

First was proposed by Adi Shamir in 1984 as one solution to the key distribution problem.[9]

[9] Adi Shamir, *Identity-Based Cryptosystems and Signature Schemes*, CRYPTO 1984

Identity-based encryption system

Public keys are derived from a known identity value such as an arbitrary ASCII string. Corresponding private keys are generated by a trusted third party called the Private Key Generator (PKG).

Before ID-based system can be used, a PKG needs to be established and its master public key made available.

If Bob wants to send an email to Alice, he computes her public key by combining the master public key with Alice's identity information.

If Alice wants to read Bob's message, she contacts PKG, which uses the master private key to generate the private key for Alice.

Issues with Identity-based encryption

There three big issues with this scheme:

1. PKG must be trusted
2. Authenticity of the private key requestor needs to be established.
3. Private key must be securely transmitted.

Additional Resources

More information on encryption with special properties: *New Technical Trends in Asymmetric Cryptography*, ECRYPT report D.AZTEC.7,

`http://www.ecrypt.eu.org/ecrypt1/documents/D.AZTEC.7.pdf`

Tutorial on homomorphic encryption by Shai Halevi presented at CRYPTO 2011. Video and slides.

`http://people.csail.mit.edu/shaih/presentations.html`