Practical Cryptography for a Peer-to-Peer Web Browsing System

A. Pokluda

Cheriton School of Computer Science
University of Waterloo

CS758 Cryptography and Network Security Project

Outline

Overview of Project Objectives

1. Perform a brief survey of the protocols and schemes used in real-world peer-to-peer systems
2.
    1. Identify the general security related requirements for a new peer-to-peer web browsing system and identify the cryptographic protocols that meet those security requirements.
    2. Identify one or two schemes to solve each problem that meet the security requirements. These may be the same schemes that are used in current peer-to-peer systems or are new schemes obtained from the literature.
3. Compare production-level implementations in C and evaluate them on a number of criteria: level of security; CPU time and memory requirements; and performance over limited bandwidth network connection.

Overview of Project Objectives

1. Perform a brief survey of the protocols and schemes used in real-world peer-to-peer systems
2.
   1. Identify the general security related requirements for a new peer-to-peer web browsing system and identify the cryptographic protocols that meet those security requirements.
   2. Identify one or two schemes to solve each problem that meet the security requirements. These may be the same schemes that are used in current peer-to-peer systems or are new schemes obtained from the literature.
3. Compare production-level implementations in C and evaluate them on a number of criteria: level of security; CPU time and memory requirements; and performance over limited bandwidth network connection.

Overview of Project Objectives

1. Perform a brief survey of the protocols and schemes used in real-world peer-to-peer systems
2.
    1. Identify the general security related requirements for a new peer-to-peer web browsing system and identify the cryptographic protocols that meet those security requirements.
    2. Identify one or two schemes to solve each problem that meet the security requirements. These may be the same schemes that are used in current peer-to-peer systems or are new schemes obtained from the literature.
3. Compare production-level implementations in C and evaluate them on a number of criteria: level of security; CPU time and memory requirements; and performance over limited bandwidth network connection.

Skype

Key Agreement RSA with 1536- to 2048-bit key lengths

Block Cipher 256-bit AES

Public-Key Infrastructure The Skype "login server" performs the role of TA and certifies user public keys

Skype is proprietary, closed-source software and all network traffic is encrypted. There have been some efforts to document the Skype peer-to-peer architecture, but not much is known about the inner-workings of Skype software.

BitTorrent

Most current BitTorrent clients use a custom encryption scheme known as “Message Stream Encryption” (MSE)

Key Agreement Diffie-Hellman with 768-bit key lengths

Block Cipher RC4

Public-Key Infrastructure None; New public keys are generated for each session

Hash Functions Content is located using `.torrent` metainfo files containing an index of data chucks needed to reconstruct a file or set of files and their SHA-1 hash values; A metainfo file itself is identified by the SHA-1 hash of the index (known as an *info hash*)

Context

Diffie-Hellman Parameters

p is a published, 768-bit safe prime, 0xFF...63
Generator G is 2
r_A and r_B are random ints between 128- and 180-bits long
Public key of A is $Y_A = G^{r_A} \mod p$
Public key of B is $Y_B = G^{r_B} \mod p$
The shared secret is $S = Y_A^{r_B} \mod p = Y_B^{r_A} \mod p$

Constants/Variables

PadA and *PadB* are random data with length 0-512 bytes
$T_{info\ hash}$ is the info hash of the torrent
VC is a *verification constant* defined to be 8 bytes set to 0

Operation

Alice		Bob
$Y_A = G^{r_A} \mod p$		
	$\xrightarrow{Y_A,\ PadA}$	
		$Y_B = G^{r_B} \mod p$
	$\xleftarrow{Y_B,\ PadB}$	
$S = Y_B^{r_A} \mod p$		$S = Y_A^{r_B} \mod p$
$K_A = H(\text{'keyA'}, S, T_{info\ hash})$		$K_A = H(\text{'keyA'}, S, T_{info\ hash})$
$K_B = H(\text{'keyB'}, S, T_{info\ hash})$		$K_B = H(\text{'keyB'}, S, T_{info\ hash})$
	$\xrightarrow{H(\text{'req1'}, S),\ H(\text{'req2'}, T_{info\ hash}) \oplus H(\text{'req3'}, S),\ e_{K_A}(VC)}$	
	$\xleftarrow{e_{K_B}(VC)}$	

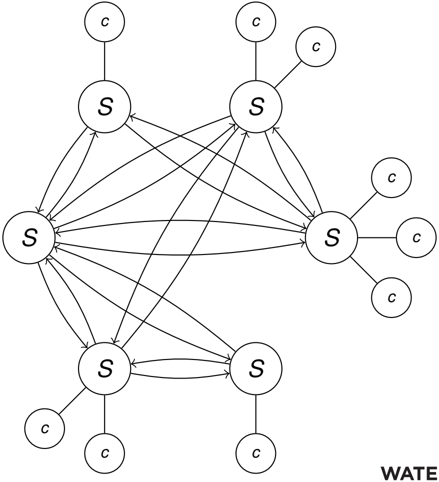
Basic Architecture

At the **outer level**

- users will use their web browsers to communicate with the peer-to-peer Web software

At the **inner level**

- instances of the peer-to-peer Web software will communicate with each other using a peer-to-peer network overlay



New Challenges

- We are designing a new system from scratch
- We can learn from existing systems, such as BitTorrent
- However, there are several fundamental differences between a web browsing system and BitTorrent from a **security perspective**:
  - BitTorrent provides no way to verify the identity of the source of a content
  - BitTorrent provides no way to update content once it has been released

What Protocols Are Needed?

Like BitTorrent,

- We can secure peer to peer communications from eavesdropping by using a key agreement scheme and block cipher
- We can locate content using a hash function

Unlike BitTorrent,

- We can bind the identity of an author to content using a signature scheme
- We need a public key infrastructure to support the verification of signatures

Side Note

In a distributed system such as this, a Web of Trust is preferable to a TA.

What Protocols Are Needed?

Like BitTorrent,

- We can secure peer to peer communications from eavesdropping by using a key agreement scheme and block cipher
- We can locate content using a hash function

Unlike BitTorrent,

- We can bind the identity of an author to content using a signature scheme
- We need a public key infrastructure to support the verification of signatures

Side Note

In a distributed system such as this, a Web of Trust is preferable to a TA.

What Protocols Are Needed?

Like BitTorrent,

- We can secure peer to peer communications from eavesdropping by using a key agreement scheme and block cipher
- We can locate content using a hash function

Unlike BitTorrent,

- We can bind the identity of an author to content using a signature scheme
- We need a public key infrastructure to support the verification of signatures

Side Note

In a distributed system such as this, a Web of Trust is preferable to a TA.

Schemes Implementing the Protocols

- **Key Agreement** Diffie-Hellman
- **Block Cipher** AES
- **Hash Function** SHA-256, MD6
- **Signature Scheme** ElGamal, DSA
- **Public Key Infrastructure** custom based on DHT

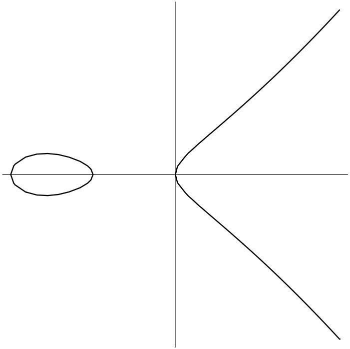
The Diffie-Hellman, ElGamal, and DSA schemes can be implemented in a **Finite Multiplicative Group** or on an **Elliptic Curve over a Finite Field**.

A Brief Introduction to Elliptic Curve Cryptography

Definition

An Elliptic Curve is the set E of solutions $(x, y) \in \mathbb{R}^2$ to the equation $y^2 = x^3 + ax + b$ together with a special point called the *point at infinity*.

An Elliptic Curve Over $\mathbb{R}^2$



A Brief Introduction to Elliptic Curve Cryptography

Point Arithmetic

We define a binary operation over E which makes E into an **abelian group**, denoted by $+$. The point at infinity $\mathcal{O}$ is the identity element, thus $\mathcal{O} + P = P + \mathcal{O} = P$ for all $P \in E$.

If $x_1 \neq x_2$ then $P + Q = (x_1, y_1) + (x_2, y_2) = (x_3, y_3)$, where

$$x_3 = \lambda^2 - x_1 - x_2$$

$$y_3 = \lambda(x_1 - x_3) - y_1$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$

Elliptic Curve vs Finite Multiplicative Group

- The primary benefit of elliptic curve cryptography is smaller key size for level of security comparable to an RSA-based system with large modulus and large key size. For example, a 256-bit ECC public key should provide comparable security to a 3072-bit RSA public key.
- The reduced key size also results in reduced storage, transmission and computational requirements. These features will likely be beneficial to our Peer-to-Peer Web Browsing System.

Work in Progress

I am currently evaluating implementations in the C programming language of the schemes mentioned previously.

Criteria: level of security; computation, storage and transmission requirements

Implementation Sources:

- Diffie-Hellman, ElGamal, DSA: I am coding these myself in FMG and EC
- AES: OpenSSL's implementation that uses hardware acceleration
- SHA-256: Crypto++ library; MD6: Rivest et al. have published C source for a reference implementation

Summary

- We briefly looked at the protocols and schemes used by real-world peer-to-peer systems
- We identified the general security requirements for a new peer-to-peer system and the cryptographic protocols that meet those security requirements
- We identified several schemes to implement the protocols
- Production-level implementation of the schemes in C are being evaluated on a number of criteria

Discussion Questions

1. What other considerations may have influenced the design of the BitTorrent encryption scheme?
2. Are there other schemes that would be more suitable for a peer-to-peer system?