Advanced Network Security

Foundations of Network Security: Part I

Thomas Martin

t.martin@mmu.ac.uk

February 16, 2017

Outline

1. Overview

2. Example Protocols

3. Mutual Authentication

Outline

1 Overview

2 Example Protocols

3 Mutual Authentication

Network Security Objectives

The primary aims of Network Security are:

- **Confidentiality**
  - **Data confidentiality:** information is not disclosed to unauthorized individuals
  - **Privacy:** Individuals control what information related to them may be collected and stored
- **Integrity**
  - **Data integrity:** information is only changed in a specified and authorized manner
  - **System integrity:** The system performs its intended function in an unimpaired manner, free from deliberate or inadvertent unauthorized manipulation
- **Availability:** Services are not denied to authorized users

Network Security Overview

The impact of the loss of any of these can be rated at different levels:

- Low: Minor degradation in organization functions
- Moderate: Significant degradation in capabilities, but primary functions can still be performed
- High: Severe degradation in capability preventing the organization from performing one of its primary functions

Beyond these objectives, two related concepts are:
Authenticity: The property of being genuine and being able to be verified and trusted; confidence in the validity of a transmission, a message, or message originator.
Accountability: All actions of an entity can be traced uniquely to that entity. This supports non-repudiation, deterrence, fault isolation, intrusion detection and prevention, and after-action recovery and legal action.

Challenges of Network Security

1. Security aims may be straightforward, but the mechanisms for achieving them are often complex
2. All known attacks must be considered, as well as potential attacks
3. Not only must security mechanisms be well designed, but they must also be correctly placed
4. Use of cryptography necessitates proper key management
5. Arms race between attacker and developer/administrator
6. Benefit from security investment is difficult to appreciate
7. Security requires constant monitoring
8. It should be built-in from the start, but is often tacked on
9. Security often viewed as unfriendly and an impediment to performing tasks

Passive Attacks

Security attacks come in two broad types: *passive attacks* and *active attacks*. Passive attacks attempt to learn information without modifying system resources, whereas active attacks attempt to alter system resources and affect their operation.

The basic passive attack is **release of message contents**. Sensitive information contained in a telephone conversation, e-mail, transferred file, etc. is obtained by an opponent.

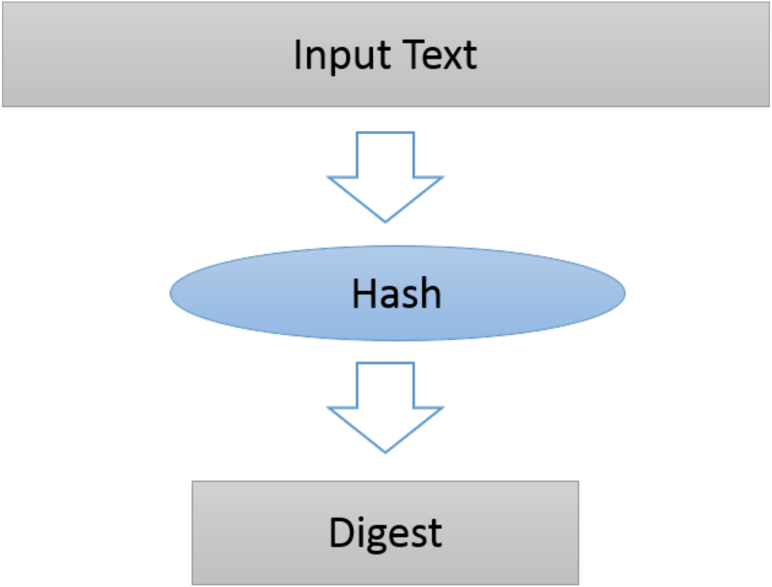
If the content is protected (encrypted), an attacker can still perform **traffic analysis**. Encryption does not protect the identities or locations of the communicating parties. An observer can obtain these details, and sometimes use leaked information to help decrypt the text.[1]

[1] Phonotactic Reconstruction of Encrypted VoIP Conversations by White et. al.

Active Attacks

Active attacks on messages in transit come in four different categories:
A **masquerade** takes place when one entity pretends to be a different entity.
A **replay** involves the passive capture of a data unit and its subsequent re-transmission to produce an unauthorized effect.
A **modification of messages** simply means that some portion of a legitimate message is altered, or that messages are delayed or reordered to produce an unauthorized effect.
A **denial of service** prevents the normal use or management of communications facilities.

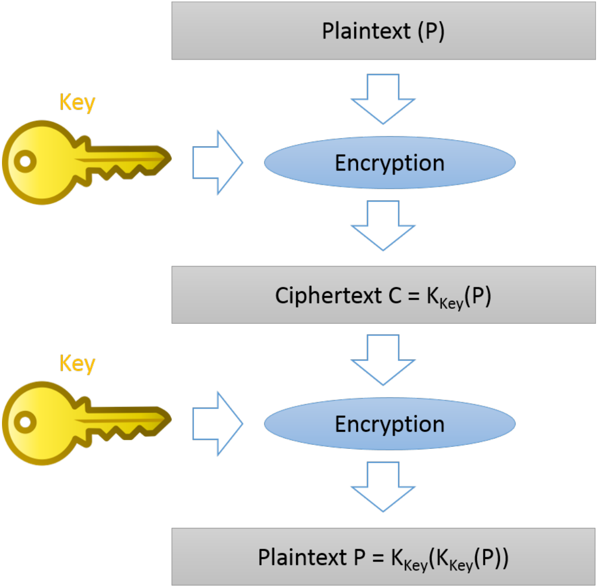
Hash Function


Input Text

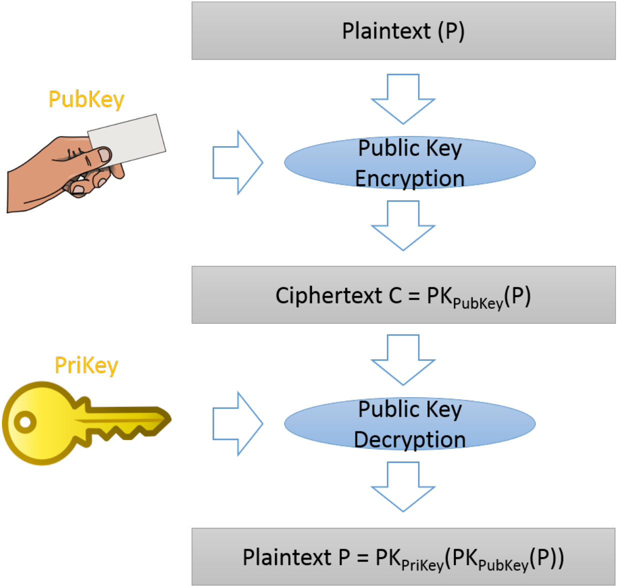
Hash

Digest

Symmetric Encryption


Plaintext (P)
Key
Encryption
Ciphertext C = $K_{Key}(P)$
Key
Encryption
Plaintext P = $K_{Key}(K_{Key}(P))$

Public Key Encryption



Plaintext (P)

PubKey

Public Key Encryption

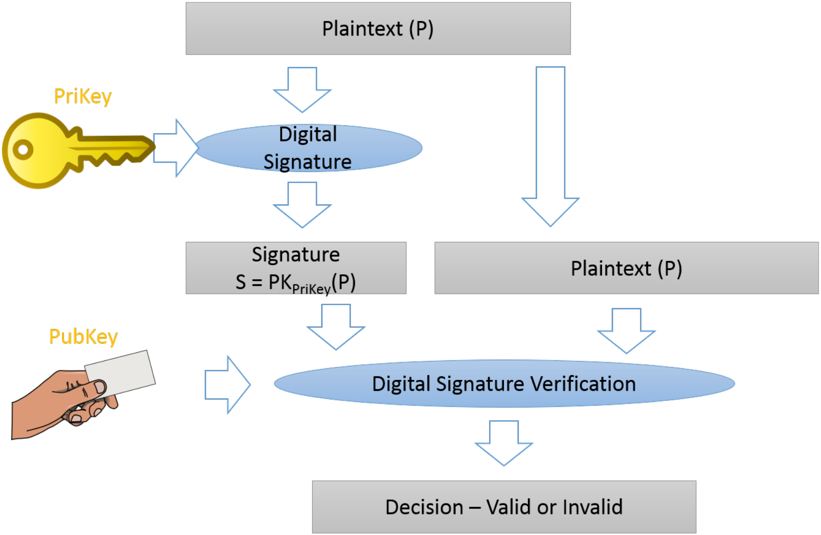
Ciphertext C = $PK_{PubKey}(P)$

PriKey

Public Key Decryption

Plaintext P = $PK_{PriKey}(PK_{PubKey}(P))$

Digital Signature


Plaintext (P)

PriKey

Digital Signature

Signature
S = PK$_{\text{PriKey}}$(P)

Plaintext (P)

PubKey

Digital Signature Verification

Decision – Valid or Invalid

Outline

1 Overview

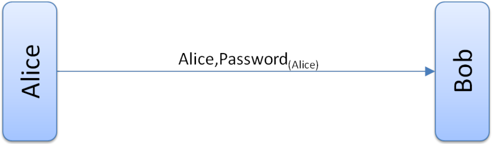
2 Example Protocols

3 Mutual Authentication

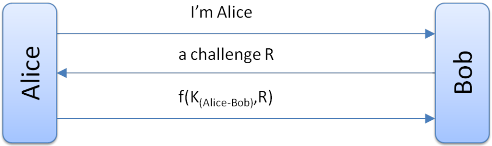
Security Handshakes

The initial communication between two parties is typically called a *handshake*. Different protocols have been built up around different possible handshakes. Assumptions can be made about what information the two participants (typically named Alice and Bob) have, what third parties they trust, etc.

The properties we use to evaluate the protocols are number of messages, processing power required, compactness of messages and security.



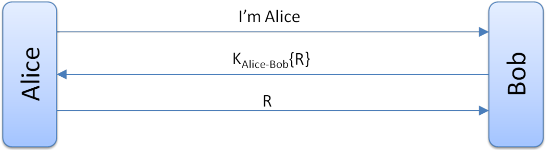
Shared Secret



If the function f is such that it cannot (practically) be reversed, this is a significant improvement on the previous protocol. However:

- Authentication is not mutual
- Eve can hijack the conversation after the initial exchange
- If the key is based on a password, then an attacker can use the information sent to attempt a brute-force/dictionary attack
- If someone compromises the key database at Bob, they can impersonate Alice (or anyone else)

Shared Secret

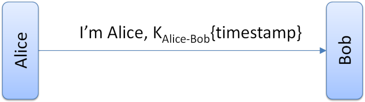


A minor variant of the previous protocol uses a symmetric encryption algorithm.

While a hash function (or MAC) could have been used for the previous protocol, the above requires a reversible encryption algorithm.

If $K_{Alice-Bob}$ is derived from a password, and R is noticeably different from random data, then Eve can brute-force the password just from obtaining any $K_{Alice-Bob}\{R\}$.

Timestamp



The handshake can be shortened by including a timestamp (string that represents current time and date).

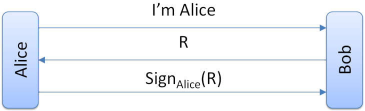
Alternative: I'm Alice, timestamp, $hash(K_{Alice-Bob}, timestamp)$

A very easy modification to make to an existing protocol for sending a cleartext password (provided clocks are synced).

Protocol is now more efficient, as only a single message is sent.

Depending on how much of a delay Bob is willing to accept, there is a window in which Alice can be impersonated (either to Bob, or to another server that shares the same key with Alice).

One-Way Public Key

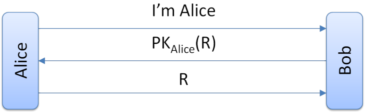


Bob will accept that the initiator is Alice if the signature is verified with Alice's public key.

Using public keys removes the need for arranging a shared secret between Alice and Bob.

Also, there is no database at Bob's server that if it were disclosed would allow an attacker to impersonate Alice. However, the database of public keys must still be protected from unauthorized modification.

One-Way Public Key



In this variant, Bob encrypts a random challenge with Alice's public key that Alice must decrypt.

One issue with this is that some public key systems only provide signatures, not encryption (DSS).

More importantly, these protocols allow an attacker to manipulate Alice into signing/decrypting any random value (if they send challenges while posing as Bob).

Eavesdrop

Even if Malory can watch the messages between Alice and Bob pass over the networks, he should not be able to:

- learn the contents of the messages between Alice and Bob
- learn information that would enable her to impersonate either Alice or Bob in a subsequent exchange
- learn information that would enable her to impersonate Alice to another replica of Bob
- learn information that would permit her to do an off-line password-guessing attack against either Alice's or Bob's secret information

Initiate a conversation pretending to be Alice

Malory can send a message to Bob and claim to be Alice and proceed at least partway through an authentication exchange. In doing so, she should not be able to:

- convince Bob she is Alice
- learn information that would enable her to do an off-line password-guessing attack against either Alice's or Bob's secret information
- learn information that would enable her to impersonate Alice on a subsequent (or interleaved) attempt
- learn information that would enable her to impersonate Bob to Alice
- trick Bob into signing or decrypting something

Lie in wait at Bob's network address and accept a connection from Alice

Malory can get at least partway through an authentication exchange. In doing so, she should not be able to

- convince Alice that Malory is Bob
- learn information that would enable her to do an off-line password-guessing attack against either Alice's or Bob's secret information
- learn information that would enable her to impersonate Bob on a subsequent attempt
- learn information that would enable her to impersonate Alice to Bob
- trick Alice into signing or decrypting something

Read Alice's database

If Malory can get all of Alice's secrets, she can convince Bob she is Alice. She can also conduct an off-line password-guessing attack against Bob's secret information (assuming Bob's secret is derived from a password), since Alice must have enough information to know if someone is really Bob. She should not be able to:

- impersonate Bob to Alice
- decrypt old recorded conversations between Alice and Bob

The same holds if Malory can get all of Bob's secrets (but with the names changed).

Sit on the channel between Alice and Bob

If Malory can examine and/or modify all messages in transit between Alice and Bob, she can prevent Alice and Bob from communicating. However, she should not be able to:

- learn information that would permit her to do an off-line password-guessing attack against either Alice's or Bob's secret information
- learn the contents of Alice's or Bob's messages
- hijack a conversation (continue a conversation started up by a legitimate party without the other side noticing the change)
- modify messages or rearrange/replay/reverse the direction of messages, causing Alice and/or Bob to misinterpret their messages to one another

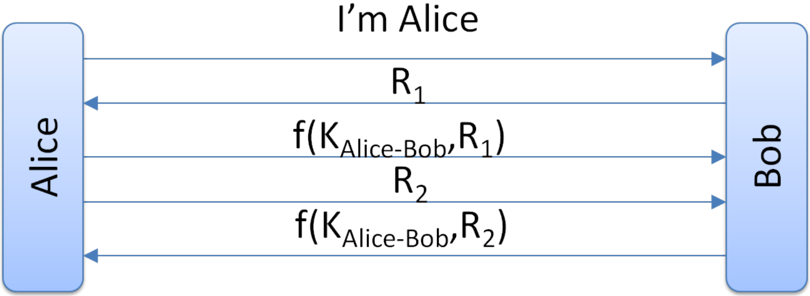
Outline

1 Overview

2 Example Protocols

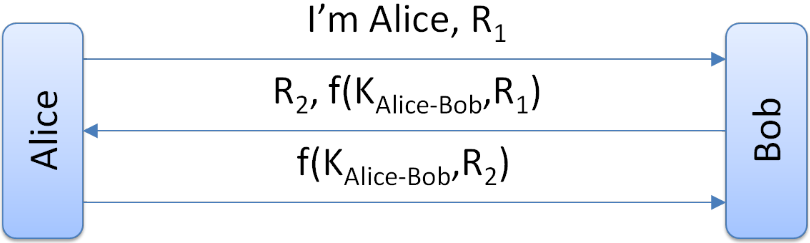
3 Mutual Authentication

Mutual Authentication


Alice
Bob
I'm Alice
R_1
$f(K_{Alice\text{-}Bob}, R_1)$
R_2
$f(K_{Alice\text{-}Bob}, R_2)$

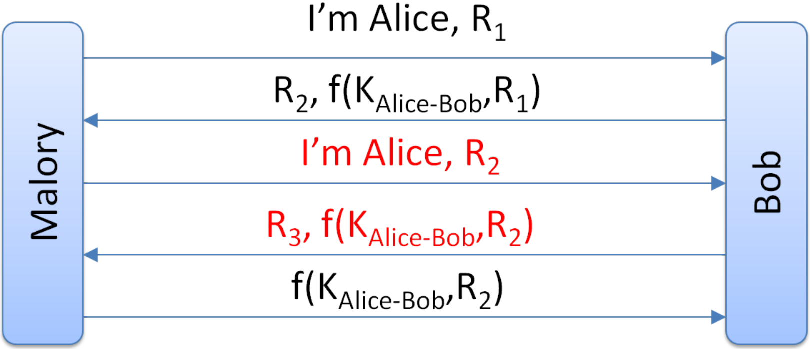
To provide authentication of Bob to Alice as well as Alice to Bob, we could make the above simple variation. After we have finished authenticating one way, we reverse the challenge/response. However, this is inefficient, and the same can be achieved with fewer messages.

Mutual Authentication


Alice
Bob
I'm Alice, R_1
R_2, $f(K_{Alice\text{-}Bob}, R_1)$
$f(K_{Alice\text{-}Bob}, R_2)$

However, this protocol is vulnerable to a replay attack, allowing someone to impersonate Alice to Bob. The attacker can start two separate sessions with Bob, and use Bob's response from one to complete the protocol in the other.

Reflection Attack



It is the similarity between the encrypted/hashed parts of messages 2 and 3 that allows this attack to occur.

Reflection Attack

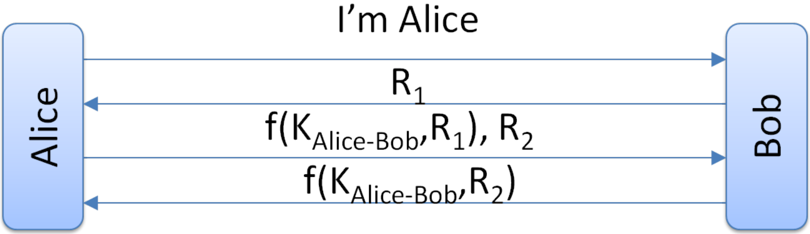
From looking at the protocol, it might seem obvious that Bob can tell the sender is being deceptive. But the security of the protocol should not rely on Bob's ability to tell that two requests are coming from the same sender (intermediaries could always be used). Also, server need to be as basic as possible, which means limiting how much is remembered about each request.

These types of attacks can be prevented by *not having Alice and Bob do exactly the same thing*:

- Use different keys: Either different keys, or easily derived keys
- Use different challenges: Add something to the random challenge before applying the function

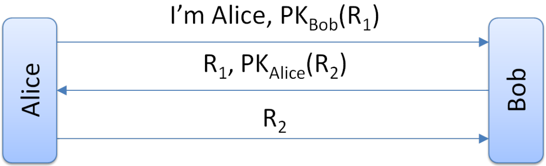
Note that the first Mutual Authentication protocol shown is not vulnerable as the initiator is the first to prove its identity (good general principle).

Password Guessing



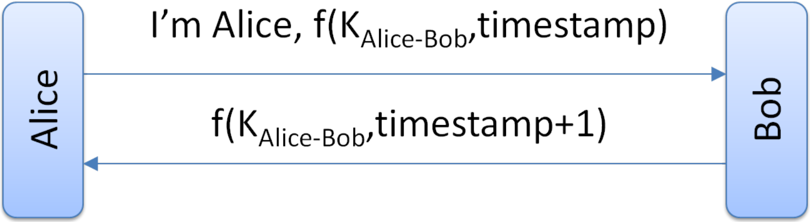
Assuming the key is password-based, this protocol makes it more difficult for an attacker to get enough information to brute-force it. Bob does not reveal anything until Alice has proven her identity. However, if Malory can impersonate Bob and get Alice to initiate a connection to her, she can get a pair $(R, f(K_{Alice-Bob}, R)$ to test passwords against.

Two-Way Public Key



Mutual Authentication can be done with public keys and three messages. However, this adds the requirement that Bob must have the public key of Alice (and of anyone else who wishes to connect). Managing the private keys is also more complex. Symmetric encryption keys can be generated from passwords, as they are simply binary strings. Asymmetric keys are more complex, and cannot be generated in the same way.

Timestamps


Alice
Bob
I’m Alice, $f(K_{Alice-Bob}, timestamp)$
$f(K_{Alice-Bob}, timestamp+1)$

There are several issues that arise when using timestamps in authentication protocols. It is important to use a different reply, even if the clock has not advanced (say it was rounded to the nearest minute). Clock skew, the difference in time at different nodes in a network must also be considered (i.e. a certain amount of skew accepted).

Thank you

Relevant chapters:

Corley and Simpson, Hands-on Ethical Hacking and Network Defense, Chapter 8.

Kaufman, Network Security, Chapters 9, 10, and 11.

Any Questions?