Access Control and Privacy Policies (8)

Email:	christian.urban at kcl.ac.uk
Office:	S1.27 (1st floor Strand Building)
Slides:	KEATS (also homework is there)

Last Week

Andrew Secure RPC Protocol: A and B share a key private K_{AB} and want to identify each other

- A sends B : A, N_A
- B sends A : $\{N_A, K'_{AB}\}_{K_{AB}}$
- A sends B : $\{N_A\}_{K'_{AB}}$

Protocols

A sends $B : \ldots$

- by convention A, B are named principals Alice... but most likely they are programs, which just follow some instructions (they are more like roles)

Protocols

$$\begin{array}{l} A \text{ sends } B : \ldots \\ B \text{ sends } A : \ldots \\ \quad \vdots \end{array}$$

- by convention A, B are named principals Alice... but most likely they are programs, which just follow some instructions (they are more like roles)

- indicates one "protocol run", or session, which specifies some order in the communication
- there can be several sessions in parallel (think of wifi routers)

Last Week

A and B share the key K_{AB} and want to identify each other

- A sends B : A, N_A
- B sends A : $\{N_A, K'_{AB}\}_{K_{AB}}$
- A sends B : $\{N_A\}_{K'_{AB}}$

Defeating Challenge-Response

A reflection attack: an intruder I impersonates B.

A sends I : A, N_A

Defeating Challenge-Response

A reflection attack: an intruder I impersonates B.

A sends I : A, N_A I sends A : B, N_A

Defeating Challenge-Response

A reflection attack: an intruder I impersonates B.

A sends I : A, N_A

I sends A : B, N_A

A sends I : $\{N_A, K'_{AB}\}_{K_{AB}}$

Defeating Challenge-Response

A reflection attack: an intruder I impersonates B.

A sends I : A, N_A

I sends A : $\{N_A, K'_{AB}\}_{K_{AB}}$

I sends A : B, N_A

A sends I : $\{N_A, K'_{AB}\}_{K_{AB}}$

Defeating Challenge-Response

A reflection attack: an intruder I impersonates B.

A sends I : A, N_A
I sends A : $\{N_A, K'_{AB}\}_{K_{AB}}$
A sends I : $\{N_A\}_{K'_{AB}}$

I sends A : B, N_A
A sends I : $\{N_A, K'_{AB}\}_{K_{AB}}$

Defeating Challenge-Response

A reflection attack: an intruder I impersonates B.

A sends I : A, N_A	I sends A : B, N_A
I sends A : $\{N_A, K'_{AB}\}_{K_{AB}}$	A sends I : $\{N_A, K'_{AB}\}_{K_{AB}}$
A sends I : $\{N_A\}_{K'_{AB}}$	I sends A : $\{N_A\}_{K'_{AB}}$

Defeating Challenge-Response

A reflection attack: an intruder I impersonates B.

A sends I : A, N_A	I sends A : B, N_A
I sends A : $\{N_A, K'_{AB}\}_{K_{AB}}$	A sends I : $\{N_A, K'_{AB}\}_{K_{AB}}$
A sends I : $\{N_A\}_{K'_{AB}}$	I sends A : $\{N_A\}_{K'_{AB}}$

Sounds stupid: “...answering a question with a counter question”

was originally developed at CMU for terminals to connect to workstations (e.g., file servers)

Identify Friend or Foe

Identify Friend or Foe

198?: war between Angola (supported by Cuba) and Namibia (supported by SA)

Identify Friend or Foe

198?: war between Angola (supported by Cuba) and Namibia (supported by SA)

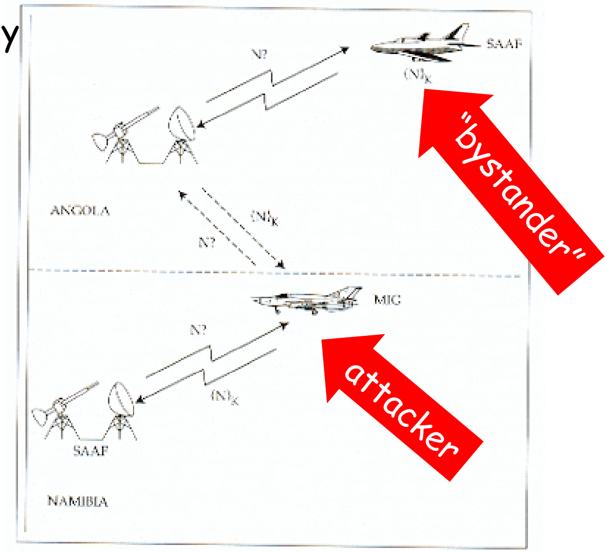


Figure 2.2 The MIG-in-the middle attack.

Identify Friend or Foe

198?: war between Angola (supported by Cuba) and Namibia (supported by SA)

being outsmarted by Angola/Cuba ended SA involvement (?)

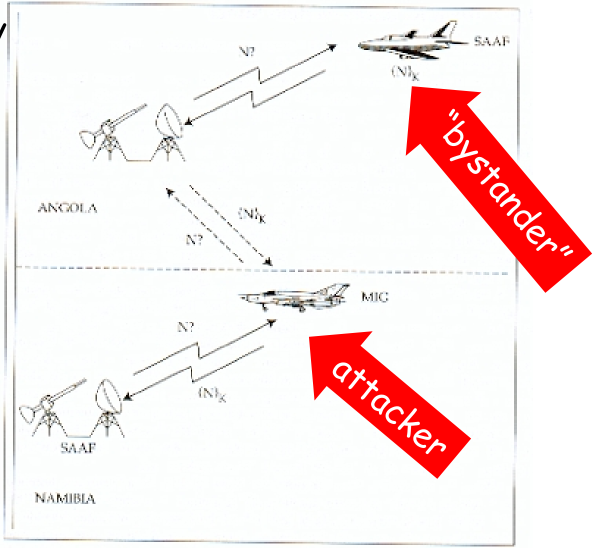


Figure 2.2 The MIG-in-the middle attack.

Identify Friend or Foe

198?: war between Angola (supported by Cuba) and Namibia (supported by SA)

being outsmarted by Angola/Cuba ended SA involvement (?)

IFF opened up a nice side-channel attack

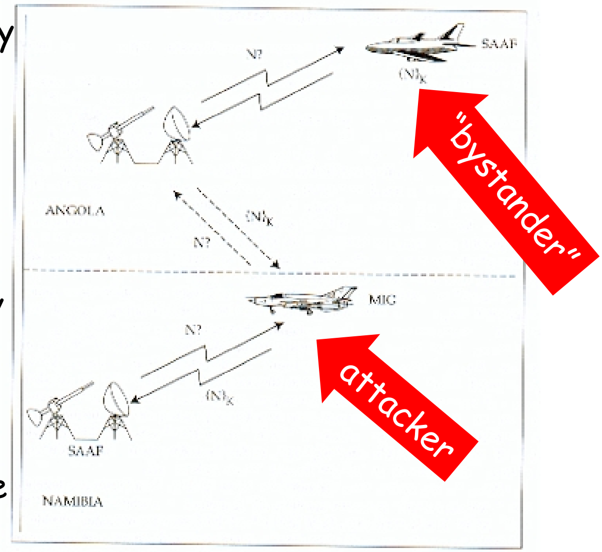


Figure 2.2 The MIG-in-the middle attack.

Encryption to the Rescue?

- A sends B : $\{A, N_A\}_{K_{AB}}$ encrypted
- B sends A : $\{N_A, K'_{AB}\}_{K_{AB}}$
- A sends B : $\{N_A\}_{K'_{AB}}$

Encryption to the Rescue?

- A sends B : $\{A, N_A\}_{K_{AB}}$ encrypted
- B sends A : $\{N_A, K'_{AB}\}_{K_{AB}}$
- A sends B : $\{N_A\}_{K'_{AB}}$

means you need to send separate "Hello" signals (bad), or worse share a single key between many entities

Protocol Attacks

- replay attacks
- reflection attacks
- man-in-the-middle attacks
- timing attacks
- parallel session attacks
- binding attacks (public key protocols)
- changing environment / changing assumptions

- (social engineering attacks)

Replay Attacks

Schroeder-Needham protocol: exchange of a symmetric key with a trusted 3rd-party S:

$$
\begin{aligned}
A &\rightarrow S : A, B, N_A\\
S &\rightarrow A : \{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}\\
A &\rightarrow B : \{K_{AB}, A\}_{K_{BS}}\\
B &\rightarrow A : \{N_B\}_{K_{AB}}\\
A &\rightarrow B : \{N_B - 1\}_{K_{AB}}
\end{aligned}
$$

Replay Attacks

Schroeder-Needham protocol: exchange of a symmetric key with a trusted 3rd-party S:

$$\begin{array}{l} A \rightarrow S : A, B, N_A \\ S \rightarrow A : \{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}} \\ A \rightarrow B : \{K_{AB}, A\}_{K_{BS}} \\ B \rightarrow A : \{N_B\}_{K_{AB}} \\ A \rightarrow B : \{N_B - 1\}_{K_{AB}} \end{array}$$

at the end of the protocol both A and B should be in the possession of the secret key K_{AB} and know that the other principal has the key

Nonces

1. I generate a nonce (random number) and send it to you encrypted with a key we share
2. you increase it by one, encrypt it under a key I know and send it back to me

I can infer:

- you must have received my message
- you could only have generated your answer after I send you my initial message
- if only you and me know the key, the message must have come from you

$$
\begin{array}{l}
A \rightarrow S : A, B, N_A \\
S \rightarrow A : \{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}} \\
A \rightarrow B : \{K_{AB}, A\}_{K_{BS}} \\
B \rightarrow A : \{N_B\}_{K_{AB}} \\
A \rightarrow B : \{N_B - 1\}_{K_{AB}}
\end{array}
$$

$$A \to S : A, B, N_A$$
$$S \to A : \{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$$
$$A \to B : \{K_{AB}, A\}_{K_{BS}}$$
$$B \to A : \{N_B\}_{K_{AB}}$$
$$A \to B : \{N_B - 1\}_{K_{AB}}$$

compromise K_{AB}

$$
\begin{aligned}
&A \rightarrow S : A, B, N_A\\
&S \rightarrow A : \{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}\\
&A \rightarrow B : \{K_{AB}, A\}_{K_{BS}}\\
&B \rightarrow A : \{N_B\}_{K_{AB}}\\
&A \rightarrow B : \{N_B - 1\}_{K_{AB}}
\end{aligned}
$$

compromise K_{AB}

$$
\begin{aligned}
&A \rightarrow S : A, B, N'_A\\
&S \rightarrow A : \{N'_A, B, K'_{AB}, \{K'_{AB}, A\}_{K_{BS}}\}_{K_{AS}}
\end{aligned}
$$

$$A \to S : A, B, N_A$$
$$S \to A : \{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$$
$$A \to B : \{K_{AB}, A\}_{K_{BS}}$$
$$B \to A : \{N_B\}_{K_{AB}}$$
$$A \to B : \{N_B - 1\}_{K_{AB}}$$

compromise K_{AB}

$$A \to S : A, B, N'_A$$
$$S \to A : \{N'_A, B, K'_{AB}, \{K'_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$$
$$I(A) \to B : \{K_{AB}, A\}_{K_{BS}}$$

replay of older run

$$A \rightarrow S : A, B, N_A$$
$$S \rightarrow A : \{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$$
$$A \rightarrow B : \{K_{AB}, A\}_{K_{BS}}$$
$$B \rightarrow A : \{N_B\}_{K_{AB}}$$
$$A \rightarrow B : \{N_B - 1\}_{K_{AB}}$$

compromise K_{AB}

$$A \rightarrow S : A, B, N'_A$$
$$S \rightarrow A : \{N'_A, B, K'_{AB}, \{K'_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$$
$$I(A) \rightarrow B : \{K_{AB}, A\}_{K_{BS}}$$

replay of older run

$$B \rightarrow I(A) : \{N'_B\}_{K_{AB}}$$
$$I(A) \rightarrow B : \{N'_B - 1\}_{K_{AB}}$$

$A \to S : A, B, N_A$
$S \to A : \{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$
$A \to B : \{K_{AB}, A\}_{K_{BS}}$
$B \to A : \{N_B\}_{K_{AB}}$
$A \to B : \{N_B - 1\}_{K_{AB}}$

compromise K_{AB}

$A \to S : A, B, N'_A$
$S \to A : \{N'_A, B, K'_{AB}, \{K'_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$
$I(A) \to B : \{K_{AB}, A\}_{K_{BS}}$ replay of older run
$B \to I(A) : \{N'_B\}_{K_{AB}}$
$I(A) \to B : \{N'_B - 1\}_{K_{AB}}$

B believes it is following the correct protocol, intruder I can form the correct response because it knows K_{AB} and talks to B masquerading as A

"On the Internet, nobody knows you're a dog."

Replay Attacks

Andrew Secure RPC protocol: exchanging a new key between A and B

$$\begin{aligned}
A \rightarrow B &: A, \{N_A\}_{K_{AB}} \\
B \rightarrow A &: \{N_A + 1, N_B\}_{K_{AB}} \\
A \rightarrow B &: \{N_B + 1\}_{K_{AB}} \\
B \rightarrow A &: \{K_{AB}^{new}, N_B^{new}\}_{K_{AB}}
\end{aligned}$$

Replay Attacks

Andrew Secure RPC protocol: exchanging a new key between A and B

$$\begin{array}{l} A \to B : A, \{N_A\}_{K_{AB}} \\ B \to A : \{N_A + 1, N_B\}_{K_{AB}} \\ A \to B : \{N_B + 1\}_{K_{AB}} \\ B \to A : \{K_{AB}^{new}, N_B^{new}\}_{K_{AB}} \end{array}$$

Assume nonces are represented as bit-sequences of the same length as keys

$A \to B : A, \{N_A\}_{K_{AB}}$
$B \to A : \{N_A + 1, N_B\}_{K_{AB}}$
$A \to I(B) : \{N_B + 1\}_{K_{AB}}$ intercepts
$I(B) \to A : \{N_A + 1, N_B\}_{K_{AB}}$ resend 2nd msg

Time-Stamps

The Schroeder-Needham protocol can be fixed by including a time-stamp (e.g., in Kerberos):

$$
\begin{aligned}
A &\rightarrow S : A, B, N_A \\
S &\rightarrow A : \{N_A, B, K_{AB}, \{K_{AB}, A, T_S\}_{K_{BS}}\}_{K_{AS}} \\
A &\rightarrow B : \{K_{AB}, A, T_S\}_{K_{BS}} \\
B &\rightarrow A : \{N_B\}_{K_{AB}} \\
A &\rightarrow B : \{N_B - 1\}_{K_{AB}}
\end{aligned}
$$

Time-Stamps

The Schroeder-Needham protocol can be fixed by including a time-stamp (e.g., in Kerberos):

$$\begin{array}{l} A \to S : A, B, N_A \\ S \to A : \{N_A, B, K_{AB}, \{K_{AB}, A, T_S\}_{K_{BS}}\}_{K_{AS}} \\ A \to B : \{K_{AB}, A, T_S\}_{K_{BS}} \\ B \to A : \{N_B\}_{K_{AB}} \\ A \to B : \{N_B - 1\}_{K_{AB}} \end{array}$$

but nothing is for free: then you need to synchronise time and possibly become a victim to timing attacks

It can also be fixed by including another nonce:

$$\begin{array}{l}
A \rightarrow B : A \\
B \rightarrow A : \{A, N_B\}_{K_{BS}} \\
A \rightarrow S : A, B, N_A, \{A, N_B\}_{K_{BS}} \\
S \rightarrow A : \{N_A, B, K_{AB}, \{K_{AB}, A, N_B\}_{K_{BS}}\}_{K_{AS}} \\
A \rightarrow B : \{K_{AB}, A, N_B\}_{K_{BS}} \\
B \rightarrow A : \{N_B\}_{K_{AB}} \\
A \rightarrow B : \{N_B - 1\}_{K_{AB}}
\end{array}$$

It can also be fixed by including another nonce:

$$\begin{array}{l}
A \to B : A \\
B \to A : \{A, N_B\}_{K_{BS}} \\
A \to S : A, B, N_A, \{A, N_B\}_{K_{BS}} \\
S \to A : \{N_A, B, K_{AB}, \{K_{AB}, A, N_B\}_{K_{BS}}\}_{K_{AS}} \\
A \to B : \{K_{AB}, A, N_B\}_{K_{BS}} \\
B \to A : \{N_B\}_{K_{AB}} \\
A \to B : \{N_B - 1\}_{K_{AB}}
\end{array}$$

but nothing is for free: then you need to synchronise time and possibly become victim to timing attacks

Binding Attacks

with public-private keys it is important that the public key is bound to the right owner (verified by a certification authority CA)

$$A \to CA : A, B, N_A$$
$$CA \to A : CA, \{CA, A, N_A, K_B^{pub}\}_{K_A^{pub}}$$

A knows K_A^{priv} and can verify the message came from CA in response to A's message and trusts K_B^{pub} is B's public key

Binding Attacks

$$
\begin{aligned}
&A \to I(CA) : A, B, N_A\\
&I(A) \to CA : A, I, N_A\\
&CA \to I(A) : CA, \{CA, A, N_A, K_I^{pub}\}_{K_A^{pub}}\\
&I(CA) \to A : CA, \{CA, A, N_A, K_I^{pub}\}_{K_A^{pub}}
\end{aligned}
$$

Binding Attacks

$$\begin{array}{l} A \rightarrow I(CA) : A, B, N_A \\ I(A) \rightarrow CA : A, I, N_A \\ CA \rightarrow I(A) : CA, \{CA, A, N_A, K_I^{pub}\}_{K_A^{pub}} \\ I(CA) \rightarrow A : CA, \{CA, A, N_A, K_I^{pub}\}_{K_A^{pub}} \end{array}$$

A now encrypts messages for B with the public key of I (which happily decrypts them with its private key)

There are plenty of other protocols and attacks.
This could go on “forever”.

There are plenty of other protocols and attacks. This could go on “forever”.

We look here on one more kind of attacks that are because of a changing environment.

Changing Environment Attacks

- all protocols rely on some assumptions about the environment (e.g., cryptographic keys cannot be broken)

Changing Environment Attacks

- all protocols rely on some assumptions about the environment (e.g., cryptographic keys cannot be broken)

- in the "good olden days" (1960/70) rail transport was cheap, so fraud was not worthwhile

Changing Environment Attacks

- all protocols rely on some assumptions about the environment (e.g., cryptographic keys cannot be broken)

- when it got expensive, some people bought cheaper monthly tickets for a suburban station and a nearby one, and one for the destination and a nearby one
- a large investment later all barriers were automatic and tickets could record state

Changing Environment Attacks

- all protocols rely on some assumptions about the environment (e.g., cryptographic keys cannot be broken)

- but suddenly the environment changed: rail transport got privatised creating many competing companies potentially cheating each other
- revenue from monthly tickets was distributed according to a formula involving where the ticket was bought...

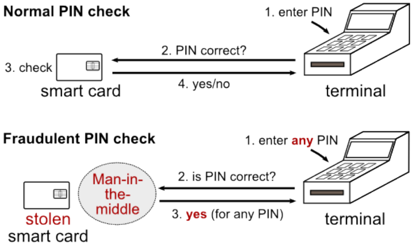
Changing Environment Attacks

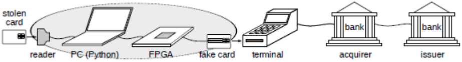
- all protocols rely on some assumptions about the environment (e.g., cryptographic keys cannot be broken)

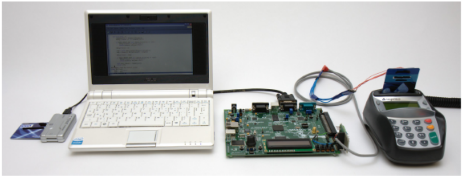
- apart from bad outsiders (passengers), you also had bad insiders (rail companies)
- chaos and litigation ensued

A Man-in-the-middle attack in real life:

- the card only says yes or no to the terminal if the PIN is correct
- trick the card in thinking transaction is verified by signature
- trick the terminal in thinking the transaction was verified by PIN







Problems with EMV

- it is a wrapper for many protocols
- specification by consensus (resulted unmanageable complexity)
- its specification is 700 pages in English plus 2000+ pages for testing, additionally some further parts are secret
- other attacks have been found
- one solution might be to require always online verification of the PIN with the bank

Problems with WEP (Wifi)

- a standard ratified in 1999
- the protocol was designed by a committee not including cryptographers
- it used the RC4 encryption algorithm which is a stream cipher requiring a unique nonce
- WEP did not allocate enough bits for the nonce
- for authenticating packets it used CRC checksum which can be easily broken
- the network password was used to directly encrypt packages (instead of a key negotiation protocol)
- encryption was turned of by default

Protocols are Difficult

- even the systems designed by experts regularly fail
- try to make everything explicit (you need to authenticate all data you might rely on)
- the one who can fix a system should also be liable for the losses
- cryptography is often not **the** answer

logic is one way protocols are studied in academia (you can use computers to search for attacks)

Public-Key Infrastructure

- the idea is to have a certificate authority (CA)
- you go to the CA to identify yourself
- CA: “I, the CA, have verified that public key P^{pub}_{Bob} belongs to Bob”

- CA must be trusted by everybody
- What happens if CA issues a false certificate? Who pays in case of loss? (VeriSign explicitly limits liability to $100.)

Privacy, Anonymity et al

Some terminology:

- secrecy is the mechanism used to limit the number of principals with access to information (eg, cryptography or access controls)
- confidentiality is the obligation to protect the secrets of other people or organizations (secrecy for the benefit of an organisation)
- anonymity is the ability to leave no evidence of an activity (eg, sharing a secret)
- privacy is the ability or right to protect your personal secrets (secrecy for the benefit of an individual)

Privacy vs Anonymity

- everybody agrees that anonymity has its uses (e.g., voting, whistleblowers, peer-review)

Privacy vs Anonymity

- everybody agrees that anonymity has its uses (e.g., voting, whistleblowers, peer-review)

But privacy?

"You have zero privacy anyway. Get over it."
Scott Mcnealy (CEO of Sun)

If you have nothing to hide, you have nothing to fear.

Privacy

private data can be often used against me

- if my location data becomes public, thieves will switch off their phones and help themselves in my home
- if supermarkets can build a profile of what I buy, they can use it to their advantage (banks - mortgages)
- my employer might not like my opinions

Privacy

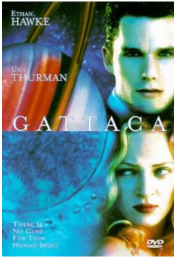
private data can be often used against me

- if my location data becomes public, thieves will switch off their phones and help themselves in my home
- if supermarkets can build a profile of what I buy, they can use it to their advantage (banks - mortgages)
- my employer might not like my opinions

- one the other hand, Freedom-of-Information Act
- medical data should be private, but medical research needs data

Privacy Problems

- Apple takes note of every dictation (send over the Internet to Apple)
- markets often only work, if data is restricted (to build trust)
- Social network can reveal data about you
- have you tried the collusion extension for FireFox?
- I do use Dropbox, store cards
- next week: anonymising data



Gattaca (1997)