Security Engineering (5)

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

Problems with Key Fobs

MONDAY 27 OCTOBER 2014 EVENING STANDARD

:ebook.com/eveningstandard
us on Twitter @standardnews

Insurers refuse to cover Range Rovers due to security flaw

Kiran Randhawa

INSURANCE companies are refusing to cover new Range Rovers in London after thieves found a way of bypassing the vehicles' keyless ignition systems.

Criminals use hand-held electronic devices, available on eBay, to get around the feature. Unless owners have secure parking, underwriters are now said to be refusing to insure them.

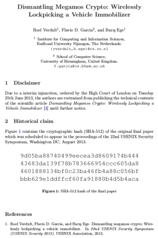
Insurers have asked to meet Jaguar Range Rover to discuss the growing problem. Thatcham Research, the motor insurers' automotive research centre, said that 294 Range Rover Evoque and Sport vehicles were stolen in London between January and July. In the same period, 63 BMW X5s, a rival to the Range Rover, were taken.

James Wasdell, co-founder of Quantum Underwriting, said: "If you are an owner of a street-parked Range Rover, nine out of 10 insurers will now say no. However, we have found a solution by combining the use of physical protection [for the car] and advising clients to insure all assets with one insurer."

Jaguar Land Rover said: "Our line-up continues to meet the insurance industry requirements. Nevertheless we are taking this issue very seriously."

Circumventing the ignition protection:

- either dismantling Megamos crypto,
- or use the diagnostic port to program blank keys



Dismantling Megamos Crypto: Wirelessly Lockpicking a Vehicle Immobilizer

Roel Verdult[1], Flavio D. Garcia[2], and Barış Ege[1]

[1] Institute for Computing and Information Sciences, Radboud University Nijmegen, The Netherlands. {rverdult,b.ege}@cs.ru.nl

[2] School of Computer Science, University of Birmingham, United Kingdom. f.garcia@cs.bham.ac.uk

1 Disclaimer

Due to a interim injunction, ordered by the High Court of London on Tuesday 25th June 2013, the authors are restrained from publishing the technical contents of the scientific article *Dismantling Megamos Crypto: Wirelessly Lockpicking a Vehicle Immobilizer* [1] until further notice.

2 Historical claim

Figure 1 contains the cryptographic hash (SHA-512) of the original final paper which was scheduled to appear in the proceedings of the 22nd USENIX Security Symposium, Washington DC, August 2013.

9d05ba88740499eecea3d8609174b444
43683da139f78b783666954ccc605da8
4601888134bf0c23ba46fb4a88c056bf
bbb629e1ddffcf60fa91880b4d5b4aca

Figure 1: SHA-512 hash of the final paper

References

1. Roel Verdult, Flavio D. Garcia, and Barış Ege. Dismantling megamos crypto: Wirelessly lockpicking a vehicle immobilizer. In *22nd USENIX Security Symposium (USENIX Security 2013)*. USENIX Association, 2013.

Protocols

- The point is that we have no control over the network
- We want to avoid that a message exchange (a protocol) can be attacked without detection

G20 Summit in 2009

- Snowden documents reveal "that during G20 meetings...GCHQ used 'ground-breaking intelligence capabilities' to intercept the communications of visiting delegations. This included setting up internet cafes where they used an email interception program and key-logging software to spy on delegates' use of computers..."
- "The G20 spying appears to have been organised for the more mundane purpose of securing an advantage in meetings."

Interlock Protocol

The interlock protocol ("best bet" against MITM):

1. $A \to B : K_A^{pub}$
2. $B \to A : K_B^{pub}$
3. $\{A, m\}_{K_B^{pub}} \mapsto H_1, H_2$
   $\{B, m'\}_{K_A^{pub}} \mapsto M_1, M_2$
4. $A \to B : H_1$
5. $B \to A : \{H_1, M_1\}_{K_A^{pub}}$
6. $A \to B : \{H_2, M_1\}_{K_B^{pub}}$
7. $B \to A : M_2$

$$A \to C : K_A^{pub}$$
$$C \to B : K_C^{pub}$$
$$B \to C : K_B^{pub}$$
$$C \to A : K_C^{pub}$$
$$\{A, m\}_{K_C^{pub}} \mapsto H_1, H_2$$
$$\{B, n\}_{K_C^{pub}} \mapsto M_1, M_2$$
$$\{C, a\}_{K_B^{pub}} \mapsto C_1, C_2$$
$$\{C, b\}_{K_A^{pub}} \mapsto D_1, D_2$$

$$A \to C : H_1$$
$$C \to B : C_1$$
$$B \to C : \{C_1, M_1\}_{K_C^{pub}}$$
$$C \to A : \{H_1, D_1\}_{K_A^{pub}}$$
$$A \to C : \{H_2, D_1\}_{K_C^{pub}}$$
$$C \to B : \{C_2, M_1\}_{K_B^{pub}}$$
$$B \to C : M_2$$
$$C \to A : D_2$$

- you have to ask something that cannot imitated (requires A and B know each other)
- what happens if m and n are voice messages?

- the moral: establishing a secure connection from "zero" is almost impossible—you need to rely on some established trust

- that is why we rely on certificates, which however are badly, badly realised (just today a POODLE attack against SSL)

Protocols

Some examples where "over-the-air" protocols are used:

- wifi
- card readers (you cannot trust the terminals)
- RFID (passports)
- car transponders

Protocols

Some examples where "over-the-air" protocols are used:

- wifi
- card readers (you cannot trust the terminals)
- RFID (passports)
- car transponders

The point is that we cannot control the network: An attacker can install a packet sniffer, inject packets, modify packets, replay messages...fake pretty much everything.

Keyless Car Transponders

- There are two security mechanisms: one remote central locking system and one passive RFID tag (engine immobiliser).
- How can I get in? How can thieves be kept out? How to avoid MITM attacks?

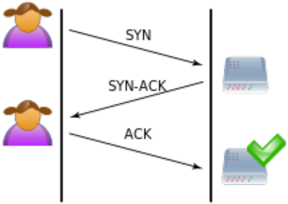
Papers: Gone in 360 Seconds: Hijacking with Hitag2,
Dismantling Megamos Crypto: Wirelessly Lockpicking
a Vehicle Immobilizer

HTTPS / GSM

- I am sitting at Starbuck. How can I be sure I am really visiting Barclays? I have no control of the access point.
- How can I achieve that a secret key is established in order to encrypt my mobile conversation? I have no control over the access points.

Handshakes

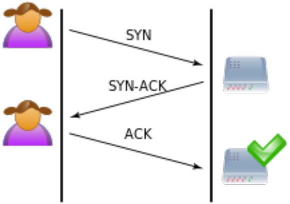
- starting a TCP connection between a client and a server initiates the following three-way handshake protocol:



Alice:	Hello server!
Server:	I heard you
Alice:	Thanks

Handshakes

- starting a TCP connection between a client and a server initiates the following three-way handshake protocol:

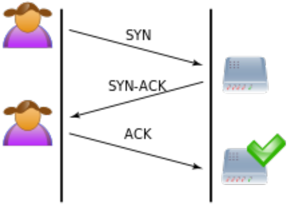


Alice:	Hello server!
Server:	I heard you
Alice:	Thanks

$$
\begin{array}{ll}
A \rightarrow S: & \text{SYN} \\
S \rightarrow A: & \text{SYN-ACK} \\
A \rightarrow S: & \text{ACK}
\end{array}
$$

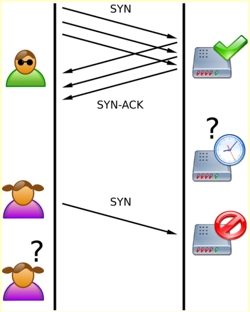
Handshakes

- starting a TCP connection between a client and a server initiates the following three-way handshake protocol:



Alice: $A \rightarrow S$: SYN
Server: $S \rightarrow A$: SYN-ACK
Alice: $A \rightarrow S$: ACK

SYNflood attacks:



Authentication

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

Knock Knock!
Who's there?
Alice.
Alice who?

Authentication Protocols

Alice (A) and Bob (B) share a secret key K_{AB}

Password transmission:

$$A \rightarrow B : K_{AB}$$

Authentication Protocols

Alice (A) and Bob (B) share a secret key K_{AB}

Password transmission:

$$A \to B : K_{AB}$$

Problems: Eavesdropper can capture the secret and replay it; B cannot confirm the identity of A

Authentication Protocols

Alice (A) and Bob (B) share a secret key K_{AB}

Simple Challenge Response (solving the replay problem):

$$\begin{array}{lll}A \rightarrow B: & \text{Hi I am A} & \\ B \rightarrow A: & N & \text{(challenge)} \\ A \rightarrow B: & \{N\}_{K_{AB}} & \end{array}$$

Authentication Protocols

Alice (A) and Bob (B) share a secret key K_{AB}

Simple Challenge Response (solving the replay problem):

$$\begin{array}{ll} A \rightarrow B: & \text{Hi I am A} \\ B \rightarrow A: & N \qquad\qquad \text{(challenge)} \\ A \rightarrow B: & \{N\}_{K_{AB}} \end{array}$$

- cannot be replayed since next time will be another challenge N
- B authenticates A, but A does not authenticate B (Eve can intercept messages from A, send random challenge and ignore last)

Authentication Protocols

Alice (A) and Bob (B) share a secret key K_{AB}

Mutual Challenge Response:

$$\begin{array}{rl} A \to B: & N_A \\ B \to A: & \{N_A, N_B\}_{K_{AB}} \\ A \to B: & N_B \end{array}$$

Authentication Protocols

Alice (A) and Bob (B) share a secret key K_{AB}

Mutual Challenge Response:

$$\begin{array}{ll} A \rightarrow B: & N_A \\ B \rightarrow A: & \{N_A, N_B\}_{K_{AB}} \\ A \rightarrow B: & N_B \end{array}$$

But requires shared secret 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

$$
\begin{array}{ll}
A \rightarrow B: & N_A \\
B \rightarrow A: & \{N_A, N_B\}_{K_{AB}} \\
A \rightarrow B: & N_B
\end{array}
$$

The attack (let A decrypt her own messages):

$$
\begin{array}{ll}
A \rightarrow E: & N_A \\
E \rightarrow A: & N_A \\
A \rightarrow E: & \{N_A, N'_A\}_{K_{AB}} \\
E \rightarrow A: & \{N_A, N'_A\}_{K_{AB}} \\
A \rightarrow E: & N'_A \;\; (= N_B)
\end{array}
$$

$$
\begin{aligned}
A \rightarrow B&: \quad N_A \\
B \rightarrow A&: \quad \{N_A, N_B\}_{K_{AB}} \\
A \rightarrow B&: \quad N_B
\end{aligned}
$$

The attack (let A decrypt her own messages):

$$
\begin{aligned}
A \rightarrow E&: \quad N_A \\
E \rightarrow A&: \quad N_A \\
A \rightarrow E&: \quad \{N_A, N'_A\}_{K_{AB}} \\
E \rightarrow A&: \quad \{N_A, N'_A\}_{K_{AB}} \\
A \rightarrow E&: \quad N'_A \;\; (= N_B)
\end{aligned}
$$

Solutions: $K_{AB} \neq K_{BA}$ or include an id in the second message

Encryption to the Rescue?

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

Encryption to the Rescue?

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

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

Trusted Third Party

Simple protocol for establishing a secure connection via a mutually trusted 3rd party (server):

$$
\begin{array}{l}
A \rightarrow S : A, B \\
S \rightarrow A : \{K_{AB}\}_{K_{AS}} \text{ and } \{\{K_{AB}\}_{K_{BS}}\}_{K_{AS}} \\
A \rightarrow B : \{K_{AB}\}_{K_{BS}} \\
A \rightarrow B : \{m\}_{K_{AB}}
\end{array}
$$

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.)

Person-in-the-Middle

"Normal" protocol run:

- *A* sends public key to *B*
- *B* sends public key to *A*
- *A* sends a message encrypted with *B*'s public key, *B* decrypts it with its private key
- *B* sends a message encrypted with *A*'s public key, *A* decrypts it with its private key

Person-in-the-Middle

Attack:

- A sends public key to B — C intercepts this message and send his own public key to B
- B sends public key to A — C intercepts this message and send his own public key A
- A sends a message encrypted with C's public key, C decrypts it with its private key, re-encrypts with B's public key
- similar the other way

Person-in-the-Middle

Prevention:

- A sends public key to B
- B sends public key to A
- A encrypts a message with B's public key, sends **half** of the message to B
- B encrypts a message with A's public key, sends **half** of the message back to A
- A sends other half, B can now decrypt entire message
- B sends other half, A can now decrypt entire message

Person-in-the-Middle

Prevention:

- *A* sends public key to *B*
- *B* sends public key to *A*
- *A* encrypts a message with *B*’s public key, sends **half** of the message to *B*
- *B* encrypts a message with *A*’s public key, sends **half** of the message back to *A*
- *A* sends other half, *B* can now decrypt entire message
- *B* sends other half, *A* can now decrypt entire message

C would have to invent a totally new message

Car Transponder (HiTag2)

1. C generates a random number N
2. C calculates $(F, G) = \{N\}_K$
3. $C \rightarrow T$: N, F
4. T calculates $(F', G') = \{N\}_K$
5. T checks that $F = F'$
6. $T \rightarrow C$: N, G'
7. C checks that $G = G'$

Car Transponder (HiTag2)

1. C generates a random number N
2. C calculates $(F, G) = \{N\}_K$
3. $C \to T$: N, F
4. T calculates $(F', G') = \{N\}_K$
5. T checks that $F = F'$
6. $T \to C$: N, G'
7. C checks that $G = G'$

This process means that the transponder believes the car knows the key K, and the car believes the transponder knows the key K. They have authenticated themselves to each other.

Person-in-the-Middle

- Border Gateway Protocol (BGP) — routers believe their neighbours
- it is possible to advertise bad routes
- can be done over continents

http://www.renesys.com/2013/11/mitm-internet-hijacking/

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)

Best Practices

Principle 1: Every message should say what it means: the interpretation of a message should not depend on the context.

Best Practices

Principle 1: Every message should say what it means: the interpretation of a message should not depend on the context.

Principle 2: If the identity of a principal is essential to the meaning of a message, it is prudent to mention the principal's name explicitly in the message (though difficult).

Best Practices

Principle 3: Be clear about why encryption is being done. Encryption is not wholly cheap, and not asking precisely why it is being done can lead to redundancy. Encryption is not synonymous with security.

Possible Uses of Encryption

- Preservation of confidentiality: $\{X\}_K$ only those that have K may recover X.
- Guarantee authenticity: The partner is indeed some particular principal.
- Guarantee confidentiality and authenticity: binds two parts of a message — $\{X, Y\}_K$ is not the same as $\{X\}_K$ and $\{Y\}_K$.

Best Practices

Principle 4: The protocol designers should know which trust relations their protocol depends on, and why the dependence is necessary. The reasons for particular trust relations being acceptable should be explicit though they will be founded on judgment and policy rather than on logic.

Example Certification Authorities: CAs are trusted to certify a key only after proper steps have been taken to identify the principal that owns it.

Formal Methods

Ross Anderson about the use of Logic:

> *Formal methods can be an excellent way of finding bugs in security protocol designs as they force the designer to make everything explicit and thus confront difficult design choices that might otherwise be fudged.*

Mid-Term

- homework, handouts, programs...

Any Questions?