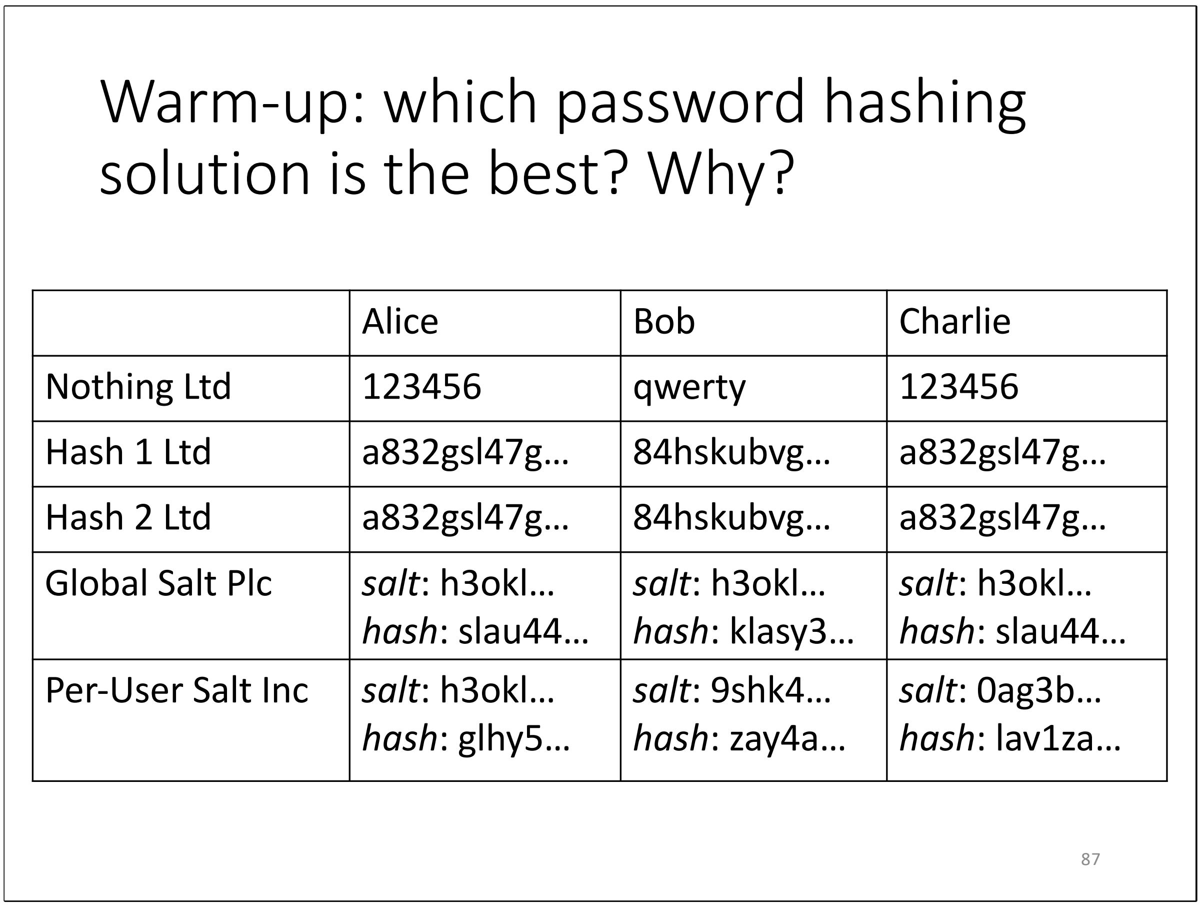
Software and Security Engineering

Lecture 4

Martin Kleppmann

martin.kleppmann@cst.cam.ac.uk

With many thanks to Ross Anderson and Alastair Beresford

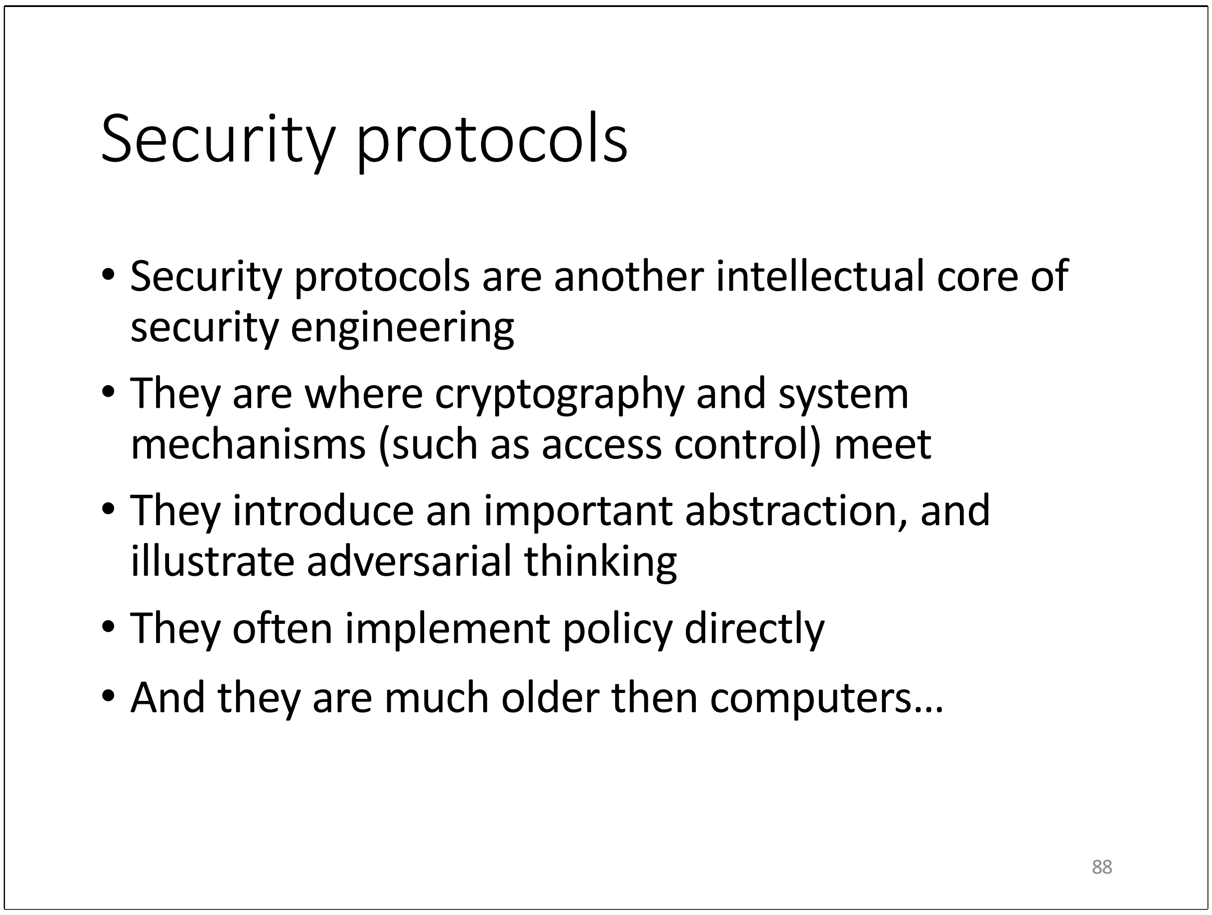


Warm-up: which password hashing solution is the best? Why?

	Alice	Bob	Charlie
Nothing Ltd	123456	qwerty	123456
Hash 1 Ltd	a832gsl47g…	84hskubvg…	a832gsl47g…
Hash 2 Ltd	a832gsl47g…	84hskubvg…	a832gsl47g…
Global Salt Plc	*salt*: h3okl… *hash*: slau44…	*salt*: h3okl… *hash*: klasy3…	*salt*: h3okl… *hash*: slau44…
Per-User Salt Inc	*salt*: h3okl… *hash*: glhy5…	*salt*: 9shk4… *hash*: zay4a…	*salt*: 0ag3b… *hash*: lav1za…

87

Alice, Bob and Charlie use the same username and password combination with five companies. Nothing Ltd uses no hashing and stores the passwords in plain text. Hash 1 and Hash 2 ltd apply a hash function to the password. Global Salt Plc applies a global salt. Per-User Salt Inc applies a per-user salt to each password entry. Which one has the best password management strategy? What are the externalities?



Security protocols

- Security protocols are another intellectual core of security engineering
- They are where cryptography and system mechanisms (such as access control) meet
- They introduce an important abstraction, and illustrate adversarial thinking
- They often implement policy directly
- And they are much older then computers…

88

Adversarial thinking is really important: there are lots of weaknesses which you can exploit which don't require pulling fingernails of a customer to get their bank account PIN. The earlier example with Matt Hanon demonstrates the failure of security protocols neatly; stealing a domain name at gun point demonstrates that (metaphorically) pulling fingernails also works.

Ordering wine in a restaurant

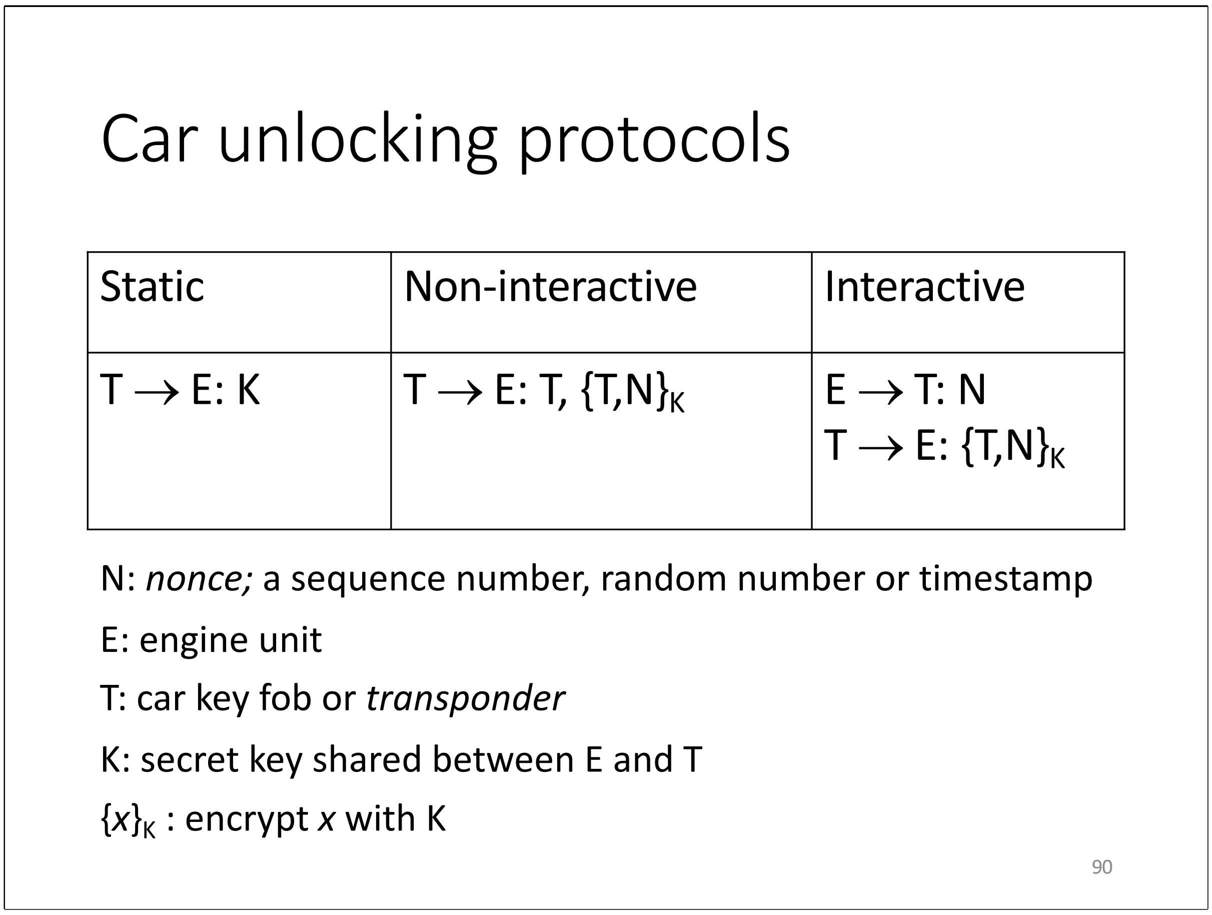
1. Sommelier presents wine list to host
2. Host chooses wine; sommelier fetches it
3. Host samples wine; then it's served to guests

Security properties?

89

[Ask the audience]

Example properties include:
- Confidentiality – of price from guests
- Integrity – can't substitute a cheaper wine
- Non-repudiation – host can't falsely complain



Car unlocking protocols

Static	Non-interactive	Interactive
$T \rightarrow E: K$	$T \rightarrow E: T, \{T,N\}_K$	$E \rightarrow T: N$ $T \rightarrow E: \{T,N\}_K$

N: *nonce;* a sequence number, random number or timestamp

E: engine unit

T: car key fob or *transponder*

K: secret key shared between E and T

$\{x\}_K$: encrypt x with K

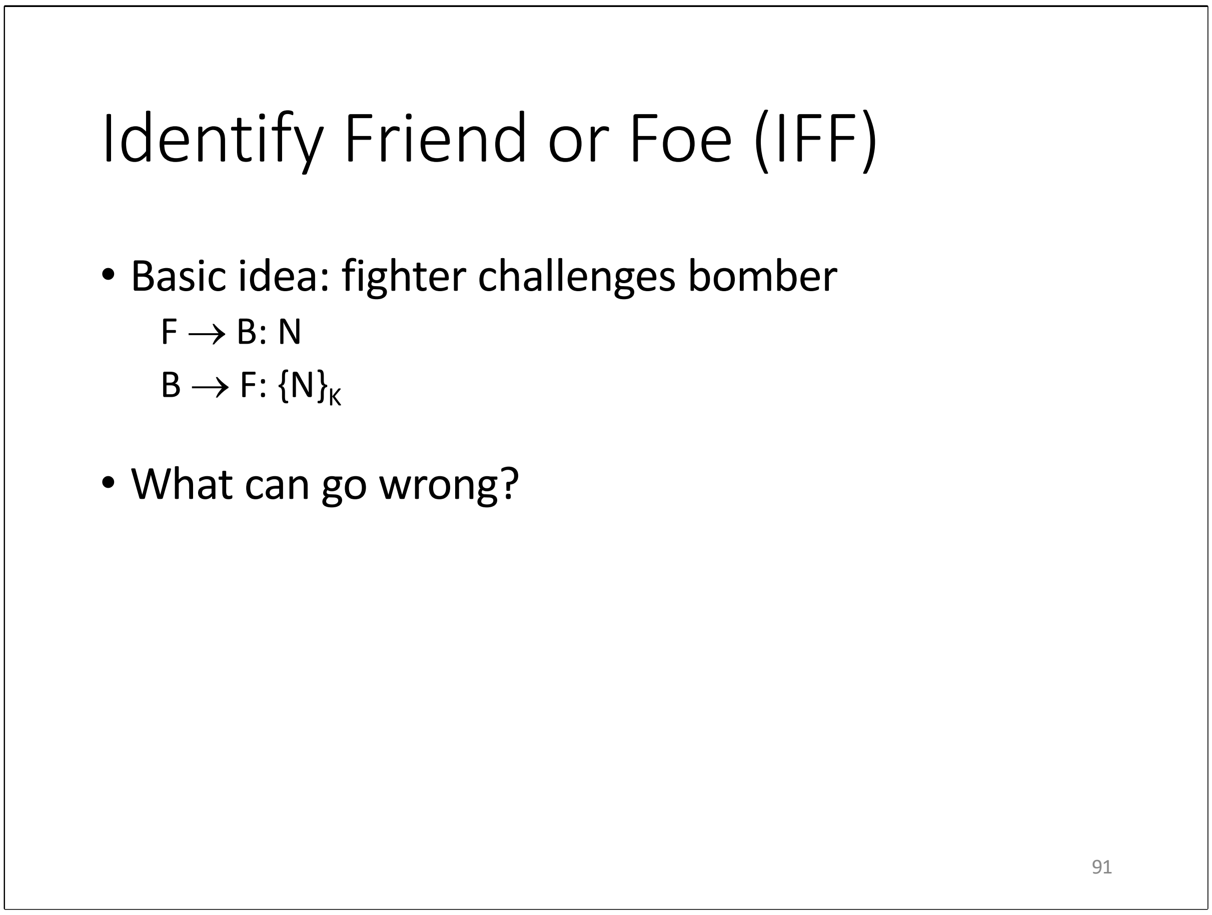
90

[Introduce the notation; explaining what is on the slide carefully.]

Static suffers from a replay attack: record the transmission of K and replay to unlock. Additionally, some systems are susceptible to brute-force attacks: some garage door openers still use the static approach with a 16-bit key, so fly a plane over Cambridge spitting out all the combinations in quick succession and watch all those garage doors go up.

The nonce is critical to the success of the other two protocols. A sequence number, a random number or a timestamp are all possible, but they need to be implemented carefully. Random requires us to keep a list of previous numbers to prevent replay attacks; sequence can go out of sync (e.g. dog presses transponder lots of times when out of range) so could look for sequences of two presses, one number apart, which suggests the user is next to the car; timestamp is okay, but problematic if clocks go out of sync or if there are time zone issues.

One problem with interactive is the relay attack. A claim, concerning keyless car keys: Audi's new key contains a motion sensor that shuts off its signal "when the key is laid down and not moving". A similar Porsche device sleeps after 30 seconds and all new Mercedes keys shut down after two minutes.
https://twitter.com/kentindell/status/1117341970068910080?s=09)



Identify Friend or Foe (IFF)

- Basic idea: fighter challenges bomber

  $F \rightarrow B: N$

  $B \rightarrow F: \{N\}_K$

- What can go wrong?

91

[Ask the audience]

Person-in-the-middle attack

(Also known as “man-in-the-middle” or MITM)

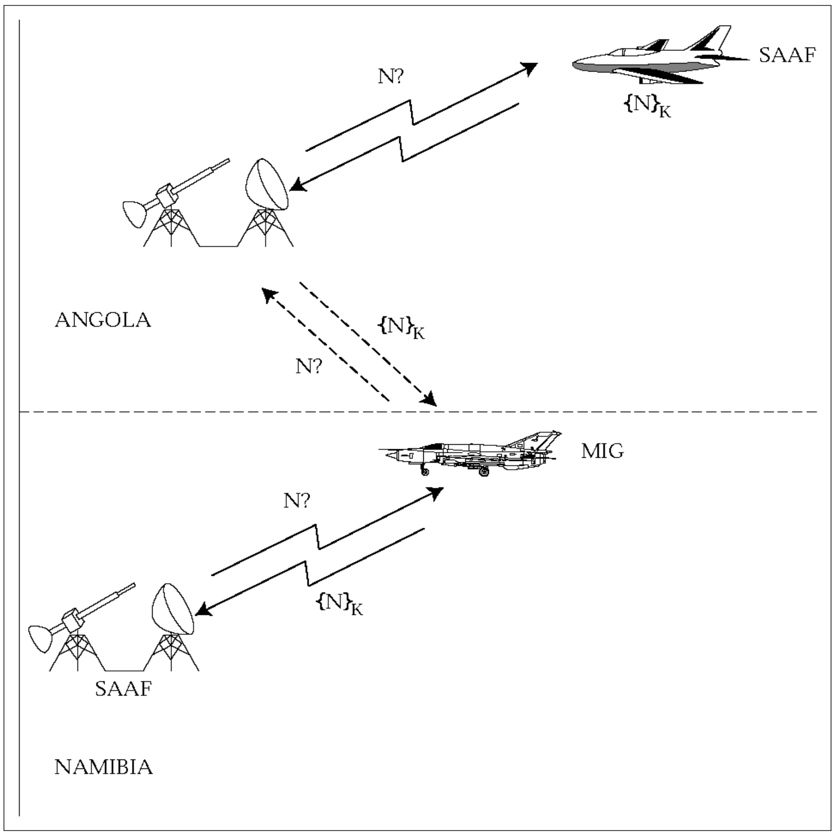
- Basic idea: fighter (F) challenges bomber (B)

  $F \rightarrow B$: N
  $B \rightarrow F$: $\{N\}_K$

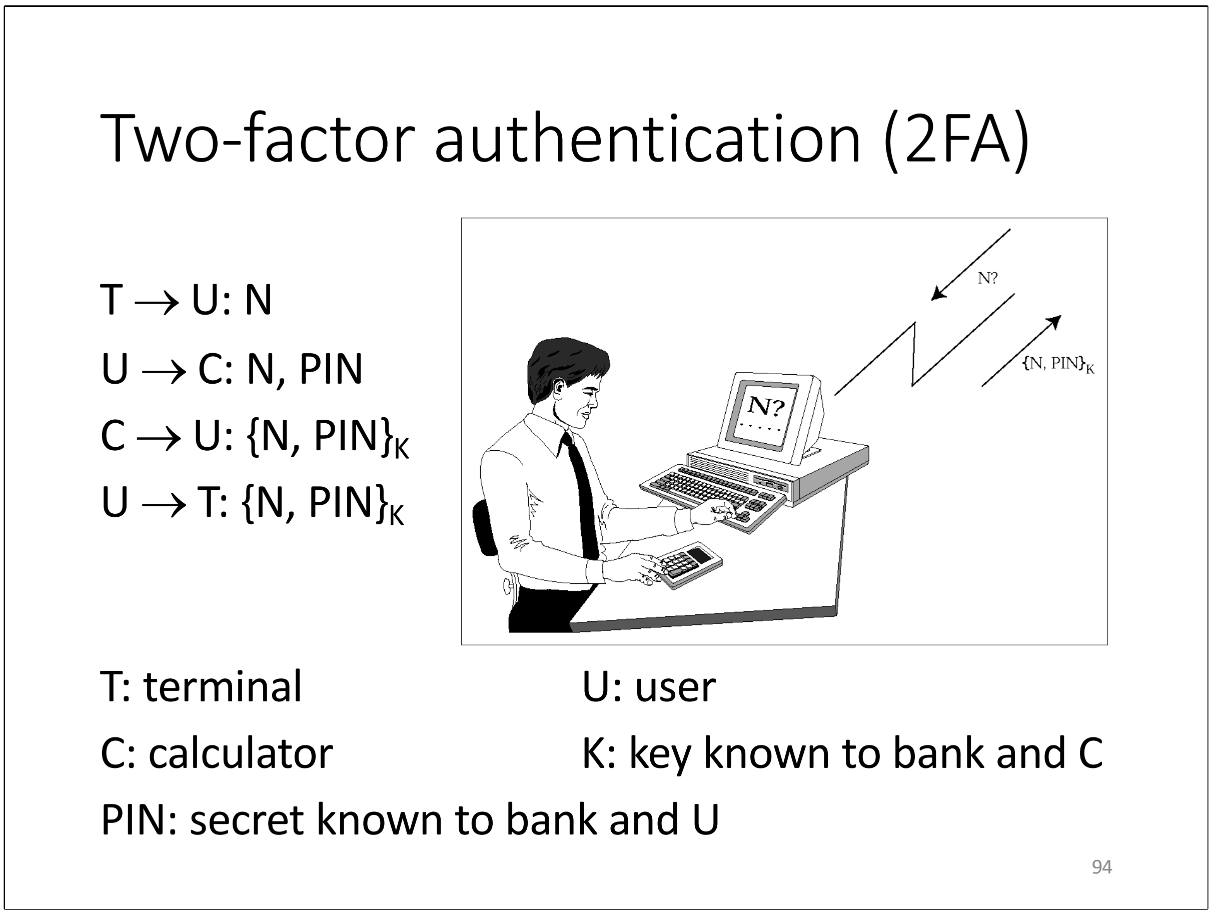
- What if the bomber reflects the challenge back at the fighter’s wingman (W)?

  $F \rightarrow B$: N
  $B \rightarrow W$: N
  $W \rightarrow B$: $\{N\}_K$
  $B \rightarrow F$: $\{N\}_K$

92



This was used against the South African Air Force in the late 1990s, when South Africa were bombing the capital of Angola. Cuba (who were helping Angola) sent in the MIG which relayed IFF to enable access to South African airspace and led to the bombing of an airport in South Africa. More detail in the course text book: Ross Anderson, Security Engineering.



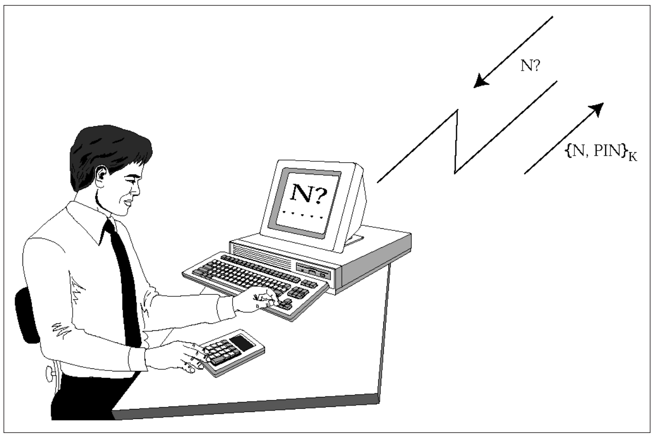
Two-factor authentication (2FA)

$T \rightarrow U$: N

$U \rightarrow C$: N, PIN

$C \rightarrow U$: $\{N, PIN\}_K$

$U \rightarrow T$: $\{N, PIN\}_K$



T: terminal	U: user
C: calculator	K: key known to bank and C

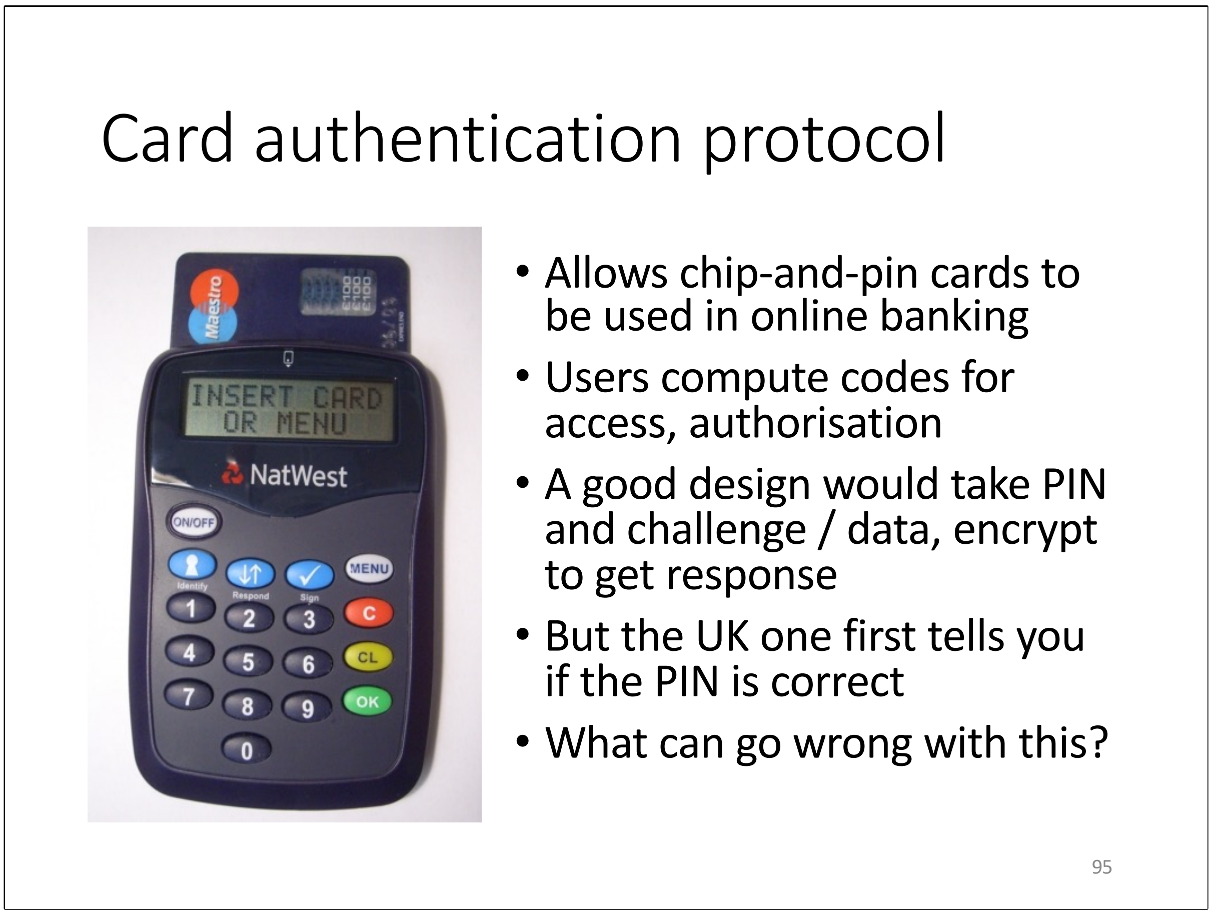
PIN: secret known to bank and U

94

Example from the 1990s. This system provided two-factor authentication where you typed in a challenge from the terminal into the calculator together with the PIN. The calculator then encrypted N and PIN under key K.

[Ask audience how they would hack it]

Hacks: MITM attack; steal the calculator; social engineering -- call up admins and convince them to give you a code; take over a session that is in progress -- the data in the 1990s was not encrypted; infect the terminal with malware; and so on. Nowadays, with SMS-based 2FA, SIM swapping attacks are used. Nevertheless, this is still much better than just passwords -- attacks don't scale well since you can't just hack into a server and steal all the passwords.

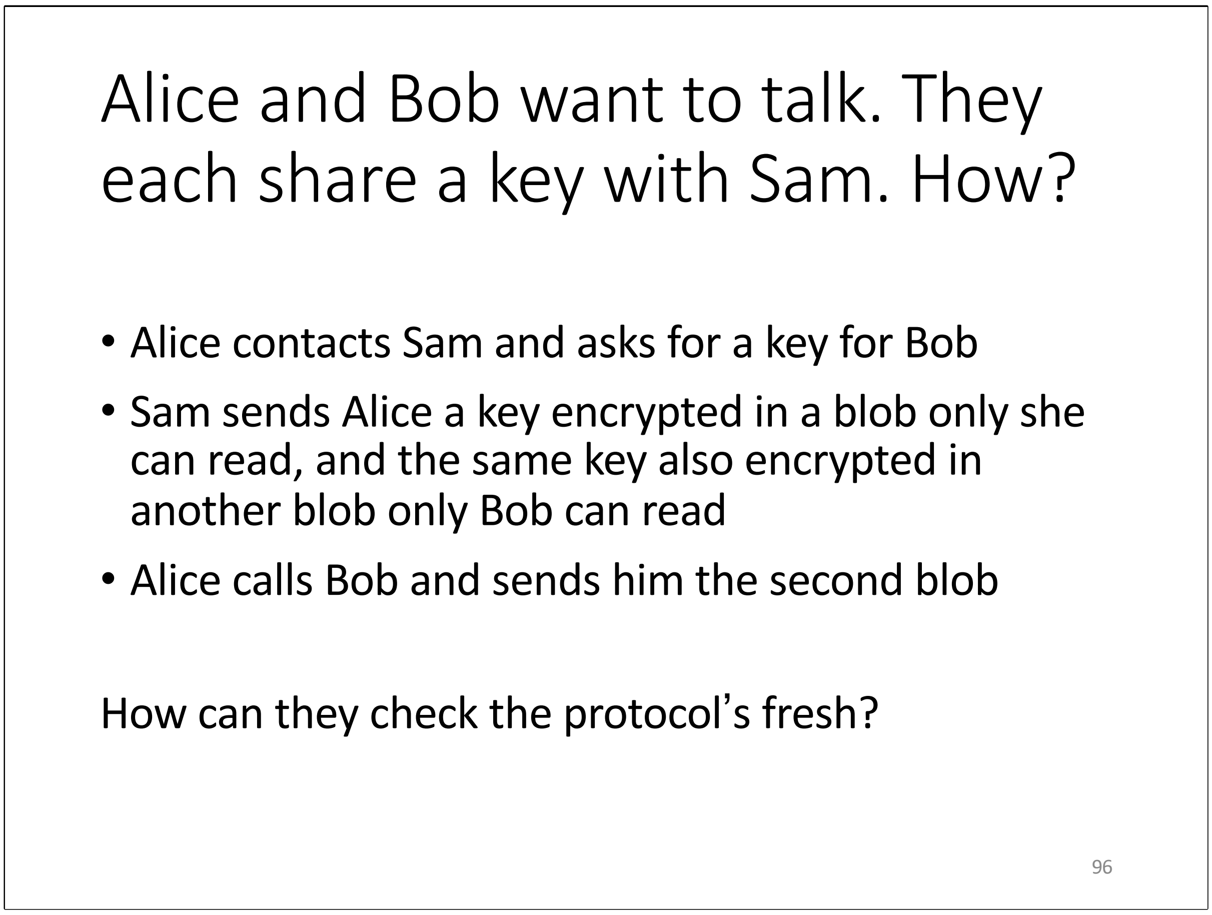


Card authentication protocol

- Allows chip-and-pin cards to be used in online banking
- Users compute codes for access, authorisation
- A good design would take PIN and challenge / data, encrypt to get response
- But the UK one first tells you if the PIN is correct
- What can go wrong with this?

95

This is a modern version of the system shown on the previous slide. Note the difference from last one – this new machine tells you whether you have got the PIN right or not. The previous version one would just spit out a random (incorrect) challenge. This appears to be superior in terms of usability until you realise that its popular with muggers. Previously a mugger would have to drag a victim to the cash machine – a risky endeavour; now a criminal can now force people at knife point to reveal and check the PIN wherever the mugging takes place.



Alice and Bob want to talk. They each share a key with Sam. How?

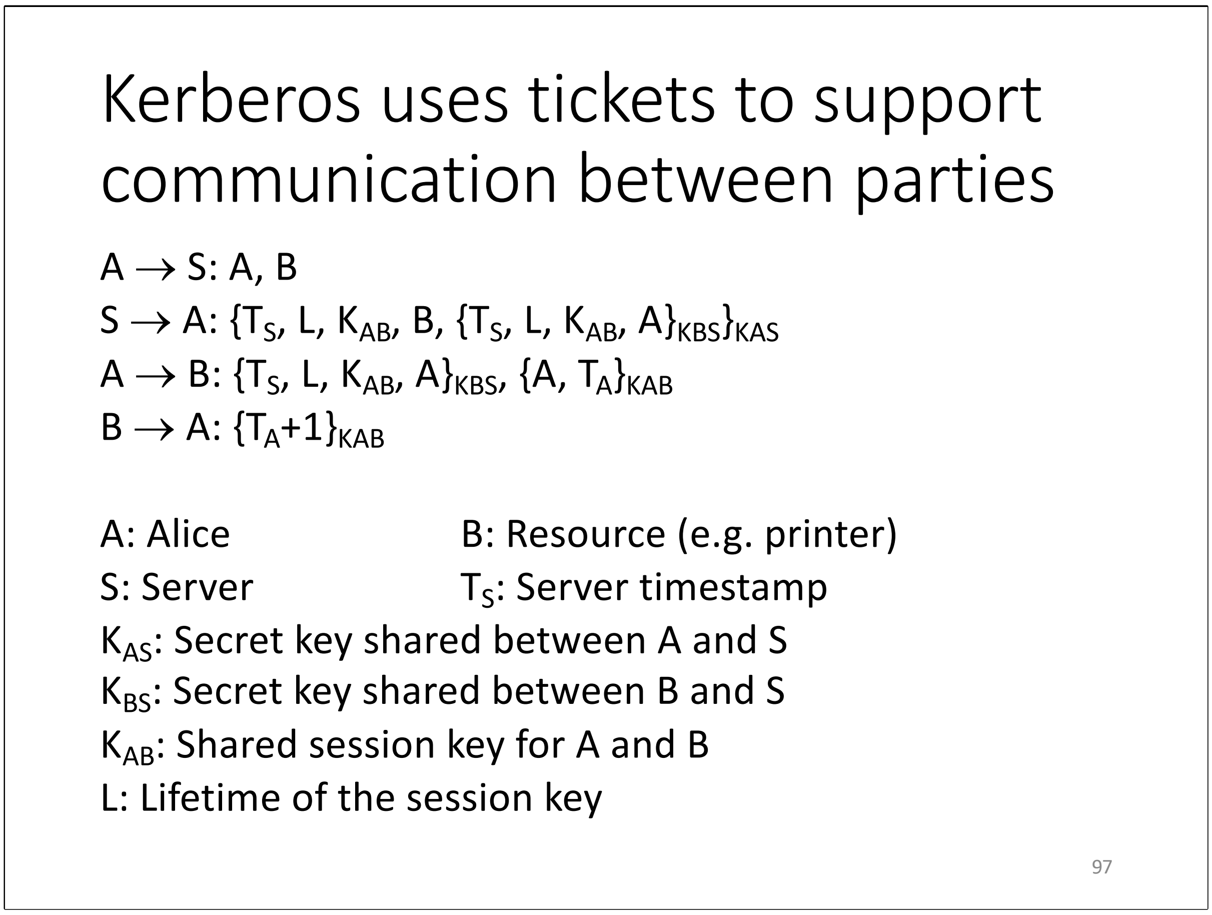
- Alice contacts Sam and asks for a key for Bob
- Sam sends Alice a key encrypted in a blob only she can read, and the same key also encrypted in another blob only Bob can read
- Alice calls Bob and sends him the second blob

How can they check the protocol's fresh?

96

This originated from the 1970s where we suddenly had network computers (e.g. at Xerox Parc). Then we want Bob, Alice, and so on to be able to communicate. Also true for other components in the system, including the printer, mail server, and so on. Having every computer or device keep a full list of all keys of everything else is going to be painful. Solution: centralise key management, but then the question is how to avoid all communications going through the central server.

In this example, we might have:
Alice = user's laptop
Bob = network printer
Sam = authentication server



Kerberos uses tickets to support communication between parties

$A \rightarrow S$: A, B
$S \rightarrow A$: $\{T_S, L, K_{AB}, B, \{T_S, L, K_{AB}, A\}_{KBS}\}_{KAS}$
$A \rightarrow B$: $\{T_S, L, K_{AB}, A\}_{KBS}, \{A, T_A\}_{KAB}$
$B \rightarrow A$: $\{T_A+1\}_{KAB}$

A: Alice	B: Resource (e.g. printer)
S: Server	T_S: Server timestamp

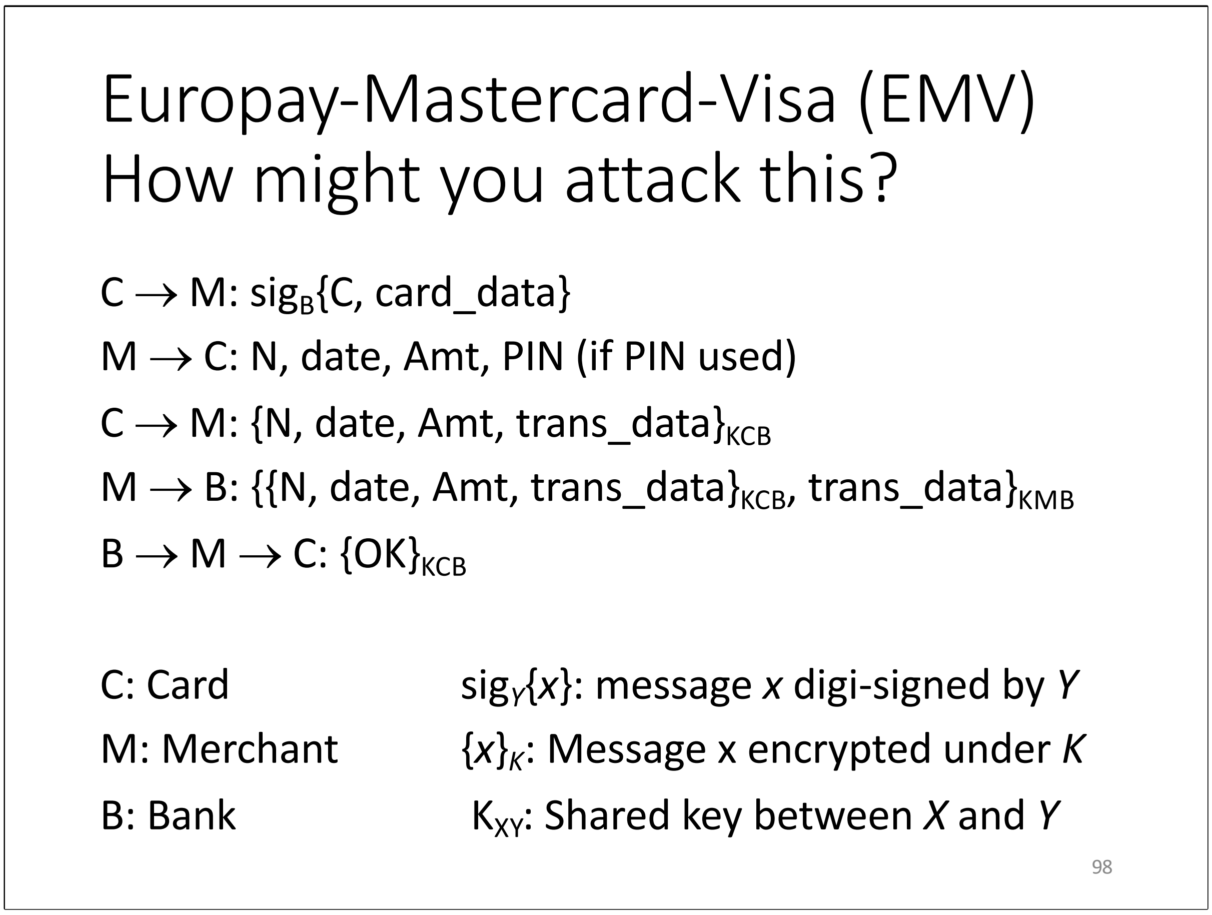
K_{AS}: Secret key shared between A and S
K_{BS}: Secret key shared between B and S
K_{AB}: Shared session key for A and B
L: Lifetime of the session key

97

We use this for access control in the Computer Laboratory. When I want to access the fileserver, I need to type in kinit before I can access my home directory. This is good for remote access: first connect to slogin.cl.cam.ac.uk, where you need an SSH key to get in (something you have) and then a password (something you know) to actually access the fileserver.

[Talk through the protocol in detail.]

There is still some trust here. For example, Alice trusts that Sam sends the right timestamps. This protocol allows things to scale: you can have different ticket granting machines (S) for different departments, and so on. There are a whole series of protocols built on top of this for distributed systems. For now, you just need to know about this protocol as an example. Later lecture courses will cover these type of things better, and also how to prove correctness and so on.



Europay-Mastercard-Visa (EMV) How might you attack this?

$C \rightarrow M$: $\text{sig}_B\{C, \text{card\_data}\}$

$M \rightarrow C$: N, date, Amt, PIN (if PIN used)

$C \rightarrow M$: $\{N, \text{date}, \text{Amt}, \text{trans\_data}\}_{KCB}$

$M \rightarrow B$: $\{\{N, \text{date}, \text{Amt}, \text{trans\_data}\}_{KCB}, \text{trans\_data}\}_{KMB}$

$B \rightarrow M \rightarrow C$: $\{OK\}_{KCB}$

C: Card	$\text{sig}_Y\{x\}$: message *x* digi-signed by *Y*
M: Merchant	$\{x\}_K$: Message x encrypted under *K*
B: Bank	K_{XY}: Shared key between *X* and *Y*

98

[Describe protocol. Ask the audience for ideas on how to attack.]

There are lots of attacks which involve replay and pre-plays which we will get to. There were a lot of attacks years ago which involve a wiretap to collect account number from a merchant device, then video PIN being typed in; then you can make a mag stripe clone of the card. Less good now as mag stripe fall back does not work in many countries.

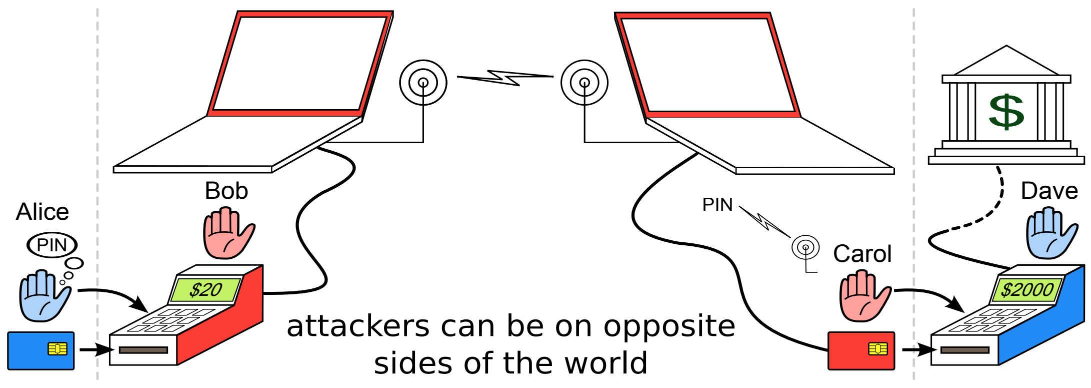
Replace insides of the terminal with your own electronics

- Capture card details and PINs from victims
- Use to perform person-in-the-middle attack in real time on a remote terminal in a merchant selling expensive goods

99

Some students in our lab bought a second-hand chip-and-pin terminal from ebay, and replaced its electronics. Here’s a picture of the terminal playing Tetris, which is a way of explaining to journalists that just because a card terminal looks real, it’s not necessarily trustworthy.

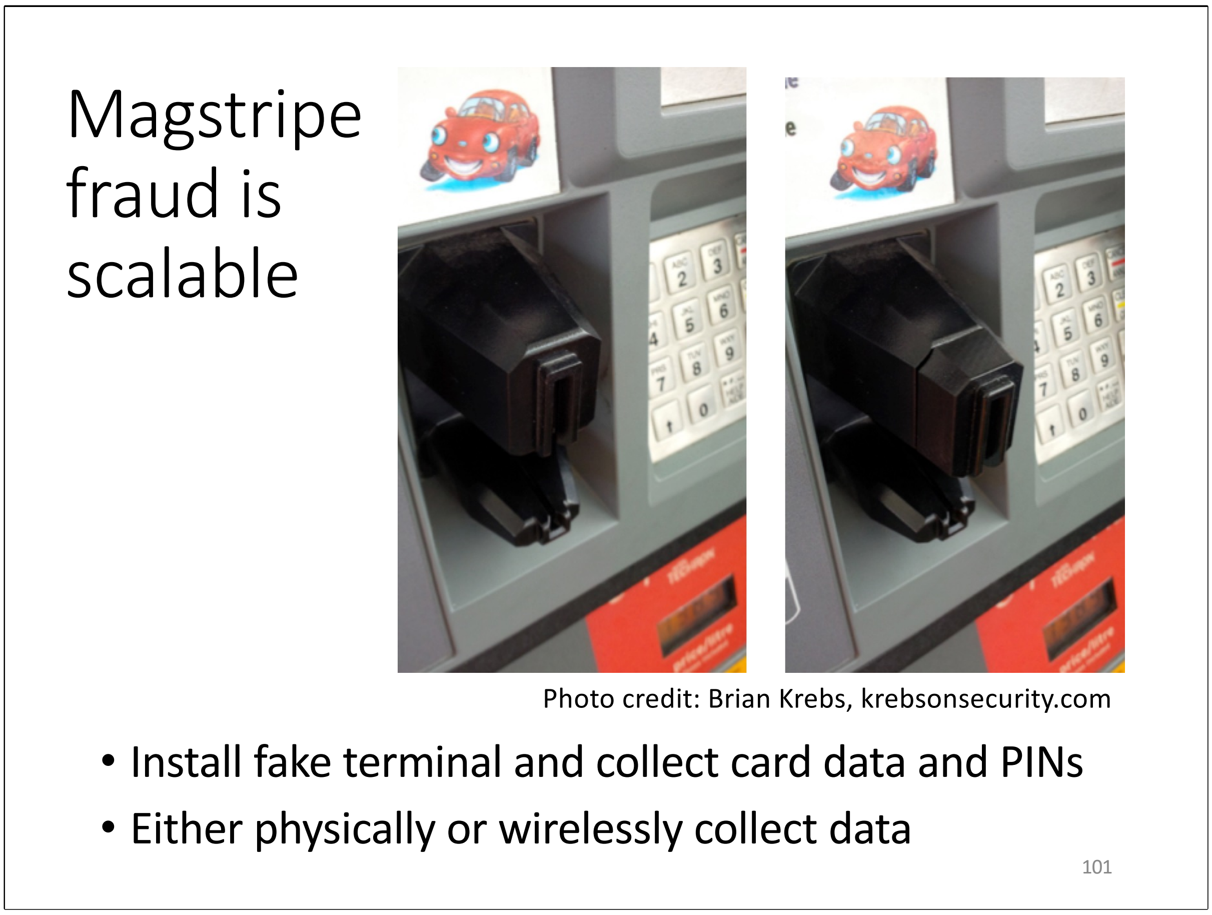
The relay attack: unstoppable but unrealistic – too hard to scale



100

Bob is a crooked coffee shop owner with a hacked chip-and-pin terminal, and when the victim Alice wants to pay for her lunch, Bob's accomplice Carol walks into another shop and buys an expensive item using a hacked card from shop owner Dave. Carol's card relays the payment request from Dave's terminal to Bob's terminal (perhaps far away), and the response from Alice's card is sent back to Carol. Alice thinks she's buying her lunch from Bob's cafe, when in fact she's buying something expensive from Dave's shop.

This attack is almost unstoppable. Steven Murdoch demonstrated this attack 10 years ago: a journalist thought they were buying a coffee, but actually bought an expensive book in another shop. This attack has not been used in real life. It just doesn't scale. The important engineering point here is that flaws need to have scale -- without that they won't be useable.



Magstripe fraud is scalable

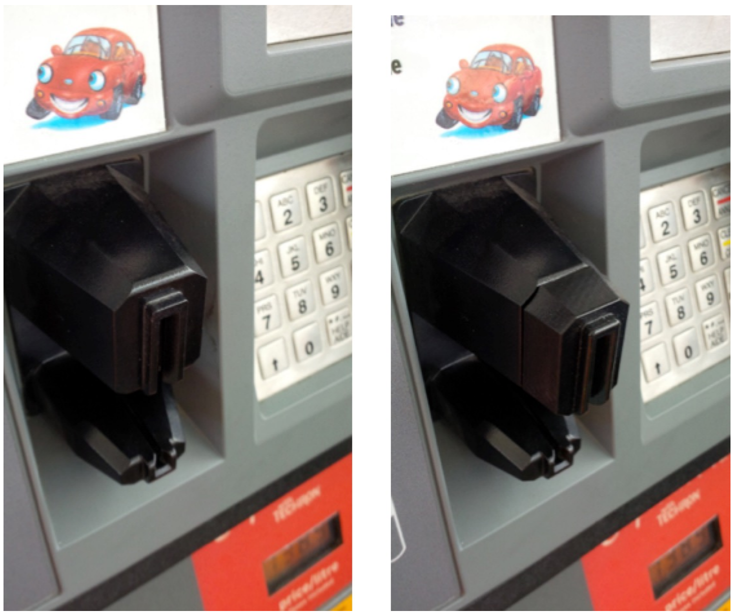


Photo credit: Brian Krebs, krebsonsecurity.com

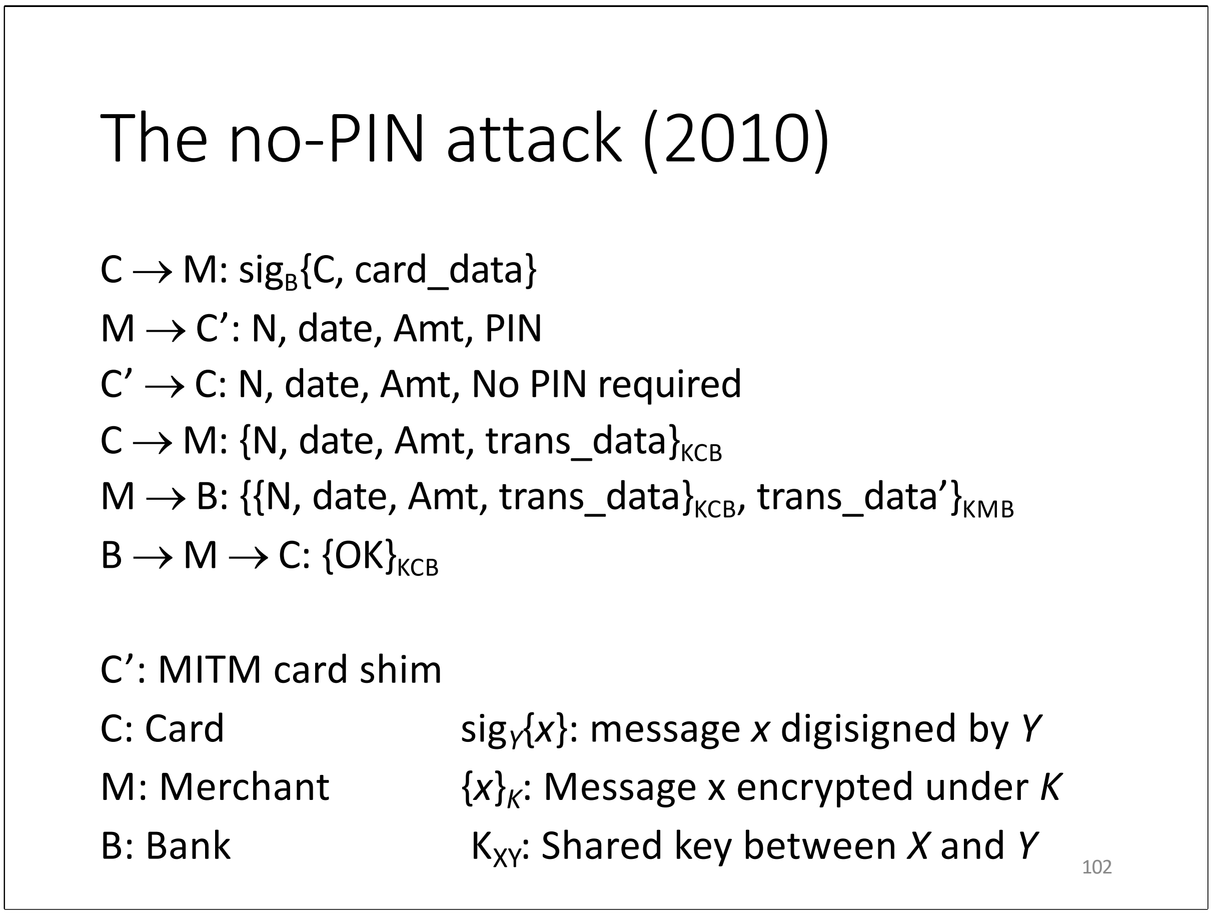
- Install fake terminal and collect card data and PINs
- Either physically or wirelessly collect data

101

Terminals (PIN entry devices) at Shell garages were doctored by malicious service engineers. Terminal supplier went bust.

Customers at BP garage in Girton in 2008 found their cards cloned and used in Thailand.

These remain big in the US, particularly when you pay at the pump. Further info on petrol pump skimmers: https://krebsonsecurity.com/tag/gas-pump-skimmers/



The no-PIN attack (2010)

C $\rightarrow$ M: sig_B\{C, card_data\}

M $\rightarrow$ C’: N, date, Amt, PIN

C’ $\rightarrow$ C: N, date, Amt, No PIN required

C $\rightarrow$ M: $\{\text{N, date, Amt, trans\_data}\}_{KCB}$

M $\rightarrow$ B: $\{\{\text{N, date, Amt, trans\_data}\}_{KCB}, \text{trans\_data'}\}_{KMB}$

B $\rightarrow$ M $\rightarrow$ C: $\{\text{OK}\}_{KCB}$

C’: MITM card shim

C: Card — $\mathrm{sig}_Y\{x\}$: message *x* digisigned by *Y*

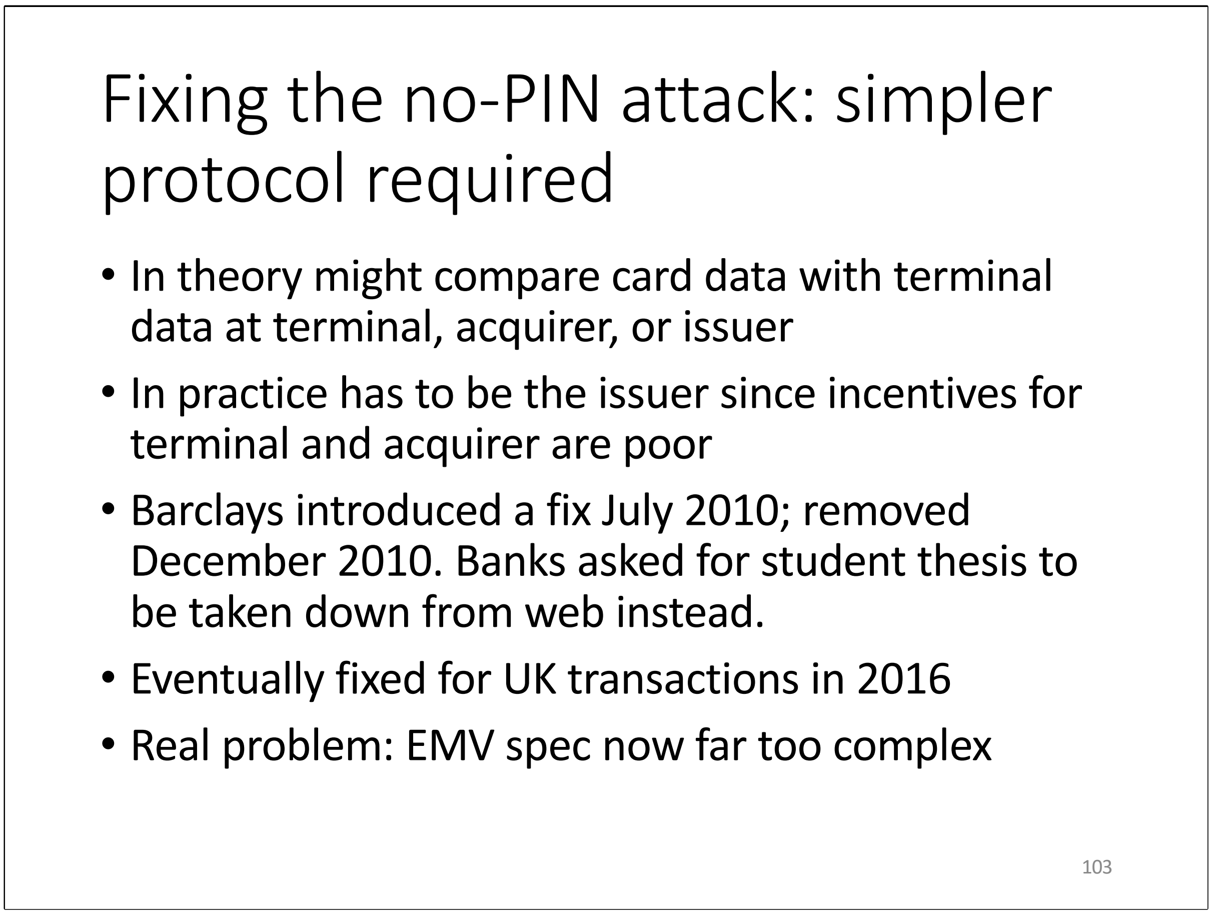
M: Merchant — $\{x\}_K$: Message x encrypted under *K*

B: Bank — K_{XY}: Shared key between *X* and *Y*

102

Apply a MITM attack to the protocol, convincing the card that it has performed a chip and signature transaction, and the terminal that it has performed a chip and PIN transaction. This allows you use a card where you don’t have the PIN.

You can now use a SIM shim (140 microns thick!) to MITM the protocol and implement the attack described.



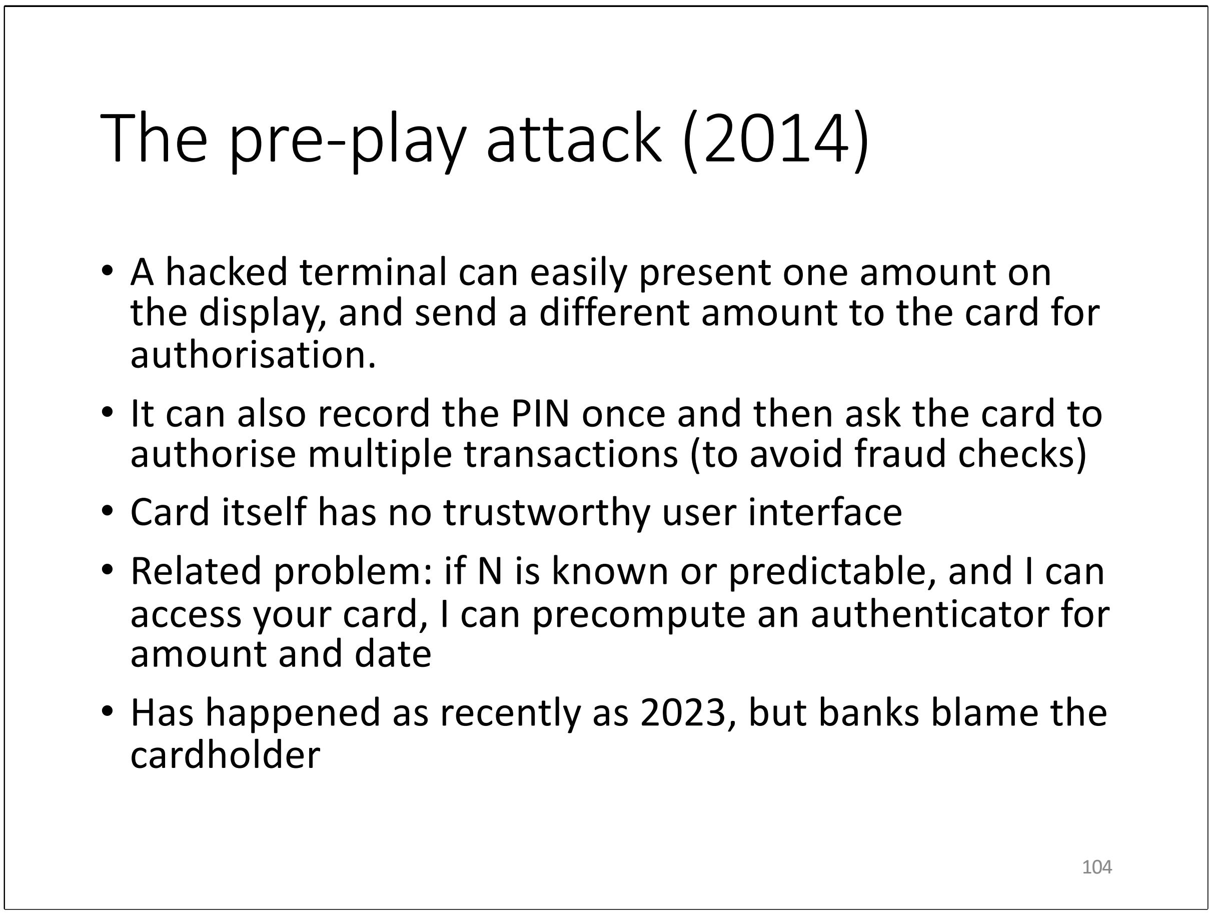
Fixing the no-PIN attack: simpler protocol required

- In theory might compare card data with terminal data at terminal, acquirer, or issuer
- In practice has to be the issuer since incentives for terminal and acquirer are poor
- Barclays introduced a fix July 2010; removed December 2010. Banks asked for student thesis to be taken down from web instead.
- Eventually fixed for UK transactions in 2016
- Real problem: EMV spec now far too complex

103

Barclays likely removed fix in December 2010 due to too many false positives. It took the banks four years to block this. Some countries still don't.

The EMV spec is 4000 pages thick. This is a real problem as there are lots of interactions between different features. This is good for the bad guys: they can exploit any and all potential feature interactions. It is a disaster for the defender since it represents a huge attack surface which is hard to check.



The pre-play attack (2014)

- A hacked terminal can easily present one amount on the display, and send a different amount to the card for authorisation.
- It can also record the PIN once and then ask the card to authorise multiple transactions (to avoid fraud checks)
- Card itself has no trustworthy user interface
- Related problem: if N is known or predictable, and I can access your card, I can precompute an authenticator for amount and date
- Has happened as recently as 2023, but banks blame the cardholder

104

Ross provided representation for a Scottish sailor who bought a round of drinks for 33 Euros, and later found he had 10 transactions of 3,300 Euros each on his card. These four transactions were made one hour apart and placed through three different acquirer banks.
https://www.lightbluetouchpaper.org/2023/06/19/the-pre-play-attack-in-real-life/
https://www.dailymail.co.uk/news/article-12190547/Falklands-war-hero-scammed-20k-holiday-denied-refund-Barclays-forced-sell-medals.html

The original paper on the pre-play attack:
https://murdoch.is/papers/oakland14chipandskim.pdf

When you think about it, you have in your wallet three or four cards, and each card may have £5000 available on it (e.g. because you can get an overdraft, or you have a large credit limit). This means you're walking around with £20k. Would you walk into a dodgy place with £20k in cash in your pocket? The problem is that people don't think this way -- they think that their PIN offers security and their bank will protect them in the case of failure. But banks have again and again responded to such frauds by claiming that the PIN was used, and therefore it’s the customer’s fault.