Software and Security Engineering

Lecture 7

Martin Kleppmann

mk428@cam.ac.uk

With many thanks to Ross Anderson

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…

187

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?

188

[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

189

[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?

190

[Ask the audience]

Person-in-the-middle attack…

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

  $F \rightarrow B$: N

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

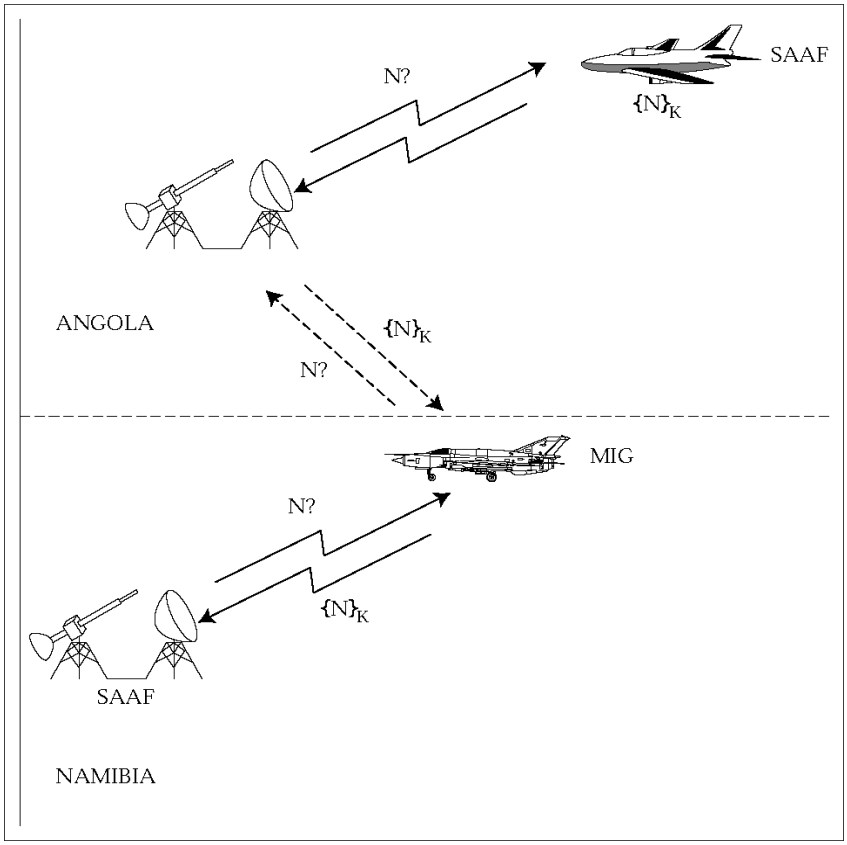
- What if the bomber reflects the challenge back at the fighter's wingman (W)?

  $F \rightarrow B$: N

  $B \rightarrow W$: N

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

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



192

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.

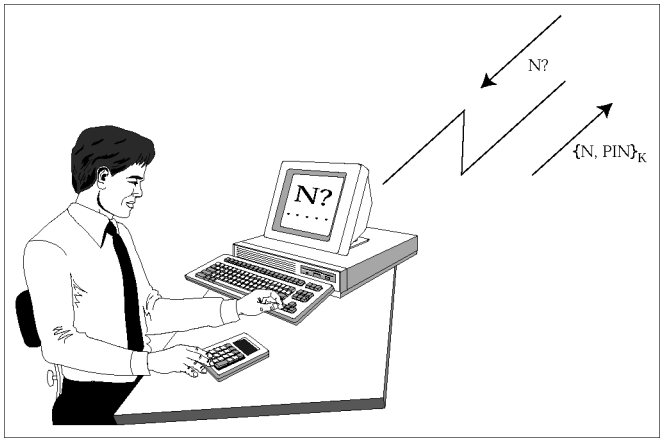
Two-factor authentication (2FA)

$T \rightarrow U$: 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

193

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: steal the calculator; MITM attack; take over a session that is in progress -- the data in the 1990s was not encrypted; infect the terminal with malware; and so on. 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 EMV 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?

194

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?

195

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.

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

196

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?

$C \rightarrow M$: $sig_B\{C, card\_data\}$

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

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

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

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

C: Card	$sig_Y\{x\}$: message *x* digisigned by *Y*
M: Merchant	$\{x\}_K$: Message x encrypted under *K*
B: Bank	K_{XY}: Shared key between *X* and *Y*

197

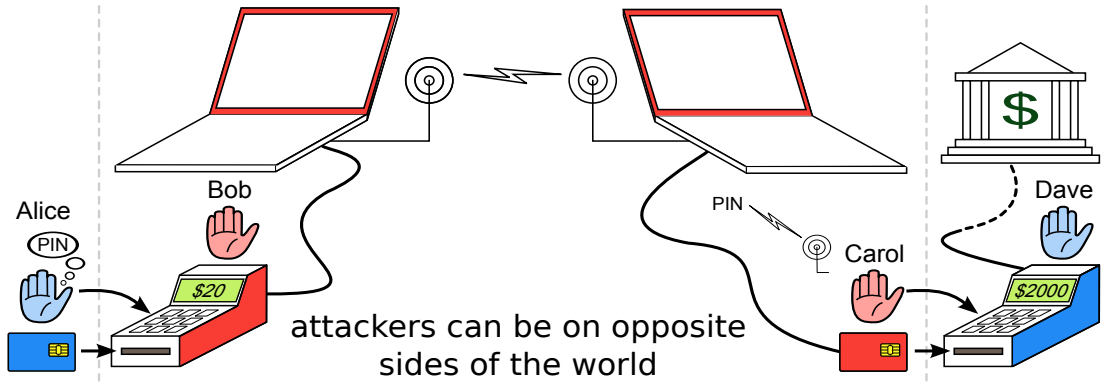
[Describe protocol. Ask the audience for ideas on how to attack.]

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

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



199

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

200

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 $\rightarrow$ Ć: N, date, Amt, PIN

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

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

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

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

Ć: MITM card shim

C: Card — $sig_Y\{x\}$: message *x* digisigned by *Y*

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

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

201

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

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.

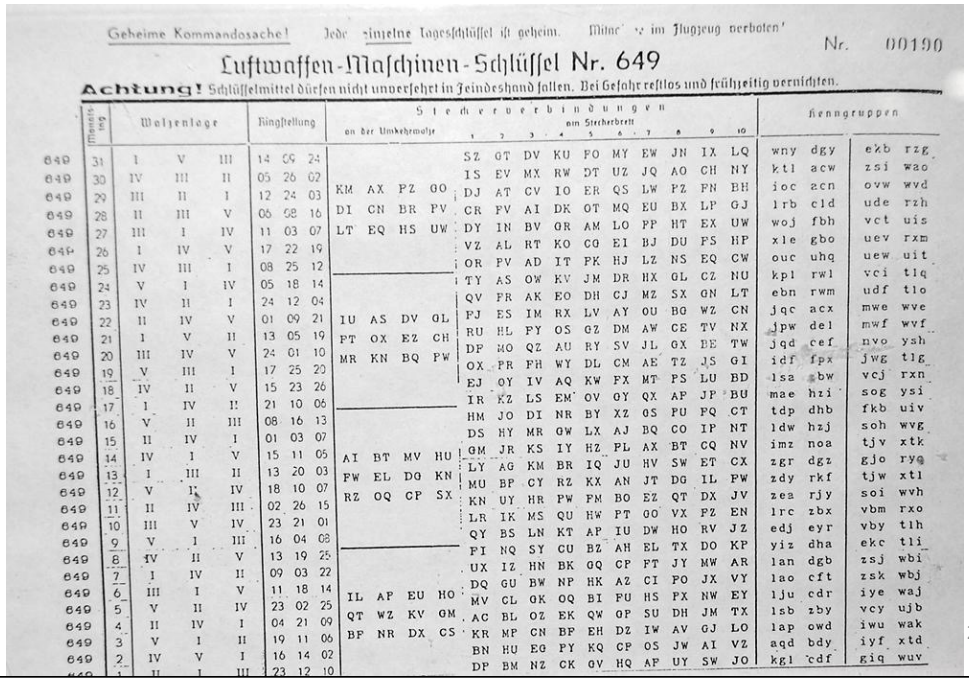
The preplay attack (2014)

- In EMV, the terminal sends a random number N to the card along with the date d and the amount Amt
- The card authenticates N, d and Amt using the key it shares with the bank, K_{CB}
- What happens if I can predict N for date d?
- Answer: if I have access to your card I can precompute an authenticator for Amt and d

203

Ross provided representation for a Scottish sailor who bought a round of drinks for 33 Euros, and later found he had 4 transactions of 3300 Euros on his card. These four transactions were made one hour apart and placed through three different acquirer banks. 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.

Symmetric key cryptography requires careful sharing of keys



Geheime Kommandosache! Jede einzelne Tagesschlüssel ist geheim. Mitnahme im Flugzeug verboten!

Nr. 00190

Luftwaffen-Maschinen-Schlüssel Nr. 649

Achtung! Schlüsselmittel dürfen nicht unversehrt in Feindeshand fallen. Bei Gefahr restlos und frühzeitig vernichten.

Monats tag	Walzenlage			Ringstellung			Steckerverbindungen an der Umkehrwalze				Steckerverbindungen am Steckerbrett										Kenngruppen			
649	31	I	V	III	14	09	24					SZ	GT	DV	KU	FO	MY	EW	JN	IX	LQ	wny	dgy	ekb	rzg
649	30	IV	III	II	05	26	02					IS	EV	MX	RW	DT	UZ	JQ	AO	CH	NY	ktl	acw	zsi	wao
649	29	III	II	I	12	24	03	KM	AX	PZ	GO	DJ	AT	CV	IO	ER	QS	LW	PZ	FN	BH	ioc	zcn	ovw	wvd
649	28	II	III	V	06	08	16	DI	CN	BR	FV	CR	FV	AI	DK	OT	MQ	EU	BX	LP	GJ	lrb	cld	ude	rzh
649	27	III	I	IV	11	03	07	LT	EQ	HS	UW	DY	IN	BV	OR	AM	LO	PP	HT	EX	UW	woj	fbh	vct	uis
649	26	I	IV	V	17	22	19					VZ	AL	RT	KO	CG	EI	BJ	DU	FS	HP	xle	gbo	uev	rxm
649	25	IV	III	I	08	25	12					OR	FV	AD	IT	PK	HJ	LZ	NS	EQ	CW	ouc	uhq	uew	uit
649	24	V	I	IV	05	18	14					TY	AS	OW	KV	JM	DR	HX	GL	CZ	NU	kpl	rwl	vci	tlq
649	23	IV	II	I	24	12	04					QV	FR	AK	EO	DH	CJ	MZ	SX	GN	LT	ebn	rwm	udf	tlo
649	22	II	IV	V	01	09	21	IU	AS	DV	GL	FJ	ES	IM	RX	LV	AY	OU	BG	WZ	CN	jqc	acx	mwe	wve
649	21	I	V	II	13	05	19	PT	OX	EZ	CH	RU	HL	PY	OS	GZ	DM	AW	CE	TV	NX	jpw	del	mwf	wvf
649	20	III	IV	V	24	01	10	MR	KN	BQ	PW	DP	HO	QZ	AU	RY	SV	JL	GX	BE	TW	jqd	cef	nvo	ysh
649	19	V	III	I	17	25	20					OX	PR	FH	WY	DL	CM	AE	TZ	JS	GI	idf	fpx	jwg	tlg
649	18	IV	II	V	15	23	26					EJ	QY	IV	AQ	KW	FX	MT	PS	LU	BD	lsa	sbw	vcj	rxn
649	17	I	IV	II	21	10	06					IR	KZ	LS	EM	OV	GY	QX	AF	JP	BU	mae	hzi	sog	ysi
649	16	V	II	III	08	16	13					HM	JO	DI	NR	BY	XZ	OS	FU	FQ	CT	tdp	dhb	fkb	uiv
649	15	II	IV	I	01	03	07					DS	HY	MR	GW	LX	AJ	BQ	CO	IP	NT	ldw	hzj	soh	wvg
649	14	IV	I	V	15	11	05	AI	BT	MV	HU	GM	JR	KS	IY	HZ	PL	AX	BT	CQ	NV	imz	noa	tjv	xtk
649	13	I	III	II	13	20	03	FW	EL	DQ	KN	LY	AG	KM	BR	IQ	JU	HV	SW	ET	CX	zgr	dgz	gjo	ryq
649	12	V	II	IV	18	10	07	RZ	OQ	CP	SX	MU	BP	CY	RZ	KX	AN	JT	DG	IL	FW	zdy	rkf	tjw	xtl
649	11	II	IV	III	02	26	15					KN	UY	HR	FW	FM	BO	EZ	QT	DX	JV	zea	rjy	soi	wvh
649	10	III	V	IV	23	21	01					LR	IK	MS	QU	HW	PT	GO	VX	PZ	EN	lrc	zbx	vbm	rxo
649	9	V	I	III	16	04	08					QY	BS	LN	KT	AP	IU	DW	HO	RV	JZ	edj	eyr	vby	tih
649	8	IV	II	V	13	19	25					FI	NQ	SY	GU	BZ	AH	EL	TX	DO	KP	yiz	dha	ekc	tij
649	7	I	IV	II	09	03	22					UX	IZ	HN	BK	GQ	CP	FT	JY	MW	AR	lan	dgb	zsj	wbi
649	6	III	I	V	11	18	14					DQ	GU	BW	NP	HK	AZ	CI	PO	JX	VY	lao	cft	zsk	wbj
649	5	V	II	IV	23	02	25	IL	AF	EU	HO	MV	CL	GK	OQ	BI	FU	HS	PX	NW	EY	lju	cdr	iye	waj
649	4	II	IV	I	04	21	09	QT	WZ	KV	GM	AC	BL	OZ	EK	QW	GP	SU	DH	JM	TX	lsb	zby	vcy	ujb
649	3	V	I	II	19	11	06	BF	NR	DX	CS	KR	MP	CN	BP	EH	DZ	IW	AV	GJ	LO	lap	owd	iwu	wak
649	2	IV	V	I	16	14	02					BN	HU	EG	FY	KQ	CP	OS	JW	AI	VZ	aqd	bdy	iyf	xtd
649	1	II	I	III	23	12	10					DP	BM	NZ	CK	OV	HQ	AP	UY	SW	JO	kgl	cdf	giq	wuv

204

Photo source: https://commons.wikimedia.org/wiki/File:Enigma_keylist_3_rotor.jpg

A list of keys for a German Enigma cipher machine.

English translation of text along the top (from Wikipedia):
“Secret Command Document! Every individual key setting is secret. Forbidden to bring on aircraft.
Luftwaffe Machine Key No.649
Attention! Key material must not fall into enemy hands intact. In case of danger destroy thoroughly and early.”