Cryptography in Industrial Embedded Systems
Our experience of needs and constraints

JP Aumasson, **Antony Vennard**

Teserakt

Who are we?

- JP Aumasson:
  - BLAKE2, NORX, password hashing
  - competition, Gravity-SPHINCS, SipHash, …
  - Blackhat, Defcon, Troopers, ...
  - He’s not in control of these slides!
- Antony Vennard
  - Software engineer (banking, defence, other)

Teserakt

E4: Encryption + key management for embedded industrial systems

(Announcements @ RWC)

HALucinator, with EPFL:

Firmware host emulation

Consulting

Trainings

Code audits, security assessments

Smartcard/HSM implementation

Why are we here?

- RFID Tags, **Industrial Controllers, Sensors, Smart cards**
- Minimum **112-**bit security level
- **~ 2000 gate implementation** (Simon/Speck paper)
- Political reasons

- Unclear/no power constraints
- Unclear/no timing constraints
- Unclear motivation to replace AES

Industry Experience
(ymmv)

Wearable devices customer

- Hardware crypto accelerator used
- Using a widely used SSL library
- P-256 and other “heavy” crypto
- Cortex M-0

Satellite coms customer

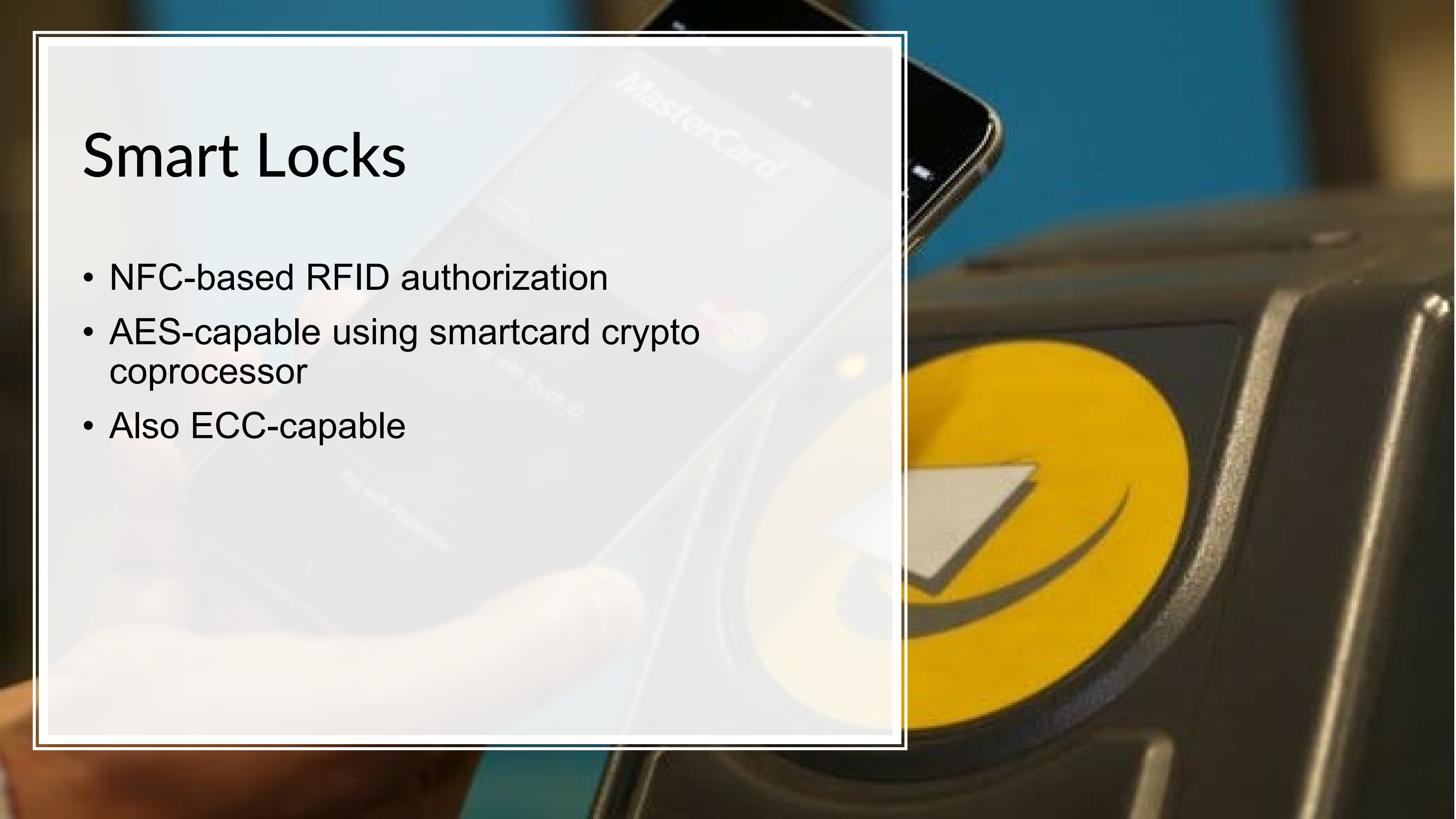
- AES + HMAC, crypto SDK
- Device powerful enough for this
- Geosynchronous orbit: limit transmission time window

Sensor Networks Customer

- Very limited payloads: can be as short as 12 bytes
- AES-based network authentication credentials
- Power + bandwidth largest concerns

Smart Locks

- NFC-based RFID authorization
- AES-capable using smartcard crypto coprocessor
- Also ECC-capable



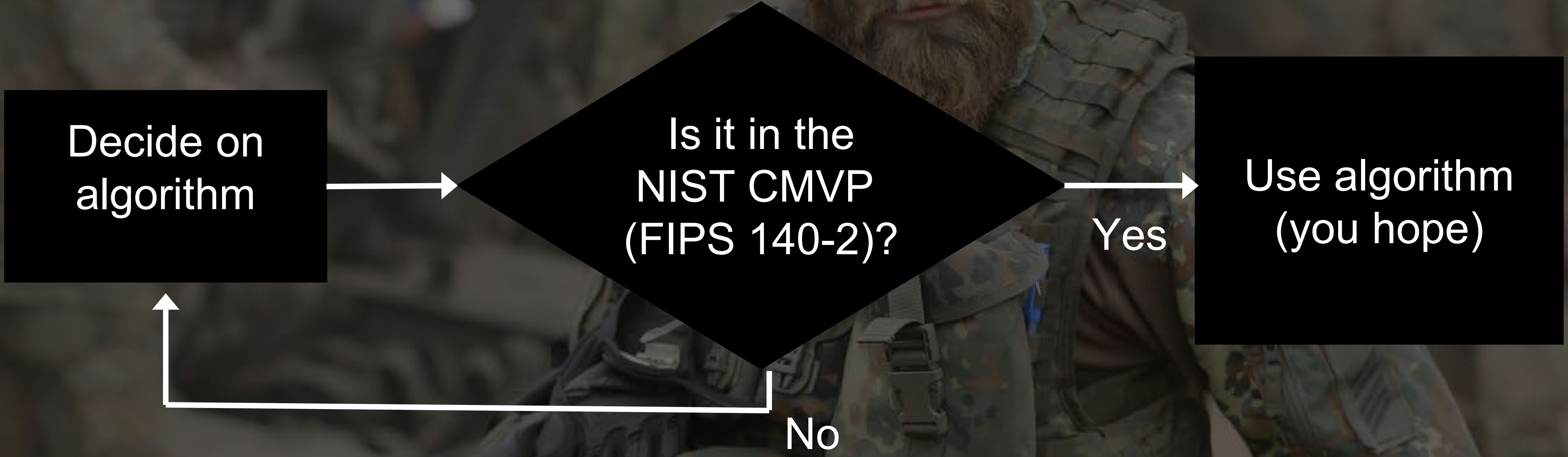
Bluetooth Low Energy

- Multiprotocol 2.4GHz radio
- 32-bit ARM Cortex M0 processor
- 256kB/128kB flash and 32kB/16kB RAM
- 128-bit AES ECB/CCM/AAR co-processor

Where	Chip details	Cost	What?
x86 Instruction Set, POWER instruction set	> 3 GHz	$500 up	Dedicated round instructions, keysched...
ARMv8-A	Variable	1 iPhone / 1 High-end Android / 1 Raspberry Pi	Dedicated round instructions, keysched, plus support for GCM and GF() operations.
AML11 ARM® Cortex®-M23 ("the new M4").	32 MHz ARM Cortex M23 Core, 64 KB Flash and 16 KB SRAM	CHF 58.96	Dedicated crypto accelerator AES-128, AES-128-GCM, SHA-256.
Infineon jTOP ID: SLJ 52GCA150CL	Symmetric Crypto Coprocessor	$6.75	AES up to 256-bit; DES, 3DES, p521, RSA-2048
NXP A71CH	AES Coprocessor, ECC Support	CHF 2.47 (less than a coffee) or CHF 50 for the Arudinuo dev kit	AES128, P-256 ECDSA/ECDH(E), HMAC
IACO Biometric Passports	AES support	Approx £100	ISO 14443 smart card, basically.

What	What if offers	Why no AES
Multos Step/One SmartCards	3DES for Static Data Authentication in EMV, SHA1	Probably pre-dates AES
IDESCO 8 CD 2.0 RFID	AES/3DES over the air RFID transmission	It does have AES!
"Fast Implementations of AES on Various Platforms", 2009	AES on 8-bit AVR	Software only but still there!
NXP UCODE® RAIN RFID (UHF)	UCODE DNA supports up to two 128-bit AES authentication keys. They are stored in the tag IC's securely guarded internal memory	It has AES
NXP QorIQ for Industrial Applications	PowerPC e6500 etc. Use case: high speed data link encryptors	It has AES

You don’t always have the luxury of choice (MILITARY GRADE CRYPTO)



Decide on algorithm

Is it in the NIST CMVP (FIPS 140-2)?

Yes

Use algorithm (you hope)

No

- Interoperability
- availability of high quality implementations on your platform
- availability of hardware-acceleration

Clear choices for devs:

- HMAC-SHA1? HMAC-MD5?
- SHA3: HMAC or not to HMAC? What do I include in the MAC?
- AES... CTR, OFB, EAX, GCM, GCM-SIV, SIV, CCM, CBC with MAC
- HMAC GMAC PMAC OMAC... Big Mac?
- I need to AES-Serpent-Twofish my hard disk yes?
- XTS sounds so cool I will use it for my network protocol!
- ChaCha20 or the totally not even an RFC XChaCha20?
- GCM has 96-bit IVs but why?
- Soooooo many curves
- So RSA why is that bad again?

Software can be chaotic, but we make it work
Expert
Trying Stuff Until it Works
O RLY?
The Practical Developer
@ThePracticalDev

Real World Constraints
(in our experience)

Stateless devices

- No storage for counters

Protocol issues

- Replay
- Retrofitting into legacy protocols
- Metadata obfuscation

Randomness

- Poor/no entropy
- Untrusted, maybe not PRNG
- K-ECDSA
- Accidental nonce reuse

Network connectivity sucks

- Transmission window
- Limited Payload
- Limited transmissions (power constraints)

Untrusted Infrastructure

- Trend towards cloud brokers and intermediate components
- Reliance on service providers for some hops: MVNOs for LTE etc

Expensive metadata

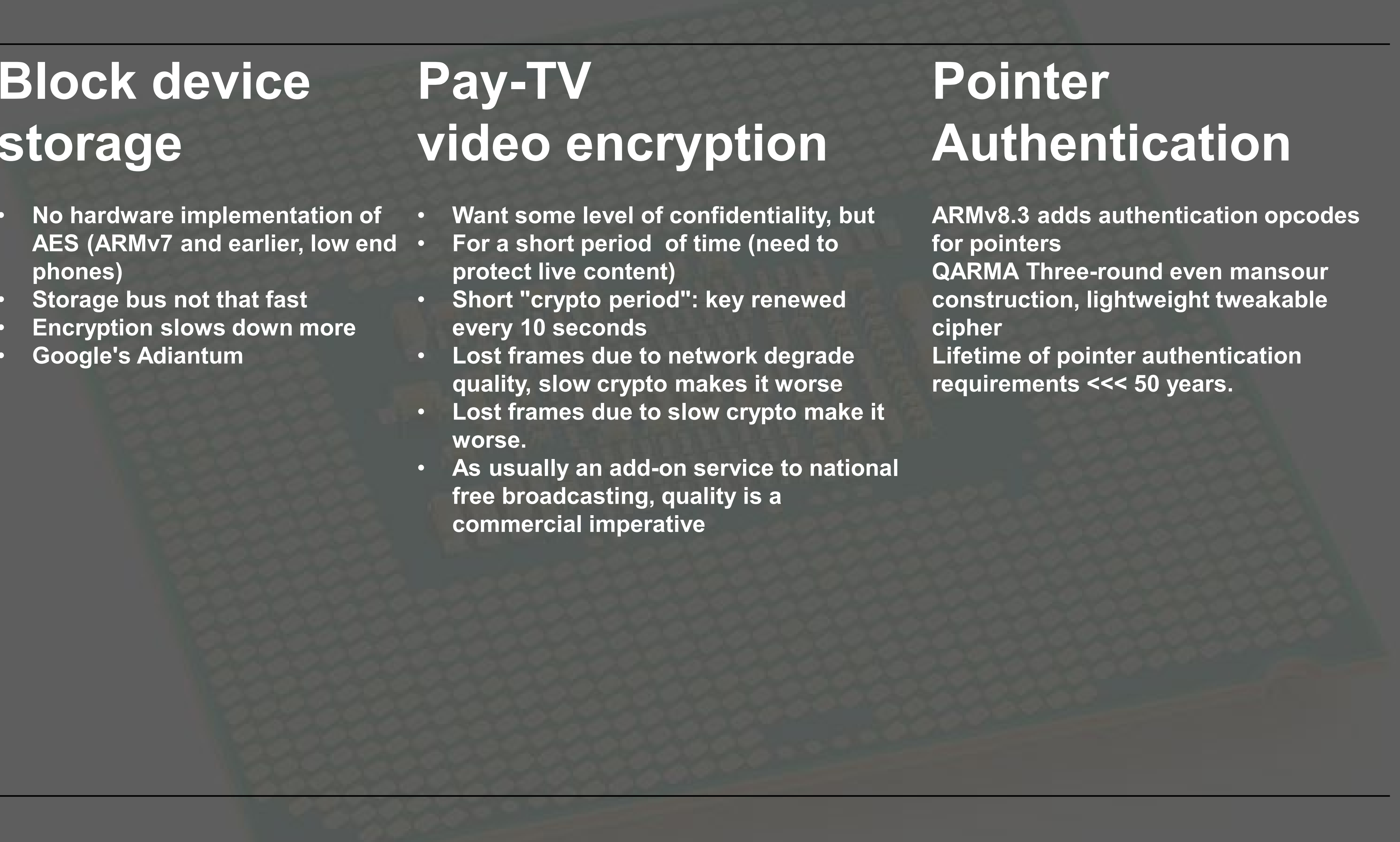
...issue with the use of Ed25519 or Ed448 in X.509 certificates is that these signature algorithms use an extra protection against collision attacks on hash functions...However, it means that the public key must be known before starting to process the signed data. ... the public key is made available only after the whole signed certificate has been received. ***Verifying a certificate path that involves use of EdDSA keys by CA thus requires buffering a complete certificate in RAM****, something which has so far been carefully avoided by BearSSL*

Thomas Pornin, BearSSL.org

Why might we want lightweight cryptography?

- What was the goal of AES?
- To protect US Federal TOP SECRET information for 50 years (lifetime before declassification review)
- Can afford to invest in hardware capable of supporting this goal

- Not all data needs guaranteed security for this length of time
- Sometimes there are much stricter availability requirements



Block device storage

- No hardware implementation of AES (ARMv7 and earlier, low end phones)
- Storage bus not that fast
- Encryption slows down more
- Google's Adiantum

Pay-TV video encryption

- Want some level of confidentiality, but
- For a short period of time (need to protect live content)
- Short "crypto period": key renewed every 10 seconds
- Lost frames due to network degrade quality, slow crypto makes it worse
- Lost frames due to slow crypto make it worse.
- As usually an add-on service to national free broadcasting, quality is a commercial imperative

Pointer Authentication

ARMv8.3 adds authentication opcodes for pointers
QARMA Three-round even mansour construction, lightweight tweakable cipher
Lifetime of pointer authentication requirements <<< 50 years.

Constructive goals, then

- **Better bounds**: what is lightweight exactly? Gate size, energy consumption, cycles/block, …? What security bounds can be afforded?
- Possibly **multiple standards** for multiple use cases, or at least **multiple security levels**.
- Need **more than primitives**: APIs, protocols, serialization, key management, etc. Industry struggles with this.
- **Clear messaging**: should we stop using AES? When do we use LWCS?

Fin! Thanks!

Praise for this talk: antony@teserakt.io

Criticism and abuse for this talk: jp@teserakt.io

Teserakt