Real World Cryptography 2015

J.H. Davenport — J.H.Davenport@bath.ac.uk

January 2015

Contents

Abstract

Chapter 1

2015 Jan 7

1.1 Virtual Currencies: Sarah Mieklejohn (UCL)

In 2012, the ECB produced a taxonomy.

Closed as in games. Black markets exist, but in theory these are separate from the "real world"

unidirectional Such as Avios, in which one can buy, but not (officially) sell.

convertible 1 e.g. Second Life dollars, controlled by a central authority.

convertible 2 e.g. Bitcoin, generated in a decentralized way.

We focus on the last. The key idea is a decentralized consensus on a globally agreed updateable but immutable artefact: the Bitcoin block chain.

1.1.1 The Hype cycle

Visibility plotted against time. Historically there are various (six) phases for a new technology.

- Technology startup.

2009 Bitcoin paper and deployed.

- Peak of inflated expectations.

2010 MtGox opens

2011 Silk Road opens. B1=$1, then shortly B1=$32.

2013 B1 goes from $13 to $1000 (the collapses).

2013 Senate hearing on Bitcoin — "a lovefest".

- Trough of disillusionment.

Late 2013 "Majority is not enough" paper [ES13]. Also problems with anonymity. Also "the killer app has yet to arrive".

Feb 2014 Bitcoin exchange MtGox collapses.

- Slope of enlightenment.

Now claims that we're entering this phase. I can hash a new paper, for example, and get its Bitcoin hash inserted in the Bitcoin block chain, thus a decentralised time stamp of authenticity. Note that this requires exactly the same properties as the Bitcoin block chain. This essentially replicates a Notary Public. I can also prove authorship, e.g. via eprints and including their signature in the block chain. This can be extended to prove history.

- Still to happen.

- Still to happen.

Q How much is decentralisation important?

A Depends on the application.

Q What happens when Bitcoin dies, i.e becomes so small that it is controllable.

A I was thinking of the more abstract Bitcoin successors.

1.2 Facebook hashing and authentication: Muffett (Facebook)

1991 `crypt` could only be iterated 3..30 times on Unix. Things have changed.

1.2.1 Facebook "Onion"

An (essentially) 8-line algorithm for password hashing.

- Uses `cryptoservice::hmac($cur)` which is an outcall to the cryptoservice. We have this to force our developers to use a serious service. This done on a sequence of hardened bastion hosts. It greatly reduces the attack surface. Also we can monitor it.

- Uses 20 bytes, rather than 12 bits as in Unix.

- It's slow (but in the part that's not in the `cryptoservice::hmac`, rather on the client. Currently provides c 1992 slowness.

Also strength: we have an additional cookie (`datr`) that stays with you (on your browser). If we don't see this, we will do more authentication. Also checks on location (TOR is painful here).

We monitor pastebin etc., also look for MD5 dumps. If we detect this [e.g. an MD5 key in a published list], and the previous factors indicate that it *is* the same user, then we force a password change at next use.

Also look for Capslock inversion, and "mobile keyboard automagically capitalises" password errors, and fix them silently (enhanced user experience).

Q Why MD5 in the first layer?

A Historical fact. We started with that, and it's grown from there.

Q Do you do further authentication once people are in

A (Boss) Yes, it's adaptive. (Questioner: awesome).

Q Can I give a set of one-time passwords to a friend, for uncoordinated users.

A Too hard (too explain to users). But we have a "recover via a group of friends" technology.

1.3 Life of a password: Mani (LinkedIn)

Talk covers Hashing, transport and storage (only).

Salted password hash is easy to get wrong. Cheap salt is dangerous. Keyed crypt has (key not stored n same place at password database) prevents dictionary attacks, but reveals shared passwords.

The logins servers are the only places able to read/write passwords from the (monitored) credential store.

Q–RJA Isn't it dangerous (in terms of rules changing) to place credit card and password details in the same store?

A Good question.

1.4 In PETS we Trust; Claudia Diaz (KU Leuven)

Privacy protections under ECHR and US Constitution ("unreasonable searches"). These are high-level and independent of technology. Note that these restrict states, and what about the private sector? Governments can be customers of data integrators. But the private sector *does* have to apply to Data Protection (or FIPS). The principle of consent is there, but people do not know what they consent to (privacy policies are notoriously hard to understand) etc. There are also subject access rights, but "de-identified" or aggregated data are not covered, so you don't have access to the decision chain.

There are those who argue for restricting use, rather than collection, but this seems even harder. Note that there is an implicit trust in data controllers. In security engineering, we see the service provider as an adversary, by contrast.

PETs should

- Handle the service provider as an adversary
- Minimise data collection
- open to public scrutiny
- Eliminate this single point of failure.

This doesn't include technologies like ad-blockers.

PETs are typically trapped in a limbo between the abstract (ECHR) and the concrete (DPA). PETs typically don't allow fro exceptions (court orders etc.).

1. PETs deployed by the provider
2. PETs deployed by the user. Here we must ensure that providers don't prevent/penalise these. Examples are e-mail encryption, anonymity.

N.B. TOR falls into this category, since the user is using it before the message reaches the provider.

3. The service provider is really the ISP — peer-peer communications. Again needs a "no block".

We believe that DP needs to adopt a PET-like approach, and view the SP as an adversary.

1.5 Protocol for Lightweight Authentication of Identity

Aimed at smart cards, Oyster etc. We are in the process of ISO standardisation. Quote from ISO website.

History AU Department of Human Services for its 26K employees. Standardised in AU in 2010. Then introduced to ISO as a "Fast Track" protocol.

We wrote a cryptographic analysis of this. We sent this to ISO, but this wasn't easy. Both DIN and BSI have submitted this as an official response.

Looked at authentication issues and privacy issues. The ISO draft claims "without the exposure of any information useful to the attacker".

Our view very weak privacy. Cards can be traced, and one can learn the capabilities of cards.

"Uncommon" design strategies.

- First terminal→card cards is not authenticated.[1]
- PKCS#1.5 padding mandated.
- Public keys encoded onto the card.
- CBC mode with a fixed IV=0.

These were actually already in the comments on V1, but are still in V2.

ISO This hasn't gone to SC27 (crypto), but SC17 (smart cards), which is "low-level encoding" only.

Q Who proposed the editor for this?

A AU.

1.6 Cryptography for everyone: ?? (W3C)

W3C has 400ish members, who vote on a charter (made by W3C team and members). Then members make IP commitments to the charter, and a sequence of drafts ending in a recommendation. At a Candidate Recommendation we must have two implementations fro every feature. Tim B-L is a great guy, but wasn't thinking defensively.

- No crypto primitives in browser.
- Must trust server-side JavaScript completely

Hence W3C Web Cryptography API.

Note that the Web forces you to trust the server's origin for remote code execution/ What happens if the server is forced to serve malicious JS code. Companies aiming to do end-to-end encryption via the Web (protonmail) are snake oil. There are native app alternatives.

LEAP architecture (my personal project) needs a heavy local client because of this. The standards (such as PGP) are rusty, and the new work is not backwards-compatible. We have a decentralised Web WG (but there's not much security there) looking at JSON-based activity standards.

It seems to us that we need some way of auditing JS (or at least JS libraries). `ttp://www.w3/org/TR/SRI`. How do we audit the auditing process and check that this has not been subverted.

Q EME?

A We need people (preferably in members) to oppose it rationally, rather than just have blogs.

[1] Editor's comment: "it's an implementation issue".

1.7 CLINT; A Cryptographic Library for IoT: Scott

There's a significant gap between academic cryptography and the real world (in which an engineer always uses a dog-eared copy of [Sch94]). Ideally this conference would help solve this.

Academic cryptography libraries tend to

- build on others (big footprint when included)
- include assembler etc. to "prove" they are faster
- ignore "real-world" issues like thread-safety.

CLINT is the opposite, available in several languages (currently working on C#) with identical behaviour.

Targets AES-128 security only. Mostly elliptic curve based.

1.8 Last talks

JHD's laptop crashes and he lost the last notes.

Chapter 2

2015 Jan 8

2.1 Short Tutorial on Symmetric Encryption: Sasha Boldyreva (Georgia Tech.)

A shared key: more precisely

K key generation

E $(M, K) \to C$

D $(C, K) \to M$

MsgSp the message space

Block ciphers only have short message spaces, and I regard them as *tools* rather than actually as encryption schemes, which are built e.g. with CBC.

So what is "good encryption". Contrasts provable security with trial-and-error methodology. Are we asking about recovery of the secret key, the whole plaintext, part of the plain text, a function of the plain text .... Informally we want no information to be leaked to a resource-limited attacker. Formally this is IND-CPA: pass in two (adversary-chosen) messages (M_0, M_1), and the oracle returns *either* $E(M_0)$ or $E(M_1)$. Secure if the adversary can't guess which with probability significantly from 1/2.

IND-CCA. We now know that IND-CCA implies IND-CPA. No deterministic stateless scheme is IND-CPA (since equality is leaked, and the attacker uses (M_0, M_1) and (M_0, M_1') and checks equality of the ciphertexts). Therefore no block cipher is IND-CPA.

CBC$ is IND-CPA if the underlying block is a pseudo-random function (PRF). A block cypher has this property if an adversary can't distinguish from a random *function* (i.e. deterministic). Note that we *assume* AES is PRF: this as not been proven.

There's also IND-CTXT, in which the adversary has to output a new valid ciphertext. IND-CPA plus IND-CTXT imply IND-CCA.

Encrypt then-MAC is IND-CCA and IND-CTXT if the encryption is IND-CPA and the MAC is UF-CMA (unforgeable under a Chosen-Message Attack).

Note that useful properties, such as deterministic, order-preserving, format-preserving can be IND-CPA. So what is the security/functionality balance in these cases?

2.2 Searching Encrypted Cloud Data: Sasha Boldyreva (Georgia Tech.)

This is a case study on academic/industry collaboration.

Academia Competitiveness (in terms of publication: novel, non-trivial, *provable security*, novel techniques, impact on future research). Therefore public

Industry Competitiveness (meaning can be sold: novel, *useful, efficient*, legislation-compliant). Therefore proprietary

Note that the definitions and proofs for provable security are hard to understand, and hard to see how they match the real world, e.g. "loss of bits of the key". Quoted example of a theorem from a paper with "practical, efficient" in the title.

We collaborated with a start-up called "skyhigh". Based on growth of cloud, but the fact is that security is a major concern. They claim that their products allow users to use existing cloud services with added security and no loss of functionality.

1. which schemes
2. How to optimize schemes without invalidating proofs
3. When can/should we trade security for functionality

Think of this as "database as a service". Exact match, range queries, fuzzy queries. Took functionality and efficiency as a given, and asked how much security can be added.

2.2.1 Order-preserving Encryption

$M_1 < M_1 \Rightarrow E(M_1) < E(M_2)$. It has a long history in the form of *one-part codes*. If we had this, we would certainly get range queries. But what about security. We know that this can't be IND-CPA: what can it be? One might think of modifying IND-CPA by insisting on the order of M_0, M_1, but this is not enough: [BCLO09]. An OPE essentially leaks relative distance.

Hence instead we ask that the OPE be Pseudo-random order-preserving secure if it is indistinguishable from a random *order-preserving* functions. There is a bijection between OPFs from $[M]$ to $[N]$ and combinations M/N.

It was a really great collaboration: they listened, came to conferences, wanted provable security, hired an academic board (Paterson, Ristenpart). Questions such as "when is frequency analysis an acceptance risk?". "Exact match search — what granularity: line/field/...?".

2.3 Protecting Credit cards: Spies (Voltage Security)

PAN Primary Account Number ("credit card number").

This is "large-scale brownfield engineering", of a system that has been around since the 1970s. There are 14G credit cards in the world, handling $3.6T transactions, with $12G of fraud (including merchant fraud etc., so this is not all preventable by technology.

Note that crypto can, and will be, misunderstood. People working in payment space are only interested in the output of standards groups. X9 is the key ANSI group for financial standards. X9.119 part 2 is tokenization, for example. X9.124 is Format Preserving Encryption. Also PCI SSC (Payments Card Industry Security Standards Council). PCI Data Security Standard is the requirement. This also charters Qualified Security Assessors, who audit merchants and processors. EMV (Was Europay/Mastercard/Visa, now includes Amex etc.): chipcard standards and payment tokenization. Federal Reserve has a "better payments initiative" to increase security and efficiency in the payments industry..

PCI DSS 1.0 (December 15 2004) was the first standard to mandate encryption of credit card data. Now at 3.0. No compliance means no processing. Because of the "brownfield", allows "compensating controls". Two main protection schemes: P2PE (point-to-Point Encryption) from swipe to host, via X9.119 part 1 guidance: needs format-preserving encryption. Tokenization is about encryption of PAN data in storage/analytics, and the creation of "limited context PANs". All major manufacturers now have crypto in the hardware.

There are actually two technologies in tokenization: payment tokenization and security tokenization.

Payment Tokenization replaces the PAN by a limited-context pseudo-PAN. The user enrols the card, and the phone receives a token. The TSP associates the token with a specific code. The transaction supplies token + cryptogram.

Security tokenization uses format-preserving encryption.Now a draft NIST: SP800. Note that the last four digits of the card number are in clear, also the first few, so in that it's only six digits that change.

Static Table Tokenization is a question about dependence on a single key. [LST12] tell us how to build a table-driven approach. Note that we are *not* dealing in asymptotics here!

2.4 Authentication Encryption and the CAESAR Computation

Want confidentiality and authenticity at the same time. Normally, MACs are used for authenticity, and we used "generic composition" to get AE. [BN00]: Encrypt and MAC, MAC then Encrypt, Encrypt the MAC (the only secure one). They also showed how to build probabilistic AE form probabilistic Enc. [BN00] also pointed out that ISO 19772 Enc then MAC got it wrong. This generic composition has been reconsidered [NRS14], especially nonce-based. Speaker concluded generic composition is very error-prone, and requires two data passes.

(A, N, M) is the input to encryption, which decrypts to $M/\perp$. want nonce-reuse (or even nonce-misuse[1]) resistant (NMR) schemes. Table of schemes, based on block ciphers or permutation (spone-like) schemes.

Another issue is "release of unverified plaintext (RUP)" (e.g. visible buffers. [Andreeevaetal2014-ASIACRYPT].

Competition for Authenticated Encryption: Security Applicability and Robustness. 57 submission in March 2014, of which 24 have been compromised. next round is probably Feb 2015. `http://homes.esat.kuleuven.be/~eandreev/caesarviz/index.html`. See `http://competitions.cr.yp.to/caesar.html`.

2.5 Smarter decisions with no privacy breaches: Bogdanov (Smartmind)

"paper-practical": the proposed technique solves a practical problem and performs rather nicely in the lab. [Bogdanov2013Thesis] "real-world practical" we had a customer, figured out the legal aspects, solved the deployment problems and made it work in a less-than-deal setting.

Comments on Estonian data: until 2011, 43% of students at 4 largest IT schools had dropped out: anecdotally "early hires". Would like to solve this problem by unifying education and employment data, but this is illegal under Estonian tax law (and EU DP). The current work is via anonymous IDs, but the "small groups" requirements leaves out 54% of Master's students and 78% of PhD students (worse if we do a gender analysis).

Hence we[2] do *real* multi-party computation across the two data bases. based on Sharemind's privacy database. This is running on *real* records. Two databases. each value is secret-shared at the source. Private data never leaves the organisation. 800K education records, 20M tax records. I claim that this is the largest secure MPC computation ever. Sharemind was sufficiently well-documented that the education/finance ministry staff installed it themselves.

We implement these data processing algorithms in a C-like language that uses privacy data types. Sharemind processes secret-shared inputs without reconstructing. Note that *not* having to hide things like the number of records

[1] e.g. M repreitiions of plain text.

[2] `https://sharemind.cyber.ee`.

actually makes life much easier (JHD didn't quite see how). Papers on the web site.

The analyst uses a R-like analysis tool. Sharemind hosts ensure that only queries in the study plan are actually executed. But the Ministries needed authorisation. So we described the private data flow to the Estonian DPA. They replied that "we are not processing personal information", so in fact (January 2014) no permission is required under Data Protection laws! However, there is still the Tax Law. January 2015: Tax Board agreed to upload actual tax records. Current phase, we have drafted agreements between the hosts and the data owners (including "non-collusion" clauses as required by the model). We believe that in the next few weeks we will actually sign the world's first secure multi-party data processing agreement.

Note that this is relatively few records (20M) and relatively simple processing. Nevertheless it's a very important precedent.

Q Is this the most efficient use of cryptographers?

A DPA is not the only use of this technology. Note that Estonia mandates full disclosure to tax authorities of inter-company financial exchanges. Only anomalies get reported to humans. Predicting satellite collisions is another application.

Q How did you get the exemption?

A There are two schools of thought — absolutist and relativist. The absolutist says that this is still private data. So the Estonians have essentially declared themselves as relativists.

JHD post See also [BJoSV15].

2.6 TinyGarble: Superfolding Garbled Circuits with Logic Synthesis: Koushanfar (Rice)

Example: Android app for "nearby people". The goal is secure function evaluation. Based on Andrew Yao's Garbled Circuits, 1986. We consider garbling as a stand-alone primitive.

Row-reduction can reduce the size of the garble truth-table by 25%. Also free-XOR [KolesnikovetalCrypto2014], Garbling with fixed-key block cipher [BellareetalSP2013]. Also execution optimisation.

We generate supercompact and scalable circuits by sequential logic description for functionality, introducing new transforms/library to enable adopting classic hardware synthesis techniques. For example we have results orders of magnitude better: SHA-3, RSA-8192.

2.6.1 Another speaker

Note that there are more transistors made every day than there are ants on the planet. Combinatorial circuits are functions of inputs, but sequential circuits are also functions of state. Describes advantage with "nearest neighbour search".

Global flow: `*.c/*.cpp` synthesised to `*.vhdl/*.v` goes through HDL logic synthesis `netlist/*.v`, which is scheduled to `*.scd`, after which we can do either garbling or evaluation. We are seeing substantial improvements[3] on circuits such as 64-bit multipliers etc., apparently by reducing the number of XOR's. K-nearest neighbour is an exciting application.

2.7 The ins and outs of Programming Cryptography in Smart Cards: Pailler (CryptoExperts)

Format is (APDU-C) header+data+Le (last is number of bytes). Response is (APDU-R) data+status word. Both contact and contactless commands (and can switch).

Native cards UART+CPU+cryptoprocessors+EPROM+RAM (maybe up to 1.8MB these days)+security features (anti-hacking etc.). Note that power is very limited. On top of this we build various APIs in C/Assembly, then all the OS-related stuff (timers etc.), then the "APDU processing" i.e. the application). Everything except the application is known as the "platform".

VM-based cards On top of the platform (as above) we have a JavaCard VM, then the JavaCard framework, APIs, add-on classes etc. Although this is Java, we don't use that for the crypto: MUCH faster to go via the APIs and the cryptoprocessors.

CPUs were 8 bits (Motorola 68C05, Intel 8051) but then went to 32-bit RISCs, ARM7-TDMI, ARM 9/11 etc.

cryptoprocessors Tends to require NDAs etc. Typically DES in 20 cycles. Good arithmetic processors have full modular arithmetic including inversion, variable operand length with automatic adjustment, exponentiation, side-channel resistance etc., but many are not that flexible. Some are basically just a Montgomery multiplier, often with massive side-channel leakage. Often bizarre, e.g. $x^{p-2} \pmod p$ is the best way of inverting.

These are a closed technology, and semi-open JavaCards have a significant slowdown factor.

Our opencard is the first truly-open card. 32-bit ARM core, 600kB memory, 18kB EPROM. To be launched Q2 2015, when we will also have `http://ww.cryptoexperts.com/opencard`.

[3] Lots of figures, but JHD didn't see quite these were measuring.

Q Good algorithms?

A DES and 3-DES, good arithmetic, but it's up to you.

Q Certification?

A No current plans.

2.8 The need for speed: Applications of HPC in Side Channel Research: Oswald (Bristol)

Until 1996, people didn't really worry about side-channels, then power/timing/EM attacks were all considered. Used for key recovery, plaintext recovery and device fingerprinting. Also acoustic attacks, as each microphone is unique. Web traffic [ChenetalIEEES&P2010] analysis allows to recover user choices. Detected mood, stress.

For a real-world AES trace, would need 2^{51} kernel computations. On Bristol's HPC setup (GPUs) could do this in days. can therefore ask what the best leakage point(s) is/are. How does combining work? MixColumns means that you have to look at 32 bits of the key at the time. on HPC/GPU can do 19M keys/second. Data from our simulation (want to look at various leakage models etc.). Combinations need roughly half as many traces. [MO12]

Next step is to attack an ARM processor. Actually, we switched to 6 £2K workstations, nVidia + AMD. Could do 2^{50} operations/seconds. [LMO+14].

2.9 Hardware Security Modules: ?? (Safenet)

"protected keys throughout their life cycle", "validated as secure by third partners", "a trust anchor" (bootstrapping trust)..

Practically, it is

- High quality source of random numbers
- a vault for holding crypto keys
- cryptographic acceleration hardware
- actual crypto algorithms (since the keys must never leave the vault!)

Can be USB, PCI, net appliance etc. in terms of connections. Practically all[4] speak PKCS#11. Java CSP, XML-DIG SIF, OpenSSL. Key management can range from "ever on" to "multiple physical keys inserted" (e.g. certificate authority's root keys).

Certification can be FIPS 140-2, PCI SSC (now have a HSM standard), Common Criteria.

Speaker noted that tamper-resistance can be overdone (all keys deleted when floor vibrates!).

[4] Except the dedicated financial services ones.

- Secure documents, as in e-passports. These have effectively smart card chips. These are build around a PKI. The keys are held by HSMs, in the passport printers and in the gates.
- Digitally signed games for games consoles. The signing keys are now (Sony!) kept in HSMs
- HSMs secure Smart Metering Systems (notably ECC because of power consumption).
- HSMs secure mobile money payments — the HSM will check "characters 2, 3 and 5 from the password" tests.
- Issuing PINs: the printer either has an HSM in it, or has one attached.
- Also smart card production, and needed at high speed.
- Payment terminals thee days (post-Target) tend to have encryption keys in them.
- SWIFT is completely driven by HSMs: every member has one.
- Five railway systems worldwide now use HSMs for the signalling infrastructure.
- Aircraft: showed a diagram of the flow of software, with 8 rôles for HSMs.
- Netflix unique keys come from an HSM.

Q-JHD HSM in the passport gate?

A In some countries, yes. Or in the (local) server controlling a bank of gates.

Followup JHD queried this privately. the reason you need an HSM in/at the gate is to prevent tampering with the gate, apparently. Clearly JHD was insufficiently paranoid.

Q To what extent has TPM eaten into your market?

A TPM has taken a fair chunk of smart card business, but we seem to have coexistence.

2.10 OpenSSL: Emilia ??

Four major attacks in 2014

goto fail

HeartBleed

Early CSS

RSA signature forgery

Of these four, HeartBleed had the lowest NSA-assigned "impact". Doesn't require an active adversary, so is high on exploitability.

So why way HeartBleed so critical, and what is the future of OpenSSL? "It had a logo"! More seriously, it was post-Snowden. Lessons.

1. We need code review.
2. But review $\neq$ audit. Few people are paid to audit critical code bases for vulnerabilities. "You get what you pay for" $\Rightarrow$ "we need to fix this" (and are).
3. Expanded team: 3 FTE plus 12 volunteers, mandatory code review, new security policy, new release strategy, new blog.

   In fact there's a new release today. This is the fifth security release since HeartBleed — very two months since then.

   We have $< 2^{-64}$ corner cases: hard to hit with unit testing.

   New-ish elliptic curve implementation: P-224, P-256, P-521: fast and constant time. Are they correct[5]? We also had a bug in bignum multiplication, which was hit by biased generation.

   There are issues with the state machine: it was written to be correct with messages in the right order, but wasn't designed with an adversary in mind.

   Also problems with ASN.1 object layer fussing: also some (primitive) open source tools in this area.

   Authenticated encryption is brittle: we need better, standardised, primitives.

2.11 PQ Key Exchange for TLS/RWE

The public-key security is based on factoring and DLOG, which could be solved by quantum computing. Hence PQC:

- Parameter sizes
- Are the key sizes sufficiently small
- can we do the operations sufficiently fast
- How to integrate?

Note that we only need authentication to be secure **now**, but the key exchange to be secure into the future.

[5] She mentioned overflow bugs.

2.11.1 LWE

Solving $M.x = r$ is easy, but adding a small error factor seems to make it hard. But to get 128-bit security we need 256kB keys. hence Ring-LWE (right meaning the matrix is composed under cyclic shifts: Toeplitz?). Can recast as a problem in $\mathbf{Z}_p[x]/(f(x))$. To get 128bit security, need 4kB keys [Peikert-PQCrypto2014]. A DH-like key exchange means that they have the same shared key, or to (different) errors.

We can integrate this into TLS, using existing authentication technology. Use Authenticated and Confidential Channel Establishment [??]. This is 1.75× slower than the EC-DH in OpenTLS, but the whole system is 1.2× slower because of other costs. But there are changes to the message order.

Note that it is possible to use a hybrid version, where the key had both a ECDH component (therefore at least as safe as today) and this component (presumably PQ).

Q preQ or postQ 128?

A preQ. I don't have an answer for postQ 128.

Q Memory usage?

A No figures. A small server seemed to have no issues here.

Q How does "moving things around" affect the DH-side of security of your hybrid scheme?

A We believe not, but see TLS 1.3.

Q Should this be standardised now?

A Not the pure scheme, as we really don't know about key sizes. If anything, the hybrid (modulo previous question).

2.12 post-Snowden ECC: (Microsoft)

Schneier I no longer trust the constants. I believe NSA has manipulated then

[**1.** Curves that regain confidence: "nothing up my sleeves". [NUMS]

2. Improved performance and security for standard ECC algorithms and protocols.

2.12.1 NUMS curve generation

1. Pick p according to well-known criteria $p = 2^{2s} - c$ where c is least such that $p \equiv 3 \pmod 4$ (and later allowed $p \equiv 1$. $s \in \{256, 384, 512\}$.
2. Find smallest $|d| > 0$ a non-square such that $\#Ed$ and $\#E'_d$ are coprime.

* Saturated or unsaturated arithmetic[6]. Depends on cost of carries (Intel desktops, ARM/NEON). Unsaturated needs better bound analysis.

* Curve TED37919, defined over $\mathbf{F}_{2^{379}-19}$. $\approx$ 188-bit security. Minimal d in twisted Edwards, safe etc.

Q-TL Now is NUMS different from our criterion [BL13]?

A Don't know.

[6] Unsaturated 10% faster on Haswell, 47% slower on AMD Bobcat. These were his extrema.

Chapter 3

2015 Jan 9

3.1 Triple Handshake: Can cryptography, formal methods and applied security be friends: Karthik Bhargavan, Markulf Kuhlwein (INRIA etc.)

3.1.1 TLS 1994–

The most widely used cryptographic protocol. Goal: a secure channel. See ACCE. Between an honest client and an honest server:

1. Hello (C→S→C)
2. KEM (RSA key transport, DHE/ECDHE etc.)
3. Finished (C→S→C)
4. AppData (MAC-Pad-Encrypt in TLS 1.2 mandated)

It's really a framework: lots of protocols. There are many obsolete constructions. RSA encryption with PKCS#1 v1.5 padding (Bliechenbacher); M-P-E (POODLE), RC4 key biases, Chained IVs in TLKS 1.0 AES-CNC (BEAST). Also implementation challenges (HeartBleed). Missing checks in a crypto program don't crash it, just cause additional packets to be accepted. There are also bugs in the implementation of the state machine (unexpected transitions, EarlyCCS, OpenSSL etc.).

So we don't we use formal methods: a type safe language would have prevented HeartBleed, and there are crypto verifiers like EasyCrypt. Hence miTLS, a reference implementation of TLS 1.2 in F#, 7kLOC, 3K lines of F# type annotations, 3k lines of EasyCrypt proof. But how does the verification link to the crypto assumptions? [Jageretal2011] Security for TLS-DHE + authenticated encryption in the standard model. [BhargavanetalCrypto2014].

Agreement If the client completes there must be a server with ...

Authenticity Each endpoint only accepts data ...

Correctness If its closed normally, all data has been accepted.

Confidentiality

Standard socket API with embedded security specification. Abstract types for confidentiality, and refinements (like pre/post conditions) for authenticity.

Theorem 1 *The ideal implementation and concrete TLS are computationally indistinguishable*

So far so good.

3.1.2 Reusing established sessions

Known in TLS as "abbreviated handshake". Most browsers etc. use it. Claim that, if the client completes an abbreviated handshake and the server in the original one was honest, and the secret has not been leaked, then we have agreement.

3.1.3 User Authentication

Outer: server-authenticated TLS, and Inner a user authentication protocol. SASL, GSSAPI etc. But if the two are not tightly bound there is a MitM attack [AsokanNiemeNyberg2002]. Martin Rex's version of the TLS renegotiation attack (2009). Need to have a channel id for the outer channel which is part of the inner handshake. EAP uses F(master secret), SASL uses F(handshake log). Hence MitM must not be able to create a matching channel id. [Giesenetal2013] shows that if an endpoint completes, then a handshake validates all the previous ones.

3.1.4 Triple Handshake and Cookie Cutters

[Bhargavanetal] `http://secure-resumption.com`.

1. Key synchronisation attack. In DHE M chooses a bad (non-prime) DH-group. Doesn't break MK's theorem, since the client did not complete with an honest server (but with MitM). This does break EAP compound authentication, as the master secret is not contributive.

2. , Transcript resynchronisation attack. After resumption, malicious M can ensure that the master secrets, keys and handshake logs on C-M and M-S are the same. Then this breaks abbreviated agreement. As above doesn't break MK's theorem.

3. User impersonation attack. Renegotiation with an honest peer implies agreement on the abbreviated handshake, but not with the original (these theorems don't compose as nicely as one would like).

- disallow server certificate change during renegotiation (preferred fix for browsers)
- Use only well-known DH groups and validate these: preferred fix for TLS libraries
- Disallow client authentication after resumption (hand to do compatibility)
- Fix the standard: add a session hash. `draft-ietf-tls-session-hash-02.txt`. This is in TLS 1.3, and should happen.

We can also verify small fragments of OpenSSL. There are many open verification problems (e.g. the above session hash idea). No analysis of side-channels yet.

Q-RJA Needham–Abadi robustness principle [AN96] would have given you the session hash anyway: you don't need formal methods.

A There's more than one way to *design* good software: we were talking about verification.

3.2 Crypto at Scale: Sniffen (Akamai)

We are a content delivery network, as in `guardian.co.uk`. 10P requests in 2014 (it really is to 6 s.f.). 60KB each. 250K machines, 15K web servers, 80K sharing one certificate (`*.akamaihd.net`) for low-value delivery. Also 50K machines use 10K vanity names (e.g. `cnn.com`, `foxnet.com`), 5K PoP. About 50M IPv4 addresses (previous two numbers multiplied together!), and 20% of web traffic.

"TLS is a tool for making fewer Web connections work". "It's been a year of raining anvils at the operations level".

1. HeartBleed. We were amazed then this went public and everyone started rotating their certificates. The CAs said that this would take eight months. In fact we rotated about 7,500 in a month, and the rest are mostly those where Akamai doesn't control the relationship with the CA.
2. Live streaming customers (due to protocol errors in these weird suppliers) can't rotate their protocols while streaming. OK for the Olympics, but really bad for permanent streamers.
3. POODLE: "our excuse to shove 3.0 over a cliff". Might work for the 80K network, but there was a lot of use on the (valuable) 50K network. A lot of web spiders turn out to be SSL 3.0. There were some customers (mostly

IoT) who were 100% SSL 3.0. But nearly all kinds of customers had a some customers who *needed* SSL 3.0 (cars which needed a dealer visit to upgrade, Nintendo WII which will never be updated etc.).

4. FOSSIL (we wanted to Fix Origin SSL) 99.4% done. Lotus Domino needs money to turn off SSL 3.0!

SNI RFC 3546. This was a third party patch for OpenSSL. Never in XP, or Android 2.2 Froyo. We were on just over 80% requests with SNI, then after the XP-end-of-life cliff it rose until 95%, but the trend doesn't hit 100% until 2030.

> "The TLS 1.3 that we are about to standardise, will still be driving round in trucks in 2030".

Can we have overlapping windows of safety across a decade? Better a change a year than a crisis every five years. We would like to steer clients based on protocol version. SRV records? A good (not RST) way of saying 'I hear you, but that's now an obsolete protocol'.

3.3 Protecting Data in Untrusted Locations — An exercise in real-world threat modelling: Jan Schaumann (Twitter)

See `https://t.co/ykdsHGV84r` for peering information.

> We knew that the World Cup was coming, and people were going to tweet from non-standard locations (e.g. from stadia)

Added a PoP in Brazil, with servers there (to do the handshake locally) and plenty of peering with network providers.

3.3.1 Threats

- *hackaris vulgaris*
- organised crime
- local governments
- foreign governments

3.3.2 Assets

- Physical equipment (vulnerable to local government)
- Local Service Access Point (vulnerable to local government)

- Access/Entry point to infrastructure (needs physical protection, traffic severely restricted and mutually authenticated)
- TLS keys

Hence picture of a site: cabinets inside wire cages floor–ceiling with sufficient air gap. Main threat is bribery/coercion rather than crypto (XKD cartoon). WE might make machines discless, but then how do we trust the netbooting service?

HSM? — expensive, and we didn't have the time to do the planning. Hence "HSM–light": TPMs. TPM wouldn't hold all the keys we needed. Hence used TPMs to bootstrap trust. Lengthy process the first time, then store a key in the image. At reboot, this is again checked by the TPM.

Defend by both raising the cost of an attack and decreasing the value of the asset (no permanent keys). Also need detection as well as prevention.

You can't just rub some crypto on it.

Q The first boot is expensive. Did you think about booting elsewhere?

A Shipping/customs. We will still need to inspect etc.

Q Why is the problem only Brazil?

A We are reverse-engineering this into our general business model/process. Also "foreign government threat" is independent of location.

3.4 Universal SSL: Nick Sullivan (CloudFlare)

HTTPS myths: "only for bankers", "only needed for authentication" "it's too hard" (there is actually some truth here). The truth is that HTTPS protects the privacy of your customers against invasive intermediaries. Also SEO. There are many sites (NYT, Amazon etc.) that don't have always-on. Reasons:

- Sysadmin tie/training
- business profess/risk
- vendor cost (CDN)
- mixed content warning from advertisements.

But there are also a load of low-end reasons.

We therefore offer "HTTPS as a service". Essentially a reverse proxy. We bring TLS termination closer to the visitor. We have c.30 PoP.

Certificate Management A complex process. We need to automate this, and handle all the interactions with CA. This needs DV (WHOIS e-mail, or DNS validation (add a TXT record, which we can do if we are the DNS provider as well), or put an index in the HTML (we can do this if we serve the HTML)), OV and EV. [JHD: I don't think he addressed the other forms of validation].

Keyless SSL (talk at last RWC) so the PK doesn't actually leave the customer. Slows the handshake slightly, but only by 1 RTT to the customer.

Scaling: CERTs One server can handle thousands of certificates. This copes with big/medium enterprises. For small ones, we can use SAN. But this only really allows a 10:1 reduction 9else the key gets unwieldy). This copes with 100K "budget" customers, but not the millions of free clients. Hence use "lazy loading" (JHD: apparently an Apache service).

Scaling: IP All this gets multiplied by #PoP. Difference between anycast and unicast. In anycast, BGP handles the routing, which we do. This reduces to 100K IP addresses. To do better, we need SNI (see above). This is a TLS extension that adds the hostname to the TLS HELLO. Not on XP, or early Android, especially on the OS browser. We turned off SSL 3.0 for all our sites after POODLE (one or two turned it back on).

Performance Westmere has AES instructions (hence practically line rate). We can reduce the server computation for handshakes by moving to EC, $\times 10$ faster than RSA. Note also that session resumption can be shared across multiple machines.

This added 10^6 HTTPS-sites to the net, was $4 \cdots 10^6$ previously, so significant. Mixed content warnings: we're working on it.

Q Customer pushback over SSL?

A Some, mostly over web crawlers, but not much.

Q XP SP3 doesn't support SNI, but there's plenty in Government.

A Indeed: requests just $< 10\%$ of all don't have SNI.

3.5 TLS 1.3: Eric Rescorla (Mozilla)

Clean up Remove unused/dangerous features.

- Static RSA Key Exchange, which doesn't have PFS. Hence take this out, but only support ephemeral RSA certificates. ECDHE has a minimal performance hit.
- Compression. Various vulnerabilities [DR12]. TLS 1.3 servers MUST FAIL if compression is offered.
- Non-AEAD ciphers. So only AES-GCM, AES-CCM and a few more. In particular no RC4.
- No custom DH/ECDH groups (see before). Client can't really validate. RFC 4402 (CFRG when it happens and various new FF groups [GH14]).
- Renegotiation. Confusion and vulnerable. But we need to support some of the uses (adding private client authentication).

***** JHD left here.

Privacy Improve by encrypting more of the handshake

Latency reduce RTTs. Want 1 for naïve clients, 0 for repeat connections.

Continuity Need to cover almost all the TLS 1.2 use cases.

3.6 Remaining talks

JHD had left.

Bibliography

[AN96] M. Abadi and R.M. Needham. Prudent Engineering Practice for Cryptographic Protocols. *IEEE Trans. Software Engineering*, 22:6–15, 1996.

[BCLO09] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill. Order-preserving symmetric encryption. In *Proceedings EUROCRYPT 2009*, pages 224–241, 2009.

[BJoSV15] D. Bogdanov, M. Jõemets, S. Siim, and M. Vaht. How the Estonian Tax and Customs Board Evaluated a Tax Fraud Detection System Based on Secure Multi-party Computation. `http://fc15.ifca.ai/preproceedings/paper_47.pdf`, 2015.

[BL13] D.J. Bernstein and T. Lange. SafeCurves: choosing safe curves for elliptic-curve cryptography [originally 2013, continual update]. `http://safecurves.cr.yp.to`, 2013.

[BN00] M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In *Proceedings Advances in Cryptology — ASIACRYPT 2000*, pages 531–545, 2000.

[ES13] I. Eyal and E.G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. `http://arxiv.org/abs/1311.0243`, 2013.

[LMO+14] J. Longo, D.P. Martin, E. Oswald, D. Page, M. Stam, and M.J. Tunstall. Simulatable Leakage: Analysis, Pitfalls, and New Constructions. *Advances in Cryptology — ASIACRYPT 2014*, pages 223–242, 2014.

[LST12] W. Landecker, T. Shrimpton, and R.S. Terashima. Tweakable blockciphers with beyond birthday-bound security. In *Proceedings CRYPTO 2012*, pages 14–30, 2012.

[MO12] L. Mather and E. Oswald. Pinpointing side-channel information leaks in web applications. *Journal of Cryptographic Engineering*, 2:161–177, 2012.

[NRS14] C. Namprempre, P. Rogaway, and T. Shrimpton. Reconsidering generic composition. In *Proceedings EUROCRYPT 2014*, pages 257–274, 2014.

[Sch94] B. Schneier. Applied Cryptography. *Wiley and Sons*, 1994.