Real World Cryptography 2017

JHD

4–6 January 2017

Contents

1 4 January 2017 **3**

1.1 Introduction 3
1.2 Software engineering and OpenSSL is not an oxymoron: Salz . . 3
1.3 Project Wycheproof — Scaling crypto testing: Duong 4
1.4 X.509 in Practice (It's worse than you think) — Camp 5
1.5 Is Crypto Software Safe Yet? Nguyen 5
1.6 NSEC5: Provably Preventing DNSSEC Zone Enumeration: Goldberg 6
1.7 Cryptographically Securing the Network Time Protocol: Franke . 7
1.8 Levchin Prize 8
1.9 White-Box Cryptography: Bos 8
1.10 NIST's Post-Quantum Cryptography Project: Peralta 9
1.11 Cryptographic Suite for Algebraic Lattices CRYSTAL: Lepointe 10
1.12 Practical post-quantum key exchange from both ideal and generic lattices: Nikolaenko 11
1.13 Supersingular Isogeny Diffie-Hellman: Naehrig 12
1.14 The Strobe protocol framework: Hamburg 12
1.15 Four**Q** based cryptography for high performance and low power applications: Longa 13

2 5 January 2017 **14**

2.1 High-Throughput Secure 3PC for Semi-Honest and Malicious Adversaries - Breaking the Billion-Gate per Second Barrier: Lindell 14
2.1.1 Achieving Malicious Security 14
2.2 Secure Multiparty Computation at Google: Kreuter 15
2.2.1 MPC: Academic and practice 15
2.2.2 Business-Business MPC 15
2.2.3 MPC on mobile/consumer devices 16
2.3 Privacy-Preserving Classication on Deep Neural Network: Phesso 16
2.4 Challenges of E2E Encryption in Facebook Messenger: Millican 17
2.5 Memories for Your Eyes Only: Yung 18
2.6 DMCA: Stoltz 19
2.7 Lightning Talks 20
2.8 Message Encryption: Perrin 21

2.9 A Formal Security Analysis of the Signal Messaging Protocol: Luke 21
2.9.1 post-compromise security . . . 22
2.9.2 model . . . 22
2.10 0-RTT Key Exchange with Full Forward Secrecy: Günther . . . 23
2.11 Towards 5G Authenticated Key-Exchange: the security and privacy of the AKA Protocol: Oneta . . . 23
2.12 Is Password InSecurity Inevitable? Cryptographic Enhancements to Password Protocols: Krawczyk . . . 24
2.13 Towards a Theory of Data-Independent Memory Hard Functions 25
2.14 The memory-hardness of Scrypt: Tessaro . . . 26
2.15 Solving the Cloudflare CAPTCHA . . . 26

3 6 January 2017 **27**
3.1 The physics of building a quantum computer: Jeffrey . . . 27
3.2 Erasing secrets from RAM: Simon . . . 27
3.3 Direct Anonymous Attestation and TPM 2.0: Getting Provably-Secure Crypto into the Real-World: Lehmann . . . 28
3.4 DPA Resistance for Real People: Handschuh . . . 29
3.5 What Else is Revealed by Order-Revealing Encryption: Cash . . 30
3.5.1 Backrounds . . . 30
3.5.2 Correlation . . . 31
3.5.3 Case Study . . . 31
3.6 Breaking Web Applications Built On Top of Encrypted Data: Grubbs . . . 31
3.7 Building web applications on top of encrypted data: Popa . . . 32
3.8 PRNG Failures and TLS Vulnerabilities in the Wild: McGrew . . 33
3.9 Concerto: A Methodology Towards Reproducible Analyses of TLS Datasets: Levillain . . . 34
3.10 Productizing TLS Attacks: The Rupture API: Sarafianou . . . 34

4 Blockchain Session **36**
4.1 Rethinking Internet-Scale Consensus: Shi . . . 36
4.2 Listening to and Silencing the Whispers of Ripple: Study and Solutions for Privacy in IOweYou Credit Networks: Moreno-Sanchez 37
4.3 Cryptography and Protocols in Hyperledger Fabric:Cachin . . . 38
4.3.1 Hyperledger Fabric . . . 38
4.4 Improving Authenticated Dynamic Dictionaries, with Applications to Cryptocurrencies: Reyzin . . . 39
4.5 Closing . . . 40

5 Conclusions **41**

Chapter 1

4 January 2017

1.1 Introduction

Tom Ristenpart: From Cornell Tech (i.e. based in NYC). Ironically, Columbia's wireless network has no password. There were 61 submissions, of which 25 were accepted. RWC 2018 will be in Zürich 10–12 January 2018.

1.2 Software engineering and OpenSSL is not an oxymoron: Salz

Speaker is Akamai/OpenSSL.

SSLeay Eric: "I've got DES and Bignum", Tim "Let's do SSL". Two guys in a garage (in Australia, free of US regulations).

OpenSSL They got day jobs, so had a group with a dozen members. "Interoperate with OpenSSL is more important than what the RFC says" was IETF reality. No real release policy etc. Code was complex and arcane: all via tables of dispatch function pointers. Project donations <2000/annum. Steve Henson did 448 commits in two years, Poliakoff did 340 — he was the assembler wizard. So barely a commit/day. All done via an extra level of indirection. And the group was living on consulting (FIPS dollars), so there were no viable plans. This all added up to a "stay dark" attitude.

CVE-2014-0160 Showed a Daily Mirror front page article as an example of impact. This was the first CVE with a logo/name.

Recovery CII created, and funded two, and donations jumped (funded two more). Actual face-to-face meeting in October 2014 (12 people × three days). Wrote release, security policies. Coding style!! Poodle helped. Met again in October 2016. CVE notification list. How to grow the team, how to get more testing. Updated roadmap and platform documentation.

Linux Foundation is the secretariat of CII. We have more tests, but Coveralls only reports 57% of lines. Using modern practices[1] like fuzzing, continuous integration (Travis). Now it's GitHub, we get more outside interest/contribution. 3889 commits in 2016. 107 new pull requests.

2016 CVEs 9 high (forces a release), 20 medium (might force a release), 28 low (just fix). We mostly met the fix/release deadlines. The major technical debt is being addressed (threads, state machine, TLS packet formats).

Future FIPS work is funded but on-hold for TLS 1.3, Licensing is moving to APLv2 (Apache). We also need to fix the RNG *portably*. We need a generic STORE facility for PKI objects.

1.3 Project Wycheproof — Scaling crypto testing: Duong

Speaker is from Google Security. Our crypto libraries are third party (OpenSSL again), then the APIs, protocols (E2E encryption, database encryption etc.) and products are ours. Bug the libraries have too many bugs for too long. There are essentially no good crypto implementation guidelines for smart software engineers without the technical background. Also, when we fix a bug and export the fix upstream, we still see the bug coming back.

80+ open source tests uncovering 40+ bugs in popular implementations of ECC, RSA, DH, DSA, AEAD etc. We have released our out-of-the-box test runners for Bouncy Castle, Spongy Castle and OpenJDK.

Notable bug in OpenJDK's DSA `signer=Signature.getInstance("DSA")` — only uses a 160-bit nonce. Also Bouncy Castle's ECDHC.

Want we want for Christmas: common crypto interfaces for C++, Python etc. (apparently Java is essentially there), robust and readable interfaces where we can switch algorithms in an existing application, show crypto properties in the code, never ask users to provide critical input such as randomness.

Q How long doe sit take to Google to update their software for a bug?

A I am not the right person to answer this. We want the window to be measured in days.

Q What do you mean by "switching algorithms"?

A We'd like to decrypt the old ciphertext then switch to the new algorithm without having to rewrite the application. In practice this means that the ciphertext needs enough metadata to describe the encryption.

Q Why not use the right algorithm in the first place? [rather than APIs?]

A Our API are meant to abstract away the technical details of the implementation.

[1] OpenSSL is *old*!

1.4 X.509 in Practice (It's worse than you think) — Camp

We've been observing certificates in the wild. Akamai top million every day, Phishtank every hour (typically 40–60 TLS/hour), Banks as defined by FDIC (twice): 27,000 institutions: the FDIC form requires the domain name (but generally poor, e.g. `foo@gmail.com`). MD5 is disappearing (slowly), Our last observation was June 2015. Version 3 is being adopted, but not really until 2015. TLS abuse dominates TLS issuance. Most phishing is hosted by cloud providers. TLS phishing is small compared to overall payment fraud, but is growing. `sinkdns` is used by 51 banks. After Heartbleed, a significant number of people changed their certificate, but not actually their key. Most certificates active at Heartbleed have been replaced recently, but that is more a matter of expiry than genuine action. She had spotted people transitioning from SHA-1 to MD5!

Siemens Smartthings key has a 12 years validity with SHA1 and RSA1024 key! Google requires 25 years lifetime with no revocation! Mother Sen.se[2] connects through any hub. All sensor data that is received by any mother goes to the cloud, and "cookies" send to the closing mother. There's a verified connection to the server, but the data is sent through an unauthenticated connection!.

I predict we'll see the last web server using SHA1 in 2020, but by Siemens definition they will be around until 2025 at least, and I'm predicting 2030 for the last sighting.

Q At Chrome, we don't think CAs are the right people to enforce the "no phishing" rule.

A But if CAs are issuing them, then the PKI infrastructure is not doing its job. The literature says "green bar is safe".

bis We intend to kill the green logo!

Q Downgrades! Was this switching providers?

A This switching is very rare. It tends to indicate a malicious certificate. The real problem is the incentive system in the infrastructure. At the moment we have a law enforcement system with only the death penalty of exclusion.

1.5 Is Crypto Software Safe Yet? Nguyen

The encryption/signature is mostly under our control, but the other end is under attacked control. Json tokens provide multiple signatures: ECDH, CBC-HMAC encryption. See RFC7515 section 4.1.3. The JWK Header Parameter is the public key that corresponds to . . . ". Square's go-jose embedded the public key in the signature. Go-hose does check the well-known "invalid curve" attack.

[2] A slide showing a really evil doll.

The sender can extract the receiver's private key. There are various 32/64 overflow issues. I can shift the boundary between the aad and the nonce by setting len(aad) to be 8.

Also problems with Galois Counter Mode. The counter is module 2^{32}. `auth_tag` is what you see on the wire, so it's under attacker's control. There's a Truncation attack where the attacker sends 1 byte. `auttag`. The counter wrap-round causes counter collisions which leaks both plain text and authentication key. Both Bouncy Castle and OpenJDK8 have this bug.

Classic Timing vulnerability on OpenJDK8. Recall Joux's Forbidden IV attack [NIST Comment 2006]. The attacker chooses collided IVs in decryption. C1 xor C2 = 1.

1.6 NSEC5: Provably Preventing DNSSEC Zone Enumeration: Goldberg

Been working on NSEC5 for several years, but many implementation results are new.

I ask my resolver for `a.com`, which goes to a server that's authoritative for `.com`. DNSSEC deals with the exchange between the resolver and server. One problem is authenticated denial of existence. Note that legacy DNS had no zone enumeration attack provided configured correctly.

NSEC (RFC4304) has a public zone signing key. Sign every consecutive (lexicographically) pair (off-line, and can keep the key away from the name server). Hence `(p.com,r.com)` proves that `q.com` doesn't exist, but it also leaks the existence of the others, and hence allows zone enumeration. Only needs n queries for n names. Problems for DDOS etc.

Hence NSEC3. We SHA-1 hash the names, and sort the hashes lexicographically, then do the same. Attacked can find all n legal hash values in n queries. Then does an offline dictionary attack.

Online signing stops offline key enumerations. We give the key to the name server. Then when a query comes in that hashes to h, i generate a denial by hashing $h-1, h+1$. Has a paper that proves you need online cryptography (at least public key signing). [NDSS2015]

NSEC5 replaces the hash function H with a verifiable Random Function (VRF) $H(\Pi(\ldots))$, We "hash" all names with the VRF. Sort results and sign pairs with secret key. In previous version, the VRF has a deterministic RSA signature s Π. use [FranklinZhang2013], which we proved was a VRF. The VRF has a public key (learned by usual DNSSEC mechanisms). Returns the NSEC5 record (hashed pair bracketing the hash) and the Π value, which can be verified because it's a VRF. The querier can therefore check this, but can't reverse engineer the hashes of the bracketing hashes.

But how do we defeat the attacker who has stolen the VRF key. Note that the proof is unique given the query and the VRF key. He doesn't know the secret ZSK, so he can't forge the NSEC5, and there's no covering NSEC5 to

replay, since he has to include the right proof. But there's still online signing (the Π).

New performance results. Uses KnotDNS and Unbound. We has the old RSA(2048) and ECC-256. The VRF proof is 641 bits. [GiebenMekking2012]. 9K lines of code. Use existing OpenSSL. NSEC2/RSA was exactly packet length, NSEC3/ECDSA was OK. NSEC5/RSA was over the MTU, but NSEC5/ECC is 800 bytes. (example used was `paypal.com`).

NSEC3 has no online work. NSEC5/ECC maxes at 64k queries/second[3], whereas NSEC3 and NSEC5/RSA peaks at 20K. (20 cores, 40 threads).

Q What about replays from stale records, back when `a.com` didn't exist?

A That's a standard DNSSEC issue, not ours.

1.7 Cryptographically Securing the Network Time Protocol: Franke

"NTP is not a high-profile network protocol but underpins much, so is a good candidate for scrutiny" — reviewer.

long list, including X.509.

Today's choices are nothing (you are here today), symmetric key (doesn't scale) and Autokey (which is broken!). Shows NTP packet header. There's a Mode, then four timestamps. t_1 origin, t_2 received, t_3 transmit, t_4 destination received. Note that $t_{1\ldots3}$ appear on the wire.$\delta = (t_4 - t_1) - (t_3 - t_2)$ and $\delta/2$ "ought to be" network latency. There's ϵ which is "everything else" (including time of clock calls etc.).

Symmetric authentication is MD5(K||M). There's a dubious story round replay protection. The same key is used in both directions, and there's a history of disastrous implementation flaws.

Note that delaying a packet changes the semantics of the packet exchange, hence crypto for NTP is hard. Can't tell if certificates have expired (checken–egg). NTP scales wierdly. A large number of clients with vey little traffic each, but the sessions are exceptionally long-lived, so we want to hold *no* state/session.

Network Time Security is an IETF effort. Conflict between statelessness and replay detection. Fortunately no mode requires both. Hence different modes are pretty different. In client/server mode, we do TLS handshake on a separate port. There's pair of EAD keys, whihc are treated like cookies, i.e. the client stores them and returns them.

Note that roughtime is a different effort. Big benefit is a publically verifiable prof of misbehaving servers (blockchain-ish). So in an ideal world you'd set your clock this way first, then fne-tune with NTP.

Q You sid this is connected to DNSSEC. Why one-yer signatures?

A Good point.

[3] An order of magnitude more than the number of queries at the root.

Q Is it really too slow to sign the responses?

A Yes.

1.8 Levchin Prize

Honors significant innovations in cryptography who have made an impact in the real world. Levchin is co-founder of Paypal. Last year the winners were Rogaway and MiTLS. This year Joan Daemen: AES and SHA–3; Marlinskipe and Perrin for Signal protocol.

Daemen I want an academic career, so this is very useful for my CV! Note that NIST chose us for AES over professional teams from the US — very open-minded. We are now seeing that permutations are very powerful tool.

Moxie Marlinspike The hardest thing has been understanding what "real" means. Contrasts 1930s fascist leaders accepting applause, while communist leaders applauded as well, as their ideology said that theu were all servants of history. Maybe technology leaders should applaud themselves as well. I once met Zuckerberg, and realised (a) I could kill him (wow!) but (b) it wouldn't matter, or change anything. So get an iPhone and use Facebook/snapchat, as that's the real world! Computers are no longer for computer people. The plan was to develop good tools for us, then convert everyone to be like us, but that's not feasible.

1.9 White-Box Cryptography: Bos

Original framework was "blakc box", but side-channel attacks (power etc.) leads more to a "grey-box" model. But in a "white-box" model we have to consider that the adversary might even be the owner of the box. The original use case was DRM. Recent trend is Host Card Emulation (HCE) used to communicate using NFC. Example, roll-back of payment information on payment cards etc. In 2014 Visa/Mastercard announced their support for HCE. Apparently by 2017 86% of PoS in N. America and 78% in Europe will support NFC. By 2018, 2/3 of all mobile phones will support NFC. [ShamirvanSomeren:Playing "Hide and Seek" withstored keys: finCrypt1999].

[ChowetlSC2002] replace every step by a look-up table: 2^{92} Tb fro AES, but a etwork of smaller tables gives 700kB. In practic ethe white-box component is a small part of the whole package: strong code obfuscation, support for traitor-tracing, mechanism for frequent updating etc. [CollbergEurocrypt2016].

All attempts in academia to build WB implementations have been broken. The attacks were, though, very WB-specific. Needed to identify what was going on, then apply algebaric attack.

Our approach merely needs ot know the algorithm. In practice all you get is a binary blob. So use Intel PIN or Valgrind. We have OpenSource plugins fro these tools.

Q Code obfusctaion?

A e don't ven see the ded code that has been dded.

Q You're not protecting the user: how do your justify it morally?

A There is a real use, protectimng true owners.

1.10 NIST's Post-Quantum Cryptography Project: Peralta

Note that symmetric cryptography is not affected in principle: we may need longer keys. Full transition may take 10+ years, hence we have to start now. There are implications for long-term privacy and security: DNA profiles of ours children etc. Hence NIST has this project, which has been runnig for five years.

- to monitor progress
- The find and standardise quantum-resistnat alternatives
- To ensure transparency
- This is not a competition (and we may standardise more than one alternative)

We hope to build a community consensus. Criteria are not written in stone (we don't know enough). We are currnetly agnostic about what techniques are best. The landscape seems to consist of

Signatures hash, code, lattive, multivariate, ...

PKE lattice, code, multivariate, ...

Key agreement PKE, lattice-based, isogeny-based, ...

Speed actually looks good, key sizes may increase significantly. Some signatures sizes look big. There may be a significant increase in ciphertext size for short plain texts. We therefor eneed industry to do an impact assessment **now**. There's quite significant disagreement over "security levels". Should we specify the number of quantum gates? This would require proposers to be experts in QC as well as PQC, so if you're worried, contact us.

Are we a committee of the NSA? There are eight of us, and NSA has no seat at that table. But we need the community to be watching us, and demanding transparency. There's a demand that future standards should make bad implementations harder. The call is open, closing in November 2017. May take another five-seven years.

Q Stateless ignatures?

A We will accept submissions both with/without state.

Q Industry assessments? What about federal agencies? Wold the Federal Government pay for one?

A I wouldn't trust any (open) Government assessment.

1.11 Cryptographic Suite for Algebraic Lattices CRYSTAL: Lepointe

We plan on submitting to the NIST call. Google combined NewHope with ECDH(X26619) in TLS, and "found noimpediments". In TLS there's a Key Exchange Module. Options might be LWe or RLWE. The point is that RLWE allows smaller messges, especially in reconciliation. Usually $\mathbf{Z}_q[x/(x^n+1)$, but there are other possibilities (x^n-1 (NTRU) or x^p-x-1 (Bernstein)).

Our suite is Crystals. We use module lattices. We want to avoid the NTRY assumption, CCA-secure KEM, no Gaussian sampling, no reconciliation. Simple parameter to increase security. d-dimensional matrices in $\mathbf{Z}_q[x]/(x^{256}+1)$. 256 is the number of bits we want to encrypt. Diaram showingthat Module lattices are half-way between Lattices and Ring lattices. d dials security up linearly, rather than the usual 2^n choice of security level.

Despite having say, 3×3 matrices, this can be generated from one seed, so is not less efficient than RLWE. So only store one seed, and the amount communicated is the same: 3 small slices rather than one slice three times the length. There is efficent mutiplication doing at NTT in 256 dimension. $D = 2/3/4$ givens security levels 98/161/227. [RegevSTOC2005] [LangloisStehléDCC2015].

We use $q = 7681$. Seed is $[0\ldots 256]^{32}$. Needs a polynomial Ψ_4^d with binomial (??). Components are SHAKE128, SHA3-256 and SHA3-512. Our binomial error distribution is the same code as NewHope (but smaller). We are two groups that came together, and still have some sorting out to do before we publish the code.

See also `https://openquantumsafe.org` as our measurement technology. Timing slide showing our scheme to be comparable with NewHope (RLWE), and betterthan BCN15 and NewHope–Frodo. The messges are slightly smaller (but 1088 bytes rather than 1824 for $A\to B$, 1152 rather than 2058 for converse: (JHD: see page 7).

Q–?? This is all work I did earlier: your references are wrong.

A I think not.

Q Generating A: how do you know these matrices aren't the product of small ones.

A If you believe in the security of SHAKE

Q In your performance slide, how do you say that these schemes are "comparable" (security)?

A Using the SVP in a certain blocksize as a measure of security.

1.12 Practical post-quantum key exchange from both ideal and generic lattices: Nikolaenko

Notes that we need *new* public key algorithms, and *longer* other keys/hashes. The porblem is the shared key agreement. In more detail,Authentication, Key agreement and payload encryption. Note that a future quantum computer will not affect past authentications, and won't necessarily break the payload encryption, but *will* be able to break recorded past key agreements, and hence the (recorded) payloads.

In october 2015, UCSB predicted 15 years. Note that NSA invested $80M (2014) and in August 2015 suggested we should move towards PQC. LWE is generally felt to be the best foundation. "Hard to find solutions to linear equations is we add some error", so I give you $A.x + e$, and it's both hard to find x, or even to tell whether a vector is one of these or random. A is $n \times n$, and x can be an n-vector or $n \times m$ matrix. Ring LWE has each row being a cyclic shoft of the row above. Hence we need only send the first row, not the whole, and can use NTT instead of general MV product. LWE is in Frodo, and RingLWE in New Hope.

Recall DH agreement g^{xy} after exchanging g^x, g^y. Here server sends $A \cdot X + E$, client says $Y \cdot A + E'$. Then both have matrices "close" to YAX. The MSB should be the shared key, except that we are working modulo q. Hnece we need a check step. Since the matrix is large, we merely transmit its generator.

NTRU. [Regev2005] introduces LWE. [Piekert2014] improved RingLWE key agreement. (R)LWE has worst/average case reduction theorems, which actually is stronger than, say, factoring. Note that Frodo is "getting rid of rings"!

Modulus q, dimension n abd distribution for erros — we use Guassian since this is what is used the worst/averag etheorem. Hnce Frodo: $q = 2^{15}$, $n = 752$, failure probability $2^{=36}$ and quantum security 130 bits, Table distribution 12 random bits/sample. New Hope is $q = 12289$, $n = 1024$ (has to be a power of 2 to allow NTT), failure probability $2^{=60}$ and quantum security 255 bits: Bonomial distribution wth 32 random bits/sample.

NTRU is 2.1KiB, New Hope 3.9KiB. It is likely (and we recommend) hybrid suits: ECDHE+NewHope (Chrome Canary). The session key is secure if at least one problem is hard. The throughpt for hybrid differs by $\times 1.5$ for small payloads and $\times 1.2$ for 100KiB pages.

1.13 Supersingular Isogeny Diffie-Hellman: Naehrig

First, let's forget about Quantum Computers, and admire the excellence of elliptic curves.

Now let $p = 2^{372} \cdot 3^{239} - 1$, $E/\mathbf{F}_{p^2} : y^2 = x63 + x$, when $\#E/\mathbf{F}_{p^2} = \left(2^{372} \cdot 3^{239}\right)^2$. Hence this fails most standard criteria. There is a large number of cyclic subgroups of maximal 2-power order. $E[2^{372}] \equiv \mathbf{Z}_{3^{239}} \times \mathbf{Z}_{3^{239}}$, and similarly in the 3-part. a finite subgroup correpsonds to a unique (up to isomorphism) curve and isogeny with that kernel. Degree of a separable isogeny is nuber of elemts in its kernel, same as its degree as a rational map.

Supersingular isogeny graphs. Vertiges all isogenous .... One for 2-, one for 3-. Notethat we can only compute large-degree isogenies if they are smooth: say $\phi = \phi_{371} \circ \phi_{370} \circ \cdots \circ \phi_0$. For this SIDH, the hard problem is to find ϕ given $E, \phi(E).\phi_A : E \to E/\langle S \rangle$ We need to expand the public kesy by the imges of the generators. So Alice sends $\phi_A(P_B), \phi_A(Q_B)$. With this, then end up sharing $E/\langle R, S \rangle$. These keys are then 564 bytes. But takes 56ms on his Haswell (currently). The best known attacks are $O(p^{1/4})$ classically and $O(p^{1/6})$ quantum, both via generic claw-finding algorithms.

We can make the key even smaller (and the protocol even slower!). Reprsnet points, not by coordinates, but by scalars w.r.t. a determinaintically-computed torsion basis. Get $\frac{7}{2} \log p$, i.e. 339 bytes.

But, if Alice uses a static key, and Bob computes honestly, but sends

$$\left(E_B, \phi_B(P_A), \phi_B(Q_A) + [2^{371}]\phi_B(??)\right),$$

which lets him compute the LSB of Alice's key, and can recover other bits similarly.

1.14 The Strobe protocol framework: Hamburg

A protocol framework for embedded systems. These want simple protocols. Simple, easy to analyse, and performance shouldn't be terrible. Status compliance is mixed: can use NIST, e.g. he uses cSHKE padding. Best practice says use TLS/IPSEC, but the real world has requirements over state, messge flow, code size requirements, so everyone builds their own custom protocol: a pain to design and analyse. Very messy diagram showing TLS 1.2 hash calls as an example of why it's hard in practice. The modern solution is to hash *everything*. See TLS 1.3, and Blinker (an inspiration for Strobe).

Hence all messages pass through Strobe at least to update the hash. Strobe has operations like "send clear", "receive clear", also initialisations ones, and "Ratchet", which is to rekey the system irreversibly. All described n terms of a few core objecst: cipher/clear to/from etc. (one important byte, always included in hashing). The stae is based ona Spoge construction. The Rate r gets xored with the input block, and the capacity c is kept separate. $(r, c) = F(r \oplus m, c)$ when we are running out of entropy. The goal is that the output of Strobe should look like a random oracle (we'll probably use RO model anyway).

This includes not just the data, but also operation and type of data, [so that punning attacks don't work]. Have various metadata operations, more precise than just "will be used to encrypt": what sort of message, and how long. This is very clear, since we probably won't be encrytping the metadata (though we can if we're worried about traffic anaysis). `https://strobe.sourceforge.io`. Simple callback based IO engine. Curve25519 code may be on indepednent interst as it has a much smaller stack than other implementations. 3.5KB code, 700B stack etc.

Q Patents?

A Keytree I can neither confirmnot deny. Otherwise not specifically patented.

Q If you're not worried about performance, why do you have a cleartext option?

A In case a smartphone, say, needs to look at headers.

1.15 FourQ based cryptography for high performance and low power applications: Longa

RFC7748 "Elliptc Curves for Security" in January 2016. Bernstein's 25519 and Hamburg's Ed448-Goldilocks. This has curve details, generation etc. NIST ECC workshop 2015 Their is real emphasis in performance and side-channel resistance. Want rigidity in the curve generation. We have uniqueness, as it's the only curve at the 128-bt level with all our properties. Speed, in kcycles shows around 2.7× over 25519 on various kinds of processor.

$$E/\mathbf{F}_{p^2} : x^2 + y^2 = 1 + dx^2y^2$$

where $p = 2^{127} - 1$, d is a specific complex number, and there are two endomorphisms: degree 2 **Q**-curve ψ, and CM by order $D = -40$: ϕ. [CostelloLongaAsiacrypt2015]

We have an optimal 4-way scalar decompisition $m \mapsto (a_1, a_2, a_3, a_4)$, which are P, $\phi(P)$, $\psi(P)$ and $\phi(\psi(P))$. There is an encoding of the a_i. Then a sequence of 64 $Q := 2Q \pm P[i]$ does multiscalar multiplication in a way that's extremely regular (presumably side-channel proof, JHD thinks he means).

Compressed keys are 32 bytes, uncompressed 64. Schnorr**Q** is a high-speed high-security signature scheme [CostelloLonga2016]. There's an optional prehashing version.

Forthcoming v3.0 will include various optimised implementations (including ARM, and AVR microcontroller), as well as portable C. Claims safe against timing, cache, eception, invalid curve and small subgroup attacks. Up to a ×10 improvement in time on AVR, also energy. However, there is almost twice as much code (35KiB versus 17KiB for curve 25519).

Four**Q**lib2.0 is integrated into OpenSSL1.1.0, bu tthis has to use OpenSSl's (slower) multiprecision operations. See `https://github.com/bifurcation/fourq`. `http://eprint.iacr.org/2015/565.pdf` and `http://eprint.iacr.org/2016/645.pdf` and `http://eprint.iacr.org/2016/569.pdf`

Chapter 2

5 January 2017

2.1 High-Throughput Secure 3PC for Semi-Honest and Malicious Adversaries - Breaking the Billion-Gate per Second Barrier: Lindell

Examples of hospitakl sharing, where all parties really want to cooperate, but have DP issues. Other examples might involve maliciious hosts. Malicious hosts are much more dificult in terms of computation. Our actual setting is 3party with honest majority (i.e. ≤ 1 dishonest). Intrested in both latency and throughput. Garbled circuit has a constant number of rounds, but poor efficiency. Alternatively secret-sharing for high-throughput setting, and this is our setting.

We have

- one bit per and gate, xor gates are free
- very simplecomputations
- ES-Ni for randomness: 1 computation for 64 AND gates.
- Each core runs 12800 in parallel, using AVX-256 (and possibly 512).

Use three servers with 10Gbps Ethernet. 2 2.3GHZ cppus totlaling 20 cores. Can get 7G and gates/sceond on 20 cores. CPU at 50%.

Active Diretcory example, bu twith passwords split across three servers. We rewrote Lerberos ticket-granting server to work with CTR mode (rather than CBC), and each login requires 32 encryotion. Latency becomes 200ms, nd a single cores supports 3000 logins/second (20 cores, 41,000).

2.1.1 Achieving Malicious Security

Prevent corrupted party form changing values. Generate a huge number of muktiplicatoin triples. Improved compbinatorial analysis. Boung by cahc emisses,

so need a new cache-efficient shuffling method. Our best protocol is 7 bits/and; also 10 bins/and but better online/offlien separation.

Get 1.1G and/second (71% CPU utilisation)

Q Are the malicious protocols just extensions of semi-honest.

A In design terms, we tend to start with semi-honest then build in non-cheating

Q What about very separated servers.

A One can get pretty low ping times between different cloud providers

Q Your measurements were AES.

A Can use idfferent circuits, but you want to be usingthe same circuit all the time.

2.2 Secure Multiparty Computation at Google: Kreuter

2.2.1 MPC: Academic and practice

Academic research concentrates on compiling any polynomial-time functoinality and compiling it. Intellectually nice. Intersted in very strong security models (e.g. protecting on ehonest participant against any number of malicious ones). We *do*use the secuirity definitions and trying to write simulators has spotted problems. Henc eI am *not* saying the academic side is useless.

The semi-honest model does have its uses: rather similar to forward security. parties can be bound by contracts. Privacy is often more important than correctness (but hard to define). The covert model is sometimes good enough.

The real problem is that the acadmeic model doesn't look at cost. Typically Gogle has lots of resources, but the network is the bottleneck. Same is true for consumer devices. Also for consumer devices we are intrested in Joules. Consumer devices may fal, which causes ORAM-model problems with a 'rewrite' cycle. Sibyl attacks are also a problem.

2.2.2 Business-Business MPC

Useful Set Intersection Functionality is really useful. Google has user data (who sees what adverts), and the business has customer data (who buys what). Hence we have an Ads Attrubution protocol. Easy to implement *and* easy to sell (customers, lawyers etc.; also software engineers, who tend to believe this is impossible).

Why not garbled circuits? Standardly: Assume a bit vector representation, so 2 AES/bit. 16bit adders are 512 bytes per operation. A further problem is that our data are very sparse, so the setting is not comparable. We ned an audit mechanism (think covert). Arbitrary lawyer-imposed things.

2.2.3 MPC on mobile/consumer devices

Machine Learning is great, but I can't sk Google to do mine, as that tells them too much. Aggregation is linear. Hence we can ransmit maasked data suchthat the maskings cancel when Google adds them. But naively sharingthese vectors between the parties would be quadratic. Use a PRG and just do key sharing.

But, if a device fails, we need to be able to take their mask away. But the problem is if the device is slow, rather than failing, we reveal their circuit. There's an open question on efficient distributed noise generation.

Q What do you need from the academics?

A Generic protocols without the huge constant factor is what Google wants; the world really wants more development of generic protocols.

Q Debugging/deployment?

A Debugigng with live data is really hard. B2B hs been deployed, the other is "real soon now".

Q How do you prevent Yahoo 2006 (?)

A Differential privacy. We need to say that excluding a small number of inputs means the output is indistinguishable. The differnetial privacy model is good, but we could do with more research here.

Q What do you do if the client sends garbage?

A We can normally tell.

Q You said there were other functions than intersection?

A No specific exmaples, but any non-affine function would be difficult.

2.3 Privacy-Preserving Classication on Deep Neural Network: Phesso

Mlis used inmedicl diagnosis, financial alnalysis etc. needs provacy presenvation. Note that an NN consits of percentrons, each of whcih can only clasify data that are linearly separable. Distibguish trainingphase form classification phase. Extended to ConvolutinalNN (CNN). These can do weighted-sum n the convolution layer. We define PP Clasification as when the sevrer has a trained CNN, and the client can get his data classified wthut the sevrer learning either the input or the output. As well as privacy, we want effciency and accuracy. The problem sithat high multiplicate depth can't be eficiency computed with FHE. ReLU and Max Pooling are issues. Replace Max Pooling with Sum Pooling, and ReLU [Rectified Linear Unit] by $x \mapsto x^2$. Cryptonets haas good performance on MNIST, with 98.95%, and 51K classification/hour.

But Deep NNs really need ReLU, and $x \mapsto x^2$ is only a good approximation over a small interval. Hnec we replace ReLU by a low-degree polynomial and add a rectification layer. We get 97.95%, against Cryptonets 98.95%. For a more comoplex CNN, noprivacy has 99.59%, and we get 99.30% at 17k clasification/hour.

Q What's wrong with "server sends the client the model"? Presumably that you lose the model. But if the client can send arbitrary questions, it can learn the model anyway. So what is the requirement.

A Confused.

Q What were the parameters for the 17k/hour.

A Confused.

2.4 Challenges of E2E Encryption in Facebook Messenger: Millican

Speaker is from Facebook. Messenger works on facebook ids, which can be present on multiple devices, but also a device maybe used by multiple identities. Would like pefect forward security. Client needs to be able to encrypt (excludes basic browsers). WhatsApp, Signal and Viber are all indexed by 'phone number. The vendor always controls the client capability. Messenger doesn't fit this model at all.

Can we completely chnage messenger, which is used by $> 10^9$ people. So what are our choices for E2E encryption? We wnated to use off-the-shellf as far as possible.

- Single or mutliple devices per thread? We went for single: simplifies the engineering and the user explanation.
- Single or mutliple devices per account?
- Which surfaces do we support?
- What end-end protocol? Signal is clearly an industry leader.
- Abuse reporting? We wanted to support at least reporting. Alice sends Bob "pay me or I'll hurt you", which Bob reports, but how do we know Bib is honest? Alice adds a random nonce and an HMAC, we add another, and then we verify Bob's report against our HMAC. Bob uploads (From: Alice, message m, auth tag a, time timestamp, seecret nonce n) from Bob. We recalculatethe franking tag and check against none.
- Attaachments. Stored spearately. We transmit the key and a SHA-256 checksum.
- ow do users verify. Can compare keys,but how many users do?

- Multiple device management. How enrolled, when unenrolled? When should we trust the device.
- Web support really compounds the multi-device problem. There is a trust problem with JavaScript crypto code, unlike apps where you can be reasonably sure that everyone is seeing the same binary.

Q Why add this, when journalists prefer touse Signal wich doesn't collect metadata.

A Protecting metadata wasn't in our model.

Q Does Facebook provide the keys for the attachment, and if so can you be compelled to reveal this?

A I can't answer policy questions. But the reason is to prevent other people from accessing data when the user is logged in.

2.5 Memories for Your Eyes Only: Yung

needs cloud storage to store the video clips from Google glasses etc. We propose cloud self-storage protocol withsecurity against servers and other users. Memories is a newer snapchat offering. Tis sytem will be extended, so we need to design it this, even though we didn't anticipate glasses.

Showed an example of Memories. Ther's a section called "My eyes only", whihc needs a PIN/password. We wanted to support mobile users across multipple devices. We wnat the content not tobe viewable by servres, and we also want to replace servers if necessary. But passwords that are strong enough to resist offline dictionary attacks are unmemorable. Looks impossible,w hcih mean sits cryptographically intersting. We willintroduce *trust domains* and use a multi-party protocol to distribute are reconstruct the key. Hence password-protected secret sharing.But tghis tends to imply PKC. However, we are on top of an existing system: can we leverage this? The initail three-party design (owner, sever, authoried viewed) is done. Needs to have room for extensions. Snaps are encryptoed with a snap key, and this key is encrytped with a master key. Once authenticaed a key is is based on a second factor.

Secret sharing is rarely used commercially (but see nuclear submarines). We have a special 3-out-of-3 application.

1. U logs in
2. U produces content
3. U deposits content, e.g. in My EyesOnly

later

4. User logs in

5. retrieves memories and opens MEO

In more detail

1. U to s: login (first factor)
2. S to U a short-term session certificate and a high entrypoy nonce N which S remembers
3. U inserts PIN P locally
4. ...

Key sharing protocol (see slides).

Q Key stretching on mobile devices?

A Yes, it's a challenge

2.6 DMCA: Stoltz

Digital Millennium Copyright Act and Security Research. Speaker is a lawyer at EFF. Notes that Israelis a country without any equivalent threat.

> No person shall circumvent a technological measure that effectively controls access to a [copyrighted] wrok: 17 U.S.C. ¶1201(1)(A).

Also (3)(A) which mentions encryption.Both criminal and civil penalties. Written with 1998 technology in mind, but written very broadly. Also (b)(1) which has been used against researchers studying the crypto system used. Sklyarov was arrested at DefCon (software on Adobe eBook files). Prof. Edward Felten's research team at SDMI was effectively censored. Also "the 2600 case". The courts were not interested in our freedom of speech points. I think there's been a lot of self-consorship as a result of this. See EFF's "unintended consequences" paper. There is a narrow exemption in (g)(2) for "good faith encryption research", but (A) is pretty constraining. "Lawyers look at text like this and visions of billable hours dance in their heads". There's also a 2017–18 temporary exemption for machines designed for personal use (including voting machines), motorized land vehicles and medical devices as long as they are not intended for patients or patient case (JHD wonders what this means). ¡p¿ We are working (Green [Matthew Green of Johns Hopkins, also Andrew "bunnie" Huang, Alphamax LLC] v DoJ) on a legal challenge to constitutionality of ¶1201. We think ¶1201 interferes with the freedom of ideas (rather than the specific expression of ideas, which is what can be copyrighted). Also that this three year licencing scheme is unconstitutional speech licencing. Note that the export licencing cases ruled that "code is speech".

Q–?? I'd like to apologise fro everyone for the fact that DMCA talks about "encryption" rather than "cryptographic". What about hash functions etc.?

2.7 Lightning Talks

Acknowledgements: Bristol for processing credit cards in pounds and therefore avoiding credit charges, also Columbia and the AV team.

Nico Core Infrastructure Initiative at Linux Foundation. I give money away to support open source security tools.

ACLU I'm their lawyer. We need crypto to support speech privacy. We have job openings.

AWS Added FPGAs, and if you want access talk to me. We also have a research programme. We like people to measure crypto costs in AWS hours. We also fund formal verification of crypto.

?? `crypto@cloudflare.com`

Galois we're hiring, in Portland Oregon. Focus on making critical systems more trustworthy.

?? We have an agile interface. Countermeasures against side-channel attacks.

Marco/Sharemind We have timing modules etc. These (seem to be) for secure multiparty computation.

IC3 Initiative for CryptoCurrency Contracts. 2day retreat in SF 23–24 February. Postdocs in NYC or Ithaca.

Zuco Zeecash is zero-cash protocol. Also a replacement for MDsum.

Mitre Lots of jobs. `D.Morse@mitre.org`. Trying to do IT asset management.

... Lots more job offers.

?? Aimed at self-sovereign identities via blockchain: no-one can take your identity away from you. `sovrin.org`.

NIST As well as PQC, we have a lightweight crypto project. See our report,which also has questions to the community. Please respond if you have a constrained device.See e-mail forum.

ACLU bis Chat apps (which means we've failed at e-mail). We want to do something about e-mail: have a road map. `autocrypt.org`. Needs to persuade MUAs to provide end-to-end encryption.

2.8 Message Encryption: Perrin

This is, of course,the classic application. Before the 20th century, manual ciphers wer used, and higher security = harder to use. Even until 1970s it was still a Key Distribution Centre. Also in the 1980s there was Symmetric Key Infrastructure, so the centre can distribute group keys. By the 1990s were were looking at PKI. This was PKI s a directory, or the PGP model, which actually didn't take off. Why: a lot of effort, and problems over how much one published. 2000s were a decade of disillusionment. 2010s caused a revival in public interest, and mobile messaging apps. Questions about Man-in-the-Middle. See Signal's out-of-band authentication Certificate transparency and CONIKS. Most current technologies are Encrypt-then-authenticate, as opposed to the previous models. This changes the user experience, and also has engineering implications. The EtA model assumes diverse trust, rather than AtE's single root of trust. 2000s also had deniable key agreement, the concept of Ratcheting, which updates forward secrecy after AKE. The 2010s added prekeys, and DH-based key agreement which made it less interactive. Also had symmetric-key ratcheting on top of DH ratcheting.

Also people want multi-party messaging.This may involve servers, and server-side fanout. Also multi-device protocols (treat each device as a party). Still metadata questions — who manages the groups, the mapping of devices to users etc.

Q Should we go straight for multi-party protocols?

A Certainlythere are ring-based protocols. But these tend to be pretty interactive. Also problems when we add authentication.

Q Yesterday there was sponge for automatic ratcheting. Ideas?

A Interesting question.

Q Problems with Signal?

A It's not widely-enough used to be spammed. Facebook has done some interesting things. While metadata are available, we can do spam detection from metadata: harder if that's encrypted.

2.9 A Formal Security Analysis of the Signal Messaging Protocol: Luke

See also Section **??**. We know how it's designed (open source) and how it's being used. But is it secure? We havethe usual question, such as M-i-t-M. We can also check for forward secrecy, etc. These are all good security problems. But it may also acheive post-compromise security: what is this?

2.9.1 post-compromise security

In some sense the opposite of forward security. Can Alice really talk to Bob after Bob's key is compromised. Looks impossible, but is solved by Alice and Bob sharing state.As they dialogue, the root key keeps evolving. Older protocols have no forward secrecy. But TLS 1.3 should have forward secrecy, and this makes the attacker's task harder. But with PCS, the adversary has to keep attacking, otherwise Alice/Bob are generating a new root key.

2.9.2 model

Adapted Bellare-Rogaway. But it's a multi-stage model Signaluses a lot of random numbers. Hence we have a very fine-grained model of the random numbers.

Theorem 1 *Assume GDH, all KDFs are random oracles. Then Signal is a secure multi-stage system.*

Very large proof tree, because of all the key stages. There's much "similarly". But note that

1. This is the abstract protocol, not any implementation
2. The medium-term key is signed by long-term key, and we didn't model this.
3. We only modeled one device/user.

`http://eprints.iacr.org/221`,1013.

Q Claims this is old idea

A Not aware of these

Q Quibble: Forward Secrecy rather than Perfect Forward Secrecy.

A That's the standard term, but I admit I don't

Q See RFC 6189 — "self-healing"

A Interesting.

Q Which theorem-prover?

A Pen-and-paper.

2.10 0-RTT Key Exchange with Full Forward Secrecy: Günther

In practice key exchange can be a bottleneck, in terms of RTTs. Ina 0RTT we can send the data with the key agreement. But there are replay issues (partly unavoidable). There's no forward secrecy (considered unavoidable, but we deny this). [CHK2003] had a hierarchical system with coarse forward secrecy — you can't decrypt messages from earlier epochs. [GM2015] had puncturable forward-secret encryption. This is the genesis of our idea.

Build generically from any HIBKEM. Can replace the involved blend of .... ssume Alice knows pk_B. Bob receives the message, and then punctues out this message, thus obtaining forward secrecy for this message. Encryption is a few milliseocnds,but the decryption can take seconds, and the puncturing a few minutesm becaus eof the expensive delegation mechanism.

Q Shared state? Load balancing.

A Within a time interval, the tree is formed along the ciphertext. Could do load balancing among multiple servers.

Q This seems to rely on the message arriving. What happens if the adversary forces message dropping.

A Correct, you don't get the guarantee in this case.

Q This seems to require forgetting data.

A Out of scope, but this is a common problem over all forward secrecy mechanisms

Q TLS 1.3 decided not to go for 0RTT.

2.11 Towards 5G Authenticated Key-Exchange: the security and privacy of the AKA Protocol: Oneta

Security is normally proved for two-aprty protocols, but in practice, norably in mobile networks, there are other parties involved. We have client, connected by radio to sever, then by a secrure channel to the operator. The phones trust the operators. So AKA has to secure the radio link, even with a semi-trusted server. Operator trusted with everything. Client with it's secret key, sk_C a function of sp_{op} and client state, The server is trusted with nothing. The operator genrates a lot of random data, and various sesison keys. How can the authentication work if the server isn't rused? The sverer is onyused as a proxy. Client has its IMSI, which should be unqiue, but there are also temporary TMSIs, which ought to be unique, but migt not be, so are coupled with locatoindatat to

identify the issuing server. When the server knowsn the IMSI, it gets a batch of authentition vectors from the operator. Then challenge-response, followed by TMSI reallocation. This last is encrypted, but noot authenticated.

[ZF03] a weak partner and a corrut server can alow replay attacks. TMSIs can be traced, but AKA doesn't guarantee privacy anyway.

So what about 5G? It still needs AKA. Butthere's no end-to-end communication in AKA: everything goes through the operator. What we need in the equivalent of the leap from TLS 1.2 to 1.3.

2.12 Is Password InSecurity Inevitable? Cryptographic Enhancements to Password Protocols: Krawczyk

Hard to overstate the importanc eof passwords,but there are problems (to say the least). They are lost/stolen n vast quantities (Yahoo $> 10^9$). Users won't choose many strong passwords,etc. Conisder "mostly blinded DH" [Chaum, FodKaliski, Boyen]. We are faced with a combination of human limitations and dictionary attacks. Offline attacks on server compromise are unaviodable. Of course, strong password stores are a helpful step. But there are often implementation problems. Ideally in a dream password store, an attacker who obtains control of the devide learns nothing (information-theoretic sense) about the master password or stored passwords.

e assume a PRF.

1. user puts pwd into client machine
2. client machine sends this to deivde
3. device returns PRF(K_d,pwd) to client machine.
4. clinet machines prduces rwd (oseudo-random, so dictionary-proof) based on this.
5. user uses rwd with server.

Similarly, but use an Oblivious PRF. Then step 2 is done via OPRF, so device only knows the encrypted form of pwd. Letsteo 2 be $a := H(pwd)^r$, step 3 be $b := a^{K_d}$. Has good performace: one encryption at device an dtwo at client machine. The only offline attack is feasible if both client machine and device are corrupted.

Passowrds stored ona single server are dngerous: want to store parts on n servers, with retrieval from $t+1$. But the server needs to authenticate the user, and this requires a pssword, and the user is going to choose the same one, which gives the same problem. [BJSL2012] Password Proetced Secret Sharing. User contacts $t+1$ servrs using the *same* password and reconstructs thesecret, and an attache dthat has compomises t servers learns nothing. Scheme requires a

total of two exponentiations for the client, independent of t and n, so cheaper than TLS.

X-PAKE. Enhanced Paassword Secureity foe the sngle-server sessing. We force the attacked to run a dictionary attack on server compromise. No precomputation prior to server compromise will help. Server never sees the password in plain text. Reduce/eliminate PKI. Offload hash iterations to the client.

Summary: all three schemes have security models and formal proofs.

2.13 Towards a Theory of Data-Independent Memory Hard Functions

Note ow fast Amazon's FPGA is at SHA-1 say. A memeory Hard functuoi is one whose computationis dominated by memory costs (defeating Amazon's FPGA). sCrypt is one such example. Butthere's a datadependent memory access pattern (hecne side-channel attacks). So we define a DAG G, and $H : \{0,1\}^{2k} \to \{0,1\}^k$. Use the notation of graph pebbling. First guess is Space–Time complexity (time $\times$ max space). However, this doesn't necessarily reflkect overlaps between different computations.

Theorem 2 ([AS15]) *High pebbling complexity implies high amortized memory cost for function.*

The honest party might use the naïve pebbling algorithm. One new pebble/round. If we pebble in topological order without discarding, we have $O(n^2)$ cost.

The vital property turns out to be *depth robustness*. We assume maximum in-degree δ (typically 2).

Definition 1 *A DAG $G = (V, E)$ is (e, d)-reducible if there exists $S \subset V$ with $|S| \leq e$ and depth $(G - s) \leq d$.*

We can attack such a graph is a combination of light phases (using few pebbles for a long time) interkeaved with (at most d) short balloon phases, using many pebbles for a short time. $en + \delta gn + \frac{n}{g}nd + nR + \frac{n}{g}nR$.

Unfortunately, popular graphs such as catena are reducibale. Ballon Hashing and Argon2i are still reducible. Cost $O(n^{1.71})$. Latest Argon has 1.77. Winner of 2015 password hashing computation. In Argon, we have a chain of nodes, with the predecessor of i being $i - 1$ and some random predecessor. [AB2016b]. This works in practice.

But any graph with bounded indegree is at least somewhat depth-reducible. We have a new result that depth-robustness is sufficnet.

Theorem 3 *Let $G = (V, E)$ be (e, d) depth robust. Then $CC(G) \geq ed$.*

Hence Argon2i is $\Omega(n^{1.66})$. We are using [ErdosGrahamSzemeredi1977]: can one do better? There 's a gap between upper and lower bound for Argon2i.

Q Isn't it good enough to be bounded on average?

A But in fact the same results apply for average indegree (questioner uncertain here).

Q

A

2.14 The memory-hardness of Scrypt: Tessaro

My focus is practice-orienedt provable security. Proof are de facto require in cmpetitions like CAESR, bu tnot in Password-Hashing Contest (PHC). Previous talk looked at new algorithms which can be proved secure with existing tools. This talk is about new tools for existing functions. Scrypt is RFC7914 (August 2016). But there's no theoory for data dependent MHFs. The core of Scrypt is ROMIx. Uses a Salsa/20 based "hash function" H. In Phase 1, we apply H n times to get X_n. Then in phase 2 we (apply H, then XOR with some X_i) n times. We want alower bound cumulative memory complexity for evaluatoing. In proofs we model H as a random oracle.

This validates a quote from [Rogaway2016].

2.15 Solving the Cloudflare CAPTCHA

CAPTCHAs embed many assumptions: culture,language, visoin/hering mobility and social class. Cloudflare protects against DDOS attacks, webscraping etc. We have no hints from TOR browsers, so most of our clues are obscured/denied. We do sometimes have to challenge real users, which we'd like not to do.

We use boring old Chaumian RSA. Modern "techniques" would use pairing s— in a broser?.

- Does this deanonanymisation, generate new attacks
- Is there a better way

`https://github.org/cloudflare.challenge-bypass-specification`.

Q I wasn't aware that CouldFlare was scrubbingthe content.

A It's really form we look for.

Q My data showed that there were only two culprits using ToR to hide attacks.

A I have no real knowledge of that.

Q Why not a plug-in that "good" guys could install?

A To make it saf ethere has to be one master key, and that's a challenge.

Chapter 3

6 January 2017

3.1 The physics of building a quantum computer: Jeffrey

Speaker is Google, ex-UCSB. Classically a bit is 0 or 1. a QuBit can be $|0\rangle$, $|1\rangle$ or $|0\rangle + |1\rangle$ etc. Gates a linear operators from $SU(2)$, equivalent to rotation on Bloch sphere. $-$ is very important.The negative sign allows cancellation of amplitudes.

UCSB/google X-mon qubit aluminium coplanar capacitor + josephson junctions. Microwave ($\sim$ 5GHz) for simple qubit rotations. Flux bias $<$ 200MSz for frequency control == two qubit operations. Single layer akminium on silicon (sapphire) fab. Simple fab is good. Extra material/layers cause decoherence. We cool to 10mK. We use a "brute force" approach. Qubit chip to the minimum, no cold classical logic, lots of coax, sophisticated software stack for compiling gates to waveforms, and calibration + characterisation. We need automation.

very good qubits have a bit error rate of 10^{-3}. The "no cloning" theorem prevents replication. Surface Code Error Correction. It only measures parity values. and corrects up to $(d-1)/2$ errors/line.

Schor for a 2048 bitnumber requires 10K logucal qubits for state registers, 250M physical qubits: 10 hours at 500ns/cycle. Current state is 9 qubits with 99.5% accuracy.

Why Google? Wea re interested in quantum simulation. 30% of non-classified DoE supercomputing goes nt quantum simultion.

Q How much money?

A EU's 1M is not enough. Might generate good research.

3.2 Erasing secrets from RAM: Simon

After P has computed, and allegedly erased, what cn we recover? Veryhard, since there are lots of components. Note that a compiler might remove a

`zeromem()` call. But there are other problems. There's no tool when we started. Static Code Analysis will not accoint fro compiler optimisations, register spill, calling conventions etc. Hence Dynamic Code aAnalysis. Weuse Taint Tracking (1=senstive). Tainted inpit sources, and then taint tracking)operators, ut also pointer arithmeti etc.). One-way functions remove taint. `https:github.com/lmrs2/secretgrind`. We actually never found compiler optimisation problems, but rather developer mistakes. I/O APIs caching optimisations (`memmap` etc.) lead to subtle problems. Recursuve functions are prone to leaving sensitive data on the stack.

We think we need compiler support. We have a plugin in Clang/LLVM. Ther is code provided buthe kernel (VDSO) `time()` etc. Also OS code `libc` etc. Signal handlers — where to they store CPy state?. Us eor not of frame pointer. Anways erasing cost an extra 280% if signal-safe, and 80% more if not. Cleaning annotated functions only cost 1%.

Even better, use tht call graph for anntate dfunctions. Computer the amximum stack that can be used.. Erase maxumim stack and all regsiters. It breaks the concept of a shared library (so workds for embedded/statically linked). Also no support for recusrion and cycles in call graph. Also non-dterministic call gaphs aren't supported.

Q C11 `memset` is surely an answer, but only implemented in one compiler.

A This doesn't solve the stack problem etc.

Q Other caches?

A Yes, there's a lot more to be done.

3.3 Direct Anonymous Attestation and TPM 2.0: Getting Provably-Secure Crypto into the Real-World: Lehmann

TPM is the root of trust. The TPM can sign attestrtions. Standard certificates would make all attestations linkable, and reveal the TPM's ID. What we want in DAA (unforgeable, anonymity, unlinkability, non-frameability). The fristversion was in TPM 1.2, Revised TPM 2.0 had elliptic curve and piring based. Over 500 TPMs sold. What is need to make DAA provably secure.

1. security model
2. Provably Secure Cryptographic Protocol
3. Implementation

[BCC2005] didi not output signatres. [CMS2006] was not realisation by any construction. Game-baed efinitions in [BrickellChenLi2009], [Chen2010] extended

with non-frameability. Same unforgeability flaw as previous. So in fact everything so far had a problem.

[CamenischDrijversLehamnn2017a] had a security model in UC framework. So this hits item 1.

TPM geneates and stores secret key. Hoststores membership credential. We thoght of the TPM as a ightweight device, but it's actually only accessible via a weak API. Our protocols were designed to avoid a staic DH oracle,. But a TPM is one, and this reduces the security of a 256bit BN curve from 128 bits to 85 bits. Via Commit/hash/sign query, we have $P^{sk} \leftarrow \cdots$.

We had a first proposal tochange this, butthis destroyed ??

Revised TPM2.0 interfaces without static DH. Re-revised provably secure PRSW/qSDH-DAA. But the next question is whether the TPM-based contributions are unforgeable and anonymous. [ChenLi2013] showed that TPM 2.0 generated SPKs are unforgeable. But [XYZF2014] showed that this was wrong. Simple fix was to add a anonce and hash. Unfortunately this adds a subliminal channel.

Nextsteps including workingto get the flawed ISO standards fixed. Also working with Intel on revision of EPID spec. Continue to work withTCG on revision of TMP2.0 API

Q–NPS We did want ISO before they started, and also others. Advice from the xperts was ignored.

A

3.4 DPA Resistance for Real People: Handschuh

Side0channel affects practically everything. Power consumption, RF emission etc Straightforward RSA has square/multiply cycle, and we can read the sequence from power trace. With AS, we can use differential power analysis. If our key guess is correct, we can spot differences in power sorted by LSB. if incorrect, we will have indistinguishable data. All safeguarding consists of changing the design (e.g. always multiplying), and this may not be allowed. So we can change the protocol instead? Protocols may alow the attaacked unlimited traces witha fixed key. Therefore we sould only apply our key rarely, and keepchanging they key. Hence we build a key tree from our shared secret. Use f_0, f_1 as two distinct one-way hash functions with low (bounded0 leakage implementations. No matter how often the keytree is traversed, no key is involved in more than three computations.

Lekage resistant encryption/decryption (KR). malicious ciphertexts including CCA attacks, Evem if the defective decryption is later detected, we have leaked infomration. dd a "validator" to the header. Encrypt in chunks The mesaage key is specific to the message, and changed every few plain-text blocks. Eaach block $_i$ is encrypted with K_i. Complicated description, but basically a hash of the whole message is used to control the walk down the key tree.

Showed an FPGA-based implementation.

Also working on product texting and Test vector Leakage assessment. We want test methods that are respeatable, precise and less subjective. Currently in the smart card world, people are doing the evaluation route (can a test lab break it) rather than validation. Hencetis shuld be a wiite-box validation. Similar to FIPS CAVP. Test vectors desiged by experts to detect likelyimplementation flaws. Specification controls the keys,da ta and any other senstive parameters. Then measure the leakage, and vert it's below a certain thrshold. Use Welch's t-test. 4.5σ is 99.999% confidnece. In order not to set the σ thresold higher, we run thetests twice. This methodlogy has a strong requirement on vendor doucmentation, includingthose ue cases that should be analysed for side-channel attacks. Time-bounded data collection and analysis. ISO/IEC 19790, 17852 nd 20085 parts 1 and 2.

19790 Security requirements for cryptographic modules. Annex F has approved non-nvasive attack mitigating test metrics. "There are none defined at this time".

17825:2016 testingmetrics ofr the mitigation of non-invaisve attack classes against cryptogrsphic modules. Covers power, EM time and llows for future ones. Level 3 is 6 hours per test, 72 max total acruisitive itme, 10k waveforms (levl 4 is 24/299/100k(.

20085 Still under deveopment Part 1 test tools/techniques.Part 2 test methods.

Q Everythingstarted with a random message number. Current solutions are to add an encrypted counter, but that has problems.

A For this use, a straight counter will do.

Q Your test vectors are still manual: have you looked at automation here?

A These can be generated systematically, but that produces far too many tests. The subtlety comes in the selection.

3.5 What Else is Revealed by Order-Revealing Encryption: Cash

Weaker than order-preserving. The advantage is that allows range queries. The problem with standard analysis is that is assumes the columns etc. are uncorrelated. We also have non-uniform data, e.g. geolocation and time stamps.

3.5.1 Backrounds

ORE is certainly less ecure. There's easy plaintext recovery with chose plaintext attachs. Hence we are really concerned with passive attackers - thouse who steal an encrypted database. There are two flavours: ideal and leaky. An ideal ORE only reveals the order, but needs iO and multilinear maps. Or interactive

protocols. But leaky ORE might tell us more, e.g. leaks some more data,e.g. statistics. There are block cipher constructions here. We want "one wayness" ideas. We can recover the plaintext if the entire comian is encrypted — just sort the cipher text.If the plain text repeats, we can do frequency analysis.

If I have correlation,then the pair of columns may be recoverable even if the columns separately. There is a data set of California road intersections. German mobile phone location history. California was bad (note that we only have a subset of the bounding box). German data was apparently worse (but JHD didn't really follow)

[BCLO2009] E is ROPF-secure if its indistinguishable from a random strinctly increasing function with the same domain/range. There is an efficient attack than given $E(k, x($ outputs x' suchthat $X - x'||2^{m/2+3}$ (i.e. can guess roughly top half bits).

3.5.2 Correlation

3.5.3 Case Study

On real data (California latitudes) we see more leakage than the bove, but not much more. However, the raw attack did not explicitly reveal any point to within 400km, bt an intelligent attack gets many points wothi 0.5km — "reconstruct california.

Q Does correlation affect other property-preserving encryptions?

A Nothing yet, but this is clearly an issue.

Q Your graph shows a sudden jump?

A Last-minute hack

3.6 Breaking Web Applications Built On Top of Encrypted Data: Grubbs

Client-server is great until there's a breach. Standard solution is to encrypt the data first. Example, orangeuser has a secret diary and shaes a document with blue user. But then seachimg requires different search tokens for each document. [PopaZeldovich2013] allows "add user" which allows orange user to convert his search tokens from private document to shared one.

Mylar wants to protectthe confidentiality of the data against full access to the servers. Snapshot passive threat s the weakest attack, then persistent pasive attacks and, strongest (claimed) is active threat model. "Mylar also allows some corrputed users". Agsinst their Chat module, we recover 100% of keyword queeries, and 70% of all encrypted documents.

Metadata is notcovered by the encryption schme, bt depends on the data,a nd may/will leak information. Example (Mylar) Alice and Bob have a Party

chatroom, and Boss shouldn't find out about the party. But the anme of the access-control is plaintext "party", and the boss will laos know who is sinvolved.

If the adversary shares a codument with the victim (or vice versa). Or the adversary corrupts a user sharing a data item. If the orange user makes a search that uses a keyword, then ....

In the hospital setting, once a nurse loses her laptop, all the private data of the documents with whon she has co-treated a patient are compromised..

Used Ubuntu chat logs as stand-in for private data in kChat. Used a 350k word dictionary. Attack dictionary precomputation took 15 minutes. search keyword recovery does not rely on access patterns. If I share a singkle document witha trusted friend who is later hacked, all my private document are compromised. Our attack is much more powerful than the "stored dictionary" attack posited by Mylar.

Conclusion: an active adversary is very powerful here.

Q There is work by IBM, Clumbia etc., that works differently. Is your passive attack prevented by encrytoing the metadata?

A Possibly, but then there's usability questions — would you accept a share you couldn't read the title of?

3.7 Building web applications on top of encrypted data: Popa

The usual problem: web servers can be compromised. Consider Mylar as an example. End-to-end encryption is important aainst apssive attacks, especially passive snapshop attacks. We also want an active attack not tobe able to tamper with client-side code and key distribution. Then we want to tampering withdata and query results. Then there's hidingmetadata (JHD: see previous) and then hdog of access patterns. Fnally, hidingoperatons performed, runtime etc.

ORAM. Garbled FHE etc. exist i theory, bt impractical. Mylar [NSDi2014]. Then Verena [IEEESP2-16], and Opaque [NSDI2017]. In Mylar, the developer specifies whihc fields are sensitive. The broser encrypts these sienlds, and the web server only sees this. But we showed that Django, Ruby on Rails etc. are not compatible with encryption. See paper. Mylar only protects the specified fields. Developer designs a *principla ra[h*.

Elet averageheart rate for Alince bween dates. By feryiftingthis proof, the doctor verifies the source of the dat (her pacemeker),, thatthe data are complete and average computing currently.

How does Verena do better. hash server. [MaziereSasha2002] with no trust assumptions and server or client connectivity, fork attacks cannot be prevented. This hash store must reliably store the hashes of recent data, to ensure freskness. The developer specifies the integrity properies of queries, not data. Each query runs in a trust context, and only members of the trust context can affect values. The server stored authenticated data structures (ADS) [LHKR2010]. We build

a Merkle hash tree. Server can prove that a partial aggregate corresponds to range in the hash tree. Verena compiles the inntegrity policy into a forest of ADS. Merkle hash roots stoed at the hash sverver.

Q I don't believe users can make security decisions. When I was young I did, but not now.

A Agreed, but who else can make these decisions?

Q Doesn't this give users a false confidence?

A We need to disabuse them?

Q Previous paper relied on order (?JHD).

A One could randomise the order.

3.8 PRNG Failures and TLS Vulnerabilities in the Wild: McGrew

Author is CISCO. [RY10] [BöcketalNonceDisrepecting]

Is encryption being used where needed. Are ther active attacks/exploits. Where are there bad certificates/keys. Where is weak crypto being used: either structurlly weak or poorimplementtaions. We are looking at the last block.

For detecting flaws, we use multisession monitoring. Note that observing $O(\sqrt{N})$ states will find collisions. `https://github.com/davidmcgrew/joy` converts PCAP into JSON. A VM snapshot is bsaiclaly a set of bits. Used,in particular, for autoscaling. In can be a volume snapshot (bootable disk), boot latency, but not vulnerable. Whereas a full snapshot has nolatency, but has [RY10] issues.

Sandbox ThreatGRID/Windows 7 is a full snapshot.

Linked clones VMWare linked clones are volume

Docker volume.

"In software, it's turtles all the way down".

TLS overview: field called `random` in initial handshake: client is 4 bytes of time +28; server 32 bytes. Typicallythere's an entropy source feeding a PRNG which feeds both the nonce generator and any crypto component. We see a variety.

Aberrant doesn't conform tospec. Often malware. Fieldsmight be fixed or repeating.

PRNG flaws repeating values, e.g. updates rarely (1ms,say). Duplicates on same IP address.

Multiple full snapshotinstances Duplicates on different IP addresses, maybe time separated.

Active netowrk scan Same hello sent many times.

These do happen, nd we can find them. Our tools helpthe good guys see what's going on. TKS shouldn't use RSA encrypted key transport. [EC]DSA shoulduse deterministic variant (RFC 6979), or stir PRF(message) inot entropy prior to signature. Robust AEAD should be used.

Q One re-use isn't a problem?

A You can cut and paste whenever the authentication key is the same.

3.9 Concerto: A Methodology Towards Reproducible Analyses of TLS Datasets: Levillain

SSL/TLS data collection.`ClientHello`, `ServerHello`, `Certificate`, `ServerHelloDone` messages.

We use full IPv4 scans, domain name scans and passive observation. Then the analysis. Methodology

- Context preparation: NSS certificate
- nswer injection
- analysis

Ther are challenges with data quality. We propose two ciphersuites, we expect either, but cal also get NULL, or `RC4_MD5`. Also some `ServerHello` missing two bytes! In 2016 we see 13% TLS 1.0, rest 1.2. Should get certificates in order, but in fact we see unordered lists, duplicates, or even useless certificates. In fact only 69% are compliant. We do not buildall chains, becaus eof X509v1 certificates generated by appliances. We saw 140,000 similar self-signed certificates. There are also mutually cross-signed CAs.

40% of web servers were still accepting SSL2.0.

Q

A

3.10 Productizing TLS Attacks: The Rupture API: Sarafianou

The victim gets a C&C channeltoan attacker. The attacker can see the length data being passed. The attacker guesses part of teh secret,uses it in reflection, and the response is shorted if the guess is right. Adaptively choosing reflectoins

can lead to full recovery. Thre are issue sof noise, antagonistic compression methods (Huffman) and unrelated static contenton page matching candidates.

Shows a diaramof her system. “Client” is a fi=airly dump piece of JavaScript. “Sniffer” looks at the data and reports ot back.These two have to be on the victim’s network. Ther’s a real time Node.js system which is the core of the attack.

Showed screenshots from an exampleyesterday. Use `nmap` as a victimselection tool. Configure the endpoint URL, secret length, secret alphabet, record structure,and a choice of serial or divide/conquer (faster but weaker).

Q Your example was gmail: does it actually work?

A Doesn’t work currently, basically because of noise.

Chapter 4

Blockchain Session

4.1 Rethinking Internet-Scale Consensus: Shi

Recall Delta's $100M August outage, also NSF outage. Hence we need replication and robustness. Hence 30 years of research in distributed systems. State-machine replication, AKA linearly ordered log or consensus. Requirements are consistency (honest nodes agree) and liveness (a transaction appears "soon"). Then, in 1AB, we had BitCoin and Blockchain — Internet-scale consensus.

People expect "maximum six months" for research results to be used in this community. What happens when you ask 300M people to vote? Only 161M turn up.

A malicious node can be arbitrary: delay/reorder messages. But we assume online honest nodes receive messages quickly (else we deem then offline). Can we achieve consensus when 51% of *active* nodes are honest? All classical protocols break down. Even 99% honest doesn't work. Some classical protocols assume synchronous delivery (problems with sleeping nodes), while asynchronous ones assume sleeping nodes are corrupt.

Community wisdom is that Nakamoto's is robust: after all it has been running for eight years. BitCoin is the "honey badger of money" because of this. But it consumes 1.5GW, 10% of all US solar energy. So what happens if we take away the expensive "proof of work".

At the risk of stating the obvious, Blockchain is a chain of blocks. Suppose Dan pays Elaine ₿50. Then he adds H(previous block, 50 BitCoin, next block). H is a random function. Honest nodes only believe the longest chain. If the transaction is sufficiently deep, Dan can't produce a longer chain denying his spend unless he has sufficient power. So proof of work is vital here.

A Blockchain is a chain of blocks. Suppose Dan pays Elaine ₿50. He computes H(previous block, 50 BitCoin, next block). But H is a random function, so this is hard to compute. Honest nodes only believe the longest chain. So if Dan wants to erase this block, he has to build a longer chain, and he can't if it's sufficiently deep unless he has sufficient power. So proof of work is vital here.

Hence what is going on is really leader election. The problem is that, once elected, the leader can sign many blocks. Hence we need time stamps in blocks to be strictly increasing, and we need honest nodes to reject blocks "in the future". Is this version secure? Actually yes, but this is non-trivial. Our version currently requires a random oracle. This is known as Sleepy consensus.

Banks really want a distributed ledger for interbank settlements. This offers easy reconfiguration, recovery and decentralised administration. We need new theoretical frameworks for this.

Q Time stamps?

A Clock synchrony or ??.

Q Nakamoto has next block, and you only have name?

A It works.

Q Transaction/second?

A Good question. We have an implementation. We are linear rather than Nakamoto's quadratic, and we are typically latency bound.

Q How do you guarantee that honest nodes always have connections to honest nodes.

A Good question (JHD: indeed: what happens if a few dishonest nodes partition the graph?).

4.2 Listening to and Silencing the Whispers of Ripple: Study and Solutions for Privacy in IOweYou Credit Networks: Moreno-Sanchez

Credit network representation of various debts. Find paths in a graph, and decrease credit along them. These matter as they are Sybil-resistant. Introducing nodes is way easier than drawing trust from well-behaved nodes. A misbehaving user's effect is bounded and localised. Examples: Ostra, Bazaar, SumUp.

Ripple is a real-life online payment system. In Ripple the nodes are users, or banks, such as CBW bank. Supports fiat currencies, BitCoin and private currencies. Transmission times are seconds and the fees are tiny. $1M trade volume and real banks use it. Transactions take place along any path with enough credit. Like BitCoin, we have public verifiability. Is privacy a real problem in Ripple? Note that it is possible to produce a Ripple transaction to mirror a BitCoin transfer. So we are linking wallets across different system. We also have hot (pocket cash) and cold (money in bank) wallets. But looking at transactions lets us map this link.

What does privacy mean? An attacker can't determine either the value or the receiver. Definitions in paper. [NDSS2015] defined PrivPay. A server

maintains the CN, and this has privacy challenges. We see minimal trusted hardware and oblivious algorithms. This provides strong privacy guarantees for the first time, and has similar transaction times.

SilentWhispers [NDSS2017]. Links are locally stored by users. Net-flow is all that matters. There is no need for a privacy-invasive ledger of proof of work. Again strong privacy guarantees. Note that BitCoin is 90GB. But these can't be deployed today. See our PathShuffle paper (in preparation). Performing several transactions simultaneously enables privacy-preserving transactions over paths sharing a common node. Similar to CoinJoin in Bit-coin. But Ripple only allows single transactions. PathShuflle is a simple smart contract. Can we do more, even though Ripple doesn't have a script language.

Note also `https://www.stellar.org`.

Q My impression is that ?? don't exist any more.

A Ripple is really dynamic and is working today.

4.3 Cryptography and Protocols in Hyperledger Fabric:Cachin

Blockchain is an egg laying wool/milk/pig! Four features: replicated ledger, cryptography, consensus and business logic.

Cryptography has been a key technology in finance. But the trust model hasn't changed despite all this. BitCoin is a more fundamental change. The promise of Blockchain is to replace trust by technology. See Cachin's "Distributing Trust on the Internet" in 2001.

Consider a state machine with validation conditions. Every transactions moves BitCoin from current owner to next. Validation is the history of past transactions. So a distributed p2p system has to create a ledger. In theory, transactions can be smart contracts.

Nodes prepare blocks. Lottery race, Solves a hard problem, selects a random winner. The bock is executes and mines a coin. Decentralised means permissionless. No central identity. Nakamoto's consensus required proof-of-work. Stability is a trade-off. This differs from consensus among known sets of peers. Problem is $O(n^2)$ communication, and $O(n)$ would still be expensive.

Hence permissioned consortium consensus. See Cachin's 2011 book. Public validation versus private state. So why do we think that Blockchain provides privacy. ZeroCash transfers use zero-knowledge proofs.

4.3.1 Hyperledger Fabric

This is technology development inside Hyperledger project (Linux Foundation). Aimed at enterprise scale consensus mechanisms. `github.com/hyperledger/fabic`. Based on IBM's OpenBlockChain, IBM Zürich heavily involved. Implemented in GO, runs smart contracts ("chaincode") within Docker containers.

Transactions can deploy new chaincode, invoke an operation, read the state. Consensus in the BFT model. Currently has Practical Byzantine Fault Tolerance [CL$^+$99]. There's a membership service which issues membership certificates. Transaction certificates are not linked to enrolment, but can be used for multiple transactions (User-Controlled Linkability: see [EKCGD14]).

Hyperledger v1 (I think he said this is forthcoming) distinguishes nodes into endorsers and consensus nodes. Chaincode specifies the endorsement policy. Consensus nodes order and validate already-validated transactions. Work flow is client produces, submitter peer executes, goes to an endorsing peer, and this sends it to a consensus peer, which orders the chain. "Sole ordered", Apache Kafka, and SBFT — a descendant of PBFT.

Why has this taken 15 years? Chaum[1] is 1980s, proof of work is 1990s. Notes the time lag from RSA to SSL, DH to PGP, or Ethernet to IEEE 802.3. Sometimes it takes a different generation to do the implementation.

Q Any interaction with ??.

A Unclear

4.4 Improving Authenticated Dynamic Dictionaries, with Applications to Cryptocurrencies: Reyzin

Code released yesterday.

So Alice pays ⁄B14 to David. The stateless (syntactic) validation is easy, but the stateful validation has to check that Alice has 14. The dictionary data structure is big and growing. If you serialise BitCoin it's 1.5GB. Even worse in multi-asset block chains. Where do you keep it? Disc is slow has has a DoS (CVE-2013-2293) or in RAM, ruling out small guys. However, we don't want the whole ledger, only Alice's amount. Then all we need is authenticated data stores. Put all the values in a Merkle tree, and put the Merkle root into the Blockchain. Only need the hashes of the Merkle siblings. Verifiers have to check the new root hash, that it matches all the transactions since the previous root hash.

We have a two-party model: provers and verifiers. The differences with prior work is how to structure-rebalance the binary tree. Skip list [PapamanthouTamassia2007]: update and insert are both $1.5H \log N$. where H is the length of a hash. Requires trusted randomness, else the tree is unbalanced (long proofs and a DoS). [MillerHicksKatzShi2015] for any generic data structures. $(H+K)\log N$ where K is the length of the PK, but insertion is $3(H+K)\log N$. We use AVL+ trees, which are $H \log N$. Compares with Etherium trie: we're about 3 times smaller. We also have good compression (buying $\sim \times 2$) for batches. `ia.cr/2016/994`

[1] JHD thinks he means [Cha85].

4.5 Closing

Thanks Columbia, students who helped, sponsors.

Survey Link on website.

Suggestions Prize, invited speakers.

Committee 4 US, 3 UK (only).

Chapter 5

Conclusions

Length of cryptotext does matter, see page 7. Statelessness matters sometimes, see page 7. Question over intended/actual demantics of green bar: page 5 and Chrome's intention.

Blockchain (especially BitCoin) is expensive: see pages 36, 38 and 39.

Bibliography

[Cha85] D. Chaum. Security without Identification: Transaction Systems to Make Big Brother Obsolete. *Comm. ACM*, 28:1030–1044, 1985.

[CL+99] Miguel Castro, Barbara Liskov, et al. Practical Byzantine fault tolerance. In *OSDI*, volume 99, pages 173–186, 1999.

[EKCGD14] A. El Kaafarani, L. Chen, E. Ghadafi, and J. Davenport. Attribute-Based Signatures with User-Controlled Linkability. In *Proceedings Cryptology and Network Security 2014*, pages 256–269, 2014.

[RY10] T. Ristenpart and S. Yilek. When Good Randomness Goes Bad: Virtual Machine Reset Vulnerabilities and Hedging Deployed Cryptography. In *Proceedings ISOC NDSS 2010*, 2010.