Trail of Bits Blog

Crypto 2019 Takeaways

- **POST**
- SEPTEMBER 11, 2019
- 1 COMMENT

This year's IACR Crypto conference was an excellent blend of far-out theory and down-to-earth pragmatism. A major theme throughout the conference was the huge importance of getting basic cryptographic primitives right. Systems ranging from TLS servers and bitcoin wallets to state-of-the-art secure multiparty computation protocols were broken when one small sub-component was either chosen poorly or misused. People need to stop using RSA (https://blog.trailofbits.com/2019/07/08/fuck-rsa/), drop AES-CBC, and make sure they're generating randomness in a cryptographically secure way.

It wasn't all attacks and bad news, though. The ascendance of cryptographic tools for privacy-preserving computation continues apace. Zero-knowledge proofs, secure multiparty computation, and secure messaging systems all had major breakthroughs this year. These areas of cryptography have become efficient enough to transcend purely theoretical interest: Companies large and small, from niche blockchain startups to tech giants like Facebook, are deploying cutting-edge cryptographic protocols. Even the city of Boston has used secure multiparty computation in a study on pay equity (https://eprint.iacr.org/2019/734.pdf).

All of the papers presented at Crypto were remarkable and groundbreaking. From this impressive pool, we've highlighted the papers we believe will have a substantial impact outside of academia in the near future.

Attacks

Breaking OCB2

The Best Paper award went to the *Cryptanalysis of OCB2: Attacks on Authenticity and Confidentiality (https://eprint.iacr.org/2018/1040.pdf)*. This amalgamation of three papers released last fall demonstrates attacks against OCB2, an ISO-standard authenticated encryption scheme, for forging signatures on arbitrary messages and full plaintext recovery. This result is especially shocking because all three versions of OCB have been standardized, and were thought to have airtight security proofs. Fortunately, OCB1 and OCB3 are not vulnerable to this attack because it relies on details specific to how OCB2 applies the XEX* mode of operation.

ECDSA Nonce Bias and Reuse

A well-known weakness in the ECDSA signature scheme is that nonces must be generated uniformly at random; otherwise, an attacker can recover the signer's private key. In *Biased Nonce Sense (https://eprint.iacr.org/2019/023.pdf)*, Breitner and Heninger demonstrate the real-world practicality of these attacks by scraping the Bitcoin and Ethereum blockchains in search of either duplicate or slightly biased nonces. These attacks allowed them to recover the private keys for over 300 Bitcoin accounts and several dozen Ethereum accounts. In most instances, these accounts had a nonzero balance, which indicates that active users of these blockchains are using libraries with bad nonce generation. This result confirms the need to move towards deterministic nonce generation in ECDSA, as specified in RFC6979 (https://tools.ietf.org/html/rfc6979)—or, even better, to stop using ECDSA entirely and use Ed25519 instead.

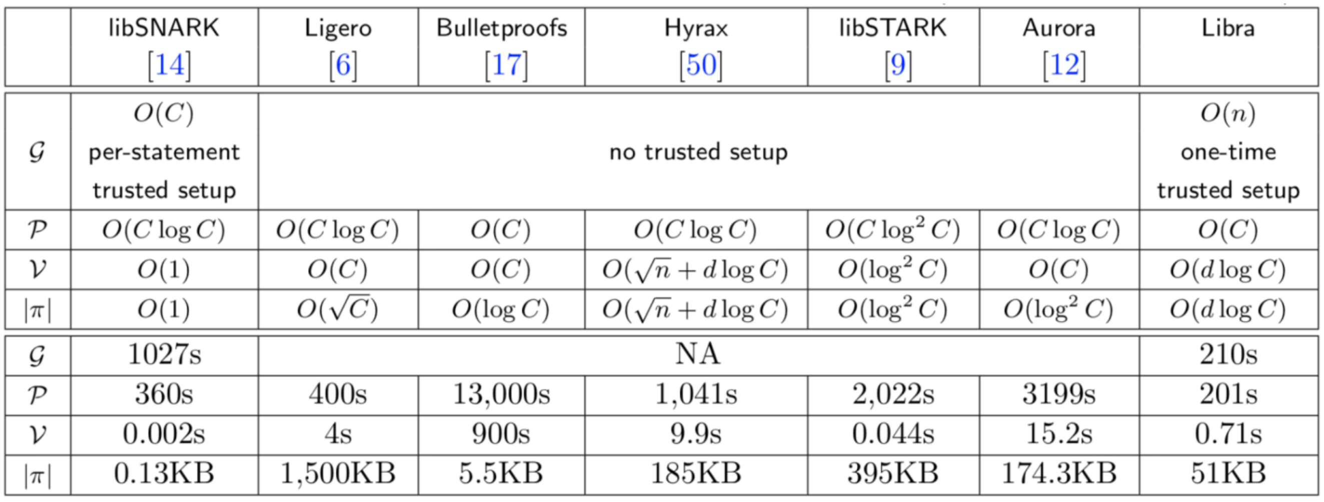
Automating TLS Padding Oracle Attack Discovery and Implementation

Developing tools for vulnerability detection proved to be a popular theme this year. One group of researchers developed a tool that automatically scans (https://www.usenix.org/system/files/sec19-merget.pdf) websites for CBC padding oracle vulnerabilities in their TLS protocol. They found that roughly 1.83% of the Alexa Top Million websites were vulnerable. Matthew Green's team at Johns Hopkins used SAT and SMT solvers to automate the development of new padding oracle attacks (https://eprint.iacr.org/2019/958.pdf) (they actually consider a broader class of attacks called format oracles), which eliminates the laborious task of discovering and writing such attacks by hand. Their tool was able to rediscover common examples of such attacks, including Bleichenbacher's attack on PKCS#1 v1.5.

Secure Computation

Efficient Zero-Knowledge Proofs

This year was huge for pushing zero-knowledge proofs further into the realm of practicality. Among the most impressive results was the development of the Libra scheme (https://eprint.iacr.org/2019/317.pdf) (no relation to the Facebook cryptocurrency). Libra is notable for several reasons. First, it has a pre-processing phase that is only linear in the witness size, not linear in the statement size like SNARKs. Second, it has a prover that runs in linear time with respect to the computation being run in zero-knowledge.



	libSNARK [14]	Ligero [6]	Bulletproofs [17]	Hyrax [50]	libSTARK [9]	Aurora [12]	Libra
$\mathcal{G}$	$O(C)$ per-statement trusted setup	no trusted setup					$O(n)$ one-time trusted setup
$\mathcal{P}$	$O(C \log C)$	$O(C \log C)$	$O(C)$	$O(C \log C)$	$O(C \log^2 C)$	$O(C \log C)$	$O(C)$
$\mathcal{V}$	$O(1)$	$O(C)$	$O(C)$	$O(\sqrt{n} + d \log C)$	$O(\log^2 C)$	$O(C)$	$O(d \log C)$
$\|\pi\|$	$O(1)$	$O(\sqrt{C})$	$O(\log C)$	$O(\sqrt{n} + d \log C)$	$O(\log^2 C)$	$O(\log^2 C)$	$O(d \log C)$
$\mathcal{G}$	1027s	NA					210s
$\mathcal{P}$	360s	400s	13,000s	1,041s	2,022s	3199s	201s
$\mathcal{V}$	0.002s	4s	900s	9.9s	0.044s	15.2s	0.71s
$\|\pi\|$	0.13KB	1,500KB	5.5KB	185KB	395KB	174.3KB	51KB

(https://trailofbits.files.wordpress.com/2019/09/image1.png)
Comparison of zero-knowledge protocols

Only Bulletproofs achieve the same asymptotic efficiency, although they run much slower in practice because they require the prover to perform many expensive cryptographic operations. On the other hand, Bulletproofs have no trusted setup phase and make somewhat more standard cryptographic assumptions. The table above, taken from the Libra paper, shows a comprehensive overview of the most performant zero-knowledge proofs.

Breaking Secure Multiparty Computation

Over in the secure multiparty computation world, Jonathan Katz delivered a keynote talk about a devastating class of vulnerabilities that affects nearly all MPC implementations. The fundamental issue is that these protocols are extremely complex and often leave low-level details up to the implementer. Since MPC protocols are extremely resource intensive, practical implementations often apply optimizations in a haphazard way.

Here's some background: MPC protocols require the computation of many hash functions, and even relatively simple functions like RSA encryption require the computation of tens of millions of hashes in this context. While we often think of SHA3 as rather fast, in extreme settings like this it's actually quite slow and is one of the main bottlenecks of the protocol. This led to researchers using fixed-key AES instead, since it's roughly 50 times faster to compute than SHA3. Originally, this optimization was placed on firm cryptographic ground in the JustGarble (https://eprint.iacr.org/2013/426.pdf) system. However, the security proof does not extend to many modern systems. In fact, Katz et al (https://eprint.iacr.org/2019/074.pdf) showed that using fixed-key AES in most widespread protocols completely undermines the privacy guarantees of MPC. However, they also showed that a simple modification to fixed-key AES was secure and equally performant.

This attack highlights the recklessness of rushing to deploy cutting-edge cryptography. These protocols are often extremely slow and complex, and few people understand the subtle details of the security proof. More work must be done to quantify the concrete security of these protocols as they are actually instantiated, not just asymptotically using idealized functionalities.

Content Moderation and Signatures

Metadata-Private Message Franking

A fundamental problem in end-to-end messaging systems is how content moderation should be handled. Facebook has been particularly concerned with this issue due to the widespread use of WhatsApp and Messenger, so they developed something called a message franking system. This system allows users to report abusive content without allowing Facebook to see the content of all users' messages. Message franking provides the following security guarantees:

- *Message privacy*: Platform should only learn messages that are reported
- *Accountability*: Moderator should always be able to verify that the alleged sender actually sent the reported message
- *Deniability*: Only the moderator should be able to verify the reported message

Unfortunately, prior work on message franking is not metadata private—the moderator is able to see the sender and recipient of every message, even those that aren't reported. This has been remedied in a new scheme (https://eprint.iacr.org/2019/565.pdf) that extends the functionality of message franking to private end-to-end encryption protocols using zero-knowledge proofs. Unlike the zero-knowledge proofs discussed above, the ones used in this franking scheme are extremely efficient and produce signatures that are only around 500 bytes.

Ring Signatures With Deniability

Another popular signature-type primitive that has been especially useful in blockchain systems is the ring signature. Ring signatures allow one member of a ring (i.e., a designated group of people) to anonymously sign messages on behalf of the entire group. For example, a ring could be a workers' union, and a ring signature would allow someone from the union to anonymously make a complaint while verifying that they're actually part of the union. It would be beneficial in some ring signature use cases if individuals could claim responsibility for signing the message. Conversely, it may also be useful for members of a ring to prove that they did not sign a given message. Park and Sealfon (https://eprint.iacr.org/2019/135) formalized these security properties and developed new ring signature constructions that satisfy them.

Post-Quantum Cryptography

As the NIST post-quantum cryptography standardization effort enters its second phase, it's essential to understand the concrete security of proposed cryptosystems. *Quantum cryptanalysis in the RAM model: Claw-finding attacks on SIKE (https://eprint.iacr.org/2019/103.pdf)*, one of the two papers chosen for the Best Young Researcher award, develops a new model for thinking about post-quantum security and applies that model to the SIKE cryptosystem. SIKE is a key encapsulation mechanism that uses supersingular isogenies (https://blog.trailofbits.com/2018/10/22/a-guide-to-post-quantum-cryptography/) to achieve post-quantum security.

One big takeaway of the paper is that SIKE is *more* secure than previously thought. But the paper's formulation of a new way to quantify post-quantum security may have a more enduring impact. This method involves thinking about quantum computers as a classical RAM machine controlling arrays of bits and qubits, then using this model to reason about time/memory tradeoffs in different attack strategies. As a result, the researchers determined that attacks previously thought to be especially potent against SIKE would require so much classical RAM that it would be more efficient to simply run the best known classical attack.

Looking Forward

Navigating the complex landscape of cryptographic protocols and parameter choices continues to be a key point of difficulty for developers. We need to agree as a community that developers should not be responsible for security-critical configuration choices, and move towards building libraries that are misuse resistant, such as Libsodium (https://libsodium.gitbook.io/doc/) and Tink (https://github.com/google/tink). The use of these libraries alone would have prevented many of the attacks on deployed systems we saw this year.

However, we realize that not all systems can realistically support a complete cryptography overhaul, and some are often stuck using a specific library or primitive for legacy reasons. While we saw lots of activity this year around automating vulnerability detection and exploit writing, we'd like to see the community emphasize bug-fixing tools as well. For example, we recently released a tool called Fennec (https://github.com/trailofbits/fennec) that can rewrite functions in compiled binaries (https://blog.trailofbits.com/2019/09/02/rewriting-functions-in-compiled-binaries/), and we used this tool to detect and repair the use of a static IV in AES-CBC without access to source.

On the theory side, we'd like to see a more stringent examination of the concrete security of cutting-edge protocols like zero-knowledge proofs and MPC. As we saw at Crypto this year, implementation details matter, and many of these complex systems are implemented in ways that either render them insecure or dramatically reduce their security level. This is made worse by the fact that many of these new protocols aren't based on standard security assumptions like factoring or the discrete log problem. For example, many blockchain companies are rushing to roll out verifiable delay functions (https://blog.trailofbits.com/2018/10/12/introduction-to-verifiable-delay-functions-vdfs/), which rely on a very new and poorly understood property of imaginary quadratic number fields. We need more thorough analyses of assumptions like this and how they impact security.

Finally, NIST will select the third round of candidates for their post-quantum cryptography (https://blog.trailofbits.com/2018/10/22/a-guide-to-post-quantum-cryptography/) standardization program in 2020. To succeed, NIST will need to rigorously assess the security of the round two candidates. We need more work like the paper on SIKE, which helps us compare classical and quantum security in a more precise way.

By Ben Perez Posted in Conferences (https://blog.trailofbits.com/category/conferences/), Cryptography (https://blog.trailofbits.com/category/cryptography/), Paper Review (https://blog.trailofbits.com/category/paper-review/)

One thought on “Crypto 2019 Takeaways”

1. Pingback: Announcing the Crytic $10k Research Prize | Trail of Bits Blog