RESEARCH STATEMENT

Viet Tung Hoang (tvhoang@engr.ucsb.edu)

Overview. My research area is cryptography, with a practice-oriented theme. The cryptography community has, because of historical reasons, evolved to two largely disconnected groups. The theory side targets *provable security*, offering aesthetically pleasing, rigorously secure results with good *asymptotic* efficiency. But those products usually have no customers, because their *concrete* running time is slow, their implementations difficult, and most importantly, the problems they solve often not of genuine interest to practitioners. In practice, developers are willing to use heuristics to achieve good performance and reduce engineering effort. My research thus aims to bridge the gap between the theory and practice of cryptography: I'm interested in delivering products of real utility that are theoretically justified, easy to implement, and even significantly *faster* than heuristic solutions.

My research papers have embodied what I call a *model-centric* approach. A cryptographic model defines what an adversary is allowed to do for its attacks, and when it successfully breaks the scheme. My thesis is that good modeling will provide clean abstraction boundaries for ease of correct use, and eventually lead to performance improvements. "*A problem well-stated is a problem half-solved*", as the adage goes. I stress that getting clean models is only the first step; there is still a big gap towards obtaining good solutions. After one nails down the definitional treatment, one has to use it as a springboard to get new schemes, tighten security analyses, and rev up performance. But the springboard has to come first.

Recognition. The material in my doctoral dissertation [**5, 6, 7**], in which I established a foundation for a 30-year-old central technique in secure distributed computing, has been used in cryptography classes at MIT, Stanford University, UC Berkeley, Aarhus University, University of Maryland College Park, Rutgers State University, and Oregon State University. In 2013, my work on security modeling of cryptographic hash functions [**3**] was invited to Journal of Cryptology as one of the top-ranked papers at CRYPTO, a tier-1 conference in my field. In 2015, I received the Best Paper Honorable Mention at EUROCRYPT, the other tier-1 cryptography venue, for my authenticated-encryption scheme AEZ [**13**], and the Best Student Paper Award at ACM CCS, a tier-1 security venue, for my work on computer-aided designs of authenticated-encryption schemes [**12**].

Research details. I have published 13 papers, 10 of which are at top cryptography and security venues (EUROCRYPT, CRYPTO, ACM CCS, and IEEE Security & Privacy). Here I'll describe some selected research topics.

Design of Authenticated Encryption Schemes

Designing cryptographic schemes is onerous and error-prone. Despite this painstaking process, the products may still have critical vulnerabilities, or the developers may miss opportunities to improve speed or achieve some desirable properties. In [**12**], we investigate how to use programming-language techniques to synthesize secure authenticated-encryption (AE) schemes.[1] We build a system that

[1] Traditionally, encryption schemes only protect privacy of messages, whereas authenticity is the task of a different primitive—message authentication code. Authenticated encryption is a primitive that aims to support both tasks to minimize misuse. It also improves efficiency by a factor of 2, compared to a generic composition of an encryption scheme and a message authentication code. Authenticated encryption is used underneath the HTTPS, IPSec, and SSH protocols to protect Internet communication. It also appears in IEEE P1619.1 standard for disk encryption, WPA2 protocol in WiFi connections, or Netflix's recent MSL protocol for streaming movie.

browses through and analyzes the security of millions of possible designs in a given template. The tool returns thousands of provably secure AE modes—most of those are new—and generates explicit attacks on other schemes. We also add some searching mechanism to shortlist modes that are fully parallelizable, or those that use the blockcipher only in the forward direction (to save chip area in hardware realization). Among the new candidates, we find a few schemes whose speed is on par with OCB, the fastest (but unfortunately patented) mode in the literature, and far faster than GCM and CCM, the current standards.[2]

On the other hand, the computer-aided design approach does have some drawbacks. First, it can only efficiently search for schemes within some simple templates, and thus will miss schemes that deviate from those blueprints. Next, the shortlisting process favors speed, which is rigged against schemes that provide extra protection that the security definition doesn't capture, such as misuse resistance. To fill this gap, in **[13]**, we build AEZ, a fast AE scheme that is robust enough to tolerate several kinds of misuse. Moreover, AEZ has better usability than existing AE schemes. Conventionally, each ciphertext must be 16-byte longer than its corresponding plaintext, but this stretch may make the ciphertext too large for some network protocols, leaving those out of the protective aegis of AE. AEZ instead lets users choose *arbitrary* values for the stretch—even different values in the same session—and gives best possible security of the corresponding stretch values. AEZ's supporting short stretch can also significantly save energy in resource-constrained settings.

We have expended a lot of cryptanalytic care to make AEZ competitive. The implementation runs at about 0.7 cycle/byte, which is comparable with OCB. Moreover, AEZ can reject invalid ciphertexts at 0.3 cycle/byte, making it attractive for network devices to thwart denial-of-service attacks. AEZ is one of the 29 round-two candidates in a NIST-sponsored competition CAESAR, and is distinguished by being the notionally strongest submission.

Format-Preserving Encryption

In 2014 we have witnessed several high-profile security breaches such as Heartbleed and Home Depot, evidencing the importance of encrypting customers' confidential data. However, legacy applications are written expecting some format of the data, for example, 16-decimal-digit numbers for credit cards. Using classical encryption modes (CBC, CTR, GCM) will result in different formats, say 128-bit strings for our credit-card example. In many cases, one can't simply change the legacy software to accommodate the new formats, because it would cause costly disruption of the service provided by those programs. It's therefore desirable to find an encryption method that preserves the format of the plaintexts to facilitate easy deployment in legacy applications. The need was realized and ad hoc solutions conceived as early as 1981, but it took 28 years for the problem to receive a proper definitional treatment [8]. Since then, the problem has received much attention from industry and standards organizations. One possible solution, suggested by [8], is to use *generalized Feistel networks*, but existing analyses are weak, stopping at inadequate security bounds. Morris, Rogaway, and Stegers [18] use the coupling technique in Markov chain analysis to derive a good security bound for a specific generalized Feistel network (known as Thorp shuffling), but their proof works for the binary domain only. The credit-card problem, however, demands a scheme that works for the decimal domain.

In **[15]** we extend the coupling method in [18] to provide a unified framework to obtain strong security bounds for all generalized Feistel networks and arbitrary domains. This implies several alternative solutions for the credit-card problem, giving users a choice of tradeoffs between running time and

[2] GCM and CCM were developed mainly to avoid the patent issue of OCB. Those modes are more complex to implement, yet quite slower. Our work thus gives ways to find simple, fast, unpatented schemes.

security guarantees. The work provides theoretical support for many existing block-cipher designs. Bellare, Rogaway, and Spies [9] supported our analyses and proposed a standard for format-preserving encryption to NIST. This standard is in draft form within NIST and ANSI X9, and has an industrial implementation by Voltage Security Inc.

The solutions based on generalized Feistel networks give reasonably good security bounds, but there is no reason to stop there. In **[14]** we design a new solution, based on a card shuffling method that we call Swap-or-Not. This has about the same running time as the Thorp shuffling, but provides a much better bound. More importantly, Swap-or-Not provides a rare alternative to create a block cipher since the Feistel networks and Substitution-Permutation networks were suggested in the 1970's.

Security Modeling of Cryptographic Hash Functions

Cryptographic hash functions, such as SHA-256, are public, deterministic functions that turn input strings of arbitrary length to short, random-looking output strings. They are one of the most fundamental cryptographic tools, with numerous applications. To justify the security of hash-based schemes, practitioners often rely on the celebrated Random-Oracle Model (ROM). In this model, the hash is assumed to have an (unrealistic) idealized behavior: it returns a random answer for each input. Theoreticians however are concerned about the soundness of this approach, pointing out (contrived) counter-examples of ROM-based security. Due to the ubiquitous use of the ROM, it's important to provide a *rigorous* justification for the security of existing ROM-based solutions. I posit that most criticism to the ROM is probably due to the lack of a realistic definition capturing how hash functions should behave. In **[3]** we propose UCE, a class of notions for (keyed) hash functions, offering provable security for a dozen practical ROM-based protocols, such as secure deduplication[3], deterministic encryption[4], or producing cryptographically strong randomness.[5]

In **[4]** we show how to build provable and practical UCE-secure hash. All current cryptographic hash designs are one-size-fits-all, aiming for a strong hash that is suitable for all applications. As a result, existing hash constructions are sequential, failing to exploit the full power of modern multi-core machines. While a conservative, sequential construction is unavoidable in some settings, this might be an overkill in others. We embrace a different philosophy, giving *two* constructions—one conservative, the other aggressive— for two corresponding notions in the UCE class to tailor to the need of applications. The first design can be easily implemented on top of SHA-256. The resulting construction inherits the strength of SHA-256, but manages to avoid some known security subtlety of the latter, at a small additional cost. The second design is *fully parallelizable*, but suffices for most of the dozen applications above. This results in a hash FastHash that is about 5 times faster than SHA-256 in the sequential setting, and the speedup may reach 24x if parallelism is available. In addition, we achieve the high performance without hurting security: the security of FastHash is reduced to the security of the compression function of SHA-256 itself. As an example for the utility, if one uses FastHash instead of SHA-256 in Bitcasa's secure deduplication protocol, the speed is improved from 22 cycles/byte to 1.1 cycles/byte.

[3] To save space, storage providers, such as DropBox or Bitcasa, perform deduplication in users' files: if two users upload the same file then clouds want to store the file only once. To ensure users' privacy, clouds have to encrypt files, but using conventional encryption methods makes deduplication impossible. A secure deduplication scheme aims to offer users a meaningful privacy protection, but still allows clouds to deduplicate.

[4] Deterministic encryption is used in the SEEED system of SAP, the Always Encrypted SQL server of Microsoft, and the Skyhigh cloud networks to provide searching over encrypted database.

[5] Cryptographic tools often need *high-quality* randomness to ensure security. In many settings, using weak randomness will lead to total security failures. Real-world random number generators such as /dev/random in Linux typically accumulate a long string of weak randomness, and then hash it to obtain a short string of high-quality randomness.

Circuit Garbling

The main goal of my doctoral dissertation is to bring *garbled circuits* (GCs), a central tool in cryptography, under the umbrella of practice-oriented provable security. GCs were originally invented as a solution for secure multiparty computation (MPC): distrustful parties need to collaborate on the combined data, but none wants to share its secret data with the partners. Thirty years after their birth, GCs have since found numerous uses[6] beyond the scope of MPC, but there remained no definition of what GCs were supposed to deliver. Consequently, each time developers implemented an instantiation of GCs, they had to prove security for a particular setting [17, 19], and thus deprived other applications of faster GCs. Moreover, the security bounds were asymptotic, providing no guidance for choosing among practical protocols. On the other hand, since GCs are complex, proving security of protocols containing this tool is a daunting task. Practitioners therefore often chose a specific instantiation of GCs and claimed that it worked for their protocols without any proof. A few papers did justify the security of their protocols, but the proofs were complex and error-prone.

In **[5]** we explain how to view GCs as a cryptographic primitive instead of a technique, give it a definitional treatment, and identify several security notions needed for most of the known applications. This modular approach gives a simple, clean interface between applications and instantiations. Protocol designers can therefore choose a suitable notion of GCs and reduce the goal of the containing protocol to the chosen security guarantee. We give several efficient instantiations of our primitive with good security bounds. Our proofs use the game-based approach, which is easy to verify.

In **[6]** we show that the proofs of two well-known and influential papers [10, 11] are buggy, because the applications need stronger security guarantee than GCs routinely deliver. These bugs are critical and subtle, collapsing the whole proofs. They also affect several other papers [1, 16]. Having a clean abstraction of GCs is the cornerstone for realizing the bugs. We then identify the new security notions needed for GCs to handle the two applications above, and then give constructions meeting these notions.

In **[7]** we give an implementation of GCs with an unprecedented speed: 7.25 ns for evaluating a gate, while the best previous works run in about 2 μs per gate. This underscores our thesis that once GCs are nicely formalized, the theory will lead to better schemes. In addition, we show that some well-known implementations [17, 19] are *insecure*, giving devastating attacks. Again, this gives compelling evidence that the whole field is in need of greater rigor and better abstraction boundaries.

Future Directions

We are living in a time when mass surveillance is widespread, yet conventional cryptographic constructions barely protect us, because there are several unprecedented challenges that those schemes are not designed for. I want to bridge cryptography and system security to improve our privacy in the context of NSA's mass surveillance. I am interested in exploring how to detect Big Brother's subversion of existing software and standards, how to build software robust enough to give us meaningful security if a part of the system is compromised, as well as revising security of important software. Our first project **[2]** studies how to encrypt data properly in the presence of a bad randomness (e.g., if one uses the subverted standard Dual EC in NIST SP 800-90A, or if one happens to use Debian's buggy random number generator). We plan to investigate the security of the other random number generators in NIST SP 800-90A to verify that those standards are free of backdoor, and find possible security and efficiency improvement. We're also working with Nick Matthewson, Tor's co-founder, to improve security of Tor's relay cell encryption.

[6] For example, GCs are used in the recent BlindBox deep-packet inspection [20] to provide meaningful privacy to network users, but still filter traffic that violates the policies.

References

[1] B. Applebaum, Y. Ishai, and E. Kushilevitz. From secrecy to soundness: Efficient verification via secure computation. *ICALP 2010*, pp. 152–163, 2010.

[2] M. Bellare and V. T. Hoang. Resisting randomness subversion: Fast deterministic and hedged public-key encryption in the standard model. *EUROCRYPT 2015*, pp. 627–656, 2015.

[3] M. Bellare, V. T. Hoang, and S. Keelveedhi. Instantiating random oracles via UCEs. *CRYPTO 2013*, pp. 398–415, 2013.

[4] M. Bellare, V. T. Hoang, and S. Keelveedhi. Cryptography from compression functions: The UCE bridge to the ROM. *CRYPTO 2014*, pp. 169–187, 2014.

[5] M. Bellare, V. T. Hoang, and P. Rogaway. Foundations of garbled circuits. *ACM CCS 2012*, pp. 784–796, 2012.

[6] M. Bellare, V. T. Hoang, and P. Rogaway. Adaptively secure garbling with applications to one-time programs and secure outsourcing. *ASIACRYPT 2012*, pp. 784–796, 2012.

[7] M. Bellare, V. T. Hoang, S. Keelveedhi, and P. Rogaway. Efficient garbling from a fixed-key blockcipher. *IEEE Security and Privacy 2013*, pp. 478–492, 2013.

[8] M. Bellare, T. Ristenpart, P. Rogaway, and T. Stegers. Format-preserving encryption. *SAC 2009*, pp. 295–312, 2009.

[9] M. Bellare, P. Rogaway, and T. Spies. The FFX mode of operation for format-preserving encryption. February 2010. Submission to NIST, available from their website.

[10] R. Gennaro, C. Gentry, and B. Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. *CRYPTO 2010*, pp. 465–482, 2010.

[11] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum. One-time programs. *CRYPTO 2008*, pp. 39–56, 2008.

[12] V. T. Hoang, A. Malozemoff, and J. Katz. Automated analysis and synthesis of authenticated encryption schemes, to appear in *CCS 2015*.

[13] V. T. Hoang, Ted Krovetz, and P. Rogaway. Robust authenticated-encryption: AEZ and the problem that it solves. *EUROCRYPT 2015*, pp. 15–44, 2015.

[14] V. T. Hoang, B. Morris, and P. Rogaway. On generalized Feistel networks. *CRYPTO 2010*, pp. 613–630, 2010.

[15] V. T. Hoang and P. Rogaway. An enciphering scheme based on a card shuffle. *CRYPTO 2011*, pp. 1–13, 2012.

[16] K. Järvinen, V. Kolesnikov, A. Sadeghi, and T. Schneider. Garbled circuits for leakage-resilience: hardware implementation and evaluation of one-time programs. *CHES 2010*, pp. 383–397, 2010.

[17] V. Kolesnikov and T. Schneider. Improved garbled circuit: Free XOR gates and applications. *ICALP 2008*, pp. 486–498, 2008.

[18] B. Morris, P. Rogaway, and T. Stegers. How to encipher messages on a small domain: deterministic encryption and the Thorp shuffle. *CRYPTO 2009*, pp. 286–302, 2009.

[19] B. Pinkas, T. Schneider, N. Smart, and S. Williams. Secure two-party computation is practical. *ASIACRYPT* 2009, pp. 250–267, 2009.

[20] J. Sherry, C. Lan, R. Popa, S. Ratnasamy. BlindBox: Deep Packet Inspection over Encrypted Traffic. SIGCOMM 2015, pp.213–226, 2015.