The Tale of Discovering a Side Channel in Secure Message Transmission Systems

Majid Ghaderi[1], Samuel Jero[2], Cristina Nita-Rotaru[3],
Hamed Okhravi[2], and Reihaneh Safavi-Naini[1]

[1] University of Calgary
[2] MIT Lincoln Laboratory
[3] Northeastern University

Abstract. Secure message transmission (SMT) systems provide information theoretic security for point-to-point message transmission in networks that are partially controlled by an adversary. This is the story of a research project that aimed to implement a flavour of SMT protocols that uses "path hopping" with the goal of quantifying the real-life efficiency of the system, and while failing to achieve this initial goal, let to the discovery a side-channel that affects the security of a wide range of SMT implementations.

1 Introduction

Security in encryption systems relies on assumptions about the adversary's capabilities. One-time-pad provides perfect secrecy against an adversary with unlimited computational power, but requires a new, fresh random key for each message, where the key length is lower bounded by the message entropy. Computationally secure encryption systems use short keys but assume the adversary's computational power is bounded. Aaron Wyner [12] pioneered the study of wiretap channels for securing message transmission where security relies on *physical layer assumptions*. In the wiretap model, the adversary is computationally unbounded and there is no shared secret key. The adversary however has a "noisy" view of the communication channel. Wyner showed that as long as the adversary has a "noisier" view of the communication channel compared to the receiver, one can use randomized coding to achieve (asymptotically) perfect secrecy. Wyner's wiretap model initiated a long line of research in the theory community as well as implementations of wiretap channels in practice.

Dolev, Dwork, Waarts, and Yung [1] used physical layer assumptions in networks to provide security in message transmission. They modeled the network between the sender

DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited. This material is based upon work supported by the Under Secretary of Defense for Research and Engineering under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Under Secretary of Defense for Research and Engineering.

and the receiver as a set of node-disjoint paths, called *wires*, and showed that perfectly secure and reliable transmission is possible if one can assume that the adversary has access to a (proper) subset of all paths. The maximum size of the subset of wires that is accessible to the adversary depends on the adversary type: passive, active, or blocking. The model was motivated by providing perfect security for message transmission in communication networks without using complexity-theoretic assumptions and led to a large body of followup works in the theoretical community. It also inspired the use of multiple path message transmission systems for security in networking community.

Consider a setting where Alice is connected to Bob by a set of n wires, up to t of which can be accessed by the adversary.

SMT systems. SMT protocols are interactive protocols in synchronous networks, consisting of rounds where each round include "phases" where in a phase one party sends a transmission to the other. Let Π be a message transmission protocol, M^A denote the message that is selected by Alice, and M^B denote the message outputted by Bob when Π is completed. Security of SMT systems is defined using two properties of privacy and reliability [2].

Privacy. Π is called ϵ-private if for any two messages m_0, m_1, and for any coin tosses r of the adversary, we have $\Sigma_c|\Pr[View_{\mathcal{A}}(m_0, r) = c] - \Pr[View_{\mathcal{A}}(m_1, r) = c]| \leq 2\epsilon$, where $View_{\mathcal{A}}(m_i, r)$ is the adversary's view when $m^A = m_i, i = 0, 1$, the probabilities are taken over the coin flips of the honest parties, and the sum is over all adversary's view.

Reliability. Π is called δ-reliable if, $\Pr(M^A \neq M^B) \leq \delta$, where the probability is taken over the choices of M^A and the coin flips of all parties.

Security. Π is called (ϵ, δ)-secure, if it is ϵ-private and δ-reliable.

It is proved that reliability requires $n \geq 2t + 1$, and perfect security and reliability require at least two phases.

1.1 SMT with Passive Adversary

Using SMT protocols in practice rules out most of the elegant optimal constructions of SMT systems because of their need for high connectivity (e.g. $n \geq 3t + 1$ for 1-phase protocols with perfect secrecy and reliability), the need for interaction (e.g. at least two phases to achieve perfect secrecy and reliability when $n \geq 2t + 1$), and processing cost (encoding and decoding in each phase) of parties. Considering passive adversaries, however, significantly improves the efficiency of the protocol. It is also well-motivated by protection against silent APT adversaries that stay dormant in the network with the goal of collecting information for multistage attacks. For passive adversaries, security can be achieved with an efficient 1-phase protocol when up to $n - 1$ wires (out of the total of n wires) are eavesdropped. A simple protocol for this setting is to encode the message m using an (n, n) secret sharing to generate a vector of n shares, $SMT.enc(m) = SS_{n,n}.Share(m) = (S_1, \cdots S_n)$, and send the share S_i over the wire w_i. Here, $SMT.enc(m)$ is the message encoding function of the SMT protocol, and the $SS_{n,n} \cdot Share$ is the share generation algorithm of a perfect (n, n) secret sharing.

The *information rate* of this SMT system, defined by the number of bits that must be transmitted for each message bit, is $1/n$.

1.2 A Moving Target Defence (MTD) Approach to SMT

To improve the information rate of SMT systems while providing security against a dynamic adversary that changes the set of eavesdropped paths over time, a Moving Target Defence (MTD) approach to SMT systems was proposed in [10].

MTD systems [7] use randomization and dynamism in the system to improve security. By randomly moving the attacker's target in consecutive time intervals, the attacker's window of opportunity is reduced, and because of the randomization, their effort in one time interval would (partially) lose its value in the next time interval. The MTD system in [10] considered transmission of a sequence of messages that are sent one per time interval. For each message, a random subset G of k target wires are chosen, and $SMT.enc(m) = SS_{k,k}.Share(m)$ is used to generate k shares that will be sent over the selected target wires. The subset G will be refreshed for each message, and as long as the elements of G are not all within the set of adversary's accessible wires, the message will remain perfectly private. This allows security for the message stream as long as the adversary can access a subset of less than n wires (although depending on the value of k some messages in the sequence may be captured).

The analysis of the system, following the theoretical framework of [10], modeled the MTD system as a stateful game between a defender and an attacker that use probabilistic strategies. The game changes state after a pair of "action" taken by the defender and the attacker, and the new state is determined by the previous state and the actions of the two players. The game has $k+1$ states, labeled by $i \in \{0, 1, \cdots, k\}$, where state i corresponds to the adversary successfully finding i target wires. State transitions characterize a Markov chain, with state k being the winning state in which the adversary learns all the target wires that carry shares of a message, and so the message is compromised.

The concrete analysis considered a defender that transmits one message in each time interval, and *acts* with probability λ, in which case they modify the set of target wires that are used in the previous time interval by randomly choosing one of the k target wires, and replacing it with another randomly selected wire from the remaining $n-k$ wires. The adversary also acts with probability μ in each time interval, in which case they randomly select one of the wires from those that they have not learnt yet. The probability μ depends on λ and a second parameter system τ that is the probability of being detected because of the move.

Security. The system evaluation uses two security measures, (i) *the expected number of times that the adversary wins in the first T time intervals,* given by $T.\pi(k)$ where $\pi(k)$ is the Markov chain stationary probability of state k, and (ii) *Expected number of time intervals until the first compromise happens*, denoted by $E^{(1)}_{win}$. These measures can be computed using transition probability matrix of the Markov chain, that is obtained using the above random strategies of the defender and the attacker.

Graphs of the numerical values of $\pi(k)$ and $E^{(1)}_{win}$ (Figures 3 and 4,[10]) for typical choices of n, k, and other system parameters showed that increasing λ resulted in the decrease of $\pi(k)$ and the increase of $E^{(1)}_{win}$, and thus higher security. This matches the intuition that faster changes in the system will "confuse" the adversary more and so "increase" security. The analysis, however, abstracts out the concrete length of hopping interval and allows the hopping probability to be close to one. That is, it does not consider

any cost for path-hopping. To find the concrete security of the system, however, one needs to estimate this cost.

2 Understanding the Cost of MTD Systems in Practice

To implement the system in practice, one needs to map the theoretical model of "wires" and the notion of "compromised wires" to real-life systems. A "wire" abstracts a network path. A path is a sequence of links that are joined by switches. Depending on the networking technology, a received packet on an incoming link is processed and forwarded over an outgoing link according to a routing protocol. Node-disjointness of paths requires that for each switch only one incoming link and one outgoing link to be used for transmission of the message. Although eavesdropping a path can happen at a switch or on a link, we considered the more probable case of switches being compromised.

Path hopping will require route planning, communication of the required information to switches along the paths, and sufficient time for switches to activate the new paths and packets on existing paths to reach their destination. In particular correct decoding of a (k, k) secret sharing based encoded message requires all shares to be received by the receiver. To determine reasonable values of λ, we had to implement the system to understand the costs. Our research question was: *What is the (delay) cost of hopping paths in networks, and what is the highest hopping rate that can be used while message recovery stays at an acceptable rate?*

2.1 Using Software Defined Networking (SDN) to Implement the System

In order to quantify the delay cost of path hopping, we needed to implement a network transport protocol based on path hopping and secret sharing and deploy it in a small-scale network. Using source routing was a logical choice to implement path hopping, as in source routing, the sender computes the entire path to reach a destination, and thus can change the path for each packet on the fly as needed in path hopping. Source routing, however, is not widely supported in practice for a variety of reasons including the additional overhead of encoding path information in each packet header.

We used SDN [4] to implement and study our research question. SDN is a paradigm that separates the network into a *control plane* that is concerned with determining how the network should forward data packets, and a *data plane* that actually forwards the data. The data plane uses programmable switches that are able to match on a variety of packet fields and perform basic forwarding, while the control plane is implemented through a logically centralized SDN controller that coordinates and controls these switches. The SDN controller provides a framework for defining, enacting, and enforcing per-flow policies in a dynamic manner. SDN is widely adopted in practice in both wide area [3] and datacenter networks [11], and SDN-capable switches are commercially available from many vendors [8], making it relatively easy to build an SDN network testbed to experiment with our path hopping protocol.

Our first implementation (2018 and 2019) used the Mininet network emulation platform. In this implementation we assumed the sender and the receiver share the seed of a pseudorandom generator that allowed them to share the set of wires that were used

for transmission. Our experiments with this emulation, however, indicated the need for a real testbed. This was because in this emulated environment the required time for path switching was negligible and reliable measurement of switching delays was not possible. In the following six months, we purchased 2 multiport SDN switches, learned how to configure and program them, and used them to set up a small 20 node network to estimate the cost of path hoping. However, we soon realized that measuring the hopping cost accurately was not straightforward. The reason is that after the SDN controller sends control commands to switches, it does not receive any feedback from switches to know when their reconfiguration is complete. So, one must actively probe switches to find out if they have already updated their configuration, which does not produce accurate results, as it takes some time to probe each switch. In our testbed implementation, we eliminated the need for the shared seed for the pseudorandom gerenrator that was used in the prior Mininet implementation, instead using a networking technique (i.e. label switching) that allowed the receiver to efficiently receive on all wires. This meant we no longer needed a shared secret random string to bootstrap the system.

Unfortunately, our experiments were still unable to estimate the delay costs. This was because we had refined our implementation and by using "proactive" routing could effectively make the path hopping cost equivalent to the switching time of a packet. In proactive routing, all potential paths between a source and a destination node are pre-installed in switches when the network starts. Then, to hop paths, the source node simply chooses a different path from the already established paths to the receiver. In our implementation, we tagged each path with an index. Then each switch was configured to forward packets based on the combination of destination IP address and packet hopping tag.

Modern switches could perform the required path switching in negligible time. This was also in part because there was no background traffic in the testbed, so switches were effectively dedicated to delivering one message stream.

Failing to achieve our goal of estimating the cost of hopping after two implementations, we re-focused the project to explore other aspects of the real-world implementation of the system. We believed the approach, despite its high transmission cost (information rate $1/k$) compared to computational approaches, had attractive properties that if it was securely and efficiently implemented, could provide a solution to post-quantum secure communication in real-world settings such as establishing a shared secret for AES. Important advantages of the system were: (i) information theoretic security which implied security against an adversary with access to a quantum computer, and no reliance on (new) computational assumptions, and (ii) future proofing, in the sense that an adversary who stored their accessible transcript of communication would not have any advantage in analyzing the stored transcript in future.

We considered the following new goals for a secure implementation. 1) Capture real network behavior in presence of the adversary. This includes lack of perfect synchronization between the sender and the adversary hop. 2) Implement an application over the SMT system using standard SDN hardware that one can buy today, to show feasibility of real life applications.

In the subsequent few months, we implemented an application for transporting arbitrary messages using path-hopping, and wrote a conference paper that reported our

experiments on the application, including the effect of reducing hopping interval on the packet loss and data leaks that would occur because of differences in the exact "action" time of the defender and attacker. We submitted the paper to a leading conference in networking and system security.

The paper was rejected. The main point of the reviews was that the feasibility of a secure implementation of a theoretically proven post-quantum secure protocol in real life was overshadowed by a plethora of detailed implementation comments and even worse, the quantum-safe property was questioned, "I feel that this paper has little to do with quantum-safe secure communication and involving quantum-safety is neither relevant nor helpful." The reviews were discouraging to the extent that we concluded the paper had no chance of being accepted in a leading conference unless we embarked on a major implementation that was very close to a real network, a daunting task.

3 Discovering a Side-channel

We had performed extensive experiments over a period of over two years on Mininet and our physical testbed. Reflecting on these experiments made us realise the importance of a security problem in the implementation that we had observed, but had not pursued because of our main focus at the time.

Our experiments had shown that in transmitting shares of a message, the receiver node must wait to receive all the shares, and this waiting time depends on the length of the longest path that is used for carrying shares. Once the shares stay on the paths for different lengths of time, this gives the attacker the opportunity to access the shares on the longer wires for a longer period of time. This, combined with the hopping ability of the adversary, could provide the opportunity for the adversary to find all shares of a message! The attack would work as long as transmission on paths was not assumed instantaneous and the attacker does not stay static during the transmission time of all shares of a message. We had found a "timing side channel" that could exist in the implementation of *all* multi-path schemes that model the network as a set of n wires, and only put a limit on the number of wires that can be accessed by the adversary without taking into account the realization of paths as a sequence of links. The attack went well beyond path-hopping systems and showed the need for the refinement of the SMT model when used in real networks.

3.1 Network Data Remanence Side-Channel

We called the side-channel, Network Data Remanence (NDR). This was inspired by data remanence side-channels that are defined in the NSA/NCSC Rainbow Series as "the residual physical representation of data that has been in some way erased" [6]. We observed that modeling the network as a set of n paths and assuming the attacker can access up to k paths effectively means that *the attacker has a single chance to capture a packet (a.k.a. a share) on a path.* In real networks, a path consists of multiple links, each with an associated delay. Thus, shares linger in the network, creating a side-channel that can be potentially exploited by an attacker to break the perfect secrecy guarantee

of the secret-sharing based message encoding. Our new research goal became studying this side-channel, and proposing mitigation strategies for it.

We followed this new direction in subsequent months and studied the information leakage of the sequence of messages. We also introduced a new message encoding algorithm that increases the number of shares (i.e. using an $(\alpha k, \alpha k)$ perfect secret sharing scheme where α is a natural number), and sent the shares over multiple time intervals, effectively spreading the message shares over space (wires) and time, and showed the effectiveness of this new encoding scheme in reducing the leakage [9]. The new submission was accepted and presented at NDSS 2021 [9].

4 Reflections

The project took over three years and became a true exploratory work: we started with an initial research goal that failed, then set a second research goal that failed also, but ended up with a surprising discovery that completely changed the outcome of the project.

Our work showed us once again the importance of implementation and experiments in the security evaluation of cryptographic systems. While the goals of estimating the (delay) cost of path hopping and evaluation of path hopping in realistic systems remain open, we showed an implementation side-channel that must be taken into account in the implementation of SMT protocols. Our work has led to the discovery of NDR side channels in other protocols [5].

Implementation of computational and quantum cryptographic systems have led to the discovery of side-channels and ensuing new requirements (e.g. protection against timing channels), as well as spawning new research areas (e.g. device-independent quantum cryptography). Our mitigation strategy against the NDR side-channel is the first step towards protecting against this side channel, and is at the cost of significantly lowering the system information rate. A more systematic treatment of the topic, and more efficient protection schemes, will be interesting future work.

Our implementation underlined the importance of vetting assumptions in real-world settings. In the network model that we considered, one needs to at least guarantee (i) paths are truly disjoint (e.g., not implemented through overlay networks that share nodes and links in lower network layer), and (ii) setting the bound on the adversary's power is supported by secondary evidences (e.g. monitoring the network activities).

Our implementation showed the crucial role of collaboration with researchers from networking and systems. Implementing cryptographic systems that rely on physical assumptions are particularly challenging (compared to the implementation of a computationally secure system) because of possible implementation choices, discipline specific requirements, and evaluation criteria and methods. We learned how to collaborate across the disciplines of cryptography, system security, and networking. The same research question finds different statements, different evaluation criteria, and different approaches to evaluation in these disciplines. In physical security, collaboration is essential, as security is tightly related to the lower level network properties.

References

1. Danny Dolev, Cynthia Dwork, Orli Waarts, and Moti Yung. Perfectly secure message transmission. *Journal of the ACM (JACM)*, 40(1):17–47, 1993.
2. M. Franklin and R. Wright. Secure communication in minimal connectivity models. *Journal of Cryptology volume*, 13(1):9–30, 2000.
3. C. Hong et al. B4 and after: Managing hierarchy, partitioning, and asymmetry for availability and scale in Google's software-defined WAN. In *Proc. SIGCOMM*, 2018.
4. HPE. SDN switches portfolio. accessed July 18, 2022. URL: `https://techlibrary.hpe.com/ie/en/networking/solutions/technology/sdn/portfolio.aspx`.
5. Pushpraj Naik and Urbi Chatterjee. Network data remanence side channel attack on spread, h-spread and reverse aodv. In *Security, Privacy, and Applied Cryptography Engineering*, pages 129–147. Springer, 2022.
6. NSA NCSC. Covert channel analysis of trusted systems (light pink book). *NSA/NCSC-Rainbow Series publications*, 1993.
7. H. Okhravi, T. Hobson, D. Bigelow, and W. Streilein. Finding focus in the blur of moving-target techniques. *Security Privacy, IEEE*, 12(2):16–26, Mar 2014. `doi:10.1109/MSP.2013.137`.
8. Open Networking Foundation. Software-defined networking (SDN) definition. accessed July 18, 2022. URL: `https://opennetworking.org/sdn-definition/`.
9. Leila Rashidi, Daniel Kostecki, Alexander James, Anthony Peterson, Majid Ghaderi, Samuel Jero, Cristina Nita-Rotaru, Hamed Okhravi, and Reihaneh Safavi-Naini. More than a fair share: Network data remanence attacks against secret sharing-based schemes. In *28th Annual Network and Distributed System Security Symposium, NDSS 2021, virtually, February 21-25, 2021*. The Internet Society, 2021.
10. Reihaneh Safavi-Naini, Alireza Poostindouz, and Viliam Lisy. Path hopping: An mtd strategy for quantum-safe communication. In *ACM Workshop on Moving Target Defense*, pages 111–114, 2017.
11. A. Singh et al. Jupiter rising: A decade of Clos topologies and centralized control in Google's datacenter network. In *Proc. SIGCOMM*, 2015.
12. A.D. Wyner. The wire-tap channel. *Bell Systems Technical J.*, 1975.