Stream Splitting Moving Target Defense
Final Report

Utsav Bhatt, Sri Harsha Lenka, Lauren Murphy, David Stenson
The University of Texas at Dallas
{*utsav.bhatt, sxl162530, lauren.murphy1, david.stenson*}*@utdallas.edu*

April 23, 2019

Abstract

This project will implement stream splitting over geographically diverse routes and physical media as a moving target defense against Man in the Middle attacks on confidentiality, integrity and anonymity of Internet communication.

Index terms — Moving Target Defense (MTD), Stream Splitting, Man in the Middle Attack, SCTP, Concurrent Multipath Transfer SCTP (CMT-SCTP)

1 Project Summary

1.1 Overview

The goal of this project is to provide a moving target defense against adversaries controlling a subset of nodes in the Internet. We will implement a program to split streams of encrypted data through several geographically diverse paths and possibly across different physical media. Our implementation will also provide speed and resiliency by balancing the load on each of the paths. However, shifting load off of congested paths conflicts with our desire to spread load over multiple paths. Users will therefore select a diversity quotient to define the balance between the efficiency of load balancing and confidentiality provided by path diversity. Finally, our program will leverage encryption to maintain integrity and confidentiality.

1.2 Team Responsibilities

- Configure and deploy network emulator
- Select a protocol e.g. Multipath TCP (MPTCP), SCTP, etc. to utilize or modify
- Implement stream splitting to thwart man-in-the-middle attack

- Implement load balancing and recovery mechanisms to provide speed and resilience
- Implement encryption to maintain the confidentiality and integrity of streams

1.3 Deliverables

- Proposal with literature review
- Progress report and presentation
- Final report and presentation
- Stream splitting program with documentation

2 Motivation

Although the Internet is publicly accessible, millions of computers, phones, and IoT devices exchange private information over it every second. Maintaining the confidentiality and integrity of these connections is paramount to commercial and national security. By splitting streams of data over a variety of routes, it is difficult for a single party to capture the streams, much less resequence the data. Stream splitting could even be used for international communications: preventing some streams from passing through the same country would prevent that country from intercepting the entire communication.

2.1 Encryption in Tandem

The impact of stream splitting is substantial. It will allow for data to be transferred via encryption that could eventually be broken, but since there is distance between the paths of all the streams, it will be very difficult for an attacker to collect all the data points of a stream. Since we are encrypting both the index and the stream data itself, decrypting and reassembling the stream via brute force methods is a much more difficult task than one would suspect. We have chosen AES encryption since it has a larger block size than DES and a larger allowable key. Brute forcing AES takes $(128)^2/2$ computations on average. If we have two encryptions— one protecting data and the other protecting the index— it should take an average computation of $(128)^2$ in order to resequence the data. While encrypting indices in this fashion also increases the strength of the encryption, we do it to prevent data collection by intermediary nodes.

3 Previous Work

3.1 Moving Target Defenses

Static systems are vulnerable to attacks as attackers can more quickly understand the specific architecture of the service or product as well as the existing mechanisms of defense. Moving target defense (MTD) strategies ensure that the attack surface of a system is dynamic, forcing attackers to expend more time and energy. Existing MTD strategies include the Multiple Operating System Rotational Environment (MORE) and Dynamic Application Rotation Environment (DARE). The former rotates the operating system running on the host. The latter rotates the platform (e.g. Nginx, Apache) a web application runs on. Unlike stream splitting, these MTD strategies are focused on protecting hosts from intrusion or exploitation rather than preserving confidentiality of communication between hosts [1].

3.2 Stream Splitting

Protocols like Multipath TCP (MPTCP), SCTP, Multipath BGP and others have leveraged stream splitting to connect across different physical media while maintaining throughput and reliability. Research suggests that stream splitting should be done across geographically diverse routes as well as different media [2]. Stream splitting can also reduce the load on intrusion detection systems (IDS) [3]. Nonetheless, forcing streams to split over routes with varying throughput can incur overhead. Users should specify a diversity quotient that determines how many different routes should be used. The higher the diversity quotient, the more confidential the connection and the higher the overhead [4].

3.2.1 SCTP

As previously mentioned, there are various protocols for and approaches to stream splitting. The Stream Control Transfer Protocol (SCTP) is one such protocol. Like TCP, SCTP is a transport-layer, connection-oriented transport protocol for unicast which provides reliable and sequential packet delivery with sliding-window style congestion control. SCTP also allows multiple streams per SCTP association for transfer of data from one node to another. A property of interest for our project is that SCTP supports multihomed devices, i.e devices that are reachable via multiple IP addresses. Once two multihomed devices form an association, they share their list of reachable IP addresses. The protocol generates multiple possible connections between the 2 communicating parties from these lists. SCTP has connection detection and fault tolerance: if the current path fails, then it gracefully switches to another path to continue transmission of data [5].

3.2.2 SCTP VS MPTCP

A previous group leveraged Multipath TCP (MPTCP). We believe SCTP to be a better alternative. MPTCP is a collection of multiple TCP collections. SCTP has the same advantages over MPTCP that it has over TCP. Concurrent Multipath Transfer SCTP (CMT-SCTP) is geared toward multihomed devices. It is a natural fit for our project as CMT-SCTP is essentially SCTP's version of TCP's MPTCP. SCTP also offers better congestion control and fault detection. SCTP imperatively supports heartbeat packets and periodically checks the status of all available paths, switching to a better connection if it is available. We also believe the chunk-based packets of SCTP are better suited for transmission and reassembly across multiple paths than the byte-based streams of MPTCP [6].

3.3 Previous Approach of Modifying CMT-SCTP

In an SCTP handshake, a multihomed client and server exchange a list of IP addresses of each of the NICs that could carry a stream. Each participant keeps a list of all possible paths based on these IP addresses and routinely checks their health in case the stream should be diverted from its current path. CMT-SCTP builds on SCTP by splitting the stream through all possible paths, rather than just one. It also performs load balancing on the paths. CMT-SCTP could have therefore provided a suitable foundation for our implementation. However, CMT-SCTP does not ensure that streams are split over geographically diverse routes or a variety of physical media.

3.3.1 Alterations to Protocol

We thought we could ensure geographic diversity by routing different packets through intermediary nodes in different locations. The source would forward a packet to one of several entry nodes, the entry node would forward the packet to one of several exit nodes and the exit node would forward the packet to the destination. We attempted to wrap the source and entry nodes in a symbolic CMT-SCTP client, with the entry nodes acting as if they were the NICs of the source node. Similarly, we thought wrap the destination and exit nodes in a symbolic CMT-SCTP server. Instead of providing the IP addresses of each of its NICs during the handshake, the source would provide the IP addresses of entry nodes and vice-versa for the destination.

3.4 Switch to SCTP Approach

Unfortunately, we encountered some difficulty understanding and modifying the userspace implementation of CMT-SCTP we selected; while its API was well-documented, its internals were understandably not as documented [7]. With the end of the semester approaching, we opted to change our approach so that we might quickly develop a prototype to partially satisfy the requirements. We decided to leverage SCTP rather than modify CMT-SCTP: once the source has selected a collection of appropriate paths through intermediate nodes to the destination, each node on the path connects to the next node with SCTP. The stream of data is split into several streams at the source, forwarded through intermediate nodes, and reassembled at the destination just as it was in our original approach.

4 Plan

4.1 Environment

The Common Open Research Emulator (CORE) will emulate the network during implementation and testing [8]. We have chosen to emulate the network because the emulator allows us to add and configure nodes and their services quickly. We can also easily adjust link bandwidth or drop links entirely to simulate interference or congestion on the link.

4.2 Stream Splitting with Load Balancing

We chose to develop our prototype in Python. We rely on the PySCTP module, which provides us bindings to kernelspace SCTP but presents a similar interface to the Python socket library [9].

4.3 Path Selection

We will add the capability to check the latency and disjointedness of paths in order to select a particular subset as well as to inform the load balancing. We thought to reference [10]. Our rudimentary means of doing so is to check the round-trip time and hops provided by ping with the record route option enabled. We will discuss the multitude of issues with this approach as well as alternatives in the section on issues we encountered.

4.3.1 Ping with RR over Traceroute

We chose ping with the IPv4 Record Route (RR) option over traceroute. One reason was its ability to pick up "hidden" nodes (i.e. those that don't increment TTL). [11] suggests that since many BGP prefixes respond to ping with RR and many are within a nine hop distance of some vantage point, RR can provide very primitive but somewhat useful insight into routes on the Internet. Nonetheless, it is heavily flawed and suitable only for our emulation.

4.4 Resilience

4.4.1 Network Degradation

We will test the program by dropping nodes or decreasing bandwidth of available links. This will allow us to evaluate its functionality in a real network.

4.4.2 Source Check

We will verify that the program can determine that data is coming from valid source. The program will implement a source check on receipt of data. We can test the source check functionality of the program by spoofing the source of a stream and checking whether the program accepted the stream.

4.4.3 Integrity Check

We will test the program's ability to detect that the stream has been modified in transit.

4.5 Encryption

AES is our encryption mechanism of choice. We will double encrypt, i.e. encrypt the data before splitting and after splitting. We will also encrypt the indices for sequencing data as well as the data itself. Only the client should decrypt the message. Each client should have its own unique key.

5 Progress

- Installed / configured CORE
- Developed algorithm for splitting a file into chunks and reassembling it in sequence
- Added functions to get the round-trip time and route between two intermediate nodes from ping with Record Route IPv4 option
- Developed rudimentary algorithm for translating diversity quotient into path selection given route and RTT for each possible path

6 Issues

6.1 Design

6.1.1 Efficiency

Our program should not be computationally expensive. We will therefore transfer much of the burden to the underlying transport protocol. Since its proposal, several extensions and optimization have been added to SCTP. Leveraging SCTP for multipath communication is an active area of research. However, none of the previous work has focused on utilizing multipath communication as a moving target defense. Furthermore, our current approach of maintaining SCTP connections is not as efficient as the CMT-SCTP approach could have been.

6.1.2 Measuring Latency

Ping is not a reliable tool for measuring latency on the Internet. Allowing nodes to be pinged arbitrarily is a well-known threat to security. Furthermore, the Internet routes and prioritizes probes differently from a stream of data flowing over a connection. We identified an alternative to pinging: "synthetic packet-pairs" (SPP). SPP measures RTT passively, meaning it does not use rely on probe packets, nor does it put load on the network. Furthermore, it does not require alteration to infrastructure — in our case, the Internet [12].

6.1.3 Measuring Path Disjointedness

Ping with the record route option is similarly not a reliable tool for identifying the path a packet follows for the same reason as RTT: ICMP messages are often routed differently from messages in an ongoing SCTP connection [12]. We suspect that the packets would likely go through the same autonomous systems, but do not know whether they go through the same routers. Since some Man in the Middle attacks occur on the AS-level, it could be productive to determine whether a path goes through the same ASes (ignoring the AS of the destination) [2]. Finally, ping with record route (and similarly traceroute) takes quite some time, meaning we can't check the route too frequently.

6.1.4 Influence of Diversity Quotient on Selecting Paths and Balancing Load

The translation of the diversity quotient to a configuration of paths is manual and rudimentary. One reason is our lack of substantial metrics i.e. only disjointedness and RTT. We offer the paths with the highest RTT and note their number of overlaps and offer the user a chance to switch to paths with lower RTT but (hopefully) less overlaps. Once the user is satisfied with their geographic diversity and RTT, they should be able to select the load on each path. The algorithm is flawed — unfortunately similar to switching levers.

6.1.5 Scaling to the Internet

Research in multipath communication has shown its promise in simulated environments. Our prototype will attempt to achieve these results in a controlled environment with paths of similar bandwidth and delay. Nonetheless, we should be wary of the challenges of scaling to the Internet. Multipath communication has documented pitfalls when used on dissimilar paths with different access technologies. Some implementations of multipath communication on multihomed devices have demonstrated throughput lower than standard single path communication. Another challenge inherited from previous work on multipath communication is that of shared bottlenecks. In simulations, we operate under the assumption that disjoint paths will exist. We may not be able to guarantee disjointedness. Our algorithm should detect this scenario and alert the user as well as resolve any fairness issue it poses.

6.1.6 Security

We aim to develop a program that leverages stream splitting as a moving target defense against man in the middle attacks. Our primary concern is the aspects of efficiency and geographical diversity. Cryptographic measures to ensure confidentiality, authentication and integrity are secondary concerns for us. We assume that these measures will not add significant overhead, but this assumption could prove incorrect under strain. We may have to add functionality to entry and exit nodes to improve performance. Alternatively, we could add more components to our trusted computing base which wouldn't have to be authenticated.

6.2 Execution

6.2.1 CORE

Our university's program liaison has previous experience with the network emulator and greatly helped us customize and replicate our setup for all members. Unfortunately, we have continued to experience bugs and issues which complicated the implementation. In particular, emulations of complex topologies crashed on guest operating systems on machines with poor computing power. Our teammates who had issues with CORE focused on aspects of the prototype that did not require CORE, e.g. the splitting and reassembly. Nonetheless, we should have brought these issues to our liaison's attention earlier instead of merely coping with them.

References

[1] N. Evans and M. Thompson, "Moving Target Defenses as a Resilience Strategy," in Proceedings of the NATO IST-152 Workshop on Intelligent Autonomous Agents for Cyber Defence and Resilience, Prague, 2017, pp. 90-99.

[2] H. Nguyen, C. Phung, S. Secci, B. Felix and M. Nogueira, "Can MPTCP secure Internet communications from man-in-the-middle attacks?" 2017 13th International Conference on Network and Service Management (CNSM), Tokyo, 2017, pp. 1-7.

[3] J. D. Judd and J. C. McEachen, "An architecture for network stream splitting in support of intrusion detection," in Proceedings of the 2003 Joint Fourth International Conference on Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia, Singapore, 2003, pp. 1717-1721 vol. 3.

[4] N. Evans and M. Thompson, "Stream Splitting Moving Target Defense," 0097784, 2018.

[5] R. Stewart "Stream Control Transmission Protocol (SCTP)," RFC 4960, Sept. 2007

[6] J. R. Iyengar, P. D. Amer and R. Stewart, "Concurrent Multipath Transfer Using SCTP Multihoming Over Independent End-to-End Paths," in IEEE/ACM Transactions on Networking, vol. 14, no. 5, pp. 951-964, Oct. 2006.

[7] B. Penoff, A. Wagner, M. Tuxen, and I. Rungeler, "Portable and Performant Userspace SCTP Stack," 2012 21st International Conference on Computer Communications and Networks (ICCCN), 2012.

[8] Comparison of CORE Network Emulation Platforms, Proceedings of IEEE MILCOM Conference, 2010, pp.864-869.

[9] E. Pftzenreuter and P. Langlois, PySCTP. P1 Security, 2018.

[10] E. Zeitler and T. Risch, "Scalable Splitting of Massive Data Streams," Database Systems for Advanced Applications Lecture Notes in Computer Science, vol. 5982, pp. 184-198, 2010.

[11] B. Goodchild, Y. Chiu, R. Hansen, H. Lua, M. Calder, M. Luckie, W. Lloyd, D. Choffnes and E. Katz-Bassett, "The record route option is an option!", Proceedings of the 2017 Internet Measurement Conference on - IMC '17, 2017.

[12] S. Zander and G. Armitage, "Minimally-intrusive frequent round trip time measurements using Synthetic Packet-Pairs", 38th Annual IEEE Conference on Local Computer Networks, 2013.

7 Biographical Sketches

7.1 Utsav Bhatt

Utsav Bhatt holds a bachelor's degree in information technology and is pursuing his master's degree in computer science with a specialization in information assurance. He is a member of the Software Security Lab at UTD. He has completed relevant coursework in data and application security, network security, information security and language-based security.

7.2 Sri Harsha Lenka

Sri Harsha Lenka holds a bachelor's degree in computer science and is currently pursuing his master's degree in computer science with a specialization in information assurance. He has completed related coursework in network security, IoT security, system security and malware analysis, cryptography, and advanced operating systems. He is preparing for CCNA and has experience with CTFs and network programming. He is interested in network security and binary analysis.

7.3 Lauren Murphy

Lauren Murphy is a fast track student completing her bachelor's degree in software engineering and starting a master's degree in computer science. Both degrees are specialized in information assurance. She interned at the Systems and Software Security Lab at UTD to develop a framework for dynamic analysis and GUI testing of suspected Android malware. She is currently taking classes on network security and digital forensics. Her research interests include network security and malware analysis.

7.4 David Stenson

David Stenson is pursuing his bachelor's and master's degrees in computer science concurrently through the UTD fast track program. He is enrolled in the information assurance focus of the master's program. He has taken classes on cryptography and networks and network security. He is currently enrolled in classes on data and application security and digital forensics. He has experience with database design and has created applications for business use. He is interested in network security, cryptography and quantum computing.