E-Mail Header Injections

An Analysis of the World Wide Web

by

Sai Prashanth Chandramouli

A Thesis Presented in Partial Fulfillment
of the Requirement for the Degree
Master of Science

Approved April 2016 by the
Graduate Supervisory Committee:

Dr. Adam Doupe, Chair
Dr. Gail-Joon Ahn
Dr. Ziming Zhao
\memberThree
\memberFour

ARIZONA STATE UNIVERSITY

May 2016

ABSTRACT

ACKNOWLEDGEMENTS

A project of this size is never easy to complete without the help and support of other people. I would like to take this opportunity to thank some of them.

This thesis would not have been possible without the help and guidance of my thesis advisor, and committee chair - the brilliant Dr. Adam Doupe. This project was his brainchild, and he held my hand through the entire project.

I would like to thank Dr. Gail-Joon Ahn, for being part of the committee, for all his help, and his valuable input on the changes to be made to make the project more impactful.

I would also like to thank Dr. Ziming Zhao, for being a part of my committee, and for the constant motivation.

I would like to thank the members of the SEFCOM, for their support, and would like to especially thank Mike Mabey, for helping set up the infrastructure for this project, and Marthony Taguinod, for helping me document this project.

TABLE OF CONTENTS

Page

LIST OF TABLES v

LIST OF FIGURES vi

CHAPTER

1 Introduction 1

2 Moving Target Defense Background 2

2.1 Introduction to Moving Target Defense 2

2.2 Related Work 2

2.3 Key Challenges to Moving Target Defense 4

2.4 Moving Target Defense Framework 5

3 Web Applications 6

3.1 Web Applications: Individual and Commercial Front Doors 6

3.2 Dissecting Modern Web Applications 8

3.3 Current Security Issues and Considerations 10

REFERENCES 12

APPENDIX

A Raw Data 13

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

3.1 A Modern Web Application Architecture and Its Running Environments. 8

Chapter 1

INTRODUCTION

Franklin *et al.* (2007)

Chapter 2

MOVING TARGET DEFENSE BACKGROUND

2.1 Introduction to Moving Target Defense

Moving Target Defense (MTD) seeks to level the asymmetric environment of attacker and defender by negating any advantages the attacker has. In order to accomplish this goal, MTD seeks to control change across various system dimensions to increase uncertainty, complexity, and cost for attackers **?**. With the assumption that perfect security is difficult to obtain, MTD focuses on enabling resilient, defensible systems that allow continued, safe operation in a compromised environment rather than developing perfectly secure systems. In addition, MTD does not remove vulnerabilities directly, rather, it reduces the attack window by limiting the temporal exposure of vulnerabilities.

Two approaches in creating such a defensible system are: disrupting reconnaissance efforts of the attacker or disrupting on-going attacks. For both MTD approaches, they aim to invalidate any information on the system that an attacker had managed to acquire previously.

2.2 Related Work

The idea and philosophy of MTD, which is to increase uncertainty and complexity for attackers, has been proposed and studied for decades **????**.

Okhravi *et al.* surveyed techniques that applied the philosophy of MTD in different cyber research domains **?**. According to them, existing techniques can be categorized into five classes based on what component to move:

1. Changing the application environment **??**
2. Changing application code dynamically or diversifying software **??**
3. Changing the representation of data **??**
4. Changing the properties of platforms **??**
5. Changing the network configurations **???**

Application environment randomization involves modifying the environment presented to the application by the system at run-time. These techniques modify configuration components such as data and instruction memory locations, heap/stack configuration, and the application's instruction set. Techniques that fall within this category typically prevent injection-attacks that seek to control the application by injecting malicious code or otherwise.

Address Space Layout Randomization (ASLR) **?** and Instruction Set Randomization (ISR) are widely adopted instances of application environment randomization in modern operating systems. Existing ASLR mechanisms randomly arrange the address space positions of key data areas such as the base executable memory location, application stack and heap, and any libraries it requires. (what-to-move) of a process when it is launched (when-to-move), including the base of the executable and the positions of the stack, heap, and libraries.

As a result, if an attacker manages to exploit some memory corruption vulnerability in the application binary, i.e. a buffer overflow attack, it would be difficult for attackers to transfer control flow to their injected code as they will be unable to accurately predict the application's memory layout.

Explain item 2, give example Code diversification - High level vs. Low level

Dynamic application randomization or code diversification involves X, seeks to prevent vulnerability Y, main drawback is Z

Explain item 3, give example Changing the application's data representation involves X, seeks to prevent vulnerability Y, main drawback is Z

Explain item 4, give example Randomizing application's platform property involves X, seeks to prevent vulnerability Y, main drawback is Z

Explain item 5, give example Network configuration randomization involves X, can prevent vulnerability Y, main drawback is Z

Further classification of techniques based on Cyber Kill Chain - Lockheed cyber kill chain Reconnaissance Delivery Exploitation

Static and Dynamic MTD techniques - Static is required to be done before run-time, Dynamic can be done during run-time In addition to the 5 classes of MTD techniques, each class can be further categorized. For instance, MTD mechanisms for programs can be categorized into two classes depending on if a program is running (*dynamic*) or not (*static*) at the time when moving happens. For instance, existing ASLR approaches are static, because the positions of code and data areas are only moved at the launch of a program but not when a program is running. On the other-hand, dynamic MTD techniques offer a wider option of choices for when-to-move, in exchange for being more difficult to implement due to other considerations: i.e. overhead cost and downtime during movement.

2.3 Key Challenges to Moving Target Defense

Provide availability to legitimate users while disrupting attackers

System design must provide the intended security benefits (attack surface must reduced)

Development of secure control system that handles Moving Target Defense system

Scalability of MTD Technique

2.4 Moving Target Defense Framework

Moving Target Defense Techniques (What-To-Move)

Moving Target Defense Approaches (When-To-Move)

Applications of Moving Target Defense

Evaluation of Moving Target Defense Systems (Effectiveness and Cost)

REFERENCES

Franklin, J., A. Perrig, V. Paxson and S. Savage, "An inquiry into the nature and causes of the wealth of internet miscreants.", in "ACM conference on Computer and communications security", pp. 375–388 (2007).

Pietraszek, T. and C. V. Berghe, "Defending against injection attacks through context-sensitive string evaluation", in "Recent Advances in Intrusion Detection", pp. 124–145 (Springer, 2005).

Zanero, S., L. Carettoni and M. Zanchetta, "Automatic detection of web application security flaws", Black Hat Briefings (2005).

APPENDIX A

RAW DATA