Protecting Web Services From Botnet Exploitations

By

Hanh Nguyen VO

A thesis submitted to Macquarie University
for the degree of
Doctor of Philosophy
Department of Computing
31 March 2012

MACQUARIE
UNIVERSITY
FACULTY OF SCIENCE

This thesis is submitted in fulfilment of the requirements of the degree of Doctor of Philosophy at Macquarie University and has not been submitted for a higher degree to any other university or institution. This thesis represents my original work and contributions. I certify that to the best of my knowledge, all sources and assistance received in the preparation of this thesis have been acknowledged.

Hanh Nguyen Vo

Acknowledgements

My sincere and deep gratitude goes to my supervisor, Professor Josef Pieprzyk, for his kind guidance and consistent support. He has guided me towards the right direction, influenced me as an active thinker, and provided me with insightful advice.

I would like to thank Dr Christophe Doche and Associate Professor Paul Watters for their help, encouragement and valuable inputs to my research.

I would particularly like to thank my family. My parents, who went overseas to assist me in difficult times, have inspired me to pursue my study. My brother and sister have never stopped supporting me in my life. Finally, there are no words to express my love and appreciation to my husband and my little baby. They make everything worthwhile.

Abstract

Botnets have attracted a significant attention and have become a primary "platform" for attacks on the Internet. As Web 2.0 evolves, especially with social networking sites becoming a dominant medium, it is no surprise that a new type of malware, Web 2.0 bot, rides on this new means of propagation and communication. For example, Koobface has been successfully leveraged social networking websites such as Facebook and Myspace to infect millions of computers and TwitterBot attempted to use Twitter as a C&C channel for communicating. Since these are legitimate sites and the communication is just like normal web traffic, this type of attack is difficult to detect by the current detection software. In addition, Captcha used for verification can be easily bypassed by modern bots as they employ the so-called relay attack. The relay attack involves a human to solve the challenges for the bots.

In this research, we solve the problem by presenting two systems: an API Verifier and an enhanced Captcha design. Since a bot must use an API (Application Programming Interface) to post information, the API Verifier will challenge a user with a Captcha if it detects that the API call is from a new computer. To recognize if the API call is from a new computer, we use the Media Access Control (MAC) address of the computer, which is globally unique. Since a bot cannot solve a Captcha challenge, it will not be able to make API calls to the Web service. Our enhanced Captcha is resistant against the relay attack as we change the static answer, which is used by majority of Captcha forms, to a dynamic one by asking the user *"where"* is the answer rather than *"what"*.

We also employ animation with a defined delay time to prevent the human solver from telling the bot where the answer is. The systems are evaluated by operating against various attacks from a modified version of Koobface, the most popular existing Web 2.0 botnet. The results show that Koobface bots fail to break our enhanced Captcha using relay attacks. Also our API Verifier successfully detects requests from Koobface bots and denies their access.

Contents

iii

Acknowledgements v

Abstract vii

List of Figures xv

List of Tables xix

1 Introduction 1

1.1 Research Challenges and Scope 6

1.2 Our Research Goals 9

1.3 Bot 2.0 Modus Operandi and Solution Overview 10

1.4 Thesis Contribution and Organisation 14

2 Botnet 17

2.1 Botnet Overview 18

2.1.1 Botnet Definition 18

2.1.2 The Botnet Life Cycle 19

2.2 Botnet Evolution 20

2.3 Bot characteristics 28

2.3.1 Definition of Malware and their Propagation Methods 29

2.3.2 Malicious Activities 31
2.3.3 Command and Control 35
2.3.4 Concealment 35
2.4 Botnet Architectures 37
2.4.1 Centralized Architecture 38
2.4.2 Decentralized Architecture 42
2.4.3 Hybrid Architecture 44
2.4.4 Random Architecture 47
2.5 Botnet Rallying Mechanism 48
2.5.1 Hard-coded IP address 48
2.5.2 Dynamic DNS Domain Name 49
2.5.3 Distributed DNS service 49
2.6 Botnet Communication Protocols 49
2.6.1 Botnets with C&C based on IRC protocol 50
2.6.2 Botnets with C&C based on HTTP Protocol 54
2.6.3 Botnets with C&C based on P2P Protocol 57
2.7 Evasion Tactics 61
2.7.1 Hiding presence techniques 61
2.7.2 Hiding Traces of Bot Communication 62
2.8 Summary 64

3 Web 2.0 Botnet and Current Botnet Detection Techniques 67

3.1 Web 2.0 botnet 68
3.1.1 Web 2.0 and Web 2.0 API 68
3.1.2 Web 2.0 Risks 72
3.1.3 Web 2.0 Botnet 73
3.1.4 Botnet 2.0 75
3.2 Current Botnet Detection Techniques 79
3.2.1 Traffic analysis 79
3.2.2 Honeypots 81

3.2.3 Detecting software vulnerabilities 82
3.2.4 Network traceback 83
3.3 Summary and Relation to Our Work 85

4 API Verifier: an application-based approach to detect Bot 2.0 89

4.1 Motivation 90
4.2 API Verifier Design 91
4.2.1 API Verifier Client 92
4.2.2 API Verifier Server 92
4.2.3 Authentication 92
4.2.4 Session Keys 93
4.2.5 Permanent MAC Address 93
4.2.6 Captcha Verification 94
4.2.7 Update of User Profile 94
4.3 API Verifier Work Flow 95
4.4 Security Analysis 97
4.4.1 Spoofing MAC Address 98
4.4.2 Protection of API Verifier Client and the secret key 100
4.4.3 Bypassing the Captcha Verification 101
4.5 Usability and Possible Enhancement 104
4.5.1 Multiple MAC addresses 105
4.5.2 Support Multiple Platforms 105
4.6 Conclusion and Possible Future Study 106
4.6.1 Alternatives for Computer Identifiers 108
4.6.2 Alternatives for Captcha 109

5 Captcha Enhancement 111

5.1 Background 112
5.2 Captcha Evolution 114
5.2.1 Text-based Scheme 114
5.2.2 Image-based Scheme 116

5.2.3 Audio-based Scheme 117
5.2.4 Animated Captcha 118
5.2.5 Problem-solving Captcha 119
5.2.6 Drag and Drop Captcha 119
5.3 Motivation 120
5.4 Related Works 124
5.5 Proposed Captcha Design 125
5.5.1 Image Database 126
5.5.2 Captcha Generator 126
5.5.3 Captcha Session 128
5.5.4 Verification Unit 129
5.5.5 Client Side 130
5.5.6 Delay Time 132
5.6 Security Analysis 136
5.6.1 Brute force Attack 137
5.6.2 Relay Attack 138
5.6.3 Image Recognition Attack 138
5.6.4 Usability 140
5.7 Conclusion 140

6 Implementation and Experiments 143

6.1 Implementation 144
6.1.1 Test of Security Goals of the API Verifier and the enhanced Captcha 144
6.1.2 Koobface 145
6.1.3 Implementation of the API Verifier and the new Captcha 149
6.1.4 Implementation of simple Koobface botnet 150
6.2 Experiments and Results 151
6.2.1 Experiment Set Up 151
6.2.2 Security Requirement 1: the API Verifier is able to distinguish between human and a computer program using Captcha verification 152

6.2.3 Security Requirement 2: the API Verifier is able to prevent spoofing MAC address attacks 155
6.2.4 Security Requirement 3: the API Verifier is able to prevent automated program from bypassing its Captcha verification phase 159
6.3 Conclusion 168
6.3.1 Summary and Achievements 168
6.3.2 Drawbacks and Possible Improvements 169

7 Conclusion **173**
7.1 Conclusion 173
7.1.1 Evaluation of Research Goals 174
7.1.2 Critical Assessments on our Solution 180
7.2 Future Work 182
7.2.1 Mobile phone botnet 182
7.2.2 Obfuscation 183
7.2.3 TPM 184

8 An Appendix **185**

Glossary **199**

List of Publications **201**

References **203**

List of Figures

1.1 Our proposed solution fitted in Web services defence system 12

2.1 A typical botnet 18
2.2 The Botnet Life Cylce 21
2.3 Rise of the web based botnets 28
2.4 Star Architecture 39
2.5 Multi-Server Architecture 40
2.6 Hierarchical Architecture 42
2.7 Decentralized Architecture 43
2.8 Hybrid P2P Architecture: slave bots also form a P2P network 45
2.9 Hybrid P2P Architecture: each servant bot is a centralized control server 46
2.10 Agobot configuration interface 52
2.11 Agobot username and password authentication 52
2.12 `achcheck.php` transaction packet capture 55
2.13 C&C GUI: Zeus Control Panel - Script Edit 57
2.14 Example of a bot information in a peer list 58
2.15 Searching other (peer) bots 59
2.16 Reply messages from other (peer) bots 59
2.17 Peer lists exchange 60
2.18 Send spam through Google's SMTP servers 60

3.1 Sample Twitter Koobface Spam 74

3.2 Fake Youtube page that leads to download Koobface worm 75
3.3 Trojan 2.0 Command and Control 76
3.4 An example of post entry from a bot 77
3.5 Sample commands posted by TwitterBots 78
3.6 Connection chain example between H_0 and H_n 84

4.1 Components of API Verifier 91
4.2 The API Verifier Workflow 95
4.3 Model of Possible Attacks 98
4.4 Overview of a HTTP Digest Authentication Scheme 106

5.1 Typical EZ-Gimpy Captchas 115
5.2 Modern Captcha: reCaptcha 116
5.3 Image-based Captcha: Asirra Captcha 117
5.4 Audio Captcha as an alternative for text Captcha 118
5.5 Animated Captcha 118
5.6 Problem solving Captcha 119
5.7 Drag and Drop Captcha 120
5.8 Relay Attack on Captcha 121
5.9 Laundry Attack on Captcha 122
5.10 Captcha Solver Job Advertisement 122
5.11 keyCaptcha 125
5.12 Captcha 2.0 125
5.13 Enhanced Captcha: system architecture 126
5.14 The enhanced Captcha GUI 127
5.15 A pseudo code of the verification and generate Captcha procedures . . 129
5.16 Client and Server exchanging new labels every 10 seconds 131
5.17 Time delays in relay attack 132
5.18 Captcha solving page 134
5.19 Communication time in the simulated relay attack 136

6.1 Koobface Social Networking Propagation 147
6.2 Koobface Captcha Breaker Component 148
6.3 Sample of Koobface fake Captcha 148
6.4 Example of Captcha verification from the API Verifier 153
6.5 Example of Koobface fake Captcha with text Captcha 163
6.6 Example of Koobface fake Captcha with our enhanced Captcha 164
6.7 Another Koobface fake Captcha with our enhanced Captcha 165

8.1 The API Verifier Sequence Diagram 186
8.2 The Enhanced Captcha Sequence Diagram 187
8.3 The Simplified Koobface Sequence Diagram 188

List of Tables

2.1 A collection of some well-known bots 22
2.2 Malware Comparison 29
2.3 Properties of Botnet Architectures 37
2.4 Some Agobot C&C commands 53
2.5 Some Koobface C&C commands 56

3.1 Methods supported in Facebook Publishing API 71

8.1 Function List of API Verifier 193
8.2 Function List of the Enhanced Captcha 195
8.3 Function List of the Simplified Koobface 197

8.4 Glossary 200