Characterizing the Running Patterns of Moving Target Defense Mechanisms

Guilin Cai, Baosheng Wang, Yuebin Luo, Sudan Li, Xiaofeng Wang

College of Computer, National University of Defense Technology, Changsha, China

cc_cai@163.com, wangbaosheng@126.com, luoyuebin@nudt.edu.cn, nudtlsd@163.com, xf_wang@nudt.edu.cn

Abstract— **Moving Target Defense (MTD) has been proposed as a game-changing theme to increase the work effort to attack as well as the security of target system. There has been proposed a multitude of MTD mechanisms. Generally, these mechanisms follow some fundamental running patterns which determine their functionalities. In this paper, we introduce three main schools of thought on MTD mechanisms systematically and categorize the related works according to them. Then we identify and define three fundamental running patterns exhibited by these MTD mechanisms. Thereafter, we use five MTD mechanisms, which belong to the three schools of thought, as cases to confirm the patterns presented. This work can help the novices of this field to understand the running behaviours of MTDs better and easier, and can also give developers design guidance of new MTD system by providing insights of the running patterns.**

Keywords— **Moving Target Defense, security, fundamental running patterns, confirmation**

I. Introduction

With the rapid growth of information technologies, the Internet has become a national key infrastructure. However, the attacks (such as IP prefix hijacking [1], botnet [2], DDoS attack [3]) can be seen everywhere and at any time, and major security incidents have been frequently reported in recent years (such as the Prism [4], the Heartbleed Bug[5] , eBay data leakage). Such security disasters are repeatedly showing that, the security of the Internet is always facing severe challenges. One of the major reasons of the severe Internet security situation is that the network configurations nowadays are typically deterministic, static, and homogeneous [6]-[7]. These features reduce the difficulties for cyber attackers scanning the network to identify specific targets and gather essential information, which gives the attackers the advantages of building up, launching and spreading attacks. Therefore, in the struggle between cyber network attack and defense, the attackers typically have asymmetric advantages and the defenders are always disadvantaged by being passive.

To alter the asymmetric situation between attacks and defenses, Moving Target Defense (MTD) is proposed as one of the "game-changing" themes in cybersecurity [6], [8]. MTD can change one or more system attributes automatically and continually, such that the attack surface area available to adversaries is unpredictable [8]. This makes attacking much more difficult for an attacker, and thus can enhance the security of target system to a certain extent.

There have been proposed a multitude of works related to the concept of MTD, and they can be divided into three research areas, named MTD theory, Mechanism, and Evaluation. MTD theory attempts to find the answers to some fundamental questions, such as how to create an effective MTD system [9]-[10], and what capabilities and features should be had by an MTD system [11]-[12]. Mechanism focuses on designing various strategies for the selected movement attribute(s) to make it/them moving (it will be discussed later). Evaluation aims at measuring the effectiveness of existing mechanisms to get some insights and provide reference for new designation, such as [13]-[16]. When just focusing on the research area of Mechanism, some fundamental running patterns emerge. Currently, there is no related research to analyze them. In this paper, we try to fill the gap by identifying and defining the fundamental running patterns exhibited in existing mechanisms to help the novices to understand and create MTDs.

In this work, we make the following contributions:

1) Categorizing existing MTD mechanisms. We introduce the three main schools of thought on MTD systematically and categorize the main MTD mechanisms according to the schools for the first time.
2) Identifying the fundamental running patterns for MTD. We describe the two main patterns, either of which would be followed by the MTD mechanisms, and an assisted pattern that can enhance the effect of the two main patterns.
3) Confirming the proposed patterns. We use five prior mechanisms as case studies, which belong to the three categories, to confirm the patterns proposed. The five case studies appear to be the epitome of all the MTD mechanisms.

II. Classification of Related Works

The goal of MTD is to increase the work effort for attackers to launch a successful attack, limit the exposure of vulnerabilities and opportunities, and enhance the resiliency of protected target. The way to achieve the goal is deploying and operating networks and systems in a manner that makes them less static, less deterministic, and less homogeneous [6]. Therefore, the MTD mechanisms should focus on designing a moving strategy for the selected movement attribute(s) to make it/them moving continuously to interrupt the static and deterministic feature of target. What's more, diversity should be also applied to the target simultaneously for making it less homogeneous.

Currently, according to the selected movement attribute(s), there are three main schools of thought providing their solutions for designing MTD mechanisms, and we call them software transformations, dynamic platform techniques, and network address shuffling.

A. Software Transformations

The first school of thought is software transformations [17]. The MTD mechanisms based on software transformations usually apply diversity transformation to the code (including transforming the value of a parameter) of software/application in different ways to create multiple variants that provide the same function but with different behaviors and features, and then dynamically shuffle among these variants in order to improve the capability against attack. In other words, the MTD mechanisms based on software transformations usually choose software/application as the movement target. There are several mechanisms proposed by different research teams in this category, including ChameleonSoft [18], Compiler-generated software diversity [19], End-to-end software diversity [20], Practical software diversification [21], SEM [22], Proactive Obfuscation [23], HMS [24], NOMAD [25] (transforming the value of name/id parameter), and the adaptive just-in-time code diversification [26]. The major difference among them is the way to diversity. The shuffling is usually realized by randomly choosing the next variant.

B. Dynamic Platform Techniques

The second school of thought is Dynamic Platform Techniques (DPT) [27]. DPT dynamically changes the properties of a computing platform in order to complicate attacks. The selected properties refer to hardware and operating system (OS) attributes such as instruction set architecture (ISA), stack direction, calling convention, kernel version, OS distribution, and machine instance [28]. In other words, the mechanisms in this category usually use the properties of the running platforms as the movement target. The representative research teams in this category include the group in MIT Lincoln Laboratory and the group in George Mason University, and the mechanisms in this category include TALENT [29] (MIT Lincoln Laboratory), SCIT [30] (George Mason University), MAS [31] (George Mason University), MTD strategy for Cloud-based services [32], and the approach of evolving computer configuration [33,34]. The major difference among them is the way to create the multiple platform instances. And now there are three ways, inherence (as in [29]), virtualization (as in [30]-[32]), and evolution-inspired techniques (as in [33]-[34]).

C. Network Address Shuffling

The third school of thought is network address shuffling [35]. Network address shuffling is a dynamic reconnaissance defense that periodically permutes the mappings between addresses and devices. For the Internet, addresses are a combination of IP and transport layer information (protocol and port numbers) and either or both types of information can be used for shuffling [35]. In other words, the mechanisms in this category usually choose IP address and/or port number as the movement target. The representative research teams in this category include the group in University of North Carolina at Charlotte, the group in Virginia Tech, and the group in George Mason University. The mechanisms in this category include DYNAT [36], NASR [37], service hopping [38], SDNA [39], MT6D [40] (Virginia Tech), OF-RHM [41] (University of North Carolina at Charlotte), RHM [42] (University of North Carolina at Charlotte), Spatio-temporal address mutation [43] (University of North Carolina at Charlotte), MORPHINATOR [44], MOTAG [45] (George Mason University), MTD-MANETs [46] (George Mason University with University of Naples Federico II), and the SDN shuffle approach [47]. The major difference among them is the way to select the next address, and the network architecture required for deployment. In addition, we should note that the virtual identity of legitimate nodes rather than IP address changes dynamically in the mechanism MTD-MANETs actually. However, as the virtual identity is used for communication between nodes like IP address in traditional network, we still take the movement attribute in it as IP address.

III. Fundamental Running Patterns

The vision of MTD can be described as follows: Create, evaluate, and deploy mechanisms and strategies that are diverse, continually shift, and change over time to increase complexity and costs for attackers, limit the exposure of vulnerabilities and opportunities for attack, and increase system resiliency [6]. From our own perspective, the key words are diversity, continually shifting and change over time. They can be seen as the common properties across existing MTDs, and shown by several fundamental running patterns exhibited in existing MTDs. In this section, we describe the two main running patterns called “hidden” pattern and “variation” pattern, and an assisted pattern called “improvement” pattern.

A. Two Main Fundamental Patterns

Now we introduce the two main fundamental running patterns across existing MTDs, “hidden” and “variation”.

First, from the related works, we can find that all the mechanisms in the three schools of thought on MTD are with multi-instances. For all the mechanisms in the category of software transformations, the multi-instances are the multiple variants of software/application. For the mechanisms in the category of DPT, the multi-instances are the multiple execution environments (each is with a different configuration for the selected properties in [29], [31]-[32], or they are with the same configuration in [30]), or the multiple configurations for the same execution environment ([33]-[34]). For all the mechanisms in the category of network address shuffling, the multi-instances are the multiple IP addresses and/or port numbers for shuffling.

We now take the mechanisms in the category of network address shuffling as an example to describe the prototype for each MTD mechanism. For simplification of description and without loss of generality, we only consider IP address here (as shown in Figure 1). From Figure 1, we can see that there is an address pool, which contains multiple valid addresses and each of them can be assigned to the target at different time. The multiple addresses must be different from one another to ensure the effectiveness of the mechanisms. In other words, they are diverse.

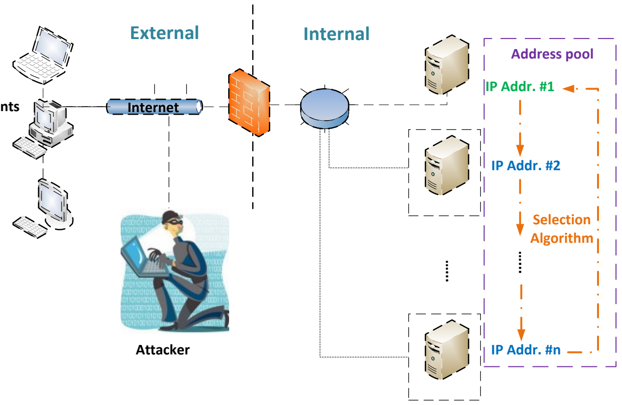
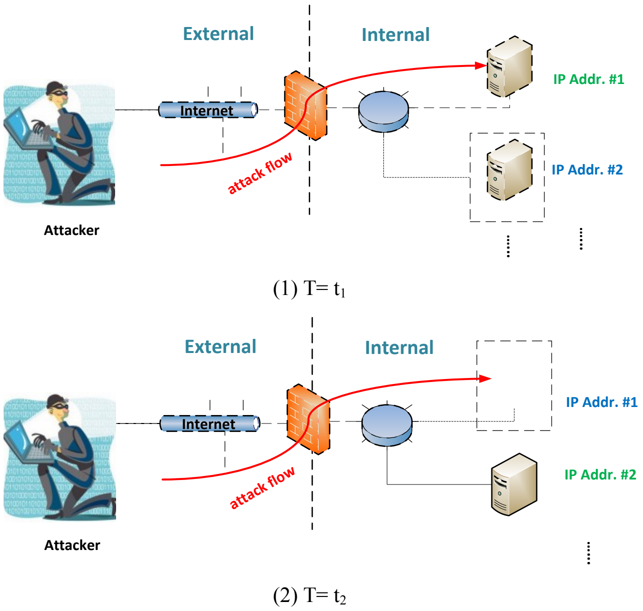


Figure 1. An example prototype of MTD

Similarly, for the mechanisms in the category of DPT, there is a platform/configuration pool with diversity; for the mechanisms in the category of software transformations, there is a variant pool with diversity. In addition, unlike the mechanisms in the category of network address shuffling, each mechanism in the category of DPT and software transformations has its own way to produce diversity. For example, in Compiler-generated software diversity [19], the compiler automatically creates multiple variants when it is translating high-level source code to low-level machine code. These variants have the same in-specification behaviour, but different out-of-specification behaviour. In Proactive Obfuscation [23], multiple server replicas having fewer shared vulnerabilities are created by using semantics-preserving code transformations. In MAS [31], virtualization technology is used to create multi-instances of protected server, and each VS (virtual server) is configured with a unique software mix and thus producing diversity. In the MTD strategy for Cloud-based services [32], virtualization technology is also used to create multiple VMs (virtual machines, i.e., the multi-instances) to deploy a service in Cloud, and enough diversity would be introduced in configuration when the VMs are created to ensure the effectiveness of MTD.

The combat between the attacks and defenses likes an arm race, in which each side hopes to be one step ahead of the other side to achieve their own goals. For the defenders, with the foundation of multi-instances with diversity, there are two running patterns to ensure them to be one step ahead by continually shifting and changing attack surface over time.

The first running pattern is the “hidden” pattern, which is shown in Figure 2.



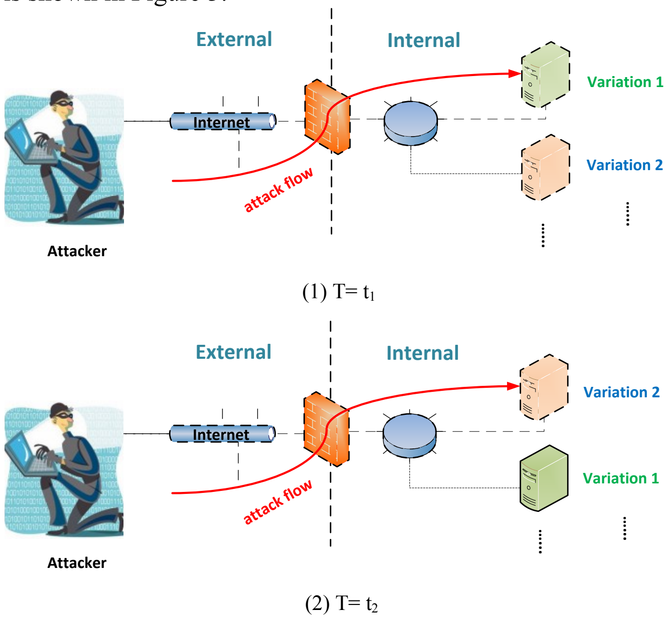
(1) T= t_1

(2) T= t_2

Figure 2. The “hidden” pattern

As shown in Figure 2, the attacker exploits the target and collects essential information at time t_1. At time t_2 ($t_2 > t_1$), the attacker wants to continue the exploitation, however, he finds that the target in the original place disappears, just like it hides itself. Therefore, this attack is aborted.

The second running pattern is the “variation” pattern, which is shown in Figure 3.



(1) T= t_1

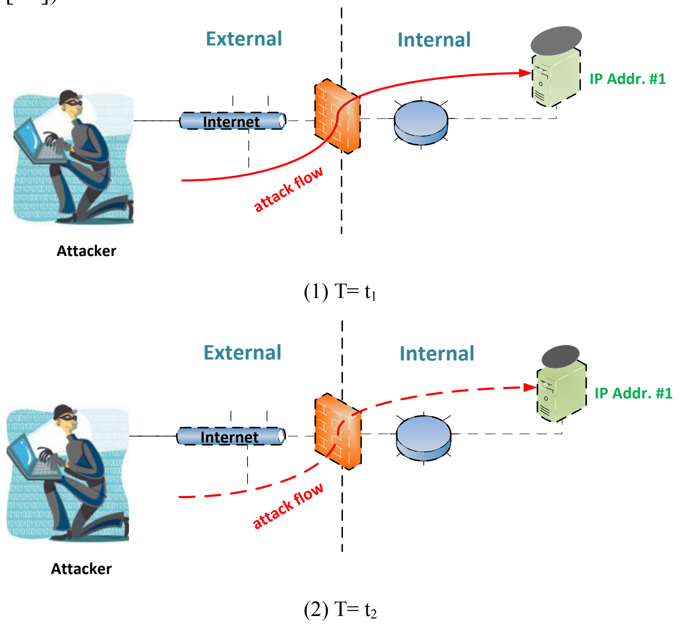
(2) T= t_2

Figure 3. The “variation” pattern

As shown in Figure 3, the attacker exploits the target and collects essential information at time t_1. At time t_2 ($t_2 > t_1$), the attacker wants to continue the exploitation, however, he finds that the target in the original place is replaced by another variation which are with different feature and vulnerabilities. Therefore, this attack is aborted.

B. *An Assisted Pattern*

Although there is still no standard definition of attack surface [9], it is the concept that relates tightly to MTD and has been frequently used in the MTDs. Existing definitions of attack surface is related to the scenarios considered by the researchers. For simplicity and without loss of generality, we refer attack surface to the definition in [48] and just treat it as the set of vulnerabilities of target. Some vulnerabilities can be removed through repairing, and thus the attack surface can be reduced and the defense effect can be improved (Intuitively, the larger the attack surface, the more insecure the system [49]).



(1) T= t_1

(2) T= t_2

Figure 4. The "improvement" pattern

The "improvement" pattern is illustrated in Figure 4. The shadow above the target implies the attack surface, which can not be empty forever because the vulnerabilities of the networking devices (software, hardware, protocols, etc.) cannot be removed completely. As shown in Figure 4, the attacker exploits the target and collects essential information for a specific vulnerability at time t_1. At time t_2 ($t_2 > t_1$), the vulnerability exploited by the attacker is repaired by defender (it is illustrated by the smaller shadow). Therefore, the attack process is interrupted for the door is closed, though the attacker wants to continue the exploitation.

Not all the mechanisms adopted the attack surface reduction, hence it is just an assisted pattern to improve the defense effect rather than a main pattern.

We have described the three fundamental running patterns (two main patterns and an assisted pattern) exhibited in the MTD mechanisms. What's more, there can be more patterns produced by combining the three fundamental patterns.

IV. Confirming Case Studies

We now use five prior MTD mechanisms as case studies to confirm the patterns presented above.

A. *ChameleonSoft*

ChameleonSoft [18] is a mechanism belongs to the category of software transformations. In this mechanism, the large missions of a huge software program are divided into smaller tasks, and each task is assigned to one or more cells which would manually or automatically generate several variants for the task. The variants have different objectives targeting different quality attributes, such as reliability, performance, robustness, and mobility. Therefore, there are multiple variants with diversity in each cell. In addition, the variants can shuffle based on a randomly adjusted timer in each cell to make the software to change over time. From the view of time, the executable variants at time t_1 and at time t_2 (t_2>t_1) are with same function, but with different-behaviour and different quality attributes. Thus it accords with the "variation" pattern.

B. *HMS*

HMS [24] is also a mechanism belongs to the category of software transformations. In this mechanism, a spatio-temporal diversity engine performs the necessary transformations on the variant selected in last generation (the first generation consists of the original input software only), and create multi-instances in current generation for shifting. When an attack is detected, the GenProg Engine would use evolutionary algorithms to create and vet candidate repair patches to repair both security-critical and non-security critical vulnerabilities of the variants. In this way, from the view of time, the running software variants at time t_1 and at time t_2 (t_2>t_1) is different, and the attack surface at time t_2 is smaller than at time t_1. Thus it complies with the combination of "variation" pattern and "improvement" pattern.

C. *TALENT*

TALENT (Trusted Dynamic Logical Heterogeneity System) [29] is a mechanism belongs to the category of DPT, which contains multiple heterogeneous physical hardware platforms and operating systems for the running critical application to migrate. When the interval expires or an anomalous event arrives, TALENT allows a running critical application to migrate to a different platform. In other words, from the time perspective, the execution environments of the application at time t_1 and at time t_2 ($t_2 > t_1$) provide the same functionality (ensuring the normal running of the critical application) but with different features. Thus it accords with the "variation" pattern.

D. *The approach of evolving computer configuration*

The approach of evolving computer configuration [33]-[34] is also a mechanism in the category of DPT. It uses evolution-inspired techniques to create multiple functional and secure configurations based on previous configurations. The crossover operation ensures that the configurations of new generation are different from the old ones, and the mutation operation is to maintain diversity across the configurations of current generation. All the configurations generated would be assessed by an assessment component to determine its security level. The new and more secure configurations would be periodically implemented. In this way, from the view of time, the configurations at time t_1 and at time t_2 ($t_2 > t_1$) is different, and the attack surface at time t_2 is smaller than time t_1 (Intuitively, the larger the attack surface, the more insecure the system [49]). Thus we can say that this approach accords with the combination of “variation” pattern and “improvement” pattern.

E. OF-RHM

OF-RHM (OpenFlow Random Host Mutation) [25] is a mechanism belongs to the category of network address shuffling. In this mechanism, each host is associated with an unused address range (i.e., the set of virtual IPs) that is assigned by the OpenFlow controller based on its specific requirement using SMT (Satisfiability Modulo Theories). A new virtual IP is chosen from the range and assigned to the host after each regular mutation interval. It is well known that the IP address of a device is equivalent to the latitude and longitude of a position in the real world, if the attacker knows the IP address of the device, he knows the position of the device, and then he can initiate the attack process to exploit and analyze the target system. If the IP address of device changes, the attacker would lose his target. It is the right thing that OF-RHM does. From the time perspective, the attacker can connect to the target at time t_1, but the target in the original position disappears at time t_2 ($t_2 > t_1$, actually it is “hidden” to another position), thus it accords with the “hidden” pattern.

What’s more, all the mechanisms in the category of network address shuffling share the similar goal and do the same thing with OF-RHM but with different ways. Therefore, all the mechanisms in the category of network address shuffling conform to the “hidden” pattern.

V. Conclusions

In this paper, we first introduce the three main schools of thought on MTD systematically and categorize the main MTD mechanisms according to the schools. And then the fundamental running patterns were identified and described, which consist of two main patterns and an assisted pattern for the MTD mechanisms running. To confirm the patterns, we use five existing MTD mechanisms which come from the three schools of thought as case studies for analyzing. The analysis shows that each case should follow one of the two main patterns, and may be with the assisted pattern, which can be seen as the epitome of existing MTD mechanisms.

Acknowledgment

This work was supported by the National Key Basic Research Program of China (973 Program). [2012CB315906, Reconfigurable Network Emulation Testbed for Basic Network Communication]

References

[1] Y. Liu, W. Peng and J. Su, "A study of IP prefix hijacking in cloud computing networks", *SECURITY AND COMMUNICATION NETWORKS*, vol.7, no.11, pp.2201-2210. 2014.

[2] T. Wang, H. Wang, B. Liu, B. Ding, J. Zhang and P. Shi, "Further Analyzing the Sybil Attack in Mitigating Peer-to-Peer Botnets", *KSII Transactions on Internet and Information Systems (TIIS)*, vol.6, no.10, pp.2731-2749. 2012.

[3] F. Wang, H. Wang, X. Wang and J. Su, "A new multistage approach to detect subtle DDoS attacks", *Math Comput Model*, vol.55, no.1, pp.198-213. 2012.

[4] *Prism*. https://en.wikipedia.org/wiki/PRISM_(surveillance_program), 2013

[5] *The Heartbleed Bug*. http://heartbleed.com/, 2014

[6] Cybersecurity Game-Change Research & Development Recommendations. Available: http://www.nitrd.gov/pubs/CSIA_IWG_%20Cybersecurity_%20GameChange_RD_%20Recommendations_20100513.pdf. 2010.

[7] Trustworthy CyberSpace: Strategic plan for the federal cybersecurity research and development program. Available: http://www.whitehouse.gov/sites/default/files/microsites/ostp/fed_cybersecurity_rd_strategic_plan_2011.pdf. 2011.

[8] National Cyber Leap Year Summit 2009 Co-Chairs ’ Report. https://www.nitrd.gov/nitrdgroups/index.php?title=Category:National_Cyber_Leap_Year_Summit_2009. 2009.

[9] R. Zhuang, S. A. DeLoach and X. Ou, "Towards a Theory of Moving Target Defense", in *Proc of MTD '14*, 2014, pp.31-40.

[10] T. Hobson, H. Okhravi, D. Bigelow, R. Rudd and W. Streilein, "On the Challenges of Effective Movement", in *Proc of MTD '14*, 2014, pp.41-50.

[11] M. Carvalho, T. C. Eskridge, L. Bunch, A. Dalton, R. Hoffman, J. M. Bradshaw, P. J. Feltovich, D. Kidwell and T. Shanklin, "MTC2: A command and control framework for moving target defense and cyber resilience", in *Proc of 6th Int Symp on Resilient Control Systems (ISRCS)* , 2013, pp.175-180.

[12] M. Green, D. C. MacFarland, D. R. Smestad and C. A. Shue, "Characterizing Network-Based Moving Target Defenses", in *Proc of MTD '15*, 2015, pp.31-35.

[13] R. Zhuang, S. Zhang, S. A. DeLoach, X. Ou and A. Singhal, "Simulation-based approaches to studying effectiveness of moving-target network defense", *National Symposium on Moving Target Research*, pp.1-12. 2012.

[14] A. Clark, K. Sun and R. Poovendran, "Effectiveness of IP address randomization in decoy-based moving target defense", in *Proc of 52nd Annual Conference on Decision and Control (CDC)*, 2013, pp.678-685.

[15] J. Xu, P. Guo, M. Zhao, R. F. Erbacher, M. Zhu and P. Liu, "Comparing Different Moving Target Defense Techniques", in *Proc of MTD '14*, 2014, pp.97-107.

[16] M. Crouse, B. Prosser and E. W. Fulp, "Probabilistic Performance Analysis of Moving Target and Deception Reconnaissance Defenses", in *Proc of MTD '15*, 2015, pp.21-29.

[17] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang and X. S. Wang, *Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats*, New York:Springer, 2011.

[18] M. Azab, R. Hassan and M. Eltoweissy, "ChameleonSoft: A moving target defense system", in *Proc of 7th Int Conf on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, 2011, pp.241-250.

[19] T. Jackson, B. Salamat, A. Homescu, K. Manivannan, G. Wagner, A. Gal, S. Brunthaler, C. Wimmer and M. Franz, "Compiler-Generated Software Diversity", *Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats*, pp.77-98, New York: Springer, 2011.

[20] M. Christodorescu, M. Fredrikson, S. Jha and J. Giffin, "End-to-End Software Diversification of Internet Services", *Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats*, pp.117-130, New York: Springer, 2011.

[21] V. Pappas, M. Polychronakis and A. Keromytis, "Practical Software Diversification Using In-Place Code Randomization", *Moving Target Defense II: Application of Game Theory and Adversarial Modeling*, pp.175-202, New York: Springer, 2013.

[22] A. Cui and S. Stolfo, "Symbiotes and defensive Mutualism: Moving Target Defense", *Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats*, pp.99-108, New York: Springer, 2011.

[23] T. Roeder and F. B. Schneider, "Proactive Obfuscation", *ACM Trans on Computer Systems*, vol.28, no.2, pp.1-4. 2010.

[24] C. Goues, A. Nguyen-Tuong, H. Chen, J. Davidson, S. Forrest, J. Hiser, J. Knight and M. Gundy, "Moving Target Defenses in the Helix Self-Regenerative Architecture", *Moving Target Defense II: Application of Game Theory and Adversarial Modeling*, pp.117-149, New York: Springer, 2013.

[25] S. Vikram, C. Yang and G. Gu, "NOMAD: Towards non-intrusive moving-target defense against web bots", in *Prod of 2013 IEEE Conference on Communications and Network Security (CNS)*, 2013, pp.55-63.

[26] A. Jangda, M. Mishra and B. De Sutter, "Adaptive Just-In -Time Code Diversification ", in *Proc of MTD '15*, 2015, pp.49-53.

[27] H. Okhravi, T. Hobson, D. Bigelow and W. Streilein, "Finding Focus in the Blur of Moving-Target Techniques", *IEEE Security & Privacy*, vol.12, no.2, pp.16-26. 2014.

[28] H. Okhravi, J. Riordan and K. Carter, "Quantitative Evaluation of Dynamic Platform Techniques as a Defensive Mechanism", *Research in Attacks, Intrusions and Defenses*. pp.405-425, 2014.

[29] H. Okhravi, A. Comella, E. Robinson and J. Haines, "Creating a cyber moving target for critical infrastructure applications using platform diversity", *International Journal of Critical Infrastructure Protection*, vol.5, no.1, pp.30-39. 2012.

[30] A. K. Bangalore and A. K. Sood, "Securing Web Servers Using Self Cleansing Intrusion Tolerance (SCIT)", in *Proc of DEPEND '09*, 2009, pp.60-65.

[31] Y. Huang and A. Ghosh, "Introducing Diversity and Uncertainty to Create Moving Attack Surfaces for Web Services", *Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats*, pp.131-151, New York: Springer, 2011.

[32] P. Wei, L. Feng, H. Chin-Tser and Z. Xukai, "A moving-target defense strategy for Cloud-based services with heterogeneous and dynamic attack surfaces", in *Proc of 2014 IEEE International Conference on Communications (ICC)* , 2014, pp.804-809.

[33] B. Lucas, E. W. Fulp, D. J. John and D. Ca N As, "An Initial Framework for Evolving Computer Configurations As a Moving Target Defense", in *Proc of CISR '14*, 2014, pp.69-72.

[34] D. J. John, R. W. Smith, W. H. Turkett, D. A. Ca N As and E. W. Fulp, "Evolutionary Based Moving Target Cyber Defense", in *Proc of the 2014 Conf Companion on Genetic and Evolutionary Computation Companion*, 2014, pp.1261-1268.

[35] T. E. Carroll, M. Crouse, E. W. Fulp and K. S. Berenhaut, "Analysis of network address shuffling as a moving target defense", in *Proc of 2014 IEEE International Conference on Communications (ICC)*, 2014, pp.701-706.

[36] D. Kewley, R. Fink, J. Lowry and M. Dean, "Dynamic approaches to thwart adversary intelligence gathering", in *Proc of DARPA Information Survivability Conference amp Exposition II*, 2001, pp.176-185.

[37] S. Antonatos, P. Akritidis, E. P. Markatos and K. G. Anagnostakis, "Defending against hitlist worms using network address space randomization", *Computer Networks-TheComputer Networks*, vol.51, no.12, pp.3471-3490. 2007.

[38] L. Shi, C. Jia and S. Lü, "DoS Evading Mechanism upon Service Hopping", in *Proc of IFIP International Conference on Network and Parallel Computing Workshops*, 2007, pp.119-122.

[39] J. Yackoski, P. Xie, H. Bullen, J. Li and K. Sun, "A Self-shielding Dynamic Network Architecture", in *Proc of MILCOM 2011*, 2011, pp.1381-1386.

[40] M. Dunlop, S. Groat, W. Urbanski, R. Marchany and J. Tront, "MT6D: A Moving Target IPv6 Defense", in *Proc of MILCOM 2011*, 2011, pp.1321-1326.

[41] J. H. Jafarian, E. Al-Shaer and Q. Duan, "Openflow Random Host Mutation: Transparent Moving Target Defense Using Software Defined Networking", in *Proc of HotSDN '12*, 2012, pp.127-132.

[42] E. Al-Shaer, Q. Duan and J. Jafarian, "Random Host Mutation for Moving Target Defense", *Security and Privacy in Communication Networks*, pp.310-327, 2013.

[43] J. H. Jafarian, E. Al-Shaer and Q. Duan, "Spatio-temporal Address Mutation for Proactive Cyber Agility Against Sophisticated Attackers", in *Proc of MTD '14*, 2014,pp.69-78.

[44] *MORPHINATOR*. http://defense-update.com/tag/morphinator, 2012

[45] Q. Jia, K. Sun and A. Stavrou, "MOTAG: Moving Target Defense against Internet Denial of Service Attacks", in *Proc of 22nd International Conference on Computer Communications and Networks (ICCCN)*, 2013, pp.1-9.

[46] M. Albanese, A. De Benedictis, S. Jajodia and K. Sun, "A moving target defense mechanism for MANETs based on identity virtualization", in *Proc of 2013 IEEE Conference on Communications and Network Security (CNS)*, 2013, pp.278-286.

[47] D. C. MacFarland and C. A. Shue, "The SDN Shuffle: Creating a Moving-Target Defense Using Host-based Software-Defined Networking", in *Proc of MTD '15*, 2015, pp.37-41.

[48] Q. Zhu and T. Başar, "Game-Theoretic Approach to Feedback-Driven Multi-stage Moving Target Defense", *Decision and Game Theory for Security*, pp.246-263, Springer International Publishing, 2013.

[49] P. Manadhata and J. Wing, "A Formal Model for a System's Attack Surface", *Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats*, pp.1-28, New York: Springer, 2011.

Guilin Cai is a doctor student in the College of Computer at National University of Defense Technology in China. She received the B.S. and M.S. degrees in computer science from National University of Defense Technology in 2005 and 2010, respectively. Her main reserch interests include cyber security and proactive defense.

Baosheng Wang has been working as a professor and PhD supervisor in the College of Computer at National University of Defense Technology in China. He received the B.S., M.S. and Ph.D. degrees in computer science from National University of Defense Technology in 1992, 1995 and 2005, respectively. His research interests include router architecture, routing protocol and cyber security.

Yuebin Luo is a doctor student in the College of Computer at National University of Defense Technology in China. He received the B.S. degree in computer science from Tianjin University in 2010, and the M.S. degree in computer science from National University of Defense Technology in 2012. His research interests include intrusion detection, active defense, and network and information security.

Sudan Li has been working as an associate professor in the College of Computer at National University of Defence Technology in China. He received B.S., M.S and Ph.D. degrees in control theory and engineering from National University of Defence Technology in 1995,1998 and 2001 respectively. His research interests include network architecture and protocol, router software designe.

Xiaofeng Wang has been working as an assistant professor in th College of Computer at National University of Defense Technology in China. He received the B.S., M.S. and Ph.D. degrees in computer science from National University of Defense Technology in 2004, 2006 and 2009, respectively. His research interests include trustworthy networks and systems, applied cryptography, network security.