International Journal of Security and Its Applications
Vol. 10, No. 1 (2016), pp.45-54
http://dx.doi.org/10.14257/ijsia.2016.10.1.05

Multilevel Decentralized Protection Scheme Based on Moving Targets

Mikhail Styugin[1] and Nikolay Parotkin[2]

[1]*Siberian State Aerospace University, Krasnoyarsk, Russia*
Siberian Federal University, Krasnoyarsk, Russia
[2]*Siberian State Aerospace University, Krasnoyarsk, Russia*
[1]*styugin@gmail.com*, [2]*nyparotkin@yandex.com*

Abstract

The present paper reviews the scheme for protection of information resources from attempts to research them by intruders on the reconnaissance stage. Current drawbacks of the popular protection technology of Moving Target Defense (MTD) are analyzed herein. The paper offers a theoretical model of a system's operation when it can diversify itself by introducing dynamic variables. Such transformation is enabled by turning each process into an "interface" for another process and distributes new interaction rules to all legal processes. Hence, the established system is not static, but dynamic and all information related to the system and obtained at one moment in time becomes irrelevant with time. A practical method for protecting a local network from scanning is offered herein. Potential difficulties and solutions for implementation of the scheme are reviewed

Keywords: *information security, protection from research, moving target defense, network scanning*

1. Introduction

Providing defense for all potential vulnerabilities at the stage of a system's design is not practicable due to structural and functional complexity of information systems today. Hence, the task of information security is mostly limited to continuous monitoring and modification of an information system based on detected vulnerabilities and unauthorized activities. Consideration of information asymmetry between an attacker and a defender is also important to in this case. A defender has limited time for system's design and analysis, whereas an attacker has practically unlimited time (except for the time periods of system changes made by a defender). Moreover, the attacker operates after a system has been designed and uses more advanced technologies, software vulnerabilities and exploits that had not been detected before.

The problem of such asymmetry can be solved by protecting a system from attacker's research at the reconnaissance stage. The main task to be solved when using that approach is defending vulnerabilities undetected at the system design stage. In the field of information security systems protection the method was represented as the Moving Target Defense Technology. A most detailed description of that technology is provided in [1] and [2]. The technology is based on permanent modification of information system's parameters, which may be used by an adversary for attacking the system. Such parameters may include IP-addresses of hosts in a network [3, 4], arrangement of virtual machines [5], distribution of processes in memory [6], structure of databases [7], *etc*. Number of publications on MTD avalanched during the previous few years and today it may be regarded as a development trend for information security systems.

The MTD technology has a significant problem: when an attacker knows the operating principle of MTD it may research and find its existing vulnerabilities. In other words MTD defends systems from the threat, which can be a threat to MTD itself. One of the

ISSN: 1738-9976 IJSIA

attempts to solve that problem was introduction of "uniqueness concept" for an information resource [7] that can be configured by a security manager, and which makes the system unique and different from other systems designed using the same technology. However, that method certainly is not a panacea as the technology used in this case remains stereotypic.

An ideal representation of such system would be an infinitely complicating system which uses unique algorithms. Establishing such system in real life is not deemed possible. However, before performing practical calculations let us try to make a theoretic model of such system's operation.

The main contributions of this paper are listed below:

1. We developed the new MTD method named Multilevel MTD
2. We adapted Multilevel MTD method and a set of MTD techniques to establish effective network address shuffling.
3. We conducted the comparative experimental evaluation of MTD VLAN realization and estimated the time of network reconstruction.
4. We implemented a set of tools for experimental evaluation of MTD VLAN technique.

The paper is structured as follows. In Section 2 we illustrate a general theoretical scheme of multilevel MTD. Section 3 presents details of a multilevel decentralized MTD scheme. Section 4 provides an experimental evaluation and a short overview of decentralized VLAN. Conclusions are presented in Section 5.

2. Theoretical Scheme of Multilevel MTD

In order to implement any dynamic parameters in a system and maintain uninterrupted functioning of all system's mechanisms it is required to implement an interface for the real process (Figure 1). The module demonstrated in the figure below transforms external (dynamic parameters) into internal static process's input and output values. All other participants of the information exchange shall either know the interface's operating rules or the interface shall be merely a transformer for an external element, which operates not according to the rules. All implementations of MTD technology come down to the scheme below.

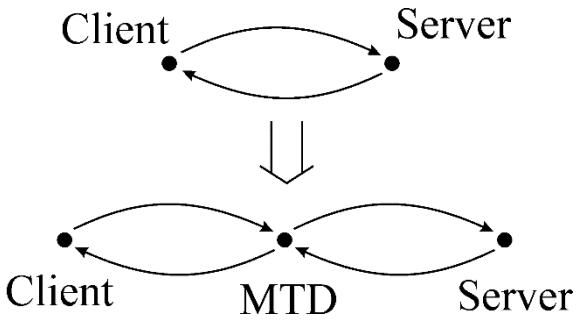

Figure 1. General Scheme of MTD

After seeing the scheme it becomes clear that MTD is actually some kind of a boundary layer via which the interaction with a system's process is carried out. There may be several layers of that type. In real life MTD systems normally it is not required to divide MTD interface into several layers as it is one single application. However, it may be useful in our case (Figure 2).

In that case MTD_1 is an interface for MTD_2 which in its turn operates with a specific process. Hence, a process's client should know transformations of every

process to generate a correct query and interpret the response and know the rules for their modification (dynamic properties), if any.

At this point we shall formalize events in the system's nodes (Figure 3). Only unidirectional data transmission without loss of generality is considered in this case.

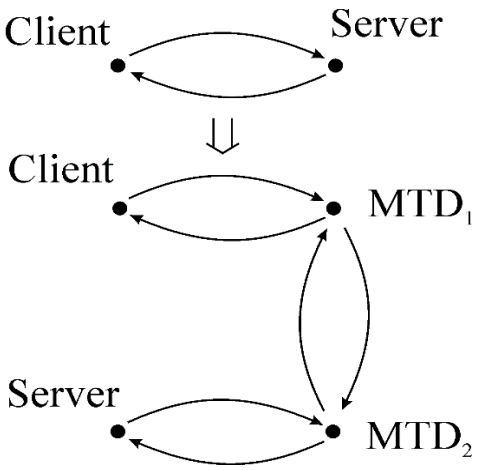

Figure 2. Double MTD Interface

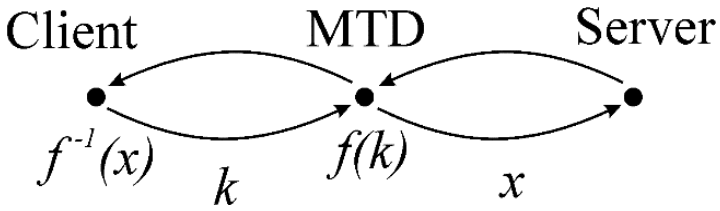

Figure 3. Functional Scheme with One MTD Node

In the above scheme the end process shall be assigned with a variable (data packet, content, *etc*.) *x*. The MTD node separates the client process form the addressee. The MTD performs a transformation *f(k)*, where *k* is what it obtains as output. Hence, the client process shall know the inverse function $f^{-1}(\cdot)$ to calculate variable *k* for a required *x*.

Scheme with two and more MTD nodes has a similar operating principle (Figure 4).

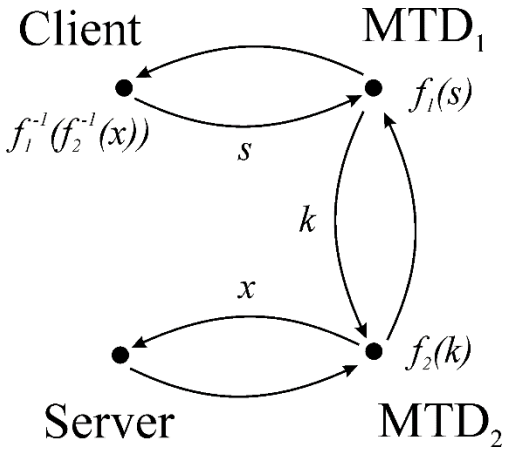

Figure 4. Functional Scheme with Two MTD Nodes

Another functional transformation is added here and consequentially the client needs to calculate two inverse functions to obtain the value of x as the result.

Let us imagine a system consisting of nodes (processes, applications, users, *etc*) and every node in that system is aware of the legal nodes of the system and the interaction interface between them. Hence, each one of the nodes in that system can undertake to function as an interface with another node after informing the node about it. The node in that case appoints a function $f(x)$ for its own interface and distributes function $f^{-1}(x)$ to all legal system nodes. The system "forgets" the real actual node and remembers only its new interface $f^{-1}(x)$.

Let us review such interface substitution in more detail. Initially we have some general system's operation scheme T. The scheme includes all nodes and legal information streams between them. Information streams are represented as sets of input and output values from nodes in relation to all other nodes. Every entity E_k included into the system has sets of input values P_k and output values F_k. Those sets can be defined by each pair of nodes. In a simplified case a scheme of information streams without sets of values was used and values were regarded typical (as in the example shown in the following paragraph). If entity E_k shall become an interface for entity E_t it has to transform the scheme of information streams $T \rightarrow T'$ in such a way that all information streams to E_t are forwarded to it. In addition to that the entity shall provide transformation rules $P_k \rightarrow P_t$ and $F_t \rightarrow F_k$ to all legal nodes of scheme T.

Let us review an example. Assume E_t is a database. Then P_t is a set of queries to the database and F_t is a set of probable results. Process E_k becomes a transmitter of queries to the database. E_k repositions all symbols in the names of tables. Being aware that legal users shall reposition all symbols in a message beforehand for obtaining a correct SQL-query as output. Then if the function for SQL-query modification is made dependent on time, *i.e*. turned into a dynamic function, we get a *standard* MTD scheme. When the interface occurrence is made dependent on time and thus becoming dynamic then a *multilevel* MTD scheme is established.

To enable the information stream reaching an addressee through several MTD nodes both common interface table I and personal history of entities I_E have to be stored.

Let us review data transmission in a system of two work objects and four entities. The two work objects may be represented for instance by databases DB_1 and DB_2. Every entity may be an interface for the databases.

The initial interface table used is as follows (Table 1).

Table 1. Initial Table of *I* Interfaces

Name	Address	Input	Output
DB_1	E_1	-	-
DB_2	E_2	-	-

The first column here is an identifier of a required database, the second column contains current addresses represented as identifiers of entities, third and fourth column are functions with input and output values. Functions are empty at the first step. All tables with history of interfaces are also empty.

At step two entity E_3 becomes an interface for database DB_1 and transforms input values using function f'. A new I interface table is obtained.

Table 2. Table of *I* Interfaces at Step 2

Name	Address	Input	Output
DB_1	E_3	f'	-
DB_2	E_2	-	-

The first line appears in I_{E3} interface table. (Table 3).

Table 3. Table of Interface History I_{E3} at Step 2

Name	Address	Input	Output
DB_1	E_1	f'	-

Assume that at step 3 entity E_4 becomes an interface for DB_1 with function f'', and at step 4 entity E_3 becomes an interface for DB_2 with function f'''. A more complex scheme is obtained shown in Tables 4, 5 and 6.

Table 4. Interface Table *I* at Step 4

Name	Address	Input	Output
DB_1	E_4	$f'f''$	-
DB_2	E_3	f'''	-

Table 5. Table of Interface History I_{E3} at Step 2

Name	Address	Input	Output
DB_1	E_1	f'	-

Table 6. Table of Interface History I_{E4} at Step 2

Name	Address	Input	Output
DB_1	E_3	f''	-

Now the scheme of query flow looks more complex as shown in Figure 5 below.

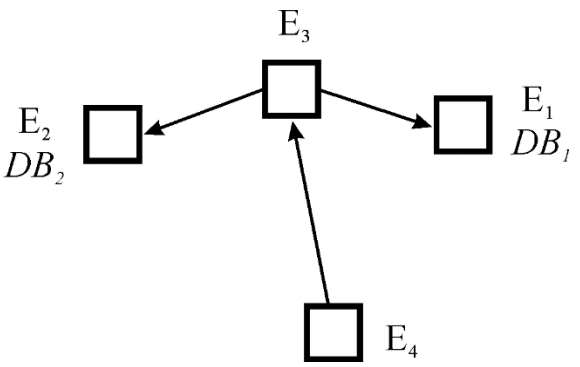

Figure 5. Scheme of Query Flow to Databases

Here entity E_3 shall now review its own interface history, redirect data and perform transformations depending on where to a query is sent. Hence, any query to DB_1 goes through three interfaces in the system and a query to DB_2 goes through two interfaces.

Several difficulties occur in such scheme:

1. How to generate the function for transformation?
2. In what form that function shall be transmitted?
3. How to limit infinitely increasing complexity of data exchange in the system?
4. How to avoid exceeding the authorization limits in assigning an interface status to a process?

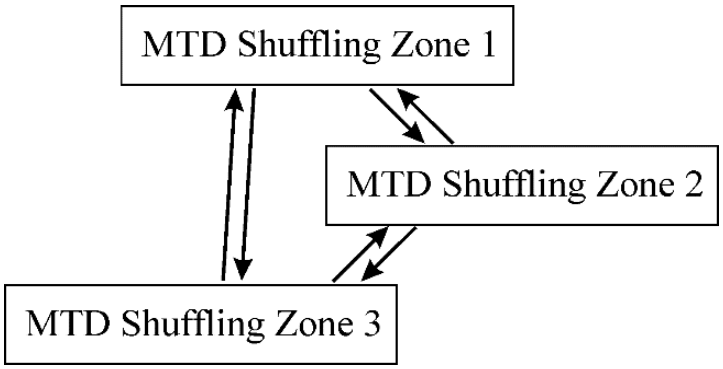

Figure 6. Separation of Shuffling Zones by Access Level

The first three problems listed above do not have a general solution as yet, but they may be solved in particular cases, for example when it is known that some specific data structures and protocols are dealt with.

In real life systems where every process has access to another process are certainly seldom to be found. However by grouping processes by access category we can define zones for shuffling such processes (MTD Shuffling Zones). That also includes an option that several systems' elements by several parameter ranges may become interfaces for one process (Figure 6).

3. Establishment of a Multilevel Decentralized MTD

Let us review a node shuffling scheme in a corporate LAN as example of practical implementation. That task has been reviewed many times from the point of MTD because remote scanning of a corporate LAN precedes most of attacks at internal corporate resources.

Address spoofing in LANs is usually associated with shuffling of host IP-addresses. Examples of such shuffling may be found in references [8-11] and [12]. The indicated papers suggest separating physical and virtual (used for current routing) host address and verification of noninfringement of the security policy which results in packet transmission through nodes which do not have access to that information [13].

All systems reviewed in the mentioned papers may be referred to centralized shuffling schemes, when centralization may be regarded as presence of one center which defines algorithms for modification and verification of hosts, or presence of a preset algorithm which governs address shuffling in a network. Such system has advantages as it allows defense from remote scanning. However, when an adversary has obtained the operation algorithm of the MTD scheme or it acquires access to one of the legal nodes of a network it will probably find out the addressing scheme.

In the MTD scheme described above, function *f(k)* shall be defined as addressing modification function and hosts of the network shall be nodes. Assume that initially there are four hosts in the network Host 1, Host 2, Host 3 and Host 4. Virtual addresses for each of the hosts are identical to their real addresses. That is when a host needs to send information to another host it shall send packets to the receiving host directly. The next moment any other host may initiate address exchange and function as the interface with another host. Host 2 and Host 3 exchange addresses on Fig. 6. In that case Host 3 sends new addresses of Host 2 and Host 3 to the others (function $f^{-1}(\cdot)$). Then Host 1 considers that Host 3 is Host 2 and sends it packets as if for Host 2. Host 3 in its turn transmits it to a real addressee (Figure 7).

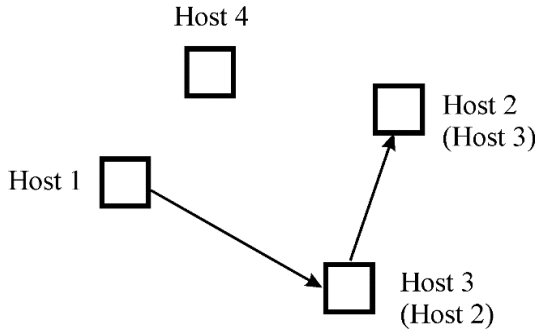

Figure 7. Host 3 and Host 2 Became Interfaces for Each Other

After that any host in the network may initiate the next exchange. For instance Host 4 exchanges addresses with Host 2 (it is the host that it sees in the routing table) by sending the new routing table to the others (Figure 8). Eventually when Host 1 sends a packet to Host 2 the packet goes through intermediate nodes (as shown on the scheme in Figure 2).

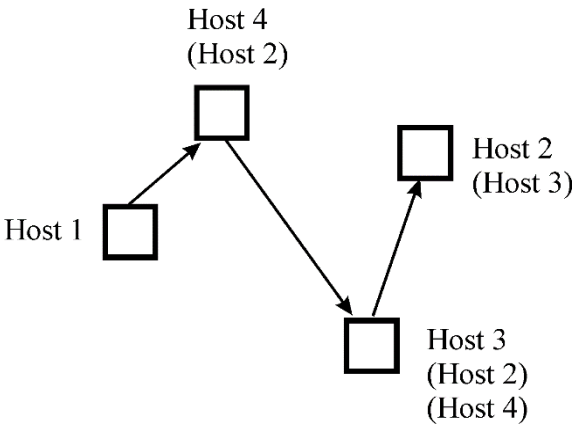

Figure 8. Host 4 and Host 2 Became Interfaces for Each Other

Such decentralized network address shuffling is undoubtedly the most efficient in terms of network's protection from research. In this case even legal hosts do not have real addresses of other hosts. A similar scheme may be implemented when a security administrator will not know the network's routing structure after that process is started. That is the system makes itself more complex and shuffles itself.

That scheme has been implemented in a real network at a software level and several problems were found, the main one was excessiveness of traffic in packet transfer. The chosen solution was to store the routing table as a distance which a

packet has to go to reach the destination addressee. Hence, that distance may be limited. When the value approaches the limit a host may either reshuffle with the last host in the chain and thus set dimensions to nil or make changes in the routing table of intermediate hosts by removing nodes from it.

4. Experiment

Implementation of the method based on segmentation of a network at the data link layer with dynamic VLANs and using a centralized MTD scheme was tested at the current stage of research. The MTD scheme is based on algorithmic shuffling for IP address with calculated address of a destination node. The list of equipment used: D-Link DGS-3120-24TC L3 level switch, RADIUS server (Elektron RADIUS server), secured node with Windows 2012 Server, and destination nodes managed by Windows 7.

Were investigated the following critical characteristics of the network:

1. Time for dynamic reconfiguration of network.
2. Number of packets lost when rebuilding the network structure.
3. The period of time required for modification of network structure.

A common ping utility based on ICMP requests was used to identify the time required to rebuild the network structure.

We considered an average number of lost requests for 20 reps. We checked availability of the site at the new IP address and request count was conducted at the same time. The counting included only the requests sent after initiation of the procedure of rebuilding the network and that were recorded in the traffic dumps on the polling and polled nodes. In this configuration of the stand and in various VLAN schemes the average number of lost packets was 5-6. That corresponds to 6-7 seconds time, which is required to rebuild the network.

In order to define intervals of time in which the network should be rebuilt we should determine how long it will take an attacker to scan a subnet to determine active nodes and ports used on them. For that purpose NMAP should be used to define the time required to identify 4 nodes on subnets of different capacity and their ports shall be checked as presented below:

1. Range: N.N.N.0-255. Availability scan: 36-40 sec. for 4 nodes. Port scan: 250-270 sec. for 1 node. Port scan: 1000-1100 sec. for 4 nodes.
2. Range: N.N.0-1.0-255. Availability scan: 42-50 sec.
3. Range: N.N.0-4.0-255. Availability scan: 61-81 sec.
4. Range: N.N.0-255.0-255. Availability scan: 3111-3305 sec.

Network scan was performed from one node in an automatic mode. With regard to the above experiment it is recommended to reconfigure a test network within 300-360 sec., in case an attacker has access to the node included in the protected subnet. MTD scheme increases time for port and availability scan for the attacker.

5. Conclusions

The present paper conceived an arrangement scheme for information systems that allows multiple changes of internal structure without disrupting operation of standard processes in the system. This design method may be regarded as multilevel and decentralized implementation of the Moving Target Defense technology. Therefore, an information system resistant to research by intruders at the reconnaissance stage is created. The scheme offered herein is more effective as compared to the existing implementations of the Moving Target Defense technology as the scheme for modification of an information system is resistant to research as well as the system.

Obtained practical schemes have high software and hardware redundancy but they may already be successfully implemented in low-load information systems in which residual risk value is highly critical.

Acknowledgments

This work was supported by the Federal Grant-in-Aid Program under Governmental Contract No. 14.574.21.0126, unique identifier RFMEFI57414X0126.

References

[1] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, X. S. Wang, "Moving Target Defense. Creating Asymmetric Uncertainty for Cyber Threats", Springer, Series: Advances in Information Security, **(2011)**.
[2] S. Jajodia, A.K. Ghosh, V. Swarup, C. Wang, X. S. Wang, "Moving Target Defense II. Application of Game Theory and Adversarial Modeling", Springer, Series: Advances in Information Security, **(2013)**.
[3] M. Carvalho and R. Ford, "Moving-target defenses for computer networks", IEEE Security and Privacy, vol 12, no. 2, **(2014)**, pp. 73-76.
[4] Q.D. JafarHaadiJafarian and Ehab Al-Shaer, "Openflow random host mutation: Transparent moving target defense using software-defined networking", Proceedings of the 1st Workshop on Hot Topics in Software Defined Networking (HotSDN), **(2012)**, pp. 127-132.
[5] A. Paulos, P. Pal, R. Schantz, B. Benyo, "Moving target defense (MTD) in an adaptive execution environment", ACM International Conference Proceeding Series. 8th Annual Cyber Security and Information Intelligence Research Workshop: Federal Cyber Security R and D Program Thrusts, CSIIRW 2013, **(2013)**, pp. 145-156.
[6] J. Li and R. Sekar, "Address-space randomization for win-dows systems", Preceedings of 2006 Annual Computer Security Applications Conference (ACSAC), **(2006)**, pp. 329-338.
[7] M. Styugin, "The New Method of Security Development for Web Services Based on Moving Target Defense (MTD) Technologies", Proceedings of the International Conference on Network Security and Communication Engineering (NSCE2014), **(2015)**, pp 55-59.
[8] E. Al-Shaer, Q. Duan and J.H. Jafarian, "Random host mutation for moving target defense", 8th International ICST Conference on Security and Privacy in Communication Networks, SecureComm, **(2012)**, pp. 310-327.
[9] T.E. Carroll, M.B. Crouse, E.W. Fulp and K.S. Berenhaut. "Analysis of network address shuffling as a moving target defense", 2014 IEEE International Conference on Communications, **(2014)**, pp. 701-706.
[10] S.A. DeLoach, Xinming Ou, R. Zhuang, and S. Zhang, "Model-driven, moving-target defense for enterprise network security", Dagstuhl Seminar 11481 on Models@run.time. LNCS, **(2014)**, pp. 137-161.
[11] J. Yackoski, P. Xie, H. Bullen, J. Li, K. Sun, "A self-shielding dynamic network architecture", Proceedings - IEEE Military Communications Conference MILCOM, **(2011)**, pp. 1381-1386.
[12] J. Yackoski, P. Xie, H. Bullen, J. Li, K. Sun, "Mission-oriented moving target defense based on cryptographically strong network dynamics", ACM International Conference Proceeding Series. 8th Annual Cyber Security and Information Intelligence Research Workshop: Federal Cyber Security R and D Program Thrusts, CSIIRW, **(2013)**, pp. 498-502.
[13] E. Al-Shaer, W. Marrero, A. El-Atawy, K. Elbadawi, "Network configuration in a box: Towards end-to-end verification of network reachability and security", Proceedings - International Conference on Network Protocols, ICNP 2009, **(2009)**, pp. 123-132.

Authors

Mikhail Styugin, he is a senior lecturer at Siberian Federal University and a scientist at Siberian State Aerospace University (Krasnoyarsk, Russia). PhD degree in computer science. Conducts research in the area of information security systems and technologies of information warfare. Owner of two companies that develop solutions in the area of information security systems on the Internet.

Nikolai Parotkin, awarded with PhD in System analysis, management and information processing by the Siberian Aerospace University (Krasnoyarsk, Russia, 2013). Main research interests are data mining, security of data networks.