A Layered Approach to Understanding Network Dependencies on Moving Target Defense Mechanisms

Sean Crosby
Sandia National Laboratories
smcrosb@sandia.gov

Marco Carvalho
Department of Computer Science
Florida Institute of Technology
Melbourne, FL
mcarvalho@fit.edu

Daniel Kidwell
US Department of Defense
dlkidw2@tycho.ncsc.mil

ABSTRACT

Moving target defense is a conceptual shift from the standard defense mechanisms used on today's statically configured networks. Most mechanisms for moving target defense are limited in the protocols and applications they can support. In this paper, we outline a simple method to identify the network dependencies these have and begin the discussion of where mitigations for dependencies broken by these mechanisms should reside.

Keywords

Moving target defense (MTD)

1. INTRODUCTION

Computer networks are typically designed to operate in relatively static environments. Services and applications are compiled, installed and configured on specific machines running pre-specified operating systems, configured in pre-defined networks, with pre-defined names. The process requires the configuration of hosts, routers, name servers, firewalls, and applications; and only then, the system is considered operational. Once deployed, the system is ready to perform its mission, and is also immediately subject to attacks and other adversarial activities such as seeking to observe, map and understand the system design in order to identify and exploit possible vulnerabilities. Defenders, at the same time, seek to win the race in locating and identifying the same vulnerabilities, hopefully fixing them before they are exploited by adversaries.

The traditional defense paradigm is based on the existence of this relatively static computer network that is the objective *target* of both attackers and defenders, on a constant race to identify and exploit (or patch) vulnerabilities. This has been, and to a great extent continues to be, the state of the art in computer network defense.

Moving target defense (MTD) proposes a conceptual shift in this paradigm. The MTD concept proposes that the target itself does not need to be static, and that a dynamic (or moving) target design can be conceived such that it maintains functionality for legitimate users, while making it difficult for adversaries to identify and exploit system vulnerabilities.

A dynamic computer network infrastructure has a changing structure and likely a changing set of vulnerabilities that can be located and exploited. Rather than focusing on the identification of vulnerabilities and fixes of a fixed system (or target), moving target defense techniques focus on the continuous changes and mobility of the target itself, as a means to protect it.

Conceptually, a moving target defense relies on sets of tools or mechanisms responsible for monitoring the state of the computer network, and a set of tools or mechanisms responsible for the mobility, or effectively changing the system.

In the recent years, a number of mobility tools and mechanisms have been proposed to implement the moving target concept. Mobility techniques have been introduced at essentially all levels of the system, ranging from the actual computing architecture and platform to operating system applications and network configurations. Each of these capabilities are usually described, illustrated and often demonstrated in isolated experiments showing their potential effectiveness for attack scenarios and applications.

Consider, for example, the moving target capability defined as Dynamic Network Address Translation (DYNAT) [1]. DYNAT introduces the concept of address hopping, which is a technique for reassigning IP addresses within a network to confuse adversaries. Michalski, et. al. [2] present a summary of DYNAT techniques, which can be implemented in hardware or in software. Like many others, DYNAT provides an example of the potential of the moving target concept for computer network networks.

However, a careful review of a large number of these tools and capabilities reveals that not much attention has been paid to the intrinsic interdependencies and side effects caused by target mobility. There are critical dependencies in services and protocols that may appear only in broader application scenarios, and are often overlooked in the analysis of specific tools and capabilities.

The practical deployment of moving target concept requires a careful review and understanding of these interdependencies, as well as the thoughtful design of broader coordination mechanisms to control and mitigate these effects.

In this paper, we bring forward an informal discussion of this problem, and identify a few examples to illustrate such dependencies. Introducing new security mechanisms into a network may increase the attack surface of the system. There is a natural trade off that must be considered in the implementation of any defense capability. While other research effort have focused primarily on the issue of complexity mitigation, for example

CSIIRW '12, Oct 30 - Nov 01 2012, Oak Ridge, TN, USA
ACM 978-1-4503-1687-3/12/10

through command and control coordination [5], this aspect of the problem is not directly relevant to our discussions and will not be addressed in this paper.

We believe that an early consideration for a holistic view of moving target defense, including its unintended effects and interdependencies, is critical for the eventual success and acceptance of the concept. In this paper, we look at moving target defense from the perspective of legitimate users, and the intended functionality of the systems to better understand the potential impacts of the concept, and envision possible ways forward to mitigate these issues.

2. RELATED WORK

The designers of various MTD mobility mechanisms have identified use-cases under which their mechanisms can fail. Repik [3] described some of the application interoperability issues associated with address hopping. Namely, applications that contain IP address and port information in the data payload, protocols that employ both a control and data port, and peer-to-peer applications may become inoperable when address hopping is employed. The DYNAT Decision Tree in [2] helps determine the compatibility of a particular network with the different types of DYNAT processes. This outlines some of these dependencies but does not put them in perspective of the network stack.

Others identified that "the underlying MTD system must have an understanding of the functional and security requirements of the system" [4]. They keep these requirements in a logical model and reference it when making changes to the network.

In this work, we focus on the functional requirements of the protected network. We identify and discuss the interdependencies of such requirements and the potential impact that moving target defenses may have on them. We posit that any system that coordinates changes within a network needs to have a sound understanding of these requirements and their interdependencies.

Numerous MTD mobility mechanisms have been proposed at network, operating system, and application levels. To simplify our discussion, and without loss of generality, we will present our argument for analysis of dependencies only in light of mobility occurring on the network level.

3. METHODOLOGY

Modern networks operate over standard protocols and services. The users (e.g. people, tools, applications) utilizing the network rely on these protocols and services to operate properly. Users can also be dependent on attributes, such as IP address (see Figure 1).

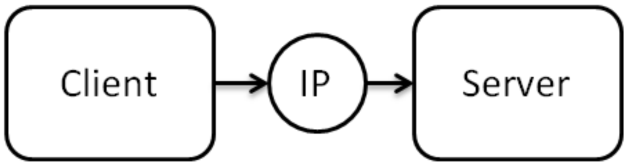

Figure 1: Illustration of a client's dependency on a server's IP address for connecting to that server.

If the IP address for a host changes a request attempting to access that host via that IP address will fail.

As dependencies can be rooted in many different components, attributes and services, a few noteworthy examples include:

- Physical connections
- Network addresses (e.g. IP, MAC)
- Security configurations (e.g. MAC port locking)
- Application state (e.g. IP addresses stored in payloads or application state)
- Inter-packet dependencies (as in TCP and other Session layer protocols)
- Encryption keys (for SSL and other encrypted protocols
- Packet timing

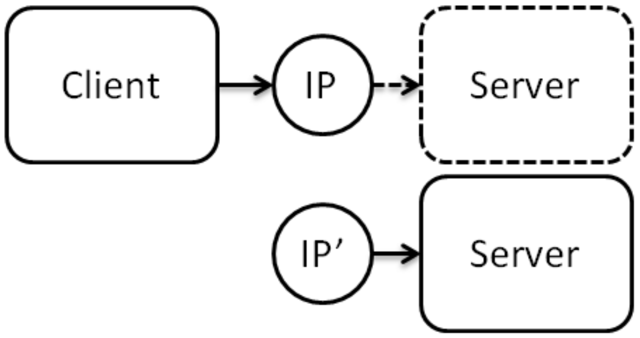

Figure 2: Illustration of a client's broken dependency after the server's IP address has changed.

Protocols, services, and attributes within a network are generally static. Hence, when an adversary gathers network information (e.g. IP addresses, host names, services, etc.) for use in an attack, he forms a dependency on this information.

The dependencies considered in this work are intrinsic to the systems under consideration. While static for a given system (including services and support infrastructure), they may be represented in different ways to users (and attackers). The conceptual role of moving target defense if to provide a perceived disruption of such dependencies to adversaries while preserving, restoring, or patching the dependencies of legitimate users.

MTD mobility mechanism design must consider three aspects of dependencies: first, identifying the dependencies that an adversary has on network protocols, services, and applications; second, breaking those dependencies to confuse, delay, or discourage the adversary; and third, designing mechanisms to mitigate the broken dependency for legitimate users. Because the adversaries and legitimate users generally utilize the same communication channels, breaking the dependencies in the network also affects legitimate users unless further action is taken. Deterring the adversaries and not the legitimate users is one of the primary difficulties in MTD mechanism design. If a mechanism breaks a dependency all network users have on the system, it must also have a function for restoring or patching the dependency for at least legitimate users.

Consider an example of IP address hopping. A server offering service *S* is configured to randomly change its statically assigned IP address every time *t*. A legitimate client has the Server's IP address and is making successful requests to *S* until time *t*. At this point the client needs to acquire the new IP address and this can be done by several mechanisms:

- The user walks over to the server administrator and asks what the new IP address is and then uses it.

- The user's client application knows the hopping pattern of the server and updates the IP address automatically.
- The user's Ethernet adapter (virtual or physical) understands the hopping pattern of the server and updates the request to include the new server IP address.
- A router, which understands the hopping pattern, uses Network Address Translation (NAT) to update the IP address in the request.

A switch operates at the Data-Link layer (layer 2) and forwards traffic based on MAC addresses, not IP addresses, and isn't the location to implement a Network layer (layer 3) mechanism.

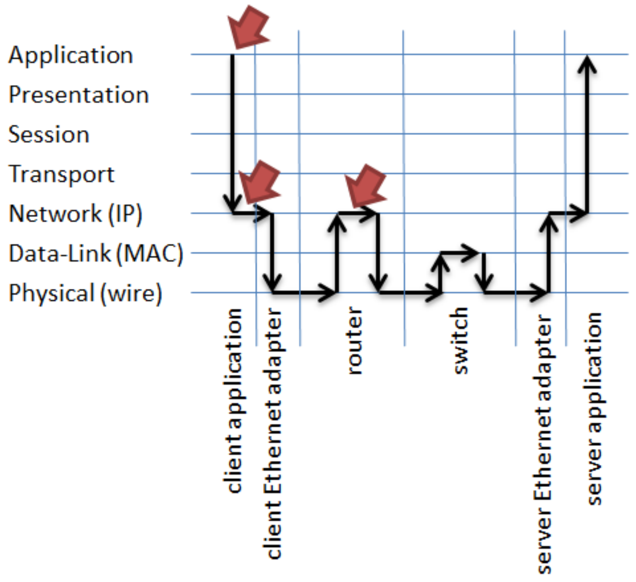

Figure 3: Points at which the IP address in the IP header of a packet can be corrected during transmission to a server.

3.1 A Layered View of Dependencies

The OSI Model (see Table 1) provides a high level abstraction of the layout of a network. Although not all protocols and applications fit within these seven categories, this is a good abstraction for understanding the dependencies between layers of the network. Each layer provides services to the layers above it. Also, layers can have dependencies on the layers below.

Table 1. The OSI Model

Layer	Name	Relevant protocols and devices
7	Application	HTTP, HTTPS, FTP, DNS
6	Presentation	SSL
5	Session	TCP
4	Transport	TCP, UDP
3	Network	Internet Protocol (IP), routers
2	Data-Link	Ethernet, MAC, switches
1	Physical	hubs

These dependencies create requirements for how the network should behave. For example, the Internet Protocol (Network layer) has a binding on MAC addresses (Data-Link layer) to forward packets. The HTTP protocol (Application layer) ultimately relies on the layers beneath it including protocols in the Session, Transport, Network, Data-Link, and Physical layers (see Figure 4).

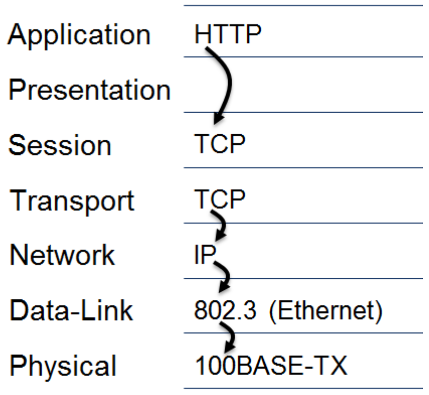

Figure 4: Basic protocol dependencies of the HTTP protocol

Each layer of the network needs to provide consistent services to the layers above it. A layer can mask the services of lower layers as long as it provides a proper emulation of those layers. An example of this is a virtualized local area network (VLAN). A VLAN does not exist at the Data-Link layer, but it offers Data-Link layer services. It is not required to exist directly on top of the Physical layer because it emulates the Physical layer services beneath it. This way, applications living in a VLAN can use it transparently because it provides all the services of the Physical and Data-Link layers.

3.2 MTD Design: a Layered Approach

MTD mobility mechanisms make a change in one (or more) network layers. This change is designed to interrupt an adversaries' service. To continue service to legitimate users, the mitigation to this change must occur at this same layer or at a higher layer where there is sufficient knowledge and visibility to perform the mitigation. For example, if an IP address is included in the payload of a packet, this correction cannot be made at the Network Layer (where IP is rooted) but at the application layer where the payload can be parsed and modified. This means that the mitigation mechanism must exist either in the client application or in another application that sits along the path of communication.

Because a protocol or application can rely on a protocol in any network layer beneath it, we must analyze the effects of breaking a dependency. If issues are caused at a higher layer, our mitigation mechanism may need to operate at that layer or higher. For example, protocols with inter-packet dependencies, such as Session layer protocols, need to be mitigated at the session layer or higher. This mitigation does not always have to take place on the client machine, but somewhere along the path of communication where there is sufficient control and visibility to perform the mitigation.

If a non-mitigated client is sending large amounts of data to the service *S* in a single TCP session and in the middle of the transmission, the server changes its IP address, transmission would fail. TCP depends on the IP address remaining the same. One solution is to instrument the client and server applications to be aware of the oncoming change, close out the TCP session, update the server IP address on both the client and the server, and begin a new TCP session that resumes the data transmission where the previous one left off.

4. CONCLUSIONS

If the concept of moving target defense is to become a reality, designers and engineers of specific MT capabilities must have the necessary tools to understand the implications of their deployment and use. Furthermore, we must identify and develop supporting mechanisms to manage and coordinate multiple defenses to maximize the defense while maintaining the functionality of the system.

We propose that dependency graphs for attacks and for the functionality of the system could be used together to allow for the deployment, configuration, and runtime management of moving target defenses. If properly managed, these capabilities could be constructed to disrupt attack dependencies, while maintaining the necessary dependencies for system functionality.

As discussed in this paper, a layered approach for the analysis of the interdependencies can also play an instrumental role in the design of such tools, enabling the proper identification of possible contention points. The proposed approach can also be instrumental for the effective deployment, management and runtime monitoring of moving target defenses in network systems.

The proposed approach for system design and management can be implemented in multiple ways. We believe that a human-assisted automation can play an important role in the management of mobility-based defense, where humans could help contextualize large-scale dependency graphs created from they system specification, and through runtime learning.

Human-agent teamwork command and control capabilities can also play an important role in the coordination of interdependencies of moving target defense, leveraging the collaborative work of humans and autonomous software agents for monitoring and control.

Nevertheless, regardless of how it could be realized, the proposed layered approach for the high-level analysis and management of defense interdependence requirements is, in our view, fundamental for the practical deployment of moving target defenses.

5. REFERENCES

[1] Kewley, D.; Fink, R.; Lowry, J.; Dean, M.; , "Dynamic approaches to thwart adversary intelligence gathering," DARPA Information Survivability Conference & Exposition II, 2001. DISCEX '01. Proceedings , vol.1, no., pp.176-185 vol.1, 2001

[2] Michalski, J.; Price, C.; Stanton, E.; Chua, E. L.; Seah, K.; Heng, W. Y.; Pheng, T. C. Final report for the network security mechanisms utilizing network address translation LDRD project. Technical Report SAND2002-3613, Sandia National Laboratories, November 2002.

[3] Repik, K. M. (2008). Defeating Adversary Network Intelligence Efforts with Active Cyber Defense Techniques. Air Force Institute of Technology.

[4] Zhuang, R.; Zhang, S.; DeLoach, S. A.; Ou, X.; Singhal, A. Simulation-based Approaches to Studying Effectiveness of Moving-Target Network Defense. National Symposium on Moving Target Research. June 11, 2012, Annapolis, MD.

[5] Carvalho, Marco, J.M. Bradshaw, Larry Bunch, Tom Eskridge, Paul J. Feltovich, Robert H. Hoffman, and Daniel Kidwell. Command and Control Requirements for Moving Target Defense. IEEE Intelligent Systems, May/June 2012 (vol. 27 iss. 3), pp. 79-85.