MAT JOURNALS

Journal of Computer Science Engineering and Software Testing

Volume 1 Issue 2

Denial of Service: Techniques of Attacks and Mitigation

Neeta Sharma[1], Mahtab Alam[1], Mayank Singh[2]

[1]Department of Computer Science Engineering, Noida International University, India
[2] Department of Computer Science Engineering, Krishna Engineering College, India
E-mail: neeta.sharma@niu.ac.in, mahtab.alam@niu.ac.in, mayanksingh2005@gmail.com

Abstract

As cloud computing technology has many advantages but cloud security or cloud software security threats and attacks at various levels are also a big concern of all the organizations. Those systems are connected to the internet in the cloud network can be effected by different types of attacks and one of the prominent attack is DoS (denial of service) attack. DoS attack has been considered as one of the important security threat in cloud computing systems at various level that has proven difficult to alleviate. This attack perpetrated in many ways such as consuming computational resources, disruption of information and obstructing the communication media. Once the attack is successful in consuming resources on the victim computers, the attacker then could control and direct them to attack as a group. This means DoS attack also allows the attacker to get the administrative control of the systems. Dos attack can be launched for sending the flood or crashes the services of the system. In this paper, we present the different types of DoS attacks and techniques for launched the DoS attack at various level and the techniques applied to mitigate the harmful effects of the DoS attack.

Keywords: *DoS, buffer overflow, cloud computing, flood, cloud server*

INTRODUCTION

Cloud computing is an important paradigm, with the potential to significantly reduce costs. The complexity and opportunity of Cloud Computing Systems weaknesses are regularly rising with the use of this technology. Due to the high growth rate of expertise of malicious users and its existing

security holes cloud computing systems going to becomes insure. The requirements for secure cloud software are concerned with nonfunctional issues such as minimizing or eliminating vulnerabilities and ensuring that the software will perform as required, even under attack. Developing secure software is based on applying the secure software design principles that form the fundamental basis for software assurance. Software must be able to resist most attacks, tolerate as many as possible of those attacks it cannot resist, and contain the damage and recover to a normal level of operation as soon as possible after any attacks it is unable to resist or tolerate [1]. For this reason we also check that our software should be able to mitigate the problem. DoS attack using Buffer Overflow techniques are the most common security intrusion attack. Since, IaaS (Infrastructure as a Service) supports multiple virtual machines; it provides an ideal platform for hackers to launch attacks like DoS attack, which require a large number of attacking instance. Currently many DoS tools are available to compromise the system first then exploit known vulnerabilities to gain the access to system which they use to launch further attacks. In other words, a remote attacker could exploit the vulnerability to execute arbitrary code or cause a denial-of-service. It can severely limit the ability of an organization to perform normal business on the internet. One recent report can be found in CERT Advisory is that the distributed denial-of-service tool called "Stacheldraht" has been discovered on multiple compromised host at several organizations [2]. One common method of attack involves saturating the target mechanism with external communications requests so much, so that it cannot respond to legitimate traffic or responds so slowly as to be rendered essentially unavailable. Such attacks usually lead to a server overload.

A DoS attack generally consists of efforts to temporarily or indefinitely interrupt or suspend services of a host connected to cloud using internet and sometimes responsible for website attacks also best known for cloud security vulnerability, as DoS attack can be performed in legacy as well as newly developed technology.

TYPES OF DOS ATTACKS

Distributed Denial of Service Attack

It occurs when multiple systems devise a synchronized DoS attack to a single target. In this attack, the target is attacked from many locations at once. In other words, the DDoS attack makes used of many different source to send a lots of useless packets to the target in very short time, which will consume target resources and make the target's services unavailable. Among all the networks attacks, the DDoS attack easy to carry out, more harmful, difficult to prevent tough to detect, so it is more serious [3].

HTTP POST DDoS Attack

In this attack, sends a complete, legitimate HTTP POST header which includes a "Content-Length" field to specify the size of the message body. Then attacker proceeds to send the actual message body at an extremely slow rate nearly 1 byte/110 seconds. Because the message being complete and correct than the target server will attempt to obey the "Content-Length" field in the header and wait for the entire body of the message to be transmitted, thus slowing it down [4].

Permanent Denial of Service

It is purely hardware targeted attack which can be much faster and required fewer resource than using botnet in DDoS. It is also known as phlashing that damages a system so badly that it requires replacement or reinstallation of hardware. Contrasting the DDoS attack, PDoS attack exploit security flows which allow remote administration on the management interfaces of the victim's hardware, such as printers, or routers etc. The attacker uses these vulnerabilities to replace a device's firmware with corrupt firmware and this process known as flashing. The potential and high probability of security exploits on (NEEDs) Network Enabled Embedded Devices. PhlashDance is a tool used to detect and demonstrate PDoS vulnerabilities [5].

XML Denial of Service (XDoS)

These are less common than unintentional XDoS attacks which occur when a programming error by trusted customer causes a handshake to go into an infinite loop. Main purpose of this attack is to shut down a web service or system running that service. It occurs when an XML message is

sent with a multitude of digital signatures and a naïve parser would look at each signature and use all the CPU cycles, eating up all resources.

Advanced Persistent DoS Attack (APDoS)

DoS attack which is simultaneous and persistent in the network is known as APDoS. This involves massive network layer DDoS attack through to focus application layer (HTTP) flood, followed by repeated SQLI and XSS attacks. It signify a clear and emerging threat needing specialized monitoring and incident response services and the defensive capabilities of specialized DDoS mitigation service providers. Attacker can use 2 to 5 attack vectors involving up to several tens of millions of requests per second and it persist for several weeks noted time 38 days. Attacker is tactically switches between several targets to create a diversion to avoid defensive DDoS countermeasures but at the same time also concentrating on primary victim. With continuous access to several very powerful network resources are capable of sustaining an extended crusade generating enormous levels of un-amplified DDoS traffic [4].

METHODS OF DOS ATTACKS

There are generally two methods of DoS attacks which are as follows:

Flooding Attacks

It occurs when the system receives too much traffic for the server to buffer, causing them to slow down and eventually stop.

- Buffer Overflow Attack-Network Level
  - SYN
  - ICMP
  - Slow Read attack
- Buffer Overflow Attack-Sever Level
  - Pointer Subterfuge
  - Arc Injection
  - Heap Smashing
  - Stack Smashing

Crashing Services

In this attack, attacker simply exploits vulnerabilities that cause the target system or service to crash. In these attacks, input is sent that takes advantage of bugs in the target that subsequently crash or severely threaten the system, so that cannot be accessed or used.

- Teardrop Attack
- NUKE Attack

Flood Attacks

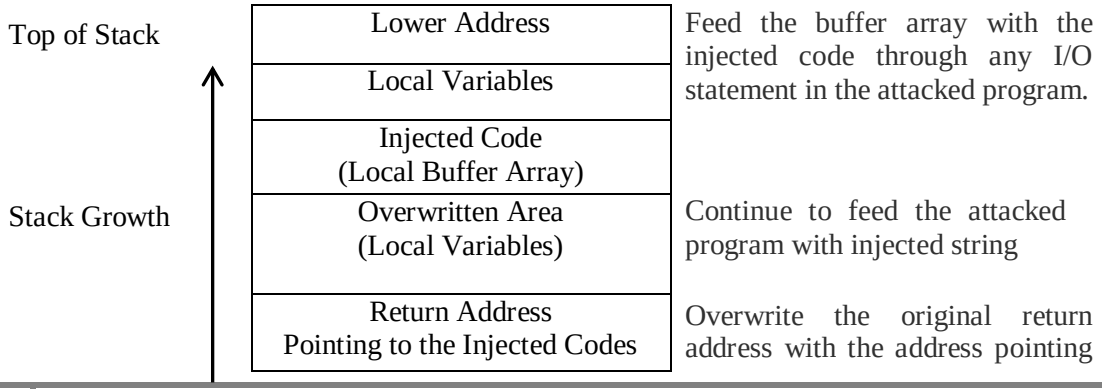
These are as follows:

Buffer Overflow Attacks (Network Level)

This is most common DoS attack which also known as application level flood. In this attack attacker send more traffic to a network address than the programmers have built the system to handle. It includes the ICMP flood and SYN flood attacks which are designed to exploit bugs specific to certain applications or networks in the cloud computing environment [6].

Buffer Overflow/Overrun Attacks (Server Level)

This attack also be consider as a type of DoS attack since it also hampers the proper functioning of the web server and application server resulting into denial of services for cloud computing environment. There seems confusion amongst some professionals in regard to the meaning of Buffer overflow and buffer overrun but essentially they are the same. Like in Buffer overrun attacks obviously occur in any program execution that allows input to be written beyond the end of an assigned buffer. Thus, it leads the data to overwrite into adjacent memory locations which are already occupied to some existing code instruction. Buffer overflow mainly consist the following three steps [7]. Planting the attack code into the program, copying into the buffer which overflows it and corrupts adjacent data structures, and hijacking the program to execute code. Commonly buffer overflow can be executed by using the stack smashing shown in the Figure 1 [8].



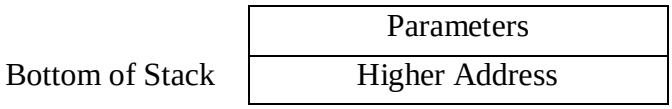


Fig. 1: *Fragment of Stack.*

CRASHING SERVICES

In this attack, attacker simply exploits vulnerabilities that cause the target system or service to crash. In these attacks, input is sent that takes advantage of bugs in the target that subsequently crash or severely threaten the system, so that cannot be accessed or used. These types of attacks are list below:

Teardrop Attacks

It involves sending mangled IP fragments with overlapping, over-sized payloads to the target machine which can crash various operating systems because of a bug in their TCP/IP fragmentation re-assembly code. Teardrop attack was referred in Windows Vista around September 2009[9].

NUKE

One of the old denial-of-Service attack against computer networks consisting of fragmented or otherwise invalid ICMP packets send to the target, achieved by using a modified ping utility to repeatedly send this corrupt date, thus slowing down the affected computer until to come to a complete stop.

METHODS OF BUFFER OVERFLOW (NETWORK LEVEL)

SYS (Synchronize) Flood

In this category, attacker sends a request to connect to server, but never complete the handshake. It continues until all open ports are saturated with requests until after the attacks ends [9].

ICMP (Internet Control Message Protocol) Flood

It leverages misconfigured network devices by sending spoofed packets that ping every computer on the targeted network, instead of just one specific machine. The network is then triggered to amplify the traffic. This attack is also known as the smurf attack, ping flood or ping of death [6].

Slow Read Attack

It ensures very slow data flow rate by sending legitimate application layer requests trying to exhaust the server's connection pool.

METHOD OF BUFFER OVERFLOW (SERVER LEVEL)

Pointer Subterfuge

Pointer Subterfuge is a common approach for exploitation by modification of pointer address to avert the control flow of a program by using function pointers as an alternative to the saved return address or change the program flow by subverting data pointers is illustrated as below [10, 11]:

```
Void Func_Name ()
{           // do something
        Statement…….1...
        ……
        Statement……..=N

}
```

```
typeset void (*FUN_PTR) (void);
int FuncMal (char *ptString)
{
        Char buf;

        strcpy (buf, szString);

FUN_PTR fp= (FUN_PTR)
(&Func_Name);
        // other code

        (*fp)();
    return 0;
    }
```

Arc Injection

It is sometimes also called as return into-libc transfer control of the code that already exists in the memory space. An exploiter uses the arc injection to invoke a number of functions in a small program that includes chained functions in sequence with arguments that are supplied by them. Following are the some functions used in arc injection buffer overrun vulnerability [12].

- excel()
- system()
- printf()

Out of three function print f is a most popular function in C programming can be used for exploitation of a program using %n, n$ and %3$n.

Heap Smashing

It overruns a heap buffer to change the control flow of a program. It allows an attacker to exploit the software by implementing some assumed variants in dynamically allocated memory and less common in practice. A typical example of heap smashing is shown in the Figure 2 [13].

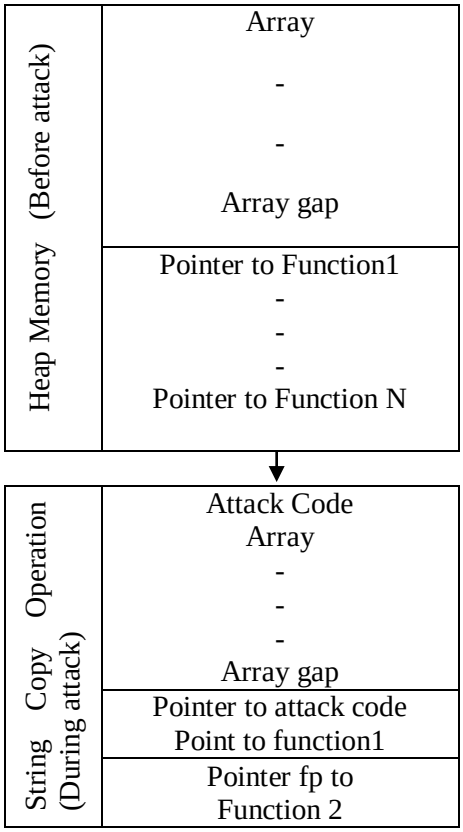


Fig. 2: *Heap Smashing Attack.*

Stack Smashing

It can be achieved by using the technique Stack buffer overflow used to gain unauthorized access to a computer. If the stack buffer is filled with the data supplied from an untrusted user than that attacker can corrupt the stack in such a way as to inject executable code into the running program and take control of the process [14–16].

MITIGATION TECHNIQUES

In this section, we have discussed all possible mitigation techniques for DoS attack at cloud application level and application development level. These include mitigation strategies at requirement and design levels of software to avoid buffer-overflows and strategies that employ static analysis techniques to find the

common coding problems that could expose buffer-overrun. Buffer overflow attack mitigation is very important at both cloud application and application development levels.

MITIGATION TECHNIQUES FOR BUFFER AT NETWORK LEVEL

A DoS attack requires a new approach that can help to detects and mitigates the effects of this attack to ensure availability of the resources. Whole DoS defense is built on concept of mitigation of attack for detection or identify traffic to preserve working continuity. The mitigation solution should include performance and architecture to deploy upstream to protect all points of vulnerability and to maintain reliable and cost efficient scalability which delivers the DoS defense. Towards this the following protection attributes are suggested:

- Enables immediate response to DDoS attack through integrated detection and blocking mechanism.
- Without announcing a failure point during attacks and enables on-demand deployment to protect the network.
- Detects and prevent every spoofed packet to guard the valid organization's transactions.
- Offers strong verification mechanism such as Intruder Detector System signatures.
- Work more designed tools to handle flood of DDoS attacks without affecting fate as secure resources.
- Use all communication standard protocols to ensure maximum consistency, reliability, interoperability and consistency.
- Try to avoid dependence on network device resources [17].
- Monitoring whole network packets carefully.

Some tools like Firewall, Intrusion Detection Systems (IDS), Application front end H/W, IPS based prevention and DoS Defense System is also available for detection the DoS attack. Other than these solutions, CISCO Systems also proposes a DDoS protection solution based on the principles of detection, diversion, verification, and forwarding to help ensure total protection.

MITIGATION TECHNIQUES FOR BUFFER OVERFLOW AT APPLICATION DEVELOPMENT LEVEL

Some other mitigation techniques for buffer overflow or overrun at application development life cycle are given below:

Traditional Approaches

Use strncpy() function instead of strcpy (). Eliminate all security bugs from program is considered infeasible. Using automatic tools which are available to find the vulnerability [7, 18–22].

Layered Approach

Buffer-overrun are mostly triggered by announcing bugs during application implementation. These types of bugs can be reduced or mitigate by using following three techniques [23]:

High Quality Code: Use high quality of code for design the application by developers.

Using Interpreted Language: Developed application with help of interpreted language that can reduce the risk for buffer-overrun problem such as java or c#.

Testing Public Interfaces: Through interface testing will also decrease the risk by providing the existence of buffer-overrun and allowing the development team to fix the problems as they are found.

Layered approach at most secure from all above mentioned techniques [24].

Compiler Approaches

It is a kind of dynamic intrusion prevention techniques at some extent. Check the range indices are most effective against this attack. This type of attack is not possible in Java programming and possible in C programming because java automatically checks array index bounds error [7]. To mitigate the possibility of the problem some safe types of compilers are designed and applied which are as follows [25]:

Stack Shield: A type of tool adding protection to the programs without changing a line of code [26]. There are two techniques as follows:

- Protection of Function Pointer
- Ret Range Check
- Global Ret Stack

Stack Guard: It was invented by Crispin Cowan with objective of dynamic detection,

prevention and stop buffer-overflow problem and return address [27, 28]. Main idea of this technique is to place an informal value known as canary between the local address and return address as shown in Figure 3. Canary save the changing return address if attacker try to overrun.

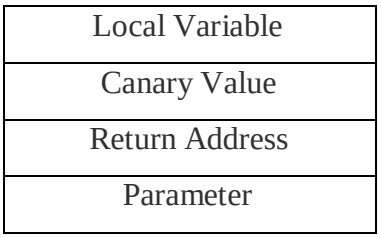


Fig. 3: *Stack Guard Frame.*

It is less secure than Stack shield and this technique only halt the buffer overflow attack against the return address.

Randomization

Randomization is another technique to mitigate the effect of stack buffer overflow by creating memory space arbitrary for executing program. This concept averts the attacker from knowing where code is store.

CONCLUSION AND FUTURE WORK

DoS attack can be launched by Buffer overflow by sending flood or creasing the services. Buffer overflow can be launch namely server level and network level and both are the most important security the breach. There are numerous techniques for detecting and preventing buffer overflow. In this paper we have try to present some mitigation techniques related to buffer overflow at both level. By using all the mentioned mitigation techniques may help to reduce the risk of buffer overflow/overruns at some degree. This paper also indicates that buffer overflow can be launch easily. Our motivation of future work is to reduce these types of attacks by proposing or making strong framework.

REFERENCES

1. Ronald L. Krutz, Russell Dean Vines. Cloud Computing: A Comprehensive Guide to Secure Cloud Computing. Wiley Publishing Inc.
2. CERT/CC and FedCIRC. CERT Coordination Center and the Federal Computer Incident Response Capability (FedCIRC). http://www-uxsup.csx.cam.ac.uk/pub/webmirrors/www.cert.org/advisories/CA-2000-01.html; 2000.
3. Jieren Cheng, Jianping Yin, Yun Liu, et al. DDoS Attack Detection Algorithm Using IP Address

Features. *Spinger Link, Lecture notes in Computer Science*. 2009; 55: 207–215p.

4. Available at: https://www.owasp.org/images/4/43/Layer_7_DDOS.pdf, Open Web Application Security Project. 18 March 2014. Retrieved 18 March 2014.
5. Available at: http://web.archive.org/web/20090201173324/http://eusecwest.com/speakers.html#PhlashDance, EUSecWest. 2008. Archived from the original on 2009-02-01.
6. Paloalto Networks. Denial of Service Attack-Prevent Dos Attacks with Palo Alto Networks. https://www.paloaltonetworks.com/resources/learning-center/what-is-a-denial-of-service-attack-dos.html; 2015.
7. Istvan Simon. A Comparative Analysis of Methods of Defense against Buffer Overflow Attacks; 2001.
8. Fu-Hau-Hsu. RAD: A Compile Time Solution to Buffer Overflow Attacks. Department of Computer Science, State University of New York at Stony Book.
9. Available at: http://www.scmagazineuk.com/video-games company-hit-by-38-day-ddos attack/article/367329/; 2015.
10. Jonathan Pincus, Brandon Baker. Beyond Stack Smashing: Recent Advances in Exploiting Buffer Overrun. *IEEE, Computer Society*. 2004; 20–27p.
11. Available at: http://blogs.msdn.com/michael_howard/archive/20 06/01/30/520200.aspx.
12. John Wilander, Mariam Kamkar. A Comparison of Publicly Available Tools for Dynamic Buffer Overflow Prevention. Published at 10th Network and Distributed System Security Symposium (NDSS); 2003.
13. Christ of Petzer Zhen Xio, "Detecting Heap Smashing Attacks Through Fault Containment Wrappers".
14. Levy, Elias (1996-11-08). Smashing the Stack for Fun and Profit". Phrack **7** (49): 14.
15. Pincus J., Baker B. "Beyond Stack

Smashing: Recent Advances in Exploiting Buffer Overruns" (PDF). *IEEE Security and Privacy Magazine*. 2004; 2(4): 20–27p. Doi:10.1109/MSP.2004.36.

16. Burebista. "Stack Overflows". (dead link). Cisco Systems. Defeating DDOS Attacks. http://www.cisco.com/c/en/us/products/collateral/security/traffic-anomaly-detector-xt-5600a/prod_white_paper0900aecd8011e927.html.
17. Ronald L. Krutz, Russell Dean Vines. Cloud Computing: A Comprehensive Guide to Secure Cloud Computing. Wiley Publishing Inc.
18. D. Larochelle, D. Evans. Statically Detecting Likely Buffer Overflow Vulnerabilities. *Proceedings of the 2001 USENIX Security Symposium, Washington DC, USA*; 2001.
19. Hal Var Flake. Auditing Binaries for Security Vulnerabilities. In Black Hat Europe Conference 2000 http://www.blackhat.com/html/bh-europe-00/bh-europe-00-speakers.html.
20. Nishad Herath (Joy). Advanced Windows NT Security. *In Black Hat Asia Conference*; 2000.
21. David Wagner, Jeffrey S Foster, Eric A. et al. A First Step Towards Automated Detection of Buffer Overrun Vulnerabilities. *Distributed System Security Symposium*; 2000.
22. Chris Evans. Nasty Security Hole in Lprm, posted in Bug Traq; 1998.
23. Jayson Taylor. Web services Risk Assesment and Recommendation. Security Innovation Commercial Tools Division 1318 S, Babcock Street, Melbourne, 2003/2004.
24. David Wheelar. Secure Programmer: Countering Buffer Overflow; 2004.
25. C. Cowan, C. Pu, D. Maier, et al. Stack Guard: Automotive Adaptive Detection and Prevention Buffer Overrun Attacks. In Proceedings of the 7th USENIX Security Conference. 1998; 63–78p.
26. Stack Shield A "stack smashing" technique protection tool for Linux: URL: http://www.angelfire.com/sk/stackshield/.
27. David Llewellyn-Jones, Madjid

Merabti, Qi Shi, et al. Buffer Overrun Prevention through Component Composition Analysis. Proceedings of the 29th Annual International Computer Software and Application Conference, (COMPSA'05) IEEE Transaction 2005.

28. C. Cowan, S. Beattie, R. Finnin Day, et al. Protecting from Stack Smashing Attacks with Stack Guard. In Linux Expo; 1999.