ON BUFFERZONE AUTOMATA NET - EXTENDED DETAILS -[1]

Răzvan Vasile

Abstract. DDoS type attacks represent the most frequent attacks on the internet. For now, no viable method to stop these has been found, this issue generating cost of millions of dollars for their prevention. In this article we provide a functionable solution to terminate these attacks. The solution is based upon the model of another security issue: the validation of a message sent from an unknown source. Thus, we shall create a BufferZone Automata Net, which will be responsible for the autheticity of every recieved message. The BufferZone Automata Net will be a secuirty interface between the browser and the server.

Keywords: BufferZone Automa Net, DDoS, attack, security, server, browser, halting

1. Introduction

The DDoS type attack represents any attempt to make the resources of a computer (i.e. server) unavailable to their final user. Though attacks differ through both means as well as objectives, the principle remains the same: one or several persons focus their efforts in order to obstruct the functioning of a website or of a temporary or permanent web service. With regards to the means, they are based upon the following idea: overload of the victim's resources, in such a degree that the victim is determined to temporarly close down that service or, if not closed down, the service to become severily malfunctioning.

The areas most prone to such attacks are the websites or services hosted on high profiled servers, like banks, gateways ofr online payments, national authorities, social networks or even search engines.

[1] First paper on BufferZone Automata Net was published in the proceedings of TAAC Conference, Kyiv. For further details please check [1], [2]

2. Some details about DDoS attacks

2.1 Symptoms

The specialists have established that the symptoms of such an attack include:

- Unusually low performance of the network.
- The unavailability of a certain website.
- The incapacity to access any website.

2.2 Attack methods

A DDoS type attack is characterized by an explicit attempt to block the users' access to a certain web service. The attacks can be directed towards any network device, mail server or DNS server.
The attacks can be launched in different ways, but the five most common types of attack are the following:

1. The saturation of computational resources such as the bandwidth or the bandwidth, or the overload of the processor.
2. The destroying of the configuration information, such as the router information.
3. The destroying of the status information, such as unsolicited resets of the TPC sessions.
4. The destroying of a network's hardware.
5. The obstruction of the communication channel between the victim and the users, so that they cannot communicate properly.

The information provided is incomplete unless we mention that the attack is distributed i.e. the attacker compromises several computers connected on the internet through viruses (usually computers with a low degree of security), so that the attack be simultaneously launched from the compromised computers. Therefore the name DDoS type distributed attack. In modern times, an attacker may have at his disposal thousands, hundreds of thousands, even millions of such compromised computer. The dimensions of such an attack may be easily imagined.

2.3 Types of attack

There are several kinds of DDoS type attacks, amongst which we will remind several[2].

Smurf type attacks - broadcast packages are sent to the entire network and any computer listening to the port from which the packages were sent shall receive them, without the packages being sent especially to him. The attack is based on sending a ping type broadcast package - with a false address - to a router, that receives such messages from the outside. Since the source address is falsified, when the router resends the package to all the computers in the network, these will respond to the falsified address, which is actually the victim's address, saturating its traffic.

The Permanent Denial of Service type attacks - is an attack that damages the system so hard that replacements or hardware reinstall are necessary. Unlike the denial-of-service type attacks, the PDoS type attacks exploit the security breaches that provide long distance access to the management interface for the victim's hardware.

Peer-to-peer attacks - the attacker acts like a "master of puppets", as he disconnects the clients of a file sharing hub, thus forcing them to connect to the victim server. The result is almost instant, since even for a medium sized hub the number of the file sharing clients may reach 100,000[3]. Even the termination of all connection attempts uses server resources and may affect it.

Distributed attacks[4] - appear when several systems saturate the bandwidth or the resources of a certain system, usually web servers. These types of attacks can be performed through various means. It should be mentioned that certain groups of persons practice blackmail and fraud by organizing such attacks[5].

3. The BufferZone Automata Net

From the examples presented in the previous chapters it can be concluded that the prevention of the DDoS type attacks is almost impossible, since it is

[2] For further details, please check the bibliography

[3] The automata put forward by us has a method of protection against this attack as well

[4] The automata net we put forward blocks mainly these types of attack

[5] The DDoS type attacks are considered breaches of the internet proper usage policies - IAB

never known who exactly is behind a massive increase in traffic; it might be a legitimate growth or it might be a malicious traffic resulting from a DDoS type attack.

Also, in order to resist an attack the costs involved are very high, since this implies multiple hardware resources, as well as security measures taken during the design.

Special attention must be paid in order to validate each browser, so that for a certain period of time we may state that behind that browser is a user and not a hacker. In order to accomplish that, two compromises must be accepted:

1. The users to be willing that once in a certain period of time they would have to enter an alphanumeric sequence from their keyboard, to provide legitimacy to the browser.

2. The second compromise consists of a buffer zone, functioning independently from any server. This buffer zone has the role to "absorb" any possible DDoS type attack. The authentication of the browser is made in this buffer zone. In case of an emergency, the buffer zone will act like an intermediary between the browser and the server.

Such an approach may be implemented using an automata net. Furthermore, we consider that an automata net can shape the model we have proposed with an increase functionality, since the communication between a client and a server is done under the form of request and answer. Thus, with an automata net we know for certain that a state may reject a request or switch to a future state, which might include the answer to the forwarded request. Following, the graphic representation of the BufferZone Automata Net we have nominated.

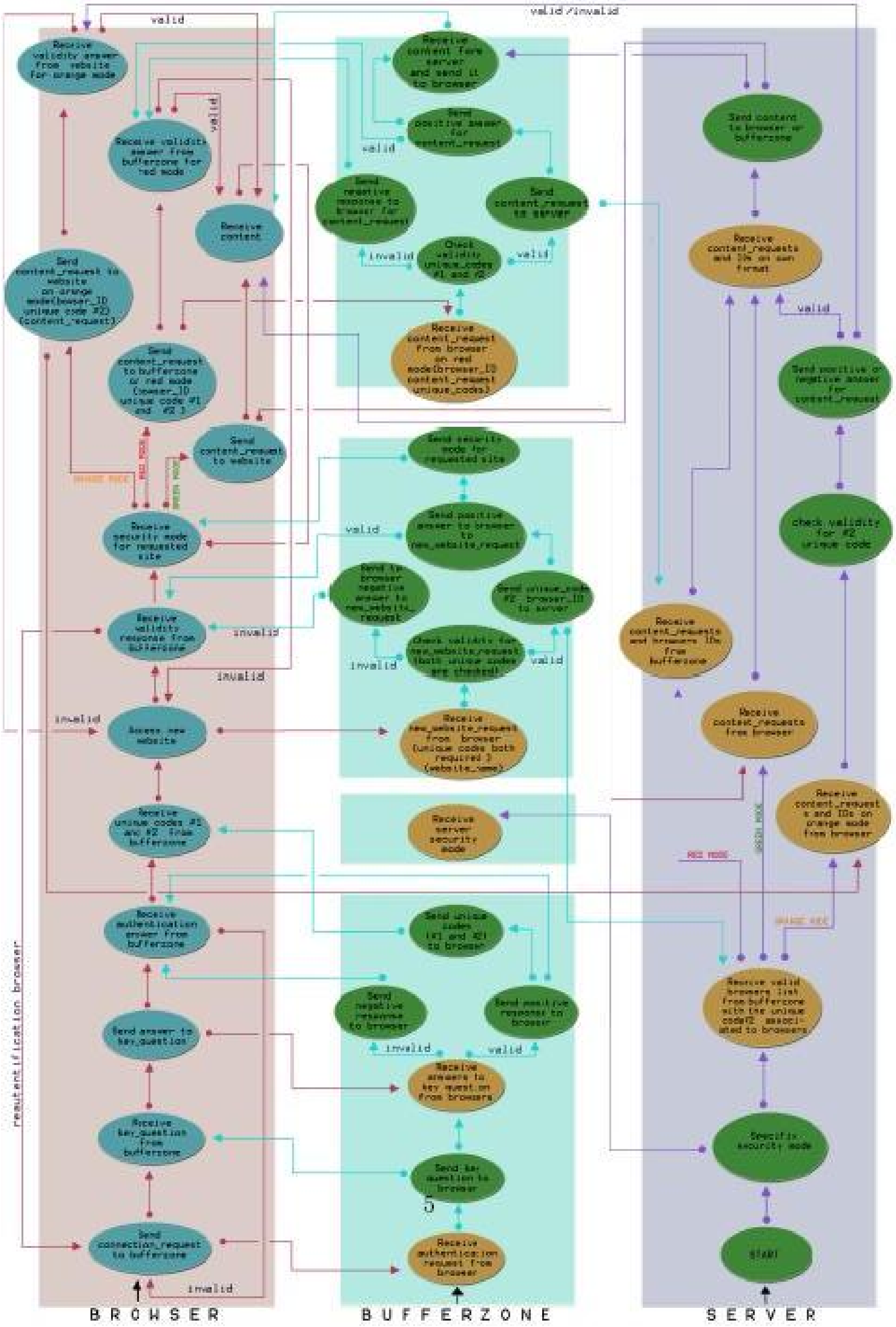
4. The implementation algorithm of the BufferZone Automata Net

Throughout this chapter we will present the implementation algorithm, structured on its three components: browser, buffer zone and server.

4.1 The implementation algorithm for the Browser

1. 1. Send a connection request to the BufferZone.

2. Receive the key question from the BufferZone.


valid
valid / invalid
Receive validity answer from website for orange mode
Receive validity answer from bufferzone for red mode
valid
Receive content
Send content_request to website on orange mode(bowser_ID unique code #2) (content_request)
Send content_request to bufferzone on red mode (bowser_ID unique code #1 and #2)
Send content_request to website
ORANGE MODE
RED MODE
GREEN MODE
Receive security mode for requested site
Receive validity response from bufferzone
invalid
invalid
invalid
Access new website
Receive unique codes #1 and #2 from bufferzone
Receive authentication answer from bufferzone
Send answer to key_question
Receive key_question from bufferzone
Send connection_request to bufferzone
invalid
reautentification browser
BROWSER
Receive content form server and send it to browser
Send positive answer for content_request
valid
Send negative response to browser for content_request
Send content_request to server
Check validity unique_codes #1 and #2
invalid
valid
Receive content_request from browser on red mode(browser_ID content_request unique_codes)
Send security mode for requested site
Send positive answer to browser to new_website_request
valid
Send to browser negative answer to new_website_request
Send unique_code #2 browser_ID to server
Check validity for new_website_request (both unique codes are checked)
invalid
valid
Receive new_website_request from browser (unique code both required) (website_name)
Receive server security mode
Send unique codes (#1 and #2) to browser
Send negative response to browser
Send positive response to browser
invalid
valid
Receive answers to key question from browsers
Send key question to browser
Receive authentication request from browser
BUFFERZONE
Send content to browser or bufferzone
Receive content_requests and IDs on own format
valid
Send positive or negative answer for content_request
check validity for #2 unique code
Receive content_requests and browsers IDs from bufferzone
Receive content_requests from browser
Receive content_requests and IDs on orange mode from browser
RED MODE
GREEN MODE
ORANGE MODE
Receive valid browsers list from bufferzone with the unique code#2 associated to browsers
Specific security mode
START
SERVER

3. Through a CapCha system, obtain the response to the key question from the user.

4. Send the response to the BufferZone.

5. If the response was correct, it obtains the two unique authentication codes; if the response was incorrect, go to step 1.

6. Send a request to access new website to the BufferZone. The request will include the following information: $< uniquecodeno1 >$, $< uniquecodeno2 >$, $< nameofthewebsite >$

7. Receive the validation response from BufferZone. In case of a negative response, go to step 1.

8. Receive the security mode for the desired website.

9. Analyze the security mode for the desired website

    - If the security mode is green, go to step 10
    - If the security mode is orange, go to step 13
    - If the security mode is red, go to step 15

10. Send content request to the website on the green mode.

11. **Receive content from the website - final state.**

12. For a new content request, go to step 9, otherwise STOP.

13. Send content request to the website on the orange mode. The request will include the following information: $< browserID >$, $< uniquecodeno1 >$, $< nameofcontentrequest >$

14. Receive validity response from the website.

    - If the response is valid, go to step 11.
    - If the response is invalid, go to step 6.

15. Send content request to the website on the red mode. The request will include the following information: $< browserID >, < uniquecodeno1 >, < uniquecodeno2 >, < nameofwebsite >, < nameofcontentrequest >$

16. Receive validity response from the website.
    - If the response is valid, go to step 11.
    - If the response is invalid, go to step 6.

4.2 The implementation algorithm for the BufferZone

The architecture of the buffer zone is made up of four parallel services. They are interconnected, but they shall be executed like separate services.

4.2.1 The key question service - I

1. Receive authentication requests from browsers. For each request, go to step 2.
2. Send the key question to the browsers.

4.2.2 The key question service - II

1. Receive responses to the key question from the browsers. For each response, go to step 2.
2. Verify the response to the key question.
    - If the response is correct, send positive response to the browser.
    - If the response is incorrect, send negative response to the browser; STOP.
3. Send to the browser the allocated ID, as well as the two generated unique codes related to it.

4.2.3 Receive the security mode from the servers.

- " Receive the security mode from the servers. STOP.

4.2.4 Send the security mode to the browser

1. 1. Receive new website requests from the Browsers. The request will include the following information: $< browserID >, < uniquecodeno1 >$ $, < uniquecodeno1 >, < nameofwebsite >$ For each, go to step 2.

2. Verify the validity of the two unique codes.
    - If the codes are valid, go to step 3.
    - If the codes are invalid, send negative response to the browser. STOP.
3. Send content request to the server on the red mode.
4. Send positive response to the browser.
5. Receive from the Server the content for the related request.
6. Send to the Browser the content for the related request.

4.3 The implementation algorithm for the Server

Same as with the BufferZone, the server's architecture is made up of several interconnected components, executed in parallel. But, unlike the BufferZone architecture, some of the server's components will be inactive - depending on the security mode configuration. More specifically, if the security mode is set on the orange mode, the other services responsible with receiving content requests will be stopped[6]. The server will behave accordingly for the two other security modes. This measure has been taken in order to prevent any type of exterior attack, thus guarantying the full security on every desired security mode.

4.3.1 Nominate the security mode

1. Start.
2. Send to the BufferZone the security mode (green, orange or red)
3. Depending on the configured security mode, execute solely the service related to the nominated security mode. STOP

4.3.2 Receive the list of valid browsers

[6] There is room for a wider debate regarding the necessity to close the content requests services while on the red mode. Depending on the desired implementation, this service can be activ at the same time with any of the other two security modes.

1. Receive the list of valid browsers, as well as the related no 2 code. STOP.

4.3.3 Receive content request on the green mode

- Receive content requests from the browser on the green mode.
- Rewrite the content request in its own format. STOP.

4.3.4 Receive content request on the orange mode

1. Receive content requests from the browser on the orange mode. For each request go to step 2.
2. Verify the validity of the unique code no 2
    - If the unique code sent by the browser is valid, send positive response to the Browser. Go to step 3.
    - If the unique code sent by the browser is invalid, send negative response to the Browser. STOP.
3. Rewrite the content request in its own format. STOP.

4.3.5 Receive content request on the red mode

1. Receive content requests from the browser on the red mode.
2. Rewrite the content request in its own format. STOP.

4.3.6 Process content request with featured format

1. Process content request with featured format.
    - If the security mode is set on red, send the content related to the request to the BufferZone.
    - If the security mode is set on green or orange - send the content related to the request to the BufferZone or Server. STOP. Final state.

5. Conclusions

The DDoS type attack represent a real danger: they are common and extremely effective. The damages from an attack are significant, given that the essence of the DDoS attacks consists of the saturation of the resources of a service, making it unusable. Also, the hidden costs related with the above mentioned attack must not be ignored, especially when the attacks are triggered against corporations or internet providers.

Currently, these attacks cannot be stopped, only somewhat managed. This approach is very expensive, involving a wide and numerous range of resources (especially hardware - servers) that require supplementing. We render that **The BufferZone Automata Net represent the first formalization which is viable and can be implemented in order to stop the DDoS attacks**. Indeed, this automaton puts forward a compromise of the users' commodity, yet, the compromise is acceptable. The increased advantages of the security in any online service represent a real gain for any user.

References

[1] Razvan Vasile, BufferZone Automata Net, Theoretical and Applied Aspects of Cybernetics. Proceedings of the International Scientific Conference of Students and Young Scientists Kyiv: Bukrek, 2011

[2] Razvan Vasile, BufferZone Automata Net, TAAC
http://taac.org.ua/en/a2011/sectionOne

[3] Armbruster B., Smith J. Cole and Park K., A Packet Filter Placement Problem with Application to Defense against Spoofed Denialof- Service Attacks, European Journal of Operational research, Vol. 176, Issue 2, pp. 1283-1292, 16 January 2007.

[4] Hole K, Denial-of-Service Attacks, Nowires research Group, Department of Informatics, University of Bergen, September 1, 2008. disponibil la www.Kjhole.com

[5] Juneja D., Chawla R. and Singh A., An Agent-Based Framework to Counter attack DDoS Attacks. International Journal of Wireless Networks and Communications, Vol. 1, No. 2, pp. 193 200, 2009.

[6] Kim Y., Lau W., Chuah M. and Chao H., PacketScore : Statistics-based Overload Control against Distributed Denial-of-Service Attacks, IEEE Transactions on Dependable and Secure Computing, Vol. 3, No. 2, pp. 141-155, April-June 2006.

[7] Kotenko I. and Ulanov A., Agent-based Simulation of Distributed Defense Against Computer Network Attacks, Proceedings 20th European Conference on Modelling and Simulation Wolfgang Borutzky, Alessandra Orsoni, Richard Zobel , ECMS 2006.

[8] Slee D., Common Denial-of-Service Attacks, published July 10, 2007.

[9] `http://en.wikipedia.org/wiki/Denial-of-service_attack`

[10] `http://en.wikipedia.org/wiki/Smurf_attack`

Răzvan Vasile
Department of Computer Science
University of Bucharest
Str. Academiei nr.14, sector 1, C.P. 010014, Bucuresti, Romania
email:*razvan.vasile@yahoo.com*