International Refereed Journal of Engineering and Science (IRJES)
ISSN (Online) 2319-183X, (Print) 2319-1821
Volume 13, Issue 2 (Mar.-Apr. 2024), PP. 234-240

Implementing Moving Target Defense for Internet Denial of Service Attacks

[1] D Devi Sri, *Student, Department of CSE, DNR College of Engineering and Technology, Bhimavaram, Andhra Pradesh, India. devidhatla@gmail.com*
[2] M. Naga Venkata Nivas Varma, *Student, Department of CSE, DNR College of Engineering and Technology, Bhimavaram, Andharapradesh, India. nivasvarmamudunuri@gmail.com*
[3] K Mounika, *Student, Department of CSE, DNR College Engineering and Technology, Bhimavaram, Andhra pradesh, India. Mounikakandiboina220@gmail.com*
[4] Ch Vijay Bhaskar, *Student, Department of CSE, DNR College of Engineering and Technology, Bhimavaram, Andhrapradesh, India. ch.vijaybhaskar14@gmail.com*
[5] Kalipindi Siva Hari Prasanna Kumar, *M. Tech., Assistant Professor, Department of CSE, D.N.R College of Engineering and Technology, West Godavari, Andhra Pradesh, India. kshpk@dnrcet.org*

Abstract: *A dynamic defensive mechanism called MOTAG is included in the project to protect application servers against insider threats and Distributed Denial of Service (DDoS) assaults. In order to stop insider assaults from figuring out the application server's IP address and port, four important components work together: an authentication server, proxy servers, the application server, and clients. Each client is dynamically assigned a proxy server by the Authentication Server using a random shuffle greedy algorithm. This adds an extra degree of protection by making it difficult for hostile users to determine the application server's real IP address and port. After checking whether the requested data is within the proxy server's processing capacity, the request is sent on to the secure application server. To improve network security, proxies may drop requests and notify clients or Authentication Servers of possible DDoS attacks based on their sizes. A client app with a GUI for file uploading to the server is part of the project. For effective monitoring and protection, the system generates visual representations of proxy assignments, successfully processed requests, and identified DDoS assaults.*
Search keywords - Distributed Denial of Service; Moving Target Defense; Secret Proxy; Insider; Shuffling.

I. INTRODUCTION

An uptick in the frequency of large-scale distributed denial-of-service (DDoS) assaults has been noted by Arbor Networks in earlier reports [1]. One hundred gigabits per second was the most bandwidth recorded in 2010 by a flood-based DDoS assault. At the same time, the price of launching a distributed denial of service (DDoS) assault is shockingly cheap. A white paper by Trend Micro [2] has shown that in the Russian underground market, a one-week DDoS service might cost as little as $150.

In the past, several different approaches have been suggested to lessen the impact of distributed denial of service assaults. In order to prevent unauthorized traffic from reaching the protected nodes, methods based on filtering [3, 4], [5] make use of filters that are widely distributed. Capability-based defensive methods [6, 7, 8, 9] aim to limit the senders' resource consumption to what the receivers allow. In order to intercept and filter out attack traffic, secure overlay systems [10], [11], [12], [13], [14], and [15] interpose an overlay network to indirect packets between clients and the protected nodes. To counter the increasingly sophisticated assaults, however, these static security measures need either a massive, powerful virtualized network or the worldwide deployment of supplementary features on Internet routers. And some of them are still susceptible to advanced assaults like adaptive floods [12] and sweeps [11].

To safeguard centralized web services, we provide MOTAG, a dynamic DDoS defensive method that uses a moving target defense technique. Online banking and efinance are examples of security-sensitive services that might benefit from MOTAG's DDoS resistance. When clients communicate with the protected application servers, MOTAG uses a layer of hidden moving proxies to mediate the conversation. When it comes to the application servers, only traffic from legitimate proxy nodes may get through the network-level filters that surround them.

There are two key features of MOTAG proxy nodes. The first thing to know is that all proxy nodes are "secret" since only authenticated clients are able to access their IP addresses, which are hidden from the public. In order to prevent any needless disclosure of information, we only supply each valid client the IP address of

one active proxy at any given moment. Protecting the client authentication channel is achieved by using pre-existing proof-of-work (PoW) techniques [16], [17], [18], [19]. Secondly, such nodes are "moving" about. In the event of an attack on an active proxy node, it is immediately replaced by a new node in a different location, and all related clients are transferred to the new proxies. We demonstrate that these features allow us to detect and punish malevolent insiders who reveal the whereabouts of hidden proxies to outsiders, in addition to reducing the impact of brute-force DDoS assaults. To do this, if a client's original proxies are under assault, we shuffle (reposition) their assignment to new proxy nodes. In order to save the majority of unsuspecting customers after every shuffle, we program algorithms that precisely predict the amount of insiders and modify the assignment of clients to proxies appropriately.

II. LITERATURE SURVEY

Due to its lack of inherent security features, the existing Internet infrastructure is susceptible to assaults and breakdowns. In instance, recent occurrences have shown how susceptible the Internet is to denial of service (DoS)[5,8,11] assaults and flash crowds, in which a single connection or many edge servers experience extreme congestion. The congestion in both denial-of-service attacks and flash crowds is not caused by an increase in traffic overall or by a single flow, but by an aggregation of a specific subset of that traffic. the third Methods for identifying and managing these kinds of high bandwidth aggregates are suggested in this work. A cooperative pushback mechanism allows a router to request that upstream routers manage an aggregate, and our system incorporates both a local technique for detecting and managing an aggregate at a single router and this mechanism. There is no silver bullet, but these safeguards have the potential to mitigate flooding-style denial-of-service assaults and flash crowds. This paper's presentation serves as a first step towards doing a more thorough assessment of these processes.

Using a filter-based defensive system (StopIt) and comparing the efficacy of filters and capabilities, this work lays out the design and execution of a DoS defense system [8, 11]. The innovative closed-control, open-service architecture is at the heart of StopIt's design. Any receiver can utilize StopIt to block unwanted traffic, and the design is resilient enough to withstand multiple strategic attacks launched by millions of bots, such as bandwidth flooding and filter exhaustion attacks, which try to disrupt the timely installation of filters. Based on our testing, StopIt is able to limit the attack traffic of several million malicious users within a few minutes, even when router memory is limited. Under simulated DoS assaults of different sorts and sizes, we compare StopIt with current filter-based and capability-based protection systems. Based on our findings, StopIt is able to protect valid communications from a variety of [5] DoS flooding assaults and performs better than current filter-based methods. In most attack situations, it likewise performs better than capability-based systems. However, capability-based systems are more successful in an attack type where the attack traffic does not reach a victim but instead congests a shared connection that the victim uses. Based on these findings, it seems that filters and capabilities are both great tools for defending against denial-of-service assaults, but that there isn't a single one that works better than the other.

To limit and avoid denial-of-service (DoS) assaults, we provide a novel strategy in this article [6]. In our design, nodes can't just transmit anything to anybody; they need "permission to send" from the receiver, which grants them tokens (or capabilities) to use with the senders whose traffic they accept. After that, the senders throw these tokens into the packets. This allows dispersed verification points around the network to verify the legitimacy of traffic by checking whether both endpoints and the route between them have certified it, and to delete any unlawful traffic thereafter. As we'll see, our method overcomes a lot of the problems with the most widely used methods for detecting and preventing denial of service attacks [8, 11] that rely on anomaly detection, traceback, and pushback. In addition, we contend that our method is compatible with current technology, can be deployed incrementally, and just necessitates the security architecture now in place to address BGP's vulnerabilities. Lastly, current DoS safeguards are limiting innovation in application and networking protocols, which our approach aims to alleviate.

Recipients of packet flows cannot prevent disruptive flows from using their network connection resources, which is one of the Internet's basic constraints. Disruption of service (DoS) assaults or flash crowds, which cause networks to experience traffic floods, pose a threat to both critical infrastructures and enterprises [5, 8, 11]. Regrettably, existing countermeasures need either establishment of an overlay system, cooperation between ISPs, or per-flow state at routers. We introduce SIFF, a Stateless Internet Flow Filter, in this study [7]. It enables an end-host to selectively block flows from entering its network, without relying on any of the aforementioned common assumptions. Priority packets [6, 7, 8, 9] that are subject to recipient control are considered privileged, whereas legacy traffic is considered unprivileged. A capability exchange handshake is used to create privileged channels. Network routers dynamically verify capabilities and may revoke them by dampening update messages to an offending site. Legacy clients and servers may see through SIFF, but only hosts that have been upgraded will be able to reap its advantages.

III. METHODOLOGY

i) Proposed Work:

A unique approach against large-scale DDoS assaults, MOTAG uses a Moving Target Defense strategy, proactive DDoS detection, and dynamic proxy assignment [7, 26]. By enhancing real-time monitoring via the client interface's visualizations, we can respond swiftly to possible risks. The network's resilience against ongoing DDoS attacks is strengthened by this all-encompassing method. Attackers will have a hard time figuring out and compromising the application server's real IP address and port because the suggested MOTAG system employs a dynamic protection mechanism via dynamic proxy assignment. Being nimble like this makes the system more resistant to cyber attacks that are always changing. Proactively identifying prospective assaults is made possible by MOTAG's proxy-based DDoS detection. The system can quickly identify and counteract DDoS attacks by evaluating the magnitude of incoming client requests, therefore lowering the possibility of service interruptions. Proxies are constantly switching between servers as part of MOTAG's Moving Target Defense technique. It becomes more difficult for prospective attackers to identify and exploit weaknesses in the network infrastructure when this method is used [10], [11], [12], [13], [14], [15]. The client app's graphical user interface shows the number of assigned proxies, the number of requests that were successful, and the number of DDoS assaults that were identified. Quick reactions to possible dangers and effective defensive techniques are made possible by this visual monitoring, which improves situational awareness.

ii) Architecture of the System:

A filter ring, an application server, an authentication server, and undisclosed proxies make up MOTAG's core architecture. Clients are allocated secret proxies after passing authentication and receiving capability tokens. The application server can only receive allowed traffic thanks to the filter ring. During assaults, proxies may dynamically switch, which might be confusing for opponents. Both proof-of-work and "secret moving proxies" function to conceal the target's IP address. As a result of its adaptable architecture, MOTAG is better able to withstand powerful DDoS assaults.

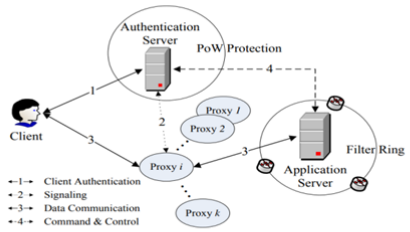

Fig 1 Proposed Architecture

iii) Processing:

Application Server: Enforces size limitations, processes client requests, and manages text file uploads. Its primary responsibility is to carry out the application's operations, guaranteeing the correct handling of legitimate client requests.

Requests from clients are received by the authentication server, which verifies their identities and then sends them to a random proxy server. The randomization of proxy assignments improves security by making it harder for malevolent users to easily identify the true server.

The application server and the clients are connected via intermediaries known as proxies. If the sizes of the requests are within the restrictions, two proxy servers will send them on for processing. An essential component in the defense of the Application Server against possible threats and in the detection and mitigation of distributed denial of service (DDoS) assaults [16...19].

• The client interface, which represents the end-user and initiates server queries. Allows users to securely upload text files and engage in secure discussions. The designated proxy server and the authentication server facilitate communication. Actively participates in protection systems such as proxy assignment and probable attack detection.

the sixth section, algorithms:

When things are "randomly shuffling" [21], it means they are being rearranged or reordered at random. This probably alludes to a way whereby the Authentication Server randomly rearranges or shuffles the available proxy servers. In its pursuit of a global optimum, a greedy algorithm iteratively produces locally optimal decisions. Once the random shuffle is complete, the greedy algorithm will aimlessly make the best-looking decisions at each stage, ignoring the big picture. By combining random shuffling with a greedy method, it may

be inferred that the Authentication Server uses a local criterion—perhaps associated with load balancing, security, or some other factor—to choose a proxy server during the proxy selection process, after randomly shuffling their order.

```
def getRandomShuffleProxy():
    proxy = randrange(2)
    return proxy

def startAuthenticationServer():
    class ClientThread(Thread):

        def __init__(self,ip,port):
            Thread.__init__(self)
            self.ip = ip
            self.port = port
            text.insert(END,'Request received from IP : '+ip+' with port no : '+str(port)+"\n")

        def run(self):
            data = conn.recv(100)
            data = data.decode()
            proxy = getRandomShuffleProxy()
            proxyport = ''
            pno = 0
            if proxy == 0:
                proxyport = 4444
                pno = 1
            if proxy == 1:
                proxyport = 5555
                pno = 2
            proxyfirst = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            conn.send(str(proxyport).encode())
            text.insert(END,'Client request assigned to proxy : '+str(pno)+'\n')
```

Fig 2 Algorithm

The authentication server has a "random shuffling greedy algorithm" that it uses to choose proxy servers on the fly. A greedy technique is used to generate locally optimum selections during proxy selection after randomly shuffling available proxy servers. This combination increases the system's resistance to harmful assaults by making the selection process more unpredictable. It prevents attackers from correctly identifying and targeting individual servers.

IV. EXPERIMENTAL RESULTS

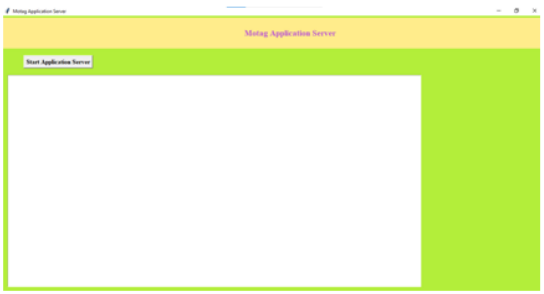

Fig 3 MOTAG application server

Fig 4 MOTAG authentication server

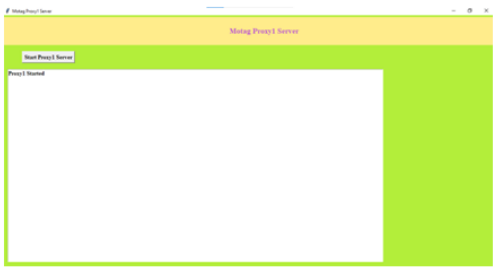

Fig 5 Proxy1 server

Fig 6 Client application

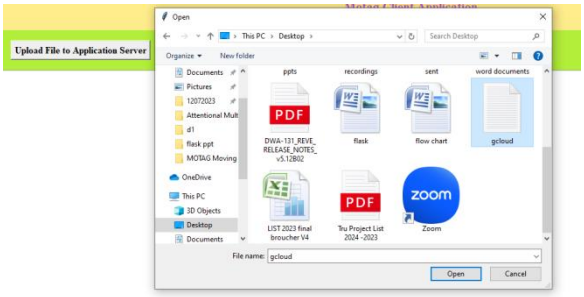

Fig 6 Upload file

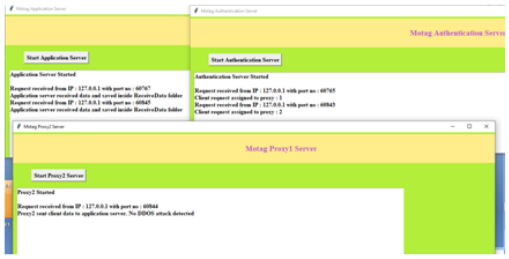

Fig 7 Processing of servers

Fig 8 Attack detection graph

V. CONCLUSION

An effective defensive system including Application Servers, Proxies, and Authentication has been put into place by the project. This strategy is designed to protect against various cyber threats in a multi-layered manner. The Authentication Server enhances security by adding a layer of complexity via dynamic proxy assignment randomization. This prevents malevolent actors from attempting to identify the real server. Overall system security and resilience are greatly enhanced by proxies' proactive involvement in identifying and mitigating possible Distributed Denial of Service (DDoS) assaults. Users are able to securely upload text files using the built Client module, which also simplifies interactions. Communication routes are kept safe even when threats are present thanks to the Authentication Server and Proxies. The project strengthens the Application Server's resiliency by processing legitimate client requests efficiently and imposing file size restrictions on incoming data. This safeguards the server's ability to continue operating normally, regardless of any hostile efforts.

8. IN THE LONG TERM

To make sure MOTAG is successful in protecting services of different sizes against heightened DDoS assaults, future research should focus on improving its scalability and performance [16], [17], [18], [19]. Additional research into using cloud environments as a proxy pool source is part of the future scope exploration. With backup proxies in place, this optimization hopes to save operating expenses without sacrificing protection. Reducing the frequency of proxy re-allocations is the primary goal of future efforts to improve MOTAG's shuffle mechanism. With this enhancement, we want to better detect and stop distributed denial of service (DDoS) assaults while keeping services running as smoothly as possible. Researching how MOTAG may be integrated with other protection mechanisms, including secure overlay networks, is an important part of exploring future possibilities. This method builds a protection strategy with many layers, making it more resilient to emerging vectors of distributed denial of service attacks.

REFERENCES

[1]. R. Dobbins and C. Morales, "Worldwide infrastructure security report vii," 2011. [Online]. Available: http://www.arbornetworks.com/report

[2]. T. Micro, "Russian underground 101," http://www.trendmicro. com/cloud-content/us/pdfs/security-intelligence/white-papers/ wp-russian-underground-101.pdf, 2012.

[3]. R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network," ACM Computer Communication Review, vol. 32, pp. 62–73, 2002.

[4]. P. Ferguson and D. Senie, "Network ingress filtering: Defeating denial of service attacks which employ ip source address spoofing," RFC 2827 (Best Current Practice), Internet Engineering Task Force, May 2000, updated by RFC 3704.

[5]. X. Liu, X. Yang, and Y. Lu, "To filter or to authorize: network-layer dos defense against multimillion-node botnets," in Proceedings of the ACM SIGCOMM 2008 conference on Data communication. New York, NY, USA: ACM, 2008, pp. 195–206.

[6]. T. Anderson, T. Roscoe, and D. Wetherall, "Preventing internet denialof-service with capabilities," SIGCOMM Comput. Commun. Rev., vol. 34, no. 1, pp. 39–44, 2004.

[7]. A. Yaar, A. Perrig, and D. Song, "Siff: A stateless internet flow filter to mitigate ddos flooding attacks," in IEEE Symposium on Security and Privacy, 2004, pp. 130–143.

[8]. X. Yang, D. Wetherall, and T. Anderson, "Tva: a dos-limiting network architecture," IEEE/ACM Trans. Netw., vol. 16, no. 6, pp. 1267–1280, 2008.

[9]. X. Liu, X. Yang, and Y. Xia, "Netfence: preventing internet denial of service from inside out," in Proceedings of the ACM SIGCOMM 2010 conference on SIGCOMM, ser. SIGCOMM '10. New York, NY, USA: ACM, 2010, pp. 255–266. [Online]. Available: http://doi.acm.org/10.1145/1851182.1851214

[10]. A. D. Keromytis, V. Misra, and D. Rubenstein, "Sos: Secure overlay services," in Proceedings of ACM SIGCOMM, 2002, pp. 61–72.

[11]. A. Stavrou and A. D. Keromytis, "Countering dos attacks with stateless multipath overlays," in Proceedings of the 12th ACM conference on Computer and communications security, ser. CCS '05. New York, NY, USA: ACM, 2005, pp. 249–259. [Online]. Available: http://doi.acm.org/10.1145/1102120.1102153

[12]. D. G. Andersen, "Mayday: distributed filtering for internet services," in USITS'03: Proceedings of the 4th conference on USENIX Symposium on Internet Technologies and Systems. Berkeley, CA, USA: USENIX Association, 2003, pp. 3–3.

[13]. R. Stone, "Centertrack: an ip overlay network for tracking dos floods," in SSYM'00: Proceedings of the 9th conference on USENIX Security Symposium. Berkeley, CA, USA: USENIX Association, 2000, pp. 15–15.

[14]. A. Mahimkar, J. Dange, V. Shmatikov, H. Vin, and Y. Zhang, "dfence: Transparent network-based denial of service mitigation," in NSDI, 2007.

[15]. C. Dixon, T. Anderson, and A. Krishnamurthy, "Phalanx: withstanding multimillion-node botnets," in Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation, ser. NSDI'08. Berkeley, CA, USA: USENIX Association, 2008, pp. 45–58. [Online]. Available: http://dl.acm.org/citation.cfm?id=1387589.1387593

[16]. T. Aura, P. Nikander, and J. Leiwo, "Dos-resistant authentication with client puzzles," in Security Protocols Workshop, 2000, pp. 170–177.

[17]. D. Dean and A. Stubblefield, "Using client puzzles to protect tls," in Proceedings of the 10th conference on USENIX Security Symposium - Volume 10, ser. SSYM'01. Berkeley, CA, USA: USENIX Association, 2001, pp. 1–1. [Online]. Available: http: //dl.acm.org/citation.cfm?id=1251327.1251328

[18]. B. Waters, A. Juels, J. A. Halderman, and E. W. Felten, "New client puzzle outsourcing techniques for dos resistance," in Proceedings of the 11th ACM conference on Computer and communications security, ser. CCS '04. New York, NY, USA: ACM, 2004, pp. 246–256. [Online]. Available: http://doi.acm.org/10.1145/1030083.1030117

[19]. B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y.-C. Hu, "Portcullis: Protecting connection setup from denial-of-capability attacks," in Proceedings of the ACM SIGCOMM, August 2007.
[20]. N. Johnson and S. Kotz, Urn Models and Their Applications: An Approach to Modern Discrete Probability Theory. New York: Wiley, 1977, ch. 1.3.2.
[21]. M. Matsumoto and T. Nishimura, "Mersenne twister: a 623- dimensionally equidistributed uniform pseudo-random number generator," ACM Trans. Model. Comput. Simul., vol. 8, no. 1, pp. 3–30, Jan. 1998. [Online]. Available: http://doi.acm.org/10.1145/272991.272995
[22]. "Iperf," http://iperf.sourceforge.net.
[23]. M. Abliz, "Internet denial of service attacks and defense mechanisms," University of Pittsburgh, Tech. Rep. TR-11-178, Mar 2011.
[24]. R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The secondgeneration onion router," in In Proceedings of the 13 th Usenix Security Symposium, 2004.
[25]. L. Overlier and P. Syverson, "Locating hidden servers," in Proceedings of the 2006 IEEE Symposium on Security and Privacy, ser. SP '06, Washington, DC, USA, 2006, pp. 100–114.
[26]. V. Kambhampati, C. Papadopoulos, and D. Massey, "Epiphany: A location hiding architecture for protecting critical services from ddos attacks," in The 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2012), 2012.
[27]. X. Wang and M. K. Reiter, "Wraps: Denial-of-service defense through web referrals," in 25th IEEE Symposium on Reliable Distributed Systems. IEEE Computer Society, 2006, pp. 51–60.
[28]. S. Antonatos, P. Akritidis, E. P. Markatos, and K. G. Anagnostakis, "Defending against hitlist worms using network address space randomization," in Proceedings of the 2005 ACM workshop on Rapid malcode, ser. WORM '05. New York, NY, USA: ACM, 2005, pp. 30– 40. [Online]. Available: http://doi.acm.org/10.1145/1103626.1103633
[29]. T. Holz, C. Gorecki, K. Rieck, and F. C. Freiling, "Measuring and detecting fast-flux service networks," in NDSS. The Internet Society, 2008.