Application-layer denial of service attacks: taxonomy and survey

Georgios Mantas, Natalia Stakhanova*,
Hugo Gonzalez, Hossein Hadian Jazi,
Ali A. Ghorbani

Information Security Centre of Excellence,
University of New Brunswick,
550 Windsor Street, ITC, Fredericton,
New Brunswick, Canada
Email: gimantas@av.it.pt
Email: g.mantas@unb.ca
Email: natalia.stakhanova@unb.ca
Email: hugo.gonzalez@unb.ca
Email: h.hadian@unb.ca
Email: ghorbani@unb.ca
*Corresponding author

Abstract: The recent escalation of application-layer denial of service (DoS) attacks has attracted a significant interest of the security research community. Since application-layer DoS attacks usually do not manifest themselves at the network level, they avoid traditional network-layer-based detection. Therefore, the security community has focused on specialised application-layer DoS attacks detection and mitigation mechanisms. However, the deployment of reliable and efficient defence mechanisms against these attacks requires the comprehensive understanding of the existing application-layer DoS attacks supported by a unified terminology. Thus, in this paper we address this issue and devise a taxonomy of application-layer DoS attacks. By devising the proposed taxonomy, we intend to give researchers a better understanding of these attacks and provide a foundation for organising research efforts within this specific field.

Keywords: denial of service attacks; application-layer attacks; taxonomy.

Reference to this paper should be made as follows: Mantas, G., Stakhanova, N., Gonzalez, H., Jazi, H.H. and Ghorbani, A.A. (2015) ‘Application-layer denial of service attacks: taxonomy and survey’, *Int. J. Information and Computer Security*, Vol. 7, Nos. 2/3/4, pp.216–239.

Biographical notes: Georgios Mantas is currently a Post-Doctoral Researcher Fellow at the 4TELL Research Group, Instituto de Telecomunicações – Aveiro, Portugal. His recent research interests include authentication mechanisms, detection and mitigation of denial of service attacks, secure network coding and privacy preserving mechanisms. He was with the Information Security Centre of Excellence, University of New Brunswick, New Brunswick, Canada. He is currently with the Instituto de Telecomunicações – Aveiro, Campus Universitário de Santiago, Portugal.

Natalia Stakhanova is the New Brunswick Innovation Research Chair in Cyber Security at the University of New Brunswick, Canada. She completed her PhD work in Network Security at the Iowa State University, USA, in 2007. She work revolves around building secure systems and includes intrusion detection, malicious software, and security evaluation and assessment. She was the recipient of McCain young Scholar Award and the Nokia Best Student Paper Award. She works closely with industry on a variety of R&D projects. She developed a number of technologies that have been adopted by high-tech companies such as IBM. She currently has three patents in the field of computer security.

Hugo Gonzalez is a PhD student at the Information Security Centre of Excellence, University of New Brunswick, Canada. He is a faculty member of the Polytechnic University of San Luis Potosi, Mexico. His current research interests include network security and malware analysis. He is a member of the Association for Computing Machinery, the IEEE Computer Society and The Honeynet Project.

Hossein Hadian Jazi is a Master student at the Information Security Centre of Excellence, University of New Brunswick, Canada. He received his BSc in Software Engineering from Isfahan University of Technology. He is an active researcher and his current research interests include network security and malware detection.

Ali A. Ghorbani is currently serves as the Dean of the Faculty of Computer Science and Director of the Information Security Centre of Excellence at the University of New Brunswick, Fredericton, Canada. He is the Co-Editor-in-Chief of *Computational Intelligence, An International Journal*. He supervised more than 150 research associates, postdoctoral fellows, and undergraduate and graduate students and authored more than 250 research papers in journals and conference proceedings and has edited 11 volumes. He is the co-inventor of three patents in the area of network security. His current research focus is cybersecurity, complex adaptive systems, critical infrastructure protection and web intelligence.

1 Introduction

The frequency and power of denial-of-service (DoS) attacks have marked the first quarter of 2013 as the worst quarter for DoS attacks in history (Prolexic, 2013). Leveraging botnets and high-speed network technologies, modern DoS attacks exceed the scale of 100 Gbps becoming a major threat on the internet (Prolexic, 2013). Being one of the oldest type of attacks, DoS attacks are known for their disruptiveness and ability to deplete the computing resources and/or bandwidth of their victims in a matter of minutes. In spite of being trivial in execution, they are easily detectable mostly due to their dynamic and voluminous attack rates. As a result, the recent years have seen a growing trend towards more sophisticated application-layer DoS attacks.

As opposed to traditional DoS attacks, application layer DoS attacks are perceived as stealthy, sophisticated and undetectable at the network layer (Xie and Yu, 2009). Focusing on specific characteristics and vulnerabilities of application layer protocols, application layer DoS attacks are capable of inflicting the same level of impact as traditional flooding DoS attacks at a much lower cost.

With the latest escalation of application-layer DoS attacks, the research community has focused its attention on defence and mitigation techniques for this type of attacks. Since effective defences require a comprehensive understanding of the existing application-layer DoS attacks, several existing studies in the field attempted to provide some classification of these attacks.

As such, one of the first attempts to classify application-layer DoS attacks was conducted by Ranjan et al. (2006). Analysing exploitable workload parameters, the authors categorised the existing attacks into three classes; request flooding attacks that employ high rates of requests to deplete server's resources; asymmetric attacks that focus on high workload requests; and repeated one-shot attacks that spread workload across multiple sessions and initiate sessions at high rates.

This classification was later adjusted by Yu et al. (2007) to distinguish session flooding attacks, request flooding attacks, and asymmetric attacks, eliminating the category of repeated one-shot attacks. Furthermore, a study by Xuan et al. (2010) confirmed a necessity to differentiate between high-rate attacks and high-workload attacks.

From the industry perspective the application-layer DoS attacks were divided into four categories: request-flooding attacks, asymmetric attacks, repeated one-shot attacks and application-exploit attacks (Arbor Networks, 2012). This classification, although does not provide necessary depth, unifies the views expressed in the academic studies mentioned above.

Several other studies attempted to narrow the classification of various types of application-layer attacks. Focusing specifically on web applications, Cambiaso et al. (2012) developed a taxonomy of slow application-layer DoS attacks. Zargar et al. (2013) looked at the flooding-based application-layer DoS attacks, offering a very broad classification into two categories: reflection/amplification-based flooding attacks and HTTP flooding attacks. The latter category also included non-volumetric attacks, such as the slow-read and slow-send HTTP attacks that are generally executed with strategically sent requests.

There were a number of surveys giving introduction into traditional DoS attacks and their defences (Mirkovic and Reiher, 2004; Beitollahi and Deconinck, 2012; Peng et al., 2007). Although providing a good analysis of network layer DoS attacks challenges, these studies do not give the depth necessary for comprehensive understanding of application-layer DoS attacks and consequently for the development of practical defence mechanisms.

In this paper, we address this problem and develop a taxonomy of application-layer DoS attacks. Along with this taxonomy we consider representative examples of various attack categories and outline characteristics of attacks.

The comprehensive understanding of the existing application-layer DoS attacks, supported by a unified terminology, is a necessary foundation for the advanced development and deployment of reliable and efficient defence mechanisms against this type of attacks.

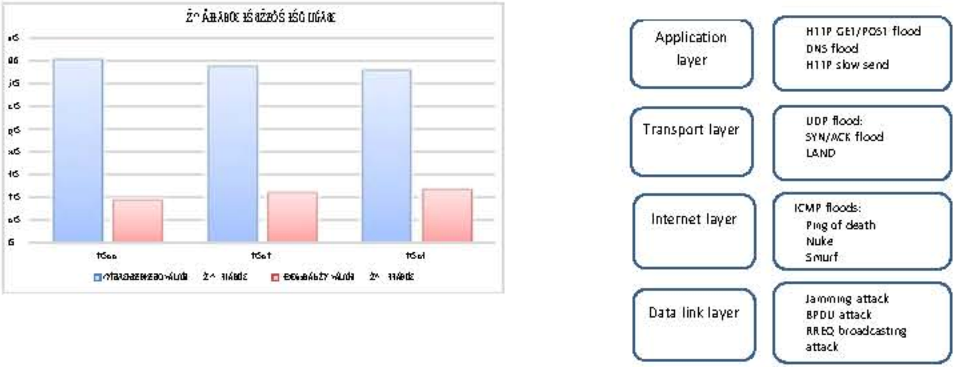
Variability of studies and close industry attention to the field have quickly revealed the gap in terminology and understanding of these attacks. Although the majority of the existing studies in the field attempt to provide some classification, they often contradict each other in defining various attack types, resorting to inconsistent terminology or simply vague description of any given attack. For example, a study of application-layer DoS attacks by Durcekova et al. (2012) referred to a type of a slow-rate attack as the Slowloris attack1. Yu et al. (2007) explored attack defences and divided application-layer DoS attacks into session flooding, request flooding, and asymmetric attacks, while Ranjan et al. (2009) referred to the same attack types as repeated-on-shot, request flooding and asymmetric attacks.

Our contributions in this work are two-fold:

- To give researchers a better understanding of the application-layer DoS attacks and defences. This taxonomy provides a comprehensive overview of the existing application-layer DoS attacks. For each category, we provide its definition, distinctive features and representative examples derived from both industry and academia.
- To provide a foundation for organising research efforts in the field of application-layer DoS attacks.

To the best of our knowledge, a comprehensive and systematic classification of application-layer DoS attacks does not exist. The goal of this paper is to provide a complete taxonomy of the existing application-layer DoS attacks accompanied by representative examples. This paper is the first attempt to organise existing research efforts in this area that, as we hope, will be extended by other researchers in the future.

Figure 1 (a) DoS attacks over the past years (b) The DoS attacks spectrum (see online version for colours)



(a) (b)

This paper is organised as follows. Section 2 provides a general background of the application-layer DoS attacks. Section 3 proposes a taxonomy and defines the adopted criteria for the classification of the application-layer DoS attacks. In Section 4, we discuss examples of the most well-known application-layer DoS attacks in relation to the proposed taxonomy. Finally, Section 5 concludes the work and outlines avenues for future research.

2 The application-layer DoS attacks

With a focus on resource exhaustion, DoS attacks spawn a broad spectrum of variants capable of depleting resources at any layer of the traditional TCP/IP architecture. Figure 1 gives examples of DoS attacks traditionally seen at various layers of the TCP/IP architecture.

Historically, DoS attacks were consistently present among internet threats. These attacks targeting network and transport layers were prevalent on networks mostly due to their effectiveness and simplicity in execution. However, with the fast development of network infrastructures, simple attacks became less effective. Although flooding is still a viable disruption method, modern DoS attacks rarely need to exceed the throughput capacity of the network as the target often fails long before the capacity is reached.

This gave a boost to development of application layer DoS attacks, that became one of the rapidly growing categories, representing more than 20% of all attacks in 2013 (Prolexic, 2013). The growth of application layer DoS attack incidents throughout the years are presented in Figure 1.

This new category of attacks specifically targets the application-layer and affects user applications without affecting network resources. There are several unique characteristics that let application-layer DoS attacks stand out among the traditional DoS attacks:

- Limited resources consumption: Traditionally, network-based DoS attacks are the flooding attacks requiring a large amount of resources (e.g., compromised ‘zombie’ devices, bandwidth) to be able to exceed network capacity at the target site. With the focus on higher levels, the need for resources on the attacker side has been diminished significantly, due to the limited capacity at the targeted applications. This opened a new possibility of executing severe DoS attacks with the use of automated scripts at a single machine. For example, slow-send attacks can be mounted by transmitting legitimate HTTP POST requests at a slow pace, occupying the resources and preventing other clients from completing their connections. This type of attacks can be effective even with the presence of a single attacker (Gonzalez et al., 2014).
- Targeted damage: The shift to the application-layer necessitates a strategic selection of a specific target (e.g., service, application) on a system. This leads to a concentrated damage incurred to that single target leaving the rest of the unrelated services intact.

- Stealthiness: This characteristic of application layer attacks was repeatedly emphasised in security domain (Xie and Yu, 2009; Tang, 2012; Beitollahi and Deconinck, 2014). The intelligent execution of the application-layer DoS attacks makes malicious traffic practically indistinguishable from the traffic generated by legitimate users. This comes mostly from the fact that the anomalous nature of the application-layer network traffic is invisible at the lower levels due to

  1 the necessity to piggyback on legitimate connections (e.g., HTTP slow-read attack requires an established TCP connection)

  2 the focus of the attacks (i.e., targeted damage) that requires the attacker to use available resources strategically, often exploiting legitimate features of application-layer protocols.

  This situation is aggravated by the lack of appropriate defences that allows these attacks to fly under the radar of traditional intrusion detection systems, most of which are deployed at the network level. Modern application-layer attacks often masquerade as human behaviour [e.g., through the use of headless browsers that mimic legitimate browser behaviour (Bains, 2014)] which allows them to blend with normal network traffic even more.

  In spite of this, there were several studies that questioned the effectiveness of certain types of application-layer DoS attacks (Gonzalez et al., 2014).

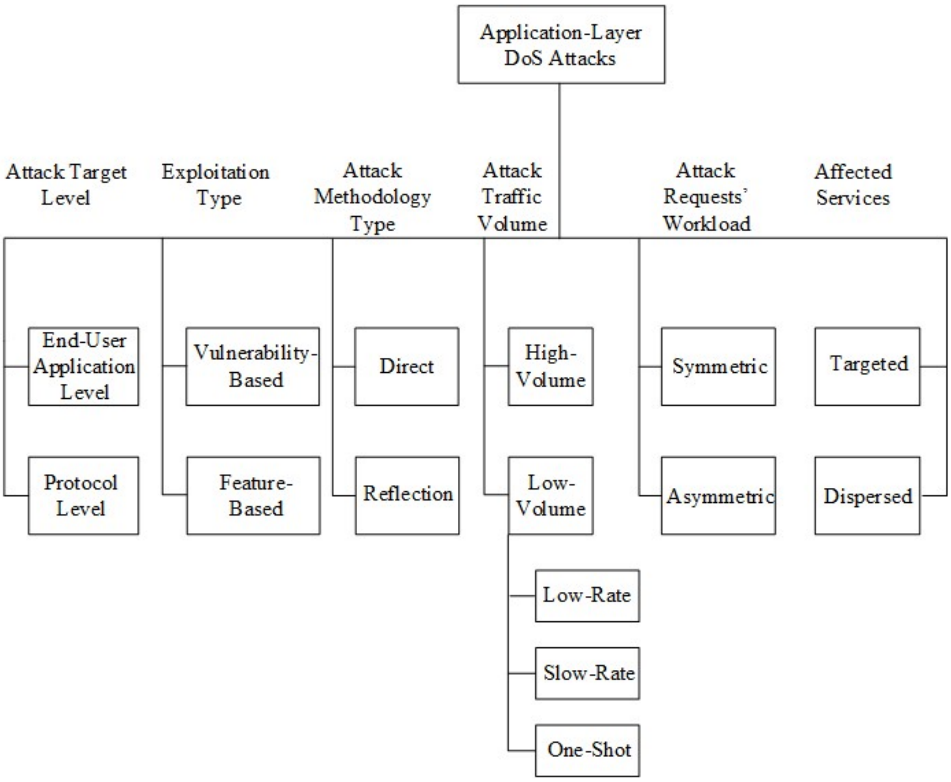
3 Taxonomy of application-layer DoS attacks

The diversity of application-layer DoS attacks calls for their detailed analysis. The critical step in characterising and understanding this variability is the definition of a set of essential parameters (i.e., features) that will provide a comprehensive classification of the currently existing attacks. We build on the existing body of knowledge about network layer DoS attacks, and in this work we only focus on discriminative features that appear at the higher levels.

To devise this classification, we consider a set of features that describe the general steps of an attack, namely, system exploitation; the attack execution; and its effect on the targeted system. Thus, we extract the following features characterising exploitation stage: attack target level and exploitation type; describing execution: attack methodology type, attack traffic volume, and attack requests' workload; and finally its effect on a victim: affected services.

These features will help researchers to navigate through the diversity of application-layer DoS attacks and as such form a foundation of the proposed taxonomy (see Figure 2).

Figure 2 Taxonomy of application-layer DoS attacks



3.1 Attack target level

Although implemented at the application-layer, DoS attacks can either target specific application protocol features or focus on components exposed at the user interface level (e.g., a search component in a content management system) and accessible through application layer. From this perspective, the attacks are divided into end-user application level attacks and protocol level attacks.

- End-user application level attacks: target weaknesses of applications running on the victim's system (e.g., web servers, database servers) in order to deplete its resources. The main premise of this category of attacks is the necessity to expose certain functions through application platforms enabling legitimate access to underlying systems. As a result, a legitimate functionality or a vulnerability is leveraged to access underlying and otherwise hidden servers. For example, a common attack against retail websites is to put a few thousand items in the shopping cart, continually adding items and refreshing the cart. The necessity to validate each of the requests would stall the underlying database server (Securosis, 2013).

This category also includes attacks exploiting application business processes, often referred to as business logic flaws. Customised for each application, business processes are unique which makes their automated detection challenging if not impossible. Exposing business logic flaws requires a deep understanding of the application and its business processes, often leading to specialised time-consuming manual testing. The examples of such exploits include the flooding of online polls, an increasingly large number of simultaneous bidders in online auctions, etc. Finally, end-user application level attacks are highly specific and damaging.

- Protocol level attacks: Specifically aim to exploit flaws of the protocols' implementation or legitimate features of the protocols' functionality. Among the most common targeted application-layer protocols are HTTP, FTP, DNS, VoIP and SMTP (Arbor Networks, 2012).

3.2 Exploitation type

Although the exploitation of an application's vulnerabilities is still a major concern, the focus of DoS attacks in the recent years has shifted towards an abuse of legitimate application features and functionality. We differentiate between vulnerability-based attacks and feature-based attacks.

- Vulnerability-based attacks: Exploit the flaw or a bug in the system design, implementation or configuration in order to exhaust the system's resources. The number of annually discovered vulnerabilities has been steadily increasing over the years. An example of the vulnerability-based attacks is the Apache Range Header attack. This is a web server specific attack where a malicious HTTP request indicating a very large overlapping range is sent to the web server, resulting in memory exhaustion and server crash. Nowadays, this attack is no longer effective since a released patch mitigates this flaw by checking the received request at the server side. When the byte range parameter is included in the received request, the sum of all ranges is calculated. In the case where the sum is larger than the requested file, the request is ignored and the entire file content is sent (Apache, 2011; Cambiaso et al., 2012).

  The DoS attacks executed with the use of malformed requests/messages are also considered under this category. In general, the malformed messages often aim to gain system control, and thus they are not typically considered as DoS attack methods. However, there is a class of malformed requests that subvert the targeted system resulting in its collapse and effectively the denial of service for the legitimate users. Theoretically, any request not adhering to the developer's expectations might lead to incorrect system behaviour and potentially, a DoS attack. Examples of these attacks seen in practice include malformed SNMP requests (Symantec, 2010; CISCO, 2002) and malformed XML data (IBM, 2013a, 2013b).

- Feature-based attacks: As opposed to vulnerability-based attacks that exploit system's defects, feature-based attacks abuse legitimate functionality of applications and protocols. Since the exploited functionality is intended, there are two common ways to perform this type of attacks: through flooding to overwhelm the allocated resources (e.g., by sending large number of requests), and through intelligent one-shot execution that occupies system resources with a large workload. Many of the end-user application level attacks fall in this category. For example, allowing a user to enter a complex query requiring a significant processing power or occupying a large chunk of memory ties up the server resources which slows down its overall performance for the legitimate users. Another example is the Snowflake search attack (i.e., a query that requests to compare a large set of values against a variable that never matches) (Securosis, 2013).

3.3 Attack methodology type

The origin of an application-layer DoS attack defines its methodology type during its execution. Based on the origin of the attack traffic, we differentiate between direct attacks and reflection attacks. Although not consistently, this category has been seen as a part of traditional DoS attacks classification (Beitollahi and Deconinck, 2012). The primary difference for application layer

- Direct attacks: Require the attack traffic to be sent directly from a true attacking source towards the targeted victim. These are generally attacks that rely on an established and legitimate connection. For example, the HTTP protocol-based attacks (e.g., HTTP POST/GET) fall into this category.
- Reflection attacks: Are executed through intermediate reflectors (i.e., legitimate devices that unknowingly participate in an attack by redirecting/reflecting the traffic to the primary target). The classic example of reflection DoS is the DNS reflection attack (also referred to as DNS amplification attack). Taking advantages of the nature of the DNS protocol that permits queries from forged IP addresses, the attack generates DNS queries that require a large response that can be effectively redirected to the victim consuming its resources. DNS reflection attack was the largest source of traffic in the notorious Spamhaus attack, labelled as the largest DDoS attack in 2013 (Prince, 2013).

  The execution of a reflection attack provides two main advantages to the attacker:

  - the ability to hide the true source of the attack traffic and increase the difficulty in the identification of the attack
  - amplify the attack traffic making it even more potent.

  As a result, reflection attacks often have a devastating effect on the victim site.

3.4 Attack traffic volume

Irrespective of how an application-layer DoS attack is being executed, any attack involves a certain amount of malicious traffic to the victim. Based on the volume of the sent traffic, we differentiate high-volume attacks and low-volume attacks.

- High-volume attacks: Also known as flooding attacks, transmit high volumes of application-layer requests (e.g., HTTP GETs, DNS queries, SIP INVITEs) to the victim in an attempt to deplete its resources. Being one of the classic attack methods associated with DoS attacks, flooding attacks, first seen in 1989 (Defence.net, 2013), have been flourished into a diverse spectrum of attacks. Traditional flooding attacks have been focused on depleting network bandwidth, however with the rapidly increasing capacities of modern network infrastructures, such direction became less effective and more resource intensive for the attackers. The situation also revealed the vulnerability of the local targets, proving that server's resources is a primary bottleneck in this type of attacks. Thus, application-layer flooding attacks intend to deplete server's resources at the application level (e.g., CPU resources through high-computation queries, memory pool through a large number of requests). SMTP flooding attack comprises a representative example of the application-layer flooding attacks. In this attack, the objective is to overwhelm the targeted SMTP server by sending to it a high volume of e-mails. When the SMTP server is under attack and the incoming traffic load exceeds a specific threshold, then its performance is affected resulting in delayed e-mail delivery or even complete system crash (Bencsáth and Rónai, 2007).

- Low-volume attacks: In contrast to flooding attacks, these attacks are not required to send a high volume of attack traffic. The low-volume attacks are generally executed with small amounts of attack flows transmitted strategically to degrade the victim's performance. There are two variations of low-volume DoS attacks: those that require a stream of attack flows in order to achieve the attack goal, and those that can be executed with a single shot (i.e., a connection, a network packet).

  The first variation of low-volume attacks includes the so called low-rate and slow-rate application-layer DoS attacks. Low-rate attacks are based on the realisation that high volumes of traffic are visible and highly suspicious. Thus sending attack traffic by periodic short-time pulses, the attacker significantly reduces the possibility of detection. Generally, low-rate attacks are characterised by a specific traffic pattern formed by the attack requests strategically sent to the victim. Rather than overwhelming target's resources, low-rate attacks aim to occupy all the allocated spots (e.g., in a service queue). Therefore, the attacker's success is often dependent on his ability to predict the appropriate point in time when the positions in the service queue become available in order to fill them before any legitimate request arrives. The timing of these positions can be generally predicted (to various degree of accuracy) through analysis of victim's behaviour and the use of fixed temporal patterns.

  Overall, low-rate attacks offer two main advantages to the attacker. On the one hand, the attack's behaviour effectively causes denial of service with minimal resources on the attacker's side. On the other hand, the intelligent execution allows the attack to remain undetectable, flying under the radar of the traditional network-layer-based intrusion detection systems.

  Similarly to low-rate attacks, slow-rate attacks exhaust server's resources causing a denial of service without sending an overabundance of network packets. An attack generally exploits one of the common properties of the application-layer protocols to reserve resources until after a completion of a connection. In this context, slow-rate attacks target specific applications with a single computer, leaving the rest of unrelated services intact.

  Although the effectiveness of both low-rate and slow-rate attacks has come under question (Gonzalez et al., 2014), their detection remains challenging.

One-shot attacks are powerful attacks that are able to inflict damage to the victim with a single connection, one request, or one network flow. These attacks generally exploit a specific weakness in order to consume excess amounts of the victim's resources. Both Snowflake search attack (Securosis, 2013) and Apache Range Header attack (SpiderLabs, 2011) are good examples of this type of attacks. Similarly to the other types of low-volume attacks, one shot attacks are almost undetectable and require customised mitigation mechanisms.

3.5 Attack requests' workload

Depending on the workload of the requests, application-layer DoS attacks are divided into symmetric attacks and asymmetric attacks.

- Symmetric attacks: Include all types of the application-layer DoS attacks sending requests that do not generate high volume of work for the targeted server.
- Asymmetric attacks: Leverage normal rates of high workload malicious requests. These requests are specifically designed to consume a significant amount of the victim server's resources. The main objective of the asymmetric attacks is to generate a large amount of work for the victim in order to consume a significant portion of its resources, with a single malicious request. Thus, in asymmetric attacks, a high rate volume of attack traffic is not required in order server resources depletion and normal service operations disruption to be achieved. This makes the detection of asymmetric attacks more difficult compared to the symmetric attacks. Additionally, the asymmetric attacks cause more damage per request than symmetric attacks (Arbor Networks, 2012; Zargar et al., 2013).

3.6 Affected services

Depending on the services simultaneously impacted by the application-layer DoS attacks, we differentiate between targeted attacks and dispersed attacks.

- Targeted attacks: target a specific service leaving the rest of unrelated services intact. An example of such attack is the Slow HTTP POST attack that transmits to the victim server incomplete HTTP requests causing the victim server to reserve resources for open connections waiting for their completion.
- Dispersed attacks: also target a particular service while affecting other services (perhaps inadvertently) at the same time. We can differentiate two variations of dispersed attacks: those that affect other services at the same layer, and those attacks that affect other services beyond one layer.

Most of the DNS attacks also fall under the second category. Due to the nature of the DNS service, specifically attacking DNS flaws, of even one server, has an immediate and damaging impact on the ability of the legitimate clients to access other unrelated servers (e.g., database server, web server).

The major challenge with this type of attacks is the difficulty in predicting the overall impact. While many of these attacks are relatively simple in detection, they are difficult to defend against.

Table 1 A summary of common application-layer DoS attacks

Level	*Attack*	*Description*	*Target focus*	*Classification*	*Existing tools*
End-user attacks	HashDoS	Hash dos attack exploits weak hash functions of hash tables and use low bandwidth rate to generate hash collisions to degrade the CPU performance.	CPU resources	End-user application level, asymmetric, direct, low-volume (one-shot)	Xenotix hash DoS tester OpenSecurity (2012)
	XDoS (Jensen et al., 2007)	Attacker exploits the XML language structure and the Service Oriented Architecture (SOA).	System or service (shut down a web service or system running that service)	End-user application level, asymmetric, direct	No specific tools
	ReDoS (Crosby and Wallach, 2003)	The attack exploits a weakness in regular expression implementation that relates to certain types of expressions requiring large computational resources (exponentially related to input size).	Specific computer or computational resources	Vulnerability-based	No specific tools
	Apache range header	In this type of attack, the attacker makes use of the byte range filter in the Apache HTTP Server to form a denial of service via a Range header that expresses multiple overlapping ranges.	Resources (exhaust the memory and CPU of the victim server)	End-user application level, vulnerability-based	Apache killer (S.A. Summit, 2014)
HTTP	High-volume: HTTP flooding	In this attack a large number of GET or POST HTTP requests are sent to the targeted server. Because of using legitimate HTTP requests, the targeted server is not able to distinguish these requests from the benign HTTP requests.	Resources (targeted server resources)	Protocol level, high-volume	High Orbit Ion Cannon (HOIC) (High Orbit Ion Cannon DDoS tool, 2011), Dirt Jumper (http://www.deependresearch.org/2011/10/), HTTP Unbearable Load King (HULK) (HTTP Unbearable Load King, 2012), bonesi (http://code.google.com/p/bonesi/), ddosim (DDOSIM-Layer 7 DDOS Simulator, 2010), XOIC (XOIC, 2012), PyLoris (PyLoris, 2009)
		Single-session attack exploits the feature of HTTP 1.1 in order to send multiple requests within a single HTTP session.	Resources (targeted server resources)	Protocol level	GoldenEye (http://packetstormsecurity.com/files/120966/GoldenEye-HTTP-Denial-Of-Service-Tool.html)
		Multiple attack creates multiple HTTP requests via integrating requests into a single packet rather than issuing them one after another during a single HTTP session.	Resources (targeted server resources)	Protocol level, asymmetric	POST-it (http://packetstormsecurity.com/files/98072/POST-it-Denial-Of-Service-Tool-1.1.0.html)

Table 1 A summary of common application-layer DoS attacks (continued)

Level	*Attack*	*Description*	*Target focus*	*Classification*	*Existing tools*
HTTP	Low-volume: LoRDAS Slow HTTP sent Slow HTTP read	In this attack, attackers use existence of a temporal deterministic behaviour in HTTP servers to predict the instants at which there are available positions in their service queues and, thus, to lunch the LoRDAS attack.	Resources (exhaust the resources of the target server)	Low-volume (low-rate), symmetric, direct, targeted damage	Slowloris (RSnake, 2009)
		This type of attacks sends legitimate incomplete HTTP requests which lead the victim server to reserve resources for open connections waiting for their completion.	Resources (targeted server resources)	Low-volume (slow-rate), symmetric, direct, targeted damage	RUDY (Raz, 2010), OWASP HTTP POST (Brenann, 2010), slowloris (RSnake, 2009), Nuclear DDoSer (Insecurity Research, 2012), Railgun (http://code.google.com/p/railgun/), Tor's Hammer (Tor's Hammer, 2011)
		The attacker lunches the attack by sending a legitimate HTTP request to the victim and then the attacker read slowly the response send by the victim.	Resources (targeted server resources)	Low-volume (slow-rate), targeted damage	Nuclear DDoSer (Insecurity Research, 2012)
DNS	DNS flooding	In this attack, DNS server receives valid but spoofed DNS request packets at a high rate and from a very large pool of source IP that the server cannot differentiate them from not spoofed requests due the fact that the content of the spoofed DNS request packets are designed to mimic actual DNS requests.	CPU resources (DNS server)	Protocol level, feature-based, high-volume, dispersed	dns-flood (http://code.google.com/p/dns-flood/source/browse/dnsflood.c)
		DNS amplification employs DNS servers to amplify attack traffic by sending DNS responses to an unsuspecting target.	CPU resources of the targeted victim	Reflection	DNS Reflection/ Amplification Attack Tool (DNS Reflection/ Amplification Attack Tool, 2013)
SIP	SIP flooding	This attack sends bogus SIP INVITEs in order to overwhelm the session initiation protocol (SIP).	Specific system and resources (exhaust the resources both of the SIP proxy server and the caller)	Protocol level, vulnerability-based	SIP Tester (SIP DoS Simulation using SIP Tester, 2013)
	SIP malformed message	This attack exploits handling of malformed SIP packets.	String buffers	Protocol level, vulnerability-based, direct, low-volume, targeted damage	SIP Tester (SIP DoS Simulation using SIP Tester, 2013)
SMTP	SMTP flooding	In this attack, the attackers send high volume of emails to the SMTP server in order to overwhelm the SMTP server.	Resources (delaying in email delivery or crashing the server)	Protocol level, high-volume, direct, symmetric	DDOSIM (DDOSIM-Layer 7 DDOS Simulator (2010), PyLoris (PyLoris, 2009)
SSL	SSL renegotiation	This attack exploits lack of cryptographic binding in SSL Renegotiation that allows the attacker to inject malicious traffic in the server-client communication or flood the server with renegotiation requests	CPU resources	Vulnerability-based, asymmetric, direct, targeted damage	THC-SSL-DOS (THS SSL DoS tool, 2011)

4 Examples of application-layer DoS attacks

In this section, we discuss examples of the most well-known application-layer DoS attacks in relation to the proposed taxonomy. A summary of the presented attacks is given in Table 1.

4.1 End-user application level attacks

End-user application level attacks (e.g., SQL injection) have been seen in the past. However, with the shift to web-based services, the end-user application level DoS attacks became more common. There have been seen a variety of end-user application level attacks mostly focused on web applications. One of the legacy attacks from this domain is a brute force attack on the application login page that aims to guess the login credentials. Among other attacks that have been recently seen are the HashDoS attack, the XDoS attack and the ReDoS attack.

HashDoS attack is a powerful asymmetric attack operating at the end-user application level. It has a large impact on the CPU performance of the victim server, by sending a low volume of carefully chosen small request messages. The attack exploits vulnerable hash functions used in hash table implementations of many programming languages (e.g., PHP, ASP.NET, Python, and Java) for web applications.

In web applications, hash tables are data structures that programmers prefer for data storage, since they store key-value pairs very efficiently. However, hash tables are not efficient in the worst-case, where all of the keys are assigned by the hash function to the same hash value, causing hash collisions. This happens when hash tables make use of weak hash functions and this is the vulnerability that attackers exploit in order to degrade the CPU performance of the targeted web servers (Crosby and Wallach, 2003).

Firstly, the attacker creates web application forms' variable names with the same hash values. Then the attacker sends, in low bandwidth rate, POST requests including many of the colliding variable names. As a result, the hash table on the web server side is overwhelmed and the web server's CPU spends all its time managing the collisions, which is a computationally intensive task (Cambiaso et al., 2012; Falkenberg et al., 2013). According to the attack methodology type, HashDoS attack is a direct attack sending directly the POST requests to the targeted victim.

Similar to HashDoS attack, XDoS attack targets servers hosting web services. This is a relatively new attack that exploits the XML language structure and the Service-Oriented Architecture (SOA) to deplete server's resources making them unavailable to the legitimate users. There are known several variations of the XDoS attack:

- XML flooding attack is one of the straightforward implementations of the XDoS attack. Although it is a truly application-layer DoS attack, it is often excluded from the XDoS category, mainly due to the lack of stealthiness and simplicity in execution.
- Oversized XML payload attack involves transmitting an excessively large payload to consume the targeted server's resources. Originally designed as compact communication format, XML messages are often processed in a buffered mode (i.e., the whole message resides in a buffer after it is received). This presents a significant challenge for the provided service if a large payload is received by the server. This

situation is often aggravated by the lack of a maximum limit on the size of the XML message that essentially opens up the server to a DoS attack. There have been two variations of this attack: single XML message and multiple XML message attacks, aiming to take down the targeted server with a single and multiple messages respectively.

- XML entity expansion (XEE) attack also known as XML Bomb or a Billion laughs attack. This attack exploits the XML message structure defined with a document type definition (DTD). Crafting a DTD with a large number of nested entities can lead to explosive growth of data when XML message is parsed. There were reported several variations of the XEE attack, namely, the Quadratic Blowup attack (Sullivan, 2009) and the Attribute Blowup attack (The Web Application Security Consortium, 2000).
- External entity reference attack exploits the XML parser property to resolve references to external resources (e.g., external URIs) during parsing and bring the requested content. This opens up a wide window for attacks. An XDoS attack can be triggered by simply specifying non-existent resources, causing parser to remain in an infinite loop. Compared to the XEE attack that occupies CPU resources, external entity reference attack only consumes a single thread of execution which has a limited impact on server's performance. A slight improvement in damage can be generated by opening multiple connections requesting to download media content (Ye, 2008).

Although only certain types of the XDoS attack has been seen in the wild (e.g., XML bomb), the potential danger of the XDoS attacks in general has been repeatedly emphasised (Padmanabhuni et al., 2006; Jensen et al., 2007; Ye, 2008; Chonka et al., 2009; Karthigeyan et al., 2012). Overall, the XDoS attacks are asymmetric attacks since only a small portion of the processing power of the attacker's machine is required in order to send an attack payload to the victim.

Also asymmetric in nature are the ReDoS attacks that make use of regular expressions (regexes) with specific patterns. Regular expression matching is a ubiquitous technique often used for input validation in web applications. The vulnerability of regexes lies in the wide adoption of backtracking algorithms [rather than the traditional deterministic finite automaton (DFA) contraction] in regular expression matchers employed in modern programming languages. As a result, crafting a malicious regex will result in the matcher taking exponentially long time to process an expression and essentially failing to terminate, causing a DoS attack. This attack was initially presented by Crosby and Wallach (2003) and has received a significant research attention since more vulnerabilities of regex matchers were discovered (Cambiaso et al., 2012; Kirrage et al., 2013).

4.2 Protocol level attacks

In spite of the variety of end-user application level attacks that are conceivable, only a handful of application-layer protocols are commonly attacked. Among them are HTTP, DNS, and SMTP protocols. However, with the emergence of voice over IP (VoIP), DoS attacks based on SIP protocol are quickly becoming one of the major threats.

4.2.1 HTTP-based attacks

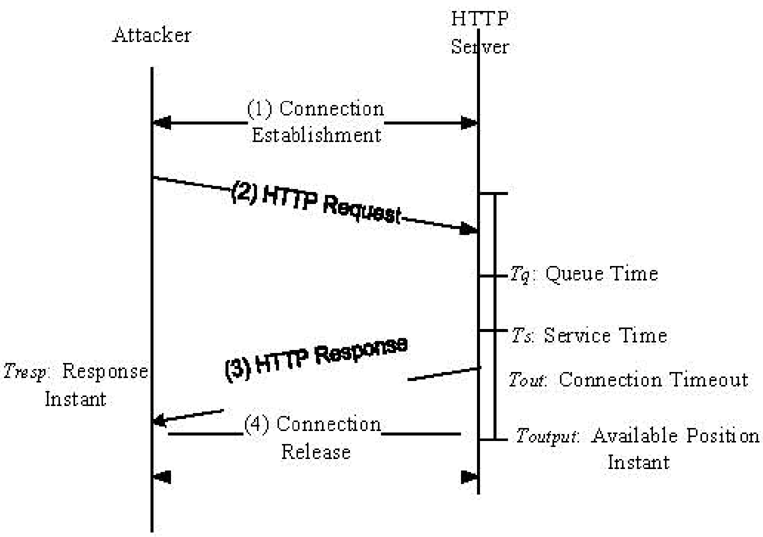
High-volume attacks: HTTP GET/POST flooding attack is perhaps the most well-known flooding application-layer DoS attack operating at the protocol level. Usually, it is a DDoS attack executed with multiple distributed attacking entities. Its main objective is to overwhelm the victim server with a high volume of valid HTTP requests leading to the depletion of its resources. The most common form of this type of attack uses GET requests but also POST requests can be used as well. Since the generated HTTP requests by the attacker have legitimate HTTP payloads and are sent via normal TCP connections, the targeted server is not able to distinguish them from the benign HTTP requests. Hence, the targeted server's resources are exhausted, since it has to handle all the received HTTP requests as benign requests resulting to completely crash of the targeted server. Thus, HTTP GET/POST flooding attack is categorised into the disruptive attacks. However, due to the fact that the malicious traffic is transmitted at a constant and high rate, the attack can be easily detectable, especially, in the case when it follows the one-to-one execution type. However, for the distributed HTTP GET/POST flooding attacks that follow the many-to-one execution type, the detection and mitigation are two challenging issues.

There are two variations of the HTTP GET/POST flooding attack: the single-session HTTP GET/POST flooding and the multiple HTTP GET/POST flooding attacks. Single-session attack exploits a legitimate functionality of the HTTP protocol to render the victim server unresponsive by consuming its resources. Specifically, this attack abuses the feature of HTTP 1.1 to allow multiple requests within a single HTTP session. Thus, the attacker achieves to deplete the server's resources with low traffic volume. The session rate of the single-session HTTP GET/POST flooding attack is limited compared to the session rate in the HTTP GET/POST flooding attack. This has as a result the attack to bypass detection mechanisms based on the session rate despite the disruptive impact on the victim server's side (Zargar et al., 2013).

Multiple HTTP GET/POST flooding attack, on the other hand, achieves its objectives by generating high volumes of workload for the victim server and consuming its resources with low attack traffic volume. For this reason, this attack serves as a representative example of the asymmetric application-layer DoS attacks. The attack is executed by creating multiple HTTP requests and integrating them into a single packet instead of transmitting them one after another during a single HTTP session. As a result, the attack causes a total disruption of the provided service to legitimate users or it brings the server completely down (Zargar et al., 2013; Ranjan et al., 2009; Arbor Networks, 2012; Das et al., 2011).

Low-volume attacks: LoRDAS attack was proposed by Macia-Fernandez et al. (2008, 2006) and is an example of a low-rate application-layer DoS attack. This is a specific low-rate attack targeting the persistent HTTP servers by transmitting attack traffic by periodic short time pulses at strategically chosen time instants. The existence of a temporal deterministic behaviour in these servers comprises a vulnerability that allows an attacker to predict the instants at which there are available positions in their service queues and, consequently, to lunch the LoRDAS attack. The case of the persistent HTTP servers can be extended to any concurrent server.

Figure 3 A process of predicting available positions in the queue of an HTTP server



The proposed process for predicting the instant T_{output} at which there is an available position in the service queue of the targeted HTTP server is illustrated in Figure 3. The attack instant, T_{attack}, which is the same as the T_{output}, is calculated by the following formula:

$$T_{attack} = T_{resp} - \overline{RTT} + T_{out}$$

The persistent connection timeout T_{out} can be easily determined with just a few tests by the attacker. In addition, the attacker can record the instant of the HTTP response reception T_{resp}. Finally, $\overline{RTT}$ is the mean value of the round trip time between the attacker and the server.

The synchronisation of the arrivals of the attack messages to the targeted server has to address two issues. Firstly, the predicted instant of an available position varies with respect to the real instant of an available position occurred in the server, since the real instant depends on the service time, which is a random variable. Secondly, the variance of the RTT between an attacker and a server also affects the synchronisation. Thus, in order to address the effect of these variations, an attacker sends a small burst of traffic including more than one attack packet and attempting to synchronise their arrival to the server around the predicted instant.

Finally, the intelligent execution of the LoRDAS attack reduces the possibilities of detection from traditional network-layer-based intrusion detection mechanisms as well as enables the attacker to degrade the targeted server's performance with minimal resources. Thus, the LoRDAS attack can be implemented successfully by following the one-to-one and the one-to-many attack execution types.

4.2.1.1 Slow HTTP sent attack

This attack comprises an example of slow-rate attacks. Its objective is to tie up server resources by slowly sending legitimate incomplete HTTP requests causing the victim server to reserve resources for open connections waiting for their completion. Any complete HTTP request ends with a line identified as \r \ n, followed by an empty line denoting the end of the request. The server waits the end of the request for a specific timeout before closing the connection. The timeout restarts when the server receives some data from the client. Thus, in the Slow HTTP Sent attack, the attacker never sends the final line in order to keep the connection open as long as possible. In addition, the attacker keeps the connection idle by slowly sending small amount of data to the targeted server and waiting a specific amount of time before the next sending. This amount of time has to be selected properly in order to prevent the connection release (Cambiaso et al., 2012). Similar to the LoRDAS attack, the Slow HTTP Sent attack can be implemented successfully by the one-to-one attack execution type as well as the one-to-many attack execution type.

Slowloris tool (RSnake, 2009) implements a variation of the Slow HTTP Sent attack by repeatedly sending a specific string 'X – a: b\ r \ n'. Some other tools, such as RUDY (Raz, 2010) and OWASP HTTP POST (Brenann, 2010) implement another variation of the Slow HTTP Sent attack – the Slow HTTP POST attack, in which HTTP payload is sent to the targeted server at a slow pace (e.g., 1 byte/min). Essentially, any website including forms accepting HTTP POST requests (e.g., uploading attachment, submitting feedback) is susceptible to this method of attack (Chee and Brennan, 2010).

4.2.1.2 Slow HTTP read attack

Another example of slow-rate attacks is the Slow HTTP Read attack that works by reading slowly the response instead of sending slowly the request. This attack starts with a legitimate HTTP request from the attacker to the victim server followed by a slow consumption of the HTTP response sent by the victim. The Slow HTTP Read attack achieves its objective by setting a smaller receive window-size than the send buffer of the victim server (Shekyan, 2012).

4.1.2.3 DNS-based attacks

Similar to the HTTP protocol, DNS protocol is one of the commonly targeted network protocols. Due to the ubiquitous nature of the DNS protocol, its disruption generally propagates beyond a single target to external services.

DNS flooding attack is an example of flooding application-layer DoS attacks operating at the protocol level and transmitting high volumes of DNS queries to the victim DNS server in order to deplete its resources. In more details, valid but spoofed DNS request packets, from a very large pool of source IP, are sent to the victim DNS server at a high rate. Due to the fact that the content of spoofed DNS request packets is designed to mimic actual DNS requests, the DNS server is not able to differentiate between malicious and legitimate packets and thus responds to all incoming requests. In its simple implementation, the attack causes the server to be overwhelmed by the requests leading to total disruption of the DNS service. A more advanced execution of the attack employs asymmetric nature of the requests requiring a DNS server to contact multiple servers to resolve the domain. In this case, the victim server is overwhelmed by attacker's requests and the responses received from other DNS servers. The most advanced form of this attack employed requests to resolve non-existent domains.

Despite the high volume of the malicious traffic that the DNS flooding attacks generate, they are difficult to detect mostly due to inability of the server to differentiate between malicious and benign requests.

Depending on the type of attack methodology, DNS flooding can be a direct attack with a primary victim being a DNS server or a reflection attack (also known as DNS amplification attack) with DNS servers being used as amplifiers by sending their responses to the target.

4.1.2.4 SIP-based attacks

The popularity of VoIP services coupled with the openness of its infrastructure to the internet triggered a wave of SIP-based attacks, most of which were DoS attacks. Such situation is mostly explained by the inherit difficulty of detection and mitigation of SIP-based DoS attacks in the context of simplicity of an attack execution. There are several variations of SIP-based DoS attacks (Ormazabal et al., 2008; Rafique et al., 2009):

- SIP flooding: SIP is vulnerable to a wide range of flooding attacks due to its open nature and the lack of robust security mechanisms (Hussain et al., 2013; Tang et al., 2012). On the application-layer, signalling floods are the most prominent. Signalling floods are distinguished by the large number of SIP INVITE or REGISTER messages sent to the targeted SIP proxy server, essentially causing excessive processing consuming the server's resources and as a result delaying or completely rejecting legitimate requests.

  Among these attacks, SIP INVITE flood is one of the most devastating attacks targeting SIP. In this attack, the caller (i.e., attacker) generates INVITE requests at a high rate in order to exhaust the resources of the SIP proxy server and the callee, since both of them are vulnerable to this flooding attack. According to the SIP protocol, the SIP proxy server must keep connection with the caller, from whom it received the INVITE request, at least for some time. This is the vulnerability of the SIP protocol that the caller (i.e., attacker) exploits by sending many new INVITE requests, without waiting for further signalling, in order to exhaust the resources of both the SIP proxy server and the callee.

- SIP malformed message attack: SIP is also vulnerable to malformed non-standard SIP packets aiming to overflow the string buffers. The attack is asymmetric in nature and as a result, it causes the server to reach an undefined state leading to call processing delays, and a complete denial of service. An example of such attack called Invite of Death was introduced by Rafique et al. (2009).

4.1.2.5 SSL-based attacks

With an overwhelming migration of web services to SSL protocol in an effort to improve security, SSL attacks are becoming more and more popular. SSL-based DoS attack exploits a computationally expensive SSL handshake that puts more resource intensive operations on the server than on the client. Due to this expensive nature, cryptographic parameters once established during original SSL handshake are not required during the following SSL renegotiation request initiated by a client side. As a result, there are two variations of SSL Renegotiation DoS attack:

- Traffic injection: the lack of cryptographic binding in SSL/TLS Renegotiation allows the attacker to inject malicious traffic in the server-client communication generating additional work for the server. Since SSL/TLS handshake requires much more processing power (i.e., 10 times to 35 times) on the server than on the client, this extra workload ties up the server's CPU resources causing the server to be unaccessible by legitimate clients. Pushdo botnet is known to generate this type of attack.

- SSL Flooding attack: on the other hand, employs a different strategy. Since the client is allowed to send renegotiation request at any time, the attacker can send multiple renegotiation requests per second in order to exhaust the targeted server's resources. SSL Renegotiation attack is a very effective DoS attack. Although it can be executed in Many-to-One mode to amplify the damage, the One-to-One attack execution type is sufficient to fully exhaust the server's resources and not allow any other legitimate client to establish a connection (TLS/SSL Renegotiation DOS, 2011).

5 Conclusions and future work

This paper presents a taxonomy of the existing application-layer DoS attacks accompanied with representative examples, derived from both industry and academia, in order to provide a foundation for organising research efforts in the field of application-layer DoS attacks. The incentive behind this effort has been the necessity for a comprehensive understanding of the existing application-layer DoS attacks, supported by a unified terminology, that will enable the advanced deployment of reliable and efficient defence mechanisms against these of attacks. This is essential due to the fact that the detection and mitigation of these attacks remain challenging issues. They are stealthier and more sophisticated compared to network-layer DoS attacks resulting in flying under the radar of traditional network-layer-based intrusion detection systems.

Furthermore, by devising the proposed taxonomy, a number of key features of application-layer DoS attacks is defined to characterise the variability of these attacks. The defined features describe the general steps of an attack (i.e., reconnaissance and execution), the attack characteristics and the attack effect on the targeted system. These features comprise a roadmap for researchers to navigate through the diversity of the application-layer DoS attacks and thus form a foundation of the proposed taxonomy.

As future work, we plan to define a set of proper metrics for the extracted features. The defined metrics will be essential for evaluating potential defence mechanisms against known application-layer DoS attacks. Moreover, the defined metrics can be used as parameters in the study of attacks that have not yet appeared but can be potential threats in the future. Finally, the set of the defined metrics can be used in the design and deployment of defence mechanisms against the new application-layer DoS attacks.

References

DDoS Botnet Simulator [online] http://code.google.com/p/bonesi/ (accessed 6 July 2014).

DNS-flood: a DNS flood tool in c/c++ [online] http://code.google.com/p/dns-flood/source/browse/dnsflood.c (accessed 7 July 2014).

GoldenEye HTTP Denial of Service Tool [online] http://packetstormsecurity.com/files/120966/GoldenEye-HTTP-Denial-Of-Service-Tool.html (accessed 12 May 2014).

Railgun [online] http://code.google.com/p/railgun/ (accessed 10 July 2014).

PyLoris (2009) [online] http://sourceforge.net/projects/pyloris/ (accessed 20 July 2014).

DDOSIM-Layer 7 DDOS Simulator (2010) [online] http://sourceforge.net/projects/ddosim/ (accessed 12 April 2014).

High Orbit Ion Cannon DDoS tool (2011) [online] http://hoic.99k.org/ (accessed 21 June 2014).

THS SSL DoS tool (2011) [online] https://www.thc.org/thc-ssl-dos/ (accessed 15 May 2014).

TLS/SSL Renegotiation DOS (2011) [online] https://www.ietf.org/mail-archive/web/tls/current/msg07553.html (accessed 6 August 2014).

Tor's Hammer (2011) [online] http://packetstormsecurity.com/files/98831/ (accessed 21 March 2014).

HTTP unbearable load king (2012) [online] http://packetstormsecurity.com/files/112856/HULK-Http-Unbearable-Load-King.html (accessed 26 April 2014).

XOIC (2012) [online] http://sourceforge.net/projects/xoic/ (accessed 17 May 2014).

DNS Reflection/Amplification Attack Tool (2013) [online] http://packetstormsecurity.com/files/122600/DNS-Reflection-Amplification-Attack-Tool.html (accessed 26 June 2014).

SIP DoS Simulation using SIP Tester (2013) [online] http://startrinity.com/VoIP/SipTester/SipDosSimulator.aspx (accessed 28 July 2014).

Apache (2011) *Range Attack: Apache HTTPD Security Advisory* [online] http://httpd.apache.org/security/CVE-2011-3192.txt (accessed 1 June 2014).

Arbor Networks (2012) *The Growing Threat of Application-Layer DDoS Attacks*.

Bains, J. (2014) 'Beware the headless browser DDoS attacks!' [online] http://itsecurityguru.org/gurus/beware- headless-browser (accessed 7 June 2014).

Beitollahi, H. and Deconinck, G. (2012) 'Analyzing well-known countermeasures against distributed denial of service attacks', *Computer Communications*, June, Vol. 35, No. 11, pp.1312–1332.

Beitollahi, H. and Deconinck, G. (2014) 'Connection score: a statistical technique to resist application-layer DDoS attacks', *Journal of Ambient Intelligence and Humanized Computing*, Vol. 5, No. 3, pp.425–442.

Bencsáth, B. and Rónai, M.A. (2007) 'Empirical analysis of denial of service attack against SMTP servers', in McQuay, W.K. and Smari, W.W. (Eds.): *CTS*, pp.72–79, IEEE.

Brenann, T. (2010) 'OWASP http post tool' [online] https://www.owasp.org/index.php/OWASP_HTTP_Post_Tool (accessed 18 June 2014).

Cambiaso, E., Papaleo, G. and Aiello, M. (2012) 'Taxonomy of slow DoS attacks to web applications', in Thampi, S., Zomaya, A., Strufe, T., Alcaraz Calero, J. and Thomas, T. (Eds.): *Recent Trends in Computer Networks and Distributed Systems Security, Communications in Computer and Information Science*, Vol. 335, pp.195–204, Springer, Berlin Heidelberg.

Chee, W.O. and Brennan, T. (2010) *H.....t.....t....p....p....o....s....t* [online] https://www.owasp.org/images/4/43/Layer_7_DDOS.pdf (accessed 25 May 2014).

Chonka, A., Zhou, W. and Xiang, Y. (2009) 'Defending grid web services from XDoS attacks by sota', *Pervasive Computing and Communications, 2009: PerCom 2009: IEEE International Conference on*, pp.1–6.

CISCO (2002) *Malformed SNMP Message-Handling Vulnerabilities for CISCO Non-ios Products*. http://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20020211-snmp-msgs-non-ios (accessed 18 March 2014).

Crosby, S.A. and Wallach, D.S. (2003) 'Denial of service via algorithmic complexity attacks', *Proceedings of the 12th Conference on USENIX Security Symposium, SSYM'03*, Vol. 12, p.3, USENIX Association Berkeley, CA, USA.

Das, D., Sharma, U. and Bhattacharyya, D.K. (2011) 'Detection of http flooding attacks in multiple scenarios', *Proceedings of the 2011 International Conference on Communication, Computing &#38; Security, ICCCS'11*, pp.517–522, ACM, New York, NY, USA.

Defence.net (2013) *DDoS Attack Timeline: The History & Changing Nature of DDoS Attacks* [online] http://www.defense.net/index.php/ddos-in-depth/ddos-timeline/ (accessed 21 July 2014).

Durcekova, V., Schwartz, L. and Shahmehri, N. (2012) 'Sophisticated denial of service attacks aimed at application layer', *Proceedings of the 9th International Conference of ELEKTRO (ELEKTRO 2012)*, pp.55–60.

Falkenberg, A., Mainka, C., Somorovsky, J. and Schwenk, J. (2013) 'A new approach towards dos penetration testing on web services', *Web Services (ICWS), 2013 IEEE 20th International Conference on*, pp.491–498.

Gonzalez, H., Gosselin-Lavigne, M-A., Stakhanova, N. and Ghorbani, A. (2014) 'The impact of application layer denial of service attacks', in Issac, B. and Israr, N. (Eds.): *Case Studies in Secure Computing – Achievements and Trends*, CRC Press, Taylor and Francis.

Hussain, I., Djahel, S., Geneiatakis, D. and Nait-Abdesselam, F. (2013) 'A lightweight countermeasure to cope with flooding attacks against session initiation protocol', *Wireless and Mobile Networking Conference (WMNC), 2013 6th Joint IFIP*, pp.1–5.

IBM (2013a) *Vulnerability IBM Java XML Parser* [online] http://www-01.ibm.com/support/docview.wss?uid=isg3T1019958 (accessed 29 June 2014).

IBM (2013b) *XML 4J Denial of Service Attack* [online] http://www-01.ibm.com/support/docview.wss?uid=swg21647217 (accessed 1 July 2014).

Insecurity Research (2012) *Denial of Service: An Investigation into 'Nuclear DDoSer'* [online] http://insecurety.net/?p=10 (accessed 28 August 2014).

Jensen, M., Gruschka, N., Herkenhoner, R. and Luttenberger, N. (2007) 'SOA and web services: new technologies, new standards-new attacks', *Web Services, 2007: ECOWS'07: Fifth European Conference on*, pp.35–44, IEEE.

Karthigeyan, A., Andavar, C. and Ramya, A.J. (2012) 'Adaptable practices for curbing XDoS attacks', *International Journal of Scientific and Engineering Research*, Vol. 3, pp.I073–I078.

Kirrage, J., Rathnayake, A. and Thielecke, H. (2013) 'Static analysis for regular expression denial-of-service attacks', in Lopez, J., Huang, X. and Sandhu, R. (Eds.): *Network and System Security, Lecture Notes in Computer Science*, Vol. 7873, pp.135–148, Springer, Berlin Heidelberg.

Maciá-Fernández, G., Díaz-Verdejo, J.E. and García-Teodoro, P. (2006) 'Assessment of a vulnerability in iterative servers enabling low-rate dos attacks', *Proceedings of the 11th European conference on Research in Computer Security, ESORICS'06*, pp.512–526, Springer-Verlag, Berlin, Heidelberg.

Maciá-Fernández, G., Díaz-Verdejo, J.E., García-Teodoro, P. and de Toro-Negro, F. (2008) 'LoRDAS: a low-rate dos attack against application servers', *Proceedings of the Second International Conference on Critical Information Infrastructures Security, CRITIS'07*, pp.197–209, Springer-Verlag, Berlin, Heidelberg.

Mirkovic, J. and Reiher, P. (2004) 'A taxonomy of DDoS attack and DDoS defense mechanisms', *ACM SIGCOMM Computer Communication*, Vol. 34, pp.39–54.

OpenSecurity (2012) *Hash Algorithm Collision DoS Attack with Xenotix Hashdos Tester.*

Ormazabal, G., Nagpal, S., Yardeni, E. and Schulzrinne, H. (2008) 'Secure SIP: a scalable prevention mechanism for DoS attacks on SIP based VoIP systems', *Proceedings of the 2nd International Conference on Principles, Systems and Applications of IP Telecommunications (IPTComm)*, pp.107–132.

Padmanabhuni, S., Singh, V., Kumar, K.M.S. and Chatterjee, A. (2006) 'Preventing service oriented denial of service (presodos): a proposed approach', *2013 IEEE 20th International Conference on Web Services*, pp.577–584.

Peng, T., Leckie, C. and Ramamohanarao, K. (2007) 'Survey of network-based defense mechanisms countering the DoS and DDoS problems', *ACM Computing Surveys*, April, Vol. 39, No. 1, p.3.

Prince, M. (2013) 'The DDoS that knocked Spamhaus offline (and how we mitigated it)' [online] http://blog.cloudflare.com/the-ddos-that-knocked-spamhaus-offline-and-ho (accessed 22 July 2014).

Prolexic (2013) *Quarterly Global DDoS Attack Report* [online] http://www.prolexic.com/knowledge-center-ddos- attack-report-2013-q1.html,Q1.

Rafique, M.Z., Akbar, M.A. and Farooq, M. (2009) 'Evaluating DoS attacks against SIP-based VoIP systems', *Proceedings of the 28th IEEE Conference on Global Telecommunications, GLOBECOM'09*, pp.6130–6135, IEEE Press, Piscataway, NJ, USA.

Ranjan, S. et al. (2006) 'DDoS-resilient scheduling to counter application layer attacks under imperfect detection', *Proceedings of IEEE INFOCOM*, pp.23–29.

Ranjan, S., Swaminathan, R., Uysal, M., Nucci, A. and Knightly, E. (2009) 'DDoS-shield: DDoS-resilient scheduling to counter application layer attacks', *IEEE/ACM Trans. Netw.*, February, Vol. 17, No. 1, pp.26–39.

Raz, R. (2010) 'Universal http DoS – are you dead yet?' [online] http://chaptersinwebsecurity.blogspot.ca/2010/11/universal- http-dos-are-you-dead-yet.html (accessed 11 March 2014).

RSnake (2009) *Slowloris HTTP DoS*.

Securosis (2013) *Database Denial of Service: Attacks* [online] https://securosis.com/blog/database-denial-of- service-the-attacks (accessed 21 March 2014).

Shekyan, S. (2012) 'Are you ready for slow reading?' [online] https://community.qualys.com/blogs/securitylabs/2012/01/05/slow-read (accessed 19 July 2014).

SpiderLabs (2011) *Mitigation of Apache Range Header DoS Attack* [online] http://blog.spiderlabs.com/2011/08/mitigation-of-apache-range-header-dos-attack.html (accessed 19 March 2014).

Sullivan, B. (2009) 'XML denial of service attacks and defenses', *MSDN Magazine*, November.

Symantec (2010) *Wireshark Malformed SNMP V1 Packet Remote Denial of Service Vulnerability* [online] http://www.securityfocus.com/bid/43197 (accessed 11 May 2014.).

Tang, J., Cheng, Y. and Hao, Y. (2012) 'Detection and prevention of sip flooding attacks in voice over IP networks', in Greenberg, A.G. and Sohraby, K. (Eds.): *INFOCOM*, pp.1161–1169, IEEE.

Tang, Y. (2012) 'Countermeasures on application level low-rate denial-of-service attack', *Proceedings of the 14th International Conference on Information and Communications Security, ICICS'12*, pp.70–80, Springer-Verlag, Berlin, Heidelberg.

The Web Application Security Consortium (2000) *XML Attribute Blowup* [online] http://projects.webappsec.org/w/page/ 13247001/XMLAttributeBlowup (accessed 12 June 2014).

Xie, Y. and Yu, S-Z. (2009) 'Monitoring the application-layer DDoS attacks for popular websites', *IEEE/ACM Trans. Netw.*, February, Vol. 17, No. 1, pp.15–25.

Xuan, Y., Shin, I., Thai, M. and Znati, T. (2010) 'Detecting application denial-of-service attacks: a group-testing-based approach', *Parallel and Distributed Systems, IEEE Transactions on*, Vol. 21, No. 8, pp.1203–1216.

Ye, X. (2008) 'Countering DDoS and XDoS attacks against web services', *Embedded and Ubiquitous Computing, 2008: EUC'08: IEEE/IFIP International Conference on*, Vol. 1, pp.346–352, IEEE.

Yu, J., Li, Z., Chen, H. and Chen, X. (2007) 'A detection and offense mechanism to defend against application layer DDoS attacks', *Networking and Services, 2007, ICNS: Third International Conference on*, p.54.

Zargar, S.T., Joshi, J. and Tipper, D. (2013) 'A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks', *Communications Surveys Tutorials*, Vol. 15, No. 4, pp.2046–2069 IEEE.

POST-it Denial of Service Tool 1.1.0 [online] http://packetstormsecurity.com/files/98072/POST-it-Denial-Of-Service-Tool-1.1.0.html (accessed 13 July 2014).

S.A. Summit (2014) *DDoS Attacks in H2 2011*, December [online] http://www.securelist.com/en/analysis/204792221/DDoS_attacks_in_H2_2011 (accessed 29 June 2014).

Dirt Jumper [online] http://www.deependresearch.org/2011/10/dirt-jumper-ddos-bot-new-versions- new.html (accessed 27 August 2014).