COMPUTERS & SECURITY 57 (2016) 67–92

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Evaluation of TFTP DDoS amplification attack

Boris Sieklik, Richard Macfarlane, William J. Buchanan *

The Cyber Academy, Edinburgh Napier University, 10 Colinton Road, Edinburgh EH10 5DT, UK

ARTICLE INFO

Article history:
Received 14 October 2014
Received in revised form 20 July 2015
Accepted 11 September 2015
Available online 30 October 2015

Keywords:
DDoS
Amplification attack
DoS
Network flood
DDoS evaluation

ABSTRACT

Web threats are becoming a major issue for both governments and companies. Generally, web threats increased as much as 600% during last year (WebSense, 2013). This appears to be a significant issue, since many major businesses seem to provide these services. Denial of Service (DoS) attacks are one of the most significant web threats and generally their aim is to waste the resources of the target machine (Mirkovic & Reiher, 2004). Distributed Denial of Service (DDoS) attacks are typically executed from many sources and can result in large traffic flows. During last year 11% of DDoS attacks were over 60 Gbps (Prolexic, 2013a). The DDoS attacks are usually performed from the large botnets, which are networks of remotely controlled computers. There is an increasing effort by governments and companies to shut down the botnets (Dittrich, 2012), which has lead the attackers to look for alternative DDoS attack methods. One of the techniques to which attackers are returning to is DDoS amplification attacks.

Amplification attacks use intermediate devices called amplifiers in order to amplify the attacker's traffic. This work outlines an evaluation tool and evaluates an amplification attack based on the Trivial File Transfer Protocol (TFTP). This attack could have amplification factor of approximately 60, which rates highly alongside other researched amplification attacks. This could be a substantial issue globally, due to the fact this protocol is used in approximately 599,600 publicly open TFTP servers. Mitigation methods to this threat have also been considered and a variety of countermeasures are proposed. Effects of this attack on both amplifier and target were analysed based on the proposed metrics. While it has been reported that the breaching of TFTP would be possible (Schultz, 2013), this paper provides a complete methodology for the setup of the attack, and its verification.

1. Introduction

DoS attacks are aimed at exhausting resources of a target in order to make the service inaccessible for legitimate users (Jin and Yeung, 2004). These attacks could have many forms, but usually they send overwhelming amounts of invalid requests to the target. If there are multiple sources of attack, this is called DDoS (Yau et al., 2005). The strength of DoS and DDoS attacks is that these attacks are easy to perform and could have significantly negative impacts upon the target. Recent growth of hacktivism, from groups such as Anonymous are commonly known to use Denial of Service attacks as a form of protest (Hampson, 2010; James, 2013).

When many traditional attacks are coped with by security architectures, the rise of the power of Reflective DoS is seen to be the next wave of attack-by-proxy methods. Denial of Service Amplification attacks (also called reflection attacks or DRDoS) are specific type of the DoS attacks. These attacks use protocol flaws and other vulnerabilities in order to amplify the amounts of transmitted data against a target system. This could effectively multiply the attacker's traffic flow many times. If this is performed in a distributed manner, it could pose a sig-

* *Corresponding author.* Tel.: +44 01314552759.
E-mail addresses: w.buchanan@napier.ac.uk (W.J. Buchanan), borissieklik@googlemail.com (B. Sieklik), r.macfarlane@napier.ac.uk (R. Macfarlane).
http://dx.doi.org/10.1016/j.cose.2015.09.006
0167-4048/

nificant threat to many services (Kambourakis, Moschos, Geneiatakis, & Gritzalis, 2013).

In contrast to other DDoS attacks, amplification attacks could use legitimate, uninfected machines or intermediate devices, which suffer from protocol or other flaws. These devices are referred to as amplifiers or reflectors (Kumar, 2007). An attacker will spoof the source IP address of a specially crafted packet and send this packet to the amplifier in order to trigger response with increased packet size. When each amplifier sends an enlarged response back to the spoofed source IP address, a denial of service could be created on the target. One of the most frequently used attacks is DNS amplification attack (Peng et al., 2006) in which the specially adjusted packets are sent to the DNS servers. Arukonda and Sinha (2015) outlined that Reflective DoS was first observed in 2000, and, in March 2013, caused an attack on Spamhus, with a peak flow of 300 Gbps, followed by others in 2014, of 421 Gbps. Also in 2014, Network Time Protocol (NTP) was used for a peak attack of 400 Gbps, along with Simple Network Management Protocol (SNMP) used to bring down gaming sites. Czyz et al. (2014) highlights that proper understanding of the threat can suppress attacks, such as in the rise and decline of NTP DDoS attacks.

Despite the fact that new forms of attacks are being developed by hackers, there appears to be lack of academic research regarding novel amplification attacks in this area. Most of the reviewed academic literature is focused on Smurf, TCP–based, NTP or DNS amplification attacks while there is limited information about other types of amplification attacks.

For instance, there are very few research papers which investigate other UDP protocols. Authors in https://www.us-cert.gov/ncas/alerts/TA14-017A, state that there are many possible UDP applications vulnerable to amplification attacks but they do not investigate this further. Hence, this work will focus on exploration of the novel amplification Denial of Service attack using a service not yet explored – TFTP.

A typical application for TFTP servers is to store device images and configuration files, such as for embedded devices including with IP phones and networked devices. With the increase in the usage of embedded systems, and with the possibilities of creating TFTP servers within the Cloud, the threat is likely to increase with the rise of the Internet of Things (IoT).

This paper shows several weaknesses in TFTP that lead to a novel amplification with achieved amplification factor of approximately 60, which rates highly alongside other researched amplification attacks. In addition, as will be shown later, the TFTP itself does not support any authentication methods and anybody can access all files if TFTP port is open. Latest Internet research shows that there are about 599 600 open TFTP servers online so this attack can be a global issue. Hence, to minimise misuse of this attack, several countermeasures were mentioned to prevent this attack to be used maliciously.

2. Literature review

2.1. *Denial of service attacks and distributed denial of service attacks*

Denial of Service attacks are typically defined as an attempt to interrupt the legitimate use of the remote service (Mirkovic and Reiher, 2004). There are variety of techniques on how to perform these attacks; however, most of them involve sending large amounts of packets to the target machine so that its processing and networking capabilities are overflowed and no legitimate users could be served (Kumar, 2007). This will result in an inaccessible service. It has been shown that Denial of Service attacks could be a major reason for financial losses (Arora et al., 2011). On the other hand, some older definitions of DoS attacks appear to be vague. For instance, the DoS attack could be defined as any attack that will result in a target machine being inaccessible for legitimate users (Gligor, 1984). These older definitions do not appear to be suitable in the current environment, where majority of attacks are executed remotely (Kambourakis et al., 2013; Kumar, 2007; Mirkovic et al., 2009). Kührer et al. (2014) highlights that the handshaking process can often be used to amplify an attack, and identified a wide range of TCP-based protocols that could be used as amplifiers, allowing amplification factors of 50 and higher.

Denial of Service attacks can use a variety of methods to make target machine or service inaccessible. The following examples are some of the most common techniques:

- The rapid increase in required processing power or transmitted traffic – There are many services prepared for the sudden and significant increase in processing power. The methods of how to increase the required processing power of the target vary; however, the most common practice is to overflow the server with invalid requests (Bogdanoski et al., 2013), for instance SYN Flood attack (Wang et al., 2002).
- The interruption of the target's communication ability by using protocol flaws – This approach focuses on the exploitation of protocol flaws instead of taking the full processing power of the target. Several machines are using vulnerable or mis-configured protocols, which could be used by an attacker to interrupt or limit the connectivity between the client and the server. An example include Optimistic ACK attacks (Savage et al., 1999).
- The manipulation of the routing path – This method changes the routing path to the target so that the new routing path is invalid. This will result in complete inaccessibility of the target. These attacks are very rare and difficult to execute; however, there are known cases where these attacks have taken place. This includes the attacks made on the key internet routing protocol – BGP (Nordstr and Dovrolis, 2004). On the other hand, these incidents could be caused unintentionally by human error (Mahajan et al., 2002).
- Physical intrusion – The majority of older Denial of Service definitions state that any attack in which the intruder could prevent authorized access to information could be considered as Denial of Service (Saltzer and Schroeder, 1975). Gligor (1984) specifically mentions the physical damage of equipment; however, he did not pursue this matter further. According to these older definitions, it means that, in theory, any physical attacks which will make the service inaccessible to users could be considered as a denial of service. An example could include an intentional electric blackout caused by attackers or physical damage to the equipment. In comparison to current research, it appears that physical intrusion is not considered to be a pressing issue in current DoS taxonomies (Mirkovic and Reiher, 2004).

Lau et al. (2000) stated that the distributed format of these attacks creates a *many-to-one* relationship, which makes DDoS attacks more difficult to protect against. Because of this relationship, the flowing traffic and taken resources are significantly higher in DDoS than in DoS. Therefore, most organizations now consider DDoS to be more a pressing threat than DoS (Mirkovic et al., 2004). In addition to this, several current statistics show that the percentage of the DoS attacks is minimal in comparison to the percentage of DDoS attacks. This paper explores the possibility of decreasing the number of attackers involved in DDoS, while increasing the transmitted traffic by using DDoS amplification attacks. This could discount several methods which are based on mitigation against large amounts of attackers.

The amount of traffic generated by DoS and DDoS attacks could vary depending on the type of attack. In the past years, the maximum attack rate has increased significantly (Arora et al., 2011). There is no single value which guarantees the success of the attack due to the fact that different targets could have different resources available to them. Hence, the particular attack rate might damage one target, but pose no threat to another. Large Distributed Denial of Service has also been noticed to be used as cover-up for sophisticated hacker attacks (Constantin, 2011). Some claim that DDoS is taking as much as 5% of the entire Internet traffic (Tung, 2010). This could result in a bandwidth of terabytes per second at any given time. These large bandwidths can have considerable effects on targeted service. One of the most recent examples of a very large DDoS attacks is when the Czech Republic was attacked by a substantial DDoS attack (The Economist, 2013). It was spread over three days and each day a different economy sector was targeted. Throughout this time, mobile operators, banking and search engines were severely affected. A government IT Security Council meeting also took place, which shows how significant the DDoS threat could be.

It is common that attackers do not have access to high-bandwidth connections. In addition to this, any attack created from this connection would be easy to track. Hence, most of the attacks are performed by the large botnets (Durno, 2011). Botnets could even have millions of machines connected to them (Stone-Gross et al., 2009). Most of the time, users do not know that their computer is being infected and being used for attack purposes.

There was a considerable growth in the number of machines connected to botnets during the past years, which resulted in one of the largest botnets – Zeus (Binsalleeh et al., 2009). However, governments and companies have lately multiplied their efforts to shut down the botnets (Stone-Gross et al., 2009) and have been relatively successful (Dittrich, 2012). These current evolvements are causing attackers to look for new ways to increase the efficiency of the current botnets. One of the most commonly used approaches of how to increase the efficiency of botnets are DDoS amplification attacks.

2.1.1. *ICMP attacks*

Ping flood, also called ICMP flood, is relatively simple attack where an attacker sends large number of ICMP Echo Request messages to the targeted machine (Criscuolo, 2012). The aim of this attack is to overflow victim's buffer or to take up the full bandwidth of a victim, so that no legitimate communication could be performed to and from the victim's machine. Currently, ping floods are not considered to be a very significant threat since many countermeasures have been developed against this type of attack. This includes bandwidth limitation for ICMP connections (Ingle and Awade, 2013), blocking ICMP messages at the edge router (Douligeris and Mitrokotsa, 2003) or turning off ICMP completely (Garg and Reddy, 2004).

Ping of death is a historical Denial of Service concept, which was previously used by attackers to generate non-standard ICMP echo messages (Lippmann et al., 2000; Peng et al., 2006). These messages could result in the failure of remote computers. One example of these messages could include ICMP echo packets longer than 65 536 bytes, which is the maximum allowed size in ICMP specification (Garg, 2011). Other definitions state that ping of death packet is separated into multiple IP fragments, which could cause buffer overflows or operating system crashes (CERT, 1996). Currently, ping of death is not a considerable threat since most of the major operating systems have been patched against this vulnerability. This attack is an example of how simple and effective countermeasures can stop the occurrence of the DoS attack completely. Nevertheless, current attacks are more complex, which requires mitigation techniques to be more complex as well.

The Purpose of the majority of Denial of Service and Distributed Denial of Service attacks is to overflow the target with large amounts of requests. This, however, requires large amount of traffic which is often quite difficult to perform. SSL **Renegotiation Denial of Service** attack is a recent evolvement in the DoS attacks in order to lower the number of connections required for the successful attack (Bhople, 2012). SSL renegotiation DoS attack is not shown in any of the reviewed taxonomies. It uses valid SSL connections to the target and then tries to use the feature of the SSL protocol to rapidly renegotiate the key pairs (Bhople, 2012). Generation of new key pairs requires a large amount of CPU resources, thus a significantly lower amount of connections is necessary for the attacker to flood the target. The original SSL renegotiation attack is intended to be performed by one user, hence it is usually classified as Denial of Service. However, the effectiveness of this attack grows rapidly when more attackers are involved. Because SSL is used by many upper-layer protocols, there are many vulnerable services (Zoller, 2011). Due to this fact, SSL Renegotiation attack is an umbrella term for several attacks used on a variety of services using SSL (Prentow and Krarup, 2009). Its core principle is to trigger key renegotiation between the client and the server. As pointed out by Prentow and Krarup (2009), SSL renegotiation exploits flaw in SSL/TLS itself and its implications depend only on the application built on top of these (Prentow and Krarup, 2009).

2.2. *Examples of DDoS tools*

There are several publicly available tools available to perform Denial of Service attacks. This work will provide brief summary and comparison of the most recent, commonly used DDoS tools. These tools provide easy to use interface and some level of attack automation. This easy to use interface allows less experienced users to perform large and complex DDoS attack

(Hampson, 2010), which makes these tools very dangerous. In addition, many attackers will buy large botnets, which they could infect with the described tools.

Recently, one of the most commonly used tools, Low Ion Orbit Cannon (LOIC), was created and used by the hacker group Anonymous (Busschers, 2010). Alomari et al. (2012) described this tool as a botnet-based DDoS tool, which produces large volumes of HTTP traffic. However, it appears that this description is not completely valid. It has been previously shown, for instance by Montoro (2011) and Pras et al. (2010), that LOIC could be installed on individual machines not necessarily belonging to any botnet. This tool was used in many wide scale attacks such as "Operation Payback" directed against credit card companies (Sauter, 2013). During this attack, a large number of novice users were attracted to use this tool and large financial losses were caused.

Low Orbit Ion Cannon is able to perform Denial of Service attack using TCP, UDP or HTTP. Busschers (2010) performed several experiments with this tool and discovered that some features are badly implemented, which can result in attacks stopping with no apparent reason (Busschers, 2010). This tool is an open-source and many contributions are made regularly; the current source code can be seen in GitHub (2013). The LOIC is also available as a web interface, often called Mobile LOIC. It allows users to visit particular web pages, select a target and perform an attack from a web browser by using JavaScript code (Imperva, 2012). This allows users to perform Denial of Service from any web browser, including a mobile phone with low processing power. The web interfaces of this tool vary. Additionally, LOIC is also available as an open-source mobile application for smart phones, which can further increase the number of attacking devices (Adib, 2013).

High Orbit Ion Cannon (HOIC) is the latest version of previously described Low Orbit Ion Cannon tool created in 2012. They both have very similar principles of functionality although they are directed to different usage scenarios. LOIC is able to perform TCP, UDP or HTTP attacks, whereas HOIC is purely HTTP Denial of Service attack tool (Barnett, 2012). HOIC is a desktop application tool. The biggest difference between LOIC and HOIC is the newly presented "Booster" feature. This is creating a randomization of requests, in which the format of these requests is changed according to the configuration file, in order to make detection of these attacks more difficult (Barnett, 2012).

An SSL renegotiation attack is a recent evolvement in DoS and DDoS attacks. Instead of trying to overflow a target with invalid packets, it tries to consume server resources by using a large number of SSL renegotiations. This attack is quite specific and is targeted towards services, which use the SSL protocol. A command-line called thc-ssl is capable of performing SSL renegotiation DoS attacks. As stated by Bhople (2012), this tool is programmed using non-blocking I/O operations. This allows program functions to be executed independently, which rapidly increases the speed. Thc-ssl is creating TCP connection with the server. This connection uses the standard TCP 3-way (packet) handshake. On the top of this connection, an SSL socket is created and as soon as successful SSL connection is created, tool asks server for renegotiation. After 50 renegotiations thc-ssl sends null data (bytes with value of zero) to the server and the cycle repeats itself. This approach is very effective because only one TCP handshake is necessary to perform many SSL renegotiations.

2.3. *Detection approaches*

Denial of Service attacks use a large variety of different approaches in order to make the target service unavailable. Hence, this variety makes these types of attacks exceptionally hard to detect. This is one of the reasons why DoS and DDoS attacks are very dangerous and their popularity in the hacker community is growing. Before there could be any mitigation processes, it is very important to be able to distinguish between normal flowing traffic and attacks; this process is called detection (Zhang et al., 2012). DoS and DDoS attacks produce traffic which is extremely similar to legitimate traffic. In the case of DDoS attack, the traffic is distributed from a large variety of sources.

2.3.1. *Statistical approaches*

Several approaches are focused on mathematical/statistical nature of Denial of Service attacks. Feinstein et al. (2003) proposed to compute the entropy and frequency distribution of the packets in order to detect Distributed Denial of Service attacks. They suggest to create the "detector" which will be used to calculate the entropy of normal conditions and compare this value to the real-time traffic (Feinstein et al., 2003). In conjunction with entropy, chi-square statistics were also used. It has been observed that during normal conditions, the entropy values for different header fields stay within a narrow range. Feinstein and colleagues claimed that during an attack, these values go over this narrow range and therefore are easy to detect (Feinstein et al., 2003). The main concerns of this approach are the unknown detection rates of the more sophisticated attacks and future tools. Additionally, this entropy calculation appears to be quite computationally expensive.

Jin and Yeung (2004) proposed slightly different statistical approach of detecting DDoS attacks. They introduce multivariate correlation analysis, which is based on the correlation between normal and abnormal patterns (Jin and Yeung, 2004). This approach presents two-variable model, where all TCP control field header values are supplied to the calculation. In the calculations pairs of TCP fields are correlated. It has been observed that this model is very effective in detecting TCP SYN Flood attacks; however, authors claim that it could be used to differentiate between any normal and attack traffic (Jin and Yeung, 2004). Despite this, it appears that this model has not been tested against other types of DoS attacks. This could pose a problem, as Denial of Service attack could vary in several parameters, not only in TCP control fields.

In comparison with the Entropy calculation mentioned above, Covariance Analysis Model has the "advantage of independence from packet distribution assumption" (Jin and Yeung, 2004, p. 4). This means that, in comparison to other models, it does not use uniform distribution of packets. Although this model appears to provide very good statistical analysis, several limitations of this model have been discovered (Jin and Yeung, 2004). Firstly, the length of the detection time interval is still unclear. Secondly, the attackers might alter the content of the packets (e.g. TCP flags) in order to avoid

detection by this model. Lastly, it remains unclear which TCP header pairs are the most suitable as input data for the analysis.

The detection approach proposed by Zhang et al. (2012) is based on analysis of user activity, aggregation and data flows. This model is based on a fact that, within certain time frame, very few users have similar behaviour on the network. This model is using following procedures. Firstly, user data are intercepted before it reaches the protected server and analysed. According to proposed analysis procedures, traffic is then aggregated to different flows. Normal traffic will be aggregated to the *normal flow* and attack traffic will be assigned to *attack flow*. The detection of attack flows comes from the assumption that most commonly used Denial of Service attack tools send the same packets repeatedly and in similar manner. Moreover, it is very difficult for these tools to reliably mimic the behaviour of normal users. As stated by Zhang et al. (2012), only a small number of users will have the same sequence of requests, hence only a few users will be incorrectly allocated to attack flow. This should assure low rates of false positives. The aggregated flows are then evaluated by using D–S evidential theory to get the probability of the attack. Once the probabilities are classified, the data are put into the queue according to their priority, with the lowest priority belonging to the attack flow. This ensures that legitimate data are served by the server with the shortest waiting time. On the other hand, several recent DDoS tools such as HOIC provide some form of randomization of request (Barnett, 2012). Hence when attacker is using this tool, the effectiveness of detection using this approach remains questionable.

Most of the previously described approaches are based on the normal behaviour of the users; therefore, these approaches could be grouped as anomaly-based approaches. Entropy calculation appears to be very similar to Covariance Analysis model, except that its calculation appears to use more resources. Aggregation and Evaluation of User data is more proactive and is placed in front of the protected server, whereas other described approaches are running on the server itself. Nevertheless, these anomaly based approaches can cause small percentage of legitimate traffic to be dropped, which creates several business issues. Hence, it is often seen that signature based approaches are used instead, even though their ability to detect new attacks is very limited (Lan et al., 2003).

Adaptive Selective verification (ASV) is a recently proposed distributed mechanism which attempts to mitigate the DoS and DDoS attacks by using selective verification (Khanna et al., 2012). It operates by introducing the new protocol for both clients and server. This protocol uses the bandwidth as an "exchange currency". In currency-based methods, the server under attack requests some currency from the connecting clients, in this case bandwidth (Alturki et al., 2008). Clients are then encouraged to use more bandwidth on a server, usually by sending "dummy" bytes on a separate channel (Walfish et al., 2013). The key concept is that an attacker is trying to the attack server with maximum possible bandwidth, whereas normal clients use only fraction of their available bandwidth. When the server is under attack, it will request more bandwidth from all clients. As normal users will have much more available bandwidth in their connection than the attackers, they should be able to win priority access to the server. This process is sometimes referred to as auction (Walfish et al., 2013). At the same time, the server implements the buffer, where a sampling algorithm is applied to incoming requests in order to verify them (Alturki et al., 2008). This buffer ensures that server load is constant regardless of the flow of traffic. Moreover, this approach is adaptive, hence the bandwidth required for auction changes according to the severity of attack. Nevertheless, it is possible for attackers to use less than maximum bandwidth in order to avoid detection by this mechanism and to win used auctions, but this approach would limit the attackers' ability to produce large amounts of traffic and decrease the chances of successful attacks.

Many of the DoS attacks are performed by spoofing the source IP address of the machine and then sending large amount of traffic from this spoofed IP addresses to the target. This makes it very difficult to track the true origin of the attack. There are several approaches which suggest possible ways of detecting the spoofed IP addresses, for instance the hop count filtering (Jin et al., 2003). This approach uses the hop count of the packets in order to detect spoofed IP addresses. It is based on the fact that attackers could change any value in the TCP/IP headers; however, they cannot typically forge the number of hop counts of the incoming packet. The initial TTL value of the packet is compared with the TTL value of the packet at destination; this will provide the hop-count value of the packet. Hop-count filtering builds IP to hop-count table using minimum amount of resources (Jin et al., 2003). There are two states of this filter: alert and action.

During the alert state, the filtering mechanism detects and marks spoofed IP packets. The action state is activated when a DDoS attack is detected and all spoofed IP packets are dropped. Jin et al. (2003) claim that this detection and filtering approach could detect and filter 90% of spoofed IP packets. A major disadvantage of this approach is the relatively high inaccuracy rates that occur when clients use the Network Address Translation (Wang et al., 2007). This could be a large issue, especially in the current environment where NAT is widely used.

There are many more technology detection approaches, such as those focused purely on ISPs (Akella et al., 2003) or in-depth traffic classification approaches (Nguyen and Choi, 2010); however, their description is outside the scope of this work.

2.4. *Mitigation approaches*

There are many approaches in both detection and mitigation techniques and these can vary significantly depending on protocol, layer or approach. However, similarly to detection approaches, it appears that there is no single approach that could be used to mitigate all types of Dos and DDoS attacks (Lee, 2013).

2.4.1. *Load balancing*

One of the easiest DoS mitigation technique is **Load Balancing**. This method increases the available connection and the performance of the protected machine to the maximum economic availability. Moreover, it also uses several intermediate devices connected in parallel in order to achieve these goals. If one device is overloaded, the other one acquires its role (Garg, 2011). This solution is quite effective and is used by large

companies such as Google and Microsoft. However, load balancing could include large financial expenses which only few businesses could afford in the large extent (Peng et al., 2006). Moreover, large-scale DDoS attacks are still effective against this approach.

2.4.2. *Bottleneck resource management*

The aim of the Denial of Service attacks is typically to find and overflow the bottleneck of the target. The approaches aimed on protection of these bottlenecks are called Bottleneck Resource Management approaches. One example of the bottleneck management is SYNkill (Schuba et al., 1996). This method injects TCP RST packets to any suspicious connections, which allows server to release the allocated resources during the TCP SYN Flood attack. One of the major drawbacks of this is again that detection algorithm needs to be very accurate in order to differentiate legitimate and attack traffic. Additionally, when the attack is sufficiently large, the TCP RST connection could also overflow the protected service.

Another approach is proposing to create history-based filtering in which, during the attack, resources are allocated only to the users who previously accessed the resource (Peng et al., 2003). In addition to these, there could be traffic classification approaches, such as class-based queuing described by Lau et al. (2000). These approaches could classify the incoming traffic and allow access to only certain predefined resources according to the type of traffic. According to Peng et al. (2006), all of these approaches are used within the commercial products aimed at DDoS protection. As Peng et al. (2006) also stated, their effectiveness is questionable due to the fact that the devices themselves could be attacked instead of directly attacking the target. When these devices are inaccessible, the target is inaccessible too, which effectively produces a denial of service.

2.4.3. *Attack source identification*

One of the commonly proposed methods on how to stop attackers performing DoS and DDoS attacks is to actively identify the source of these attacks, also called the trace-back method (Peng et al., 2006). This method is more focused on the incident response and is considered to be more aggressive. Peng et al. (2006) stated that there are three main approaches to traceback: Active interaction, Probabilistic packet matching schemes and Hash-based schemes (Peng et al., 2006). In the active interaction, network administrator of the victim server tries to determine the source of the attack and shut it down (Khanna et al., 2012). Authorities or ISPs could be used in order to track and filter packets coming from the attackers. Probabilistic packet matching is proposing many techniques, which could calculate the probable path of the packet, as described by Park and Lee (2000). Hash-based approach is based on a new scheme, where routers store the recent packets they transmitted in a form of cache. This cache could be accessed by the hosts under attack in order to determine the source of an attack (Snoeren et al., 2001).

All of the previously described processes are trying to actively search and shut down the attackers machine or route path. This could be a very effective approach, but it faces several key limitations, such as spoofed IP addresses, legal issues, botnets, cooperation of ISPs (Peng et al., 2006). As it can be seen, there is no single approach that would be effective against all types of Denial of Service attacks. Instead, combination of several methods is used in order to provide maximum possible protection. Several of mitigation approaches rely on accurate detection techniques, which often do not provide required accuracy. Evaluation of these countermeasures is directly related to the evaluation of DoS and DDoS attacks.

2.5. *Evaluation of denial of service attacks*

Denial of Service can be evaluated in several different ways. As stated by Mirkovic et al. (2009), all the metrics used in measurement must be clearly quantified (Mirkovic et al., 2009). Commonly used metrics are focused on how the particular attack will affect the normal operations of the target machine (Nguyen and Choi, 2010; Wang et al., 2007). However, a major problem when evaluating Denial of Service attacks is the lack of commonly accepted framework or set of metrics. This could be a significant problem as it is often very difficult to compare results from one study to another (Mirkovic et al., 2006).

Research has found that the used metrics are often "not specifically DDoS-centric; rather, they are straightforward applications of well–known metrics used by researchers and practitioners in networking, performance, and quality of service evaluations" (Hussain et al., 2006). This means that many of the used metrics are not directly designed for the Denial of Service attacks. Despite these issues, several metrics have been repeatedly used in the reviewed literature. Research by Hussain et al. (2006) proposes two types of metrics – extrinsic and intrinsic (Hussain et al., 2006). Extrinsic metrics can be observed and computed by the external parties, such as latency to the target. Whereas intrinsic metrics could be only observed on the attacked machine, for instance CPU usage or RAM queues.

There are several metrics which are used to examine the effect of the particular Denial of Service attack to the target, for example request/response delay of the involved machines (Schuba et al., 1996), packet drop rate, packet losses, packet header overhead. Internal consumption of computer resources such as CPU or RAM consumption is often widely used (Kambourakis et al., 2008). Additional examples of intrinsic metrics include network buffer sizes and HDD read/write requests. Further important metric which is widely used when evaluating Denial of Service attack is Total Throughput (Peng et al., 2003; Savage et al., 1999). Total throughput represents the number of bytes successfully delivered during the certain time frame (Hussain et al., 2006). This intrinsic metric appears to be one most popularly used metric, due to the fact that the main aim of majority of Denial of Service attacks is to overflow target's bandwidth or other resources. Hence measuring the total amount of transmitted traffic on the target can determine the severity of the attack.

2.5.1. *Amplification factor*

Attack Amplification Factor (AAF) can be generally used to measure the amount of bandwidth generated by the amplifying systems and the amount of the bandwidth used initially by the attacker (Kumar, 2007). It measures the effectiveness of the amplification attack and it is also a widely used metric for evaluation of Denial of Service attacks (Abramov and Herzberg, 2013; Kambourakis et al., 2008). In general, any Denial of Service

attack which achieves amplification factor greater than 1 could be considered to be an amplification attack.

However, due to the lack of the common framework, it appears that several of the reviewed research papers create different definitions on how to measure this factor. Many papers adjust equations to better reflect the particular attack (Kumar, 2007) or using high-level definitions (Kambourakis et al., 2008). Others use straightforward, logical, definitions where the amplification attack is simply defined as the ratio of data received on the target and data sent by attacker, as demonstrated by Cowperthwaite and Somayaji (2010). In addition to this, many papers do not define amplification factor, but they still measure similar values, for example Abramov and Herzberg (2013) and Deshpande et al. (2002). This lack of uniform definition of amplification factor raises several issues when evaluating and comparing different amplification attacks. Nevertheless, amplification factor is a very important measurement. The bigger this factor is, the more bandwidth and resources the attack consumes (Kambourakis et al., 2013). Since the aim of most Denial of Service attacks is to consume resources of the target, this is an important measurement value of any amplification attack. There are many ways on how to measure amplification factor.

The equation presented by Kumar (2007) appears to be mostly designed for the ICMP amplification attacks, which could bring several challenges when trying to apply this to different amplification attacks. For instance, this equation has three main factors: Q, N and B_p. Q represents the number of broadcast domains used for amplification, N represents number of hosts in broadcast domain and B_p represents initial bandwidth used by attacker. This is shown in Equation (1). However, not all amplification attacks will use whole broadcast domains and not all hosts in broadcast domain will respond. Another key drawback of this equation is that it does not account for any retransmissions made by computers or packet losses. If attacker is able to generate one packet which will make the server perform several retransmissions, these will not be included in the equation's results.

$$T_a = (M1 + M2) \times \sum_{j=1}^{Q} \frac{N \times B_p}{M1 + M2} = Q \times N \times B_p$$

Equation (1) – Kumar's Amplification Attack Factor Equation

The equation presented by Kambourakis et al. (2008) appears to be more general and could be applied to a larger variety of amplification attack. This equation is presented as a part of DNS amplification attack evaluation. However its advantage is that it could be applied to almost any amplification attack. The format of the equation can be seen in Equation (2). On the other hand, this equation does not contain any variable for retransmission cases or packet losses. This same equation is used in most of the reviewed literature; for instance, in Geva et al. (2013). The equation has the following format:

$$Amplication\ Factor\ (A) = \frac{Size\ of\ Response}{Size\ of\ Request}$$

Equation (2) – Kambourakis et al.'s Amplification Attack Factor Equation

Amplification factors could be measured in very different ways, hence its comparison could be difficult. Moreover, there are several issues present in previously described equations. For instance, Geva et al. (2013) stated that with high-transmission connections, the packet loss could be a major issue to the attacker. None of the reviewed equations had account for this issue.

Due to the limitations discovered in the found amplification factor equations, this work investigated the new equation based on the Kambourakis et al. (2008) approach. There will need to be changes in this equation in order to accommodate for retransmission cases (assuming that retransmitted response has the same size as original response), which can be seen in Equation (3). Additionally, to achieve the maximum accuracy, packet loss will also be taken into account in the proposed equation, which is visible on the right hand of Equation (3) below. The suggested equation has the following format:

$$Amplication\ Factor\ (AAF) = \frac{\sum size\ of\ all\ responses}{size\ of\ request} \times \left(1 - \frac{Packet\ Loss\ in\ \%}{100}\right)$$

Equation (3) – Proposed Amplification Attack Factor Equation

Taking into account that most of the reviewed literature uses the simple equation introduced by Kambourakis et al. (2008), this work will use this equation as the main metric when measuring amplification factors. Additionally, newly proposed equation will also be present in order to assess its accuracy. All of the previously mentioned equations are the important metric when evaluating amplification attacks.

2.6. *Amplification attacks*

Denial of Service Amplification Attacks (also called reflection attacks or DRDoS) are specific types of DoS and DDoS, where the attacker is using intermediate devices to amplify his traffic (Sairam et al., 2010). Geva et al. (2013) differentiates between the amplification and reflection attacks (Geva et al., 2013). They state that the reflection attacks are simpler whereas the amplification attacks could cause a sender or receiver to retransmit packets multiple times, making them more complex. It is believed that this distinction could cause confusion due to the fact that most of the reviewed literature does not refer to these as separate types of attacks. Hence, to avoid confusion this work will consider amplification and reflection attacks to be synonymous.

The devices used to multiply the transmitted traffic are called amplifiers, agents or reflectors (Kumar, 2007). These devices are usually legitimate hosts which are unaware of being an amplifier. In comparison to botnets, amplifiers are usually uninfected and uncontrolled; however, the attacker uses protocol flaws and other vulnerabilities in order to make these devices multiply the traffic. In amplification attacks the *request–reply* relationship is a key principle (Tsunoda et al., 2008). When attacker sends *request* to the reflector, the reflector will send back a *reply*. The attacker is sending packets to the amplifiers with the spoofed IP address of the victim. Amplifiers then increase the amounts of packets and/or packet size and send the reply to the spoofed IP address. Because of a spoofed IP address,

all of this amplified traffic is directed towards the target, which could result in denial of service.

This brings several difficulties to the traditional countermeasures. Firstly, the origin of the attack is very difficult to trace, due to the fact that essentially it is the amplifiers who unwillingly execute the attack. Hence, most of the traditional methods could indicate that amplifiers are the source of the attack, which is not correct (Tsunoda et al., 2008). Moreover, the attackers with small bandwidth could effectively multiply their original traffic many times. Amplification attacks are usually used within two possible scenarios. Either the attacker uses his own computer to multiply his/her traffic or the intruder uses a botnet to amplify the traffic of all botnet machines and direct it towards the target. The latter could lead to a very large traffic flows.

2.6.1. *ICMP (Smurf attack)*

The Smurf attack was the first publicly known DDoS amplification attack (CERT, 1998), and is considered to be an important evolvement in the Denial of Service attacks (Fig. 1). Current taxonomies clearly define this attack (Mirkovic and Reiher, 2004). Smurf is a network layer attack and it uses the resources of other computers in order to amplify the attacker bandwidth. Smurf attack creates an ICMP Echo request packet with the spoofed source IP address of the victim and this packet is then sent to the *directed-broadcast* address of the amplifying network(s) (Kumar, 2007). Directed-broadcast is a specific network prefix (address) where public computers could send messages to the private broadcast domain (Baker, 1995). The packet is then broadcasted throughout the amplifying network(s). All the receiving devices examine the packet and send all the responses to the source IP address of the initial packet, which is the victim IP. Therefore, a single adjusted ICMP request could create large amounts of traffic from multiple computers.

2.6.2. *DNS amplification attack*

Domain Name Service amplification attack is one of the most popular amplification attacks. In fact, most of the reviewed research in the area of amplification attacks is focused on this attack (Deshpande et al., 2002; Kambourakis et al., 2008; Ye and Ye, 2013). Nevertheless, due to the fact that this attack has started to be used widely only recently, the reviewed taxonomy does not show this attack (Mirkovic and Reiher, 2004). Domain Name Service (DNS) is a key, hierarchical UDP-based system used for name resolution in the internet. DNS DDoS amplification attack is an application layer attack which uses widely available DNS servers to amplify the attacking traffic (Geva et al., 2013). The attacker creates a number of DNS queries with spoofed source IP address which are directed towards DNS servers. Impact on these amplifying servers is minimal, whereas traffic directed to the target could cause much more damage (Ye and Ye, 2013).

The DNS queries could be approximately 40-byte in size; however, the response could have 4000 bytes and more (Deshpande et al., 2002). This establishes an attack amplification factor of 100. Thus, if attacker is using a botnet of 100 machines each with 1 Mbps connection, the total traffic after amplification will be 10 Gbps. This is the process of DNS amplification (Fig. 2). Frighteningly, several botnets with capacity of hundreds of thousands of machines were discovered (Geva

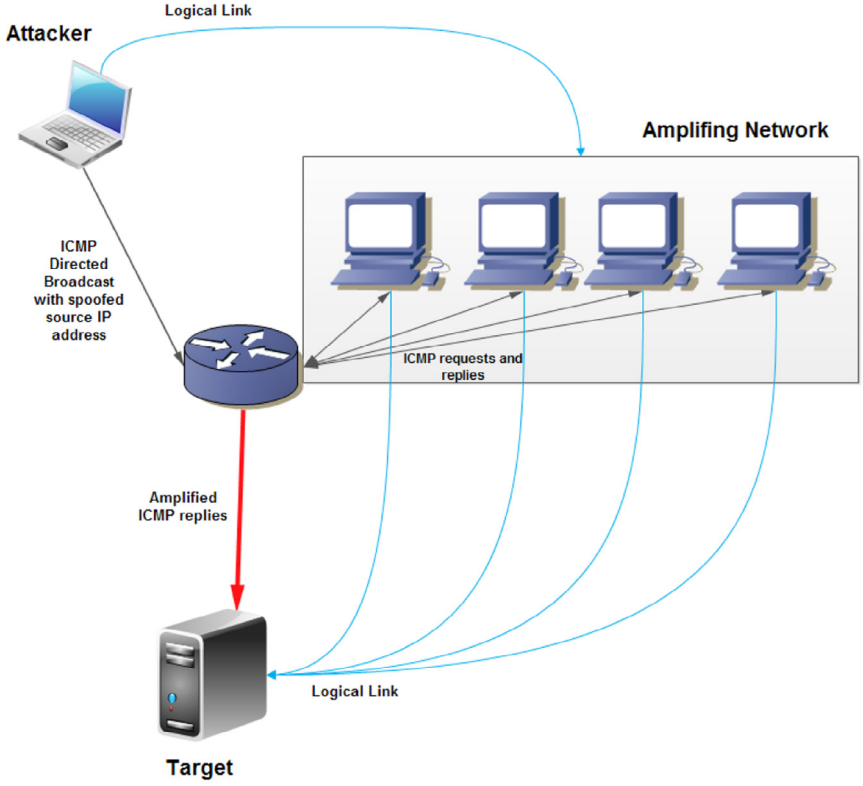

Fig. 1 – Smurf Attack.

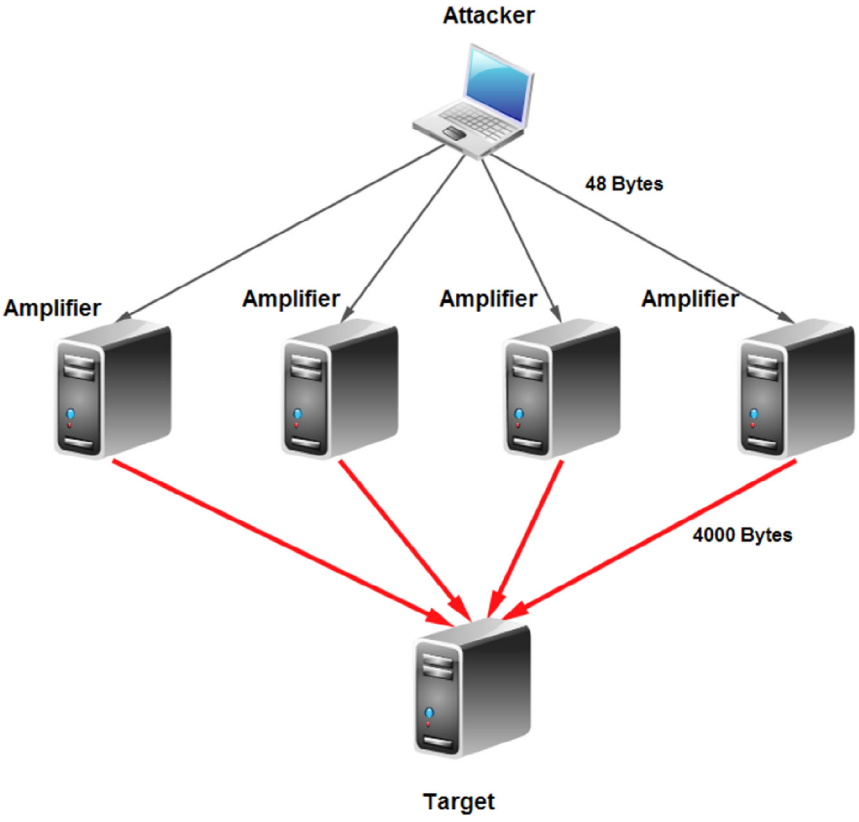

Fig. 2 – DNS Amplification Attack.

et al., 2013). This shows that DNS amplification attacks could be a very big issue in today's environment, especially when performed from large botnets. Many of the recent attacks were using DNS amplification, for instance attacks on the alternative DNS providers such as DNSimple or easyDNS (Constantin, 2013).

The key assumption for the server to work as a DNS amplifier is to allow recursive queries (Kambourakis et al., 2013). Due to the nature of DNS protocol, there are many servers that allow this; for instance in 2005 there were 7.5 million vulnerable servers (Ye and Ye, 2013). MacFarland et al. (2015) outline that methods for restricting DNS attacks focus on upstream ISP, and made queries to DNS servers for 129 million DNS domains which are registered in nice top-level domains (TLDs) and then characterised the requests to create an attack of 444 Mbytes/sec for a sending rate of 44 Mbytes/sec. In their research, they proposed a novel Query Rate-Limiting parameter to limit the attacks.

One of the key prevention techniques against these attacks is spoof detection (Guo et al., 2006). If the amplifying device is able to detect that the source IP address of the request is spoofed, it would be very easy to filter and ignore these requests. A mechanism proposed by Ye and Ye (2013) suggests one-to-one mapping between the DNS requests and responses. Device providing this method would be placed in front of the local DNS server and match each reply to the previous request. If there is any generated reply for which no valid request was received, this could be blocked.

The next generation of Domain Name Service – DNSSEC – is designed to improve authentication and data integrity. However, in the current standard, there is no protection against DNS amplification attacks (Ye and Ye, 2013). Moreover, DNSSEC is criticised to increase the probability of these attacks (Cowperthwaite and Somayaji, 2010). In order to mitigate the risk of these attacks, ICANN, the internet's main domain provider, proposes three high-level recommendations (ICANN, 2006).

Firstly, some mechanism to identify spoofed source addresses should be present. Secondly, clear mandatory procedures should be defined for each DNS root operator. In addition, any blocked IP address must demonstrate that its infrastructure has been protected against this type of attack in the future in order to be reconnected to ISP. Thirdly, open recursion should be disabled and DNS queries should only be processed form trusted sources. As discussed by Ye and Ye (2013), these measures could be very effective; however, ISPs have low initiative to deploy these due to various reasons (Ye and Ye, 2013). Another countermeasure is to deploy black-hole forwarding for the machine under attack, such that all traffic to the victim is routed to the black-hole which will completely deny access to the target (Geva et al., 2013). This will block all the traffic directed to the victim, which could potentially help the achieve attacker's intention. However, this is only used when network bandwidth providers themselves have a trouble handling the traffic amounts. Black-hole forwarding is faster than ACL processing and more aggressive (Geva et al., 2013).

2.6.3. *SNMP amplification*

Even though SNMP request–reply behaviour has been described previously, it has not been considered a threat in relation to amplification attacks (Paxson, 2001). It has been only recently that SNMP amplification attacks started to be used more widely. Due to the recent nature of this attack, minimum academic literature has been found regarding these attacks and they are not mentioned in the reviewed taxonomy (Mirkovic and Reiher, 2004).

SNMP is a UDP-based protocol used to manage network devices (Case et al., 1990). During the SNMP amplification attack, an attacker needs to obtain a list of vulnerable SNMP hosts (Prolexic, 2013b). This could be performed by network scanning or other techniques. The attacker will send an SNMP request, for instance GetBulkRequest, to the host with a spoofed source IP address of a target. The reply from the host will be larger and directed towards that target. The amplification factor of this attack is approximately 3; however, it could be extended to approximately 7.44 (Prolexic, 2013b).

2.6.4. *TCP-based amplification attacks*

There are several amplification attacks focused on TCP vulnerabilities. In Optimistic ACK flood, attacker is rapidly acknowledging the data, even if these have not yet arrived on the target (Savage et al., 1999). This leads to a rapid increase in the TCP congestion window. This could damage data integrity; however, if an attacker does not care about data losses, it could effectively multiply the used bandwidth many times. In Duplicate ACK floods, the attacker's aim is to both increase the congestion window and also to retransmit the current segment, even if this segment was safely delivered (Savage et al., 1999). During this attack, the intruder is transmitting many duplicate Acknowledgement packets, which is interpreted by the server side as a need to increase the congestion window and to retransmit the current segment.

This causes the server to generate new traffic and to increase the allocated bandwidth to the attacker (Kumar and Sisalem, 2004). It has been shown that both Optimistic ACK and Duplicate ACK floods could be used towards multiple targets at the same time (Sherwood et al., 2005). During these attacks, zombie/botnet computers are used in order to initiate seemingly legitimate connections to the server and then rapidly increase the congestion window and consequently used bandwidth. This approach could also be distributed which could lead to very large amplification effects (Sherwood et al., 2005). Both Optimistic ACK and Duplicate ACK attacks are not shown in the reviewed DoS taxonomy.

The Optimistic ACK and Duplicate ACK attacks are both based on the TCP algorithm and are relatively well described. Therefore, these attacks could not be applied to any different protocols and their testing is outside the scope and purpose of this work. In addition, these attacks do not appear to be used widely (Prolexic, 2013a).

2.7. *Requirements of novel DoS amplification attack*

The majority of literature is focused on Smurf, TCP-based or DNS amplification attacks, apart from newer work which highlights the risks of application attacks in the Cloud (Kuker). Tsunoda et al. (2008) stated that there are many possible UDP applications vulnerable to amplification attacks but they do not investigate this further. It is also discussed that almost any protocol could be vulnerable to these attacks if it uses a *request–reply* relationship, as described above. However, the reply is protocol-specific, hence not every protocol will provide amplification. Moreover, some applications may produce very different responses to different requests; some could have many requests and one reply.

UDP itself does not implement any response mechanisms; these are built into higher-level protocols which use UDP (Paxson, 2001). Paxson (2001) shows a table (Fig. 3) with the available amplification attacks in which he states that "Other UPD applications" have unknown amplification factors. This indicates that there are more vulnerable UDP services.

Prolexic discovered several new services susceptible to amplification attacks – NTP, SMTP and CHARGEN (Prolexic, 2013b). There are several indications of new attack in the services using UDP protocol. User Datagram Protocol (UDP) is the most widely connectionless protocol, hence this section will explore possible new vulnerabilities in services using UDP. Connectionless nature of UDP protocol makes amplification attacks easier to perform. There are thousands of UDP ports (IANA, 2013); hence the scope of this work cannot closely explore each and every available service. Therefore, this work will focus on the most

ICMP request/reply	Likely not difficult to filter out. Includes *smurf* attacks.
ICMP problem	Likely not difficult to filter out.
TCP source port	If filtered, no general access to remote server of given type.
TCP SYN ACK	If filtered, no general access to remote servers.
TCP RST	If filtered, state will accumulate over time.
TCP guessable seq. no.	**Major threat.**
T/TCP	Would be significant threat but easily filtered due to limited deployment.
UDP	No threat due to no inherent reply mechanism.
UDP length	Insignificant.
UDP checksum	Insignificant.
DNS query/response	Can be filtered by opening up holes to specific remote servers.
Recursive DNS queries	**Major threat to name servers.**
SNMP request/response	Generally can be filtered out with little impact on victim.
HTTP proxy caches	A significant threat, but likely easily traced back to slave.
Gnutella "push"	**Major threat.**
Other TCP applications	Will in general be traceable to slave if application server keeps logs.
Other UDP applications	*Unknown*.
Other overlay networks	*Unknown*.

Fig. 3 – Paxson's (2001) Table of Amplification Attacks.

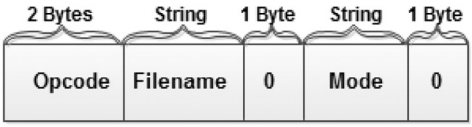

Fig. 4 – RRQ/WRQ Packet.

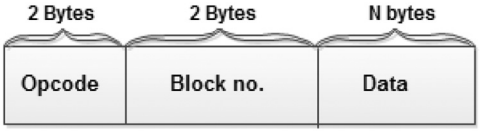

Fig. 5 – Data Packet.

commonly used services which indicate they may be vulnerable to these types of attacks.

From the current RFC review, it appears that several new services could be vulnerable to amplification attacks. The most important requirement is that the vulnerable protocol needs to be stateless and predictably react to a particular *request* with larger *reply*.

2.8. *The TFTP protocol*

Trivial File Transfer Protocol (TFTP) is a simple protocol, which allows the downloading or uploading of files (Sollins, 1992). In comparison to quite robust File Transfer Protocol (FTP), TFTP provides a very simple implementation. It is generally used on top of UDP, so that it can take advantage of the fast transmission rates and the stateless operation; however, as stated in RFC (Sollins, 1992), TFTP is capable of using almost any transport protocol.

Trivial File Transfer Protocol provides very basic functionality, which is limited to two different types of transmissions binary or ASCII. The TFTP service is available on the well-known UDP port of 69. Unlike FTP, TFTP does not contain any method to display the content of a directory. There are several vulnerabilities known in TFTP. Firstly, buffer overflow could occur when attacker is trying to read/write very long names from/to the server. Secondly, TFTP is also susceptible to format string vulnerabilities (Liu and Zhang, 2008). In this vulnerability, attacker is sending the special string as a file name, which could execute the harmful code or display the protected values such as memory stack.

In the past, TFTP services were prone to denial of service vulnerability (Singh et al., 2008), when an attacker tried to read/write the special file name. For instance, on the Windows machines names such as CON or COM1 could cause so-called "blue screen of death", resulting in a denial of service for valid users. Currently, servers are patched against these vulnerabilities; however, due to the restrictions in the standard, there is very little additional security that could be provided for TFTP services. There are five types of packets present in TFTP (Sollins, 1992), and these are described in the following sections.

2.8.1. *Read request*

Read Request (RRQ) packets are used by a sender to download the file placed on the server. The format of the Read Request can be seen in Fig. 4. First two bytes are allocated for the Opcode. A value of 1 (01 h or 0000000000000001 respectively) determines that this packet is a Read Request.

The file name is a sequence of ASCII (netascii) characters. The size of a file name field is not defined in a standard.

The zero byte is serving as delimiter to determine the end of a file name.

Mode field is used to determine the current mode of a protocol. A protocol has three modes: *netascii*, *octet* and *mail*. These modes are transmitted in netascii format and could be sent in any combination of upper or lower case. The modes determine the format in which the files are stored. Currently, it is assumed that most of the file servers would use netascii due to the best compatibility. The zero byte is again serving as a delimiter to distinguish the end of the mode field.

2.8.2. *Write request*

Write Request (WRQ) packets are used by sender to upload (write) the file placed on the server. The format of the Write Request can be seen in Fig. 4 above. Write Request is very similar to previously described Read Request. Only difference is that the Opcode value is set to 2 (02 h or 0000000000000002 respectively) instead of 1.

2.8.3. *Data packet*

Data packets are used to transmit the data across the network. Their format can be seen in Fig. 5. The Opcode value in data packet is a value of 3 (3 h or 0000000000000003 respectively). This uniquely identifies the packet as a data packet. The block number identifies the currently transmitted segment of data. The maximum block number is not defined in the standard, which in theory allows the transfer of unlimited file sizes. However, this is often limited by running TFTP software. Each new block will have an incremented block number by one. For instance, the first block starts with 1, the second block has block number 2 and so on. This allows unique identification of the blocks.

The data field could have values ranging from 0 to 512 bytes. This field is actually transmitting the required data. If its size is 512 bytes, it is defined that this block is not the last block and that new blocks will follow. If the block size is less than 512 bytes, it means that this is the last block and it also signals the end of transfer.

2.8.4. *Acknowledgement packet*

Acknowledgement packets are used for acknowledgement or termination, unless the timeout occurs. The format of an acknowledgement can be seen in Fig. 6. The Opcode value in acknowledgement packets is 4 (4 h or 0000000000000004). The Block value contains the block number of the Data packet which is being acknowledged.

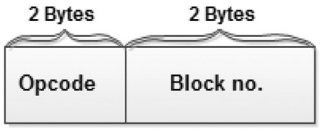

Fig. 6 – Acknowledgment Packet.

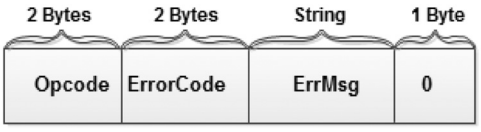

Fig. 7 – Error Packet.

WRQ requests are acknowledged with Block value of 0. Each time a receiver successfully receives data packet, acknowledgement will be sent to the server. After successfully receiving the acknowledgement by the server, a new block of data is sent. If no acknowledgement packet is received, data can be retransmitted; however, amount of retransmissions is not defined in the standard and it purely depends on the running TFTP software. Acknowledgement packets are used as a reply to both Data packets and Write Request packets. RRQ packets are acknowledged either by Data or Error packets.

2.8.5. *Error packet*

Error packets are used to notify the remote party that error has occurred. These packets could serve as a type of negative acknowledgement for any other TFTP packet. The format of Error packet can be seen in Fig. 7. Error packets have Opcode value 5 (5 h or 0000000000000005). ErrorCode value is an integer which specifies the type of an error.

ErrMsg is a sequence of ASCII (netascii) characters. It is used to transmit a human-readable error message. The size of the error message is not defined in standard. The zero byte has a purpose of being delimiter to mark the end of the ErrMsg and packet itself.

2.9. *Novel TFTP amplification attack*

During the research and exploration of RFCs, it was discovered that Trivial File Transfer Protocol has several features which indicate that it could be used for Denial of Service amplification attacks (Sollins, 1992). Several fields in TFTP packets do not have pre-defined sizes. TFTP is a purely stateless protocol with small number of packet formats and clear request–reply behaviour. One of the disadvantages of TFTP is that there is no authentication method. Unless other filtering methods are present, any user is able to connect to the TFTP server and download the data. For instance, the highest-degree of security that the Tftpd32 implementation provides is “read–only” files, which does not prevent attackers reading the content of stored files. In order to mitigate these risks, Unix systems often offer IP filtering or allow the TFTP service to operate only within one directory.

The discovered TFTP attack is using newly discovered TFTP protocol flaw, hence it should be applicable to any TFTP implementation, which is compliant with the TFTP standard. In high level view, the attacker sends a smallest possible request for the file on the TFTP server with a spoofed IP address, which triggers a larger response directed towards the target. This flaw is using specific RRQ TFTP request to generate larger response together with retransmissions and error codes which will be directed towards the target, which significantly amplifies the response, which in turn results in large amplification.

2.9.1. *Format of the attack packet*

The attack will require creation of special adjusted RRQ TFTP request. In order to achieve maximum amplification and best efficiency of an attack, the smallest possible valid RRQ TFTP packet is created. This specially crafted packet has spoofed source IP address. Hence, when amplification server receives this request, it will send all the responses to this target machine. Based on the TFTP standard. The Opcode of this packet is kept the same – 1. The Filename needs to be previously known to the attacker. Because of the fact that amplifying server will send all the responses to the origin IP address and the source port, this attack could be directed towards any UDP service with varying effects on the target machine.

3. Design

DDoS amplification attacks could be a serious threat. Most of the reviewed research in amplification attacks is focused on the traditional protocols such as ICMP (Kumar, 2007), TCP-based attacks (Sherwood et al., 2005) or DNS (Kambourakis et al., 2013). There is very little research focused on exploration of effects of amplification attacks on different protocols. Several indications were found that different protocols could also be vulnerable to these attacks. One candidate for an amplification attack is the Trivial File Transfer Protocol. The aim of the testing is to generate accurate data so that the selected metrics could be measured and compared to other research papers.

3.1. *Testing environment*

There is a lack of common testing and evaluation framework or methodology for DoS techniques (Mirkovic and Reiher, 2004). Despite this fact, there are several similar testing environments in the reviewed literature. The simplest topology used by Lau et al. (2000) consists of several clients connected to the one router and a target connected to the same router, but on different network. Mansfield et al. (2000) used medium size Internet exchange point consisting of three routers with three different networks and one measure point. This approach seems to be very common in literature for measuring amplification attacks, for instance in Peng et al. (2006) and Abramov and Herzberg (2013). Rastegari et al. (2009) used high level description of the proposed system (Rastegari et al., 2009). This approach could be adjusted to better explain the running systems. This is due to the fact that it allows the precise description of measurements tools and their placements. In addition, high-level description of involved system is also provided. Gilad and Herzberg (2012) used the topology of five networks connected to one router (Gilad and Herzberg, 2012).

This variety of testing environments shows that the comparison of the results between different research papers could be difficult. This is caused by different technologies, methodologies and topologies used in different papers. The comparison of different research works is also depending on selected metrics. There are large varieties of intrinsic and extrinsic metrics. In order to provide comparable results, this work will use the most commonly used approach consisting of three routers with three networks as shown in Abramov and Herzberg

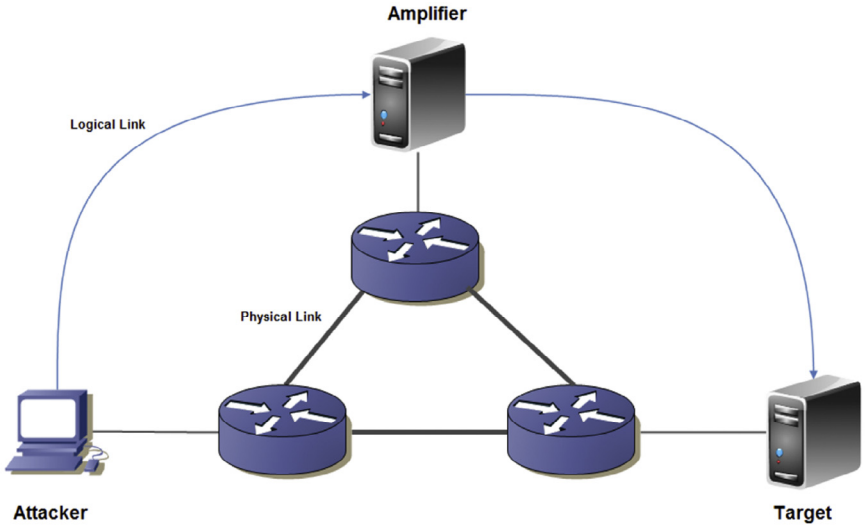

Fig. 8 – Experimental DoS amplification testing topology.

(2013). In addition, to better describe running systems and their components, an approach used by Rastegari et al. (2009) will also be used.

The amplification factor is a very important metric when measuring amplification attacks as it can measure increase in traffic specific for the particular amplification attack (Kambourakis et al., 2008). This appears to be the main metric when comparing amplification attacks, hence this work will focus mainly on measuring amplification factor. The equations used to calculate an amplification factor can vary; however, their measurement methods appear to stay the same.

3.1.1. *Topology*

The created topology consists of three routers, one computer and two servers. The amplification server will run the TFTP service, whereas the target server could run any UDP service. As mentioned previously, this topology was found to be most common among the performed research. The three routers will simulate the Internet connections between involved computers. Routers will be using routing protocol in order to route the traffic. Each router will provide one independent network to which the computer/servers are connected. For the simplicity and accuracy of results, each network will only have one machine connected (Fig. 8). The more precise analysis of all involved systems is present in the following sections below.

3.1.2. *DoS attacking system*

The attacker's system will be composed of Network Interface Card (NIC), operating system and packet-adjusting software. From research it was shown that typically, Windows operating systems block methods of changing/spoofing packet fields (Menzies, 2002), therefore an attacker would usually have to use the other types of operating system. Additionally, special packet adjusting software needs to be run on this machine. Due to the nature of an attack, the attacker will not receive any confirmation of packets which arrived to amplifier, therefore all the communication from the attacker's machine will be one directional (Fig. 9).

3.1.3. *DoS amplifier system*

The amplifying system will be based on the separate and remote network. It will consist of several sub-components. First component is Network Interface Card (NIC), capable of transmitting and receiving traffic. When packet arrives, this network card will send it to the running operating system. After operating system receives this, it will pass the traffic to the running TFTP software. There will be several TFTP servers tested – each implementation could have different effect. These implementations are described in sections below. Once packet is processed by the TFTP server, the response will be passed to the operating system and network interface, which will send the response back according to the source IP address. During these processes, several measurements will be performed on both operating system and Network Interface Card (NIC). These measurements will be semi-automatic and will be active for the precise time. Their results will be stored locally and accessed at the end of testing (Fig. 10).

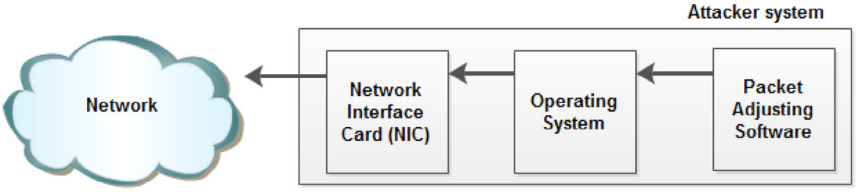

Fig. 9 – Attacking System Components.

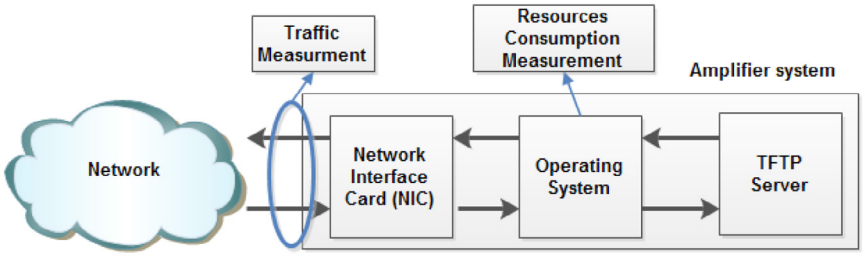

Fig. 10 – Amplifier System Components.

3.1.4. *DoS target system*

The target server will have very similar settings. There will be a Network Interface Card (NIC), which will forward all the received traffic to the operating system. Operating system will then pass this traffic to the appropriate UDP service; several UDP services will be tested. Additionally, it is assumed that at least one UDP service will be running on the target. It could be the TFTP; however, this is not a requirement. Consistently with the amplifier system, during this process several measurements will be taken. The result of these semi-automated measurements will be stored locally (Fig. 11).

3.1.5. *TFTP RRQ format*

This specially crafted packet will have spoofed source IP address. This spoofed IP address will be set to a 192.168.205.128, which is the IP address of a target. Hence, when amplification server receives this request, it will send all the responses to this target server. The special format of this packet could be seen in Fig. 12 and is described in paragraph below.

The Opcode of this packet is kept the same – 1. During performed tests, the file name will be set to "*A.pdf*" value. A.pdf is a PDF version of GNU/GPL licence and has the total size of 33.5 KB. Mode of the transmission will be send to octet.

Because of the fact that amplifying server will send all the responses to the origin IP address and the source port, this attack could be directed towards any UDP service. Hence, by changing the source port of the crafted packet, the attacker could effectively direct the attack towards any open UDP service. There is a large number of UDP services, hence attacks against different UDP services could have different effects on the target. Because the target will be running Windows XP operating system; default windows XP UDP services will be attacked and effect on the target will be observed. Then these results will be compared in order to test the presented theory that attacks directed towards different UDP services could have different effects on resource consumption.

3.2. *Testing procedure*

In the testing scenario, the DoS attack system will try to attack external target by spoofing the source IP address of a packet. This packet will be sent to the TFTP amplifying server, which will send the response to the spoofed IP address of a victim machine on desired port. This port will vary according to tested service – the target machine does not need to run TFTP server. The capturing software is run in all involved parties to record all flowing traffic. Each test will be performed three times to ensure its accuracy. Analysis of this data will mostly show all three performed tests; however, some tests, such as CPU usage, will use averages created from these values.

The testing procedure will be performed from one attacker to one target (Fig. 13), hence it will simulate Denial of Service attack. Simulation of Distributed Denial of Service attacks is outside of the scope of this work; however, it is very probable that Distributed Denial of Service would have larger effect. Another goal is to measure the effects of this attack on a target machine by recording the metrics. These results could then be extended by using simple extrapolation in order to predict the behaviour of distributed denial of service attack.

3.3. *Experimental metrics*

There are a large variety of metrics which is used to evaluate DoS and DDoS attacks. Due to the fact that there is a lack of common metrics and framework for DDoS testing (Mirkovic and Reiher, 2004), this work will use metrics from several research papers in order to provide comparable results. The metrics are also chosen in such a way so that both intrinsic

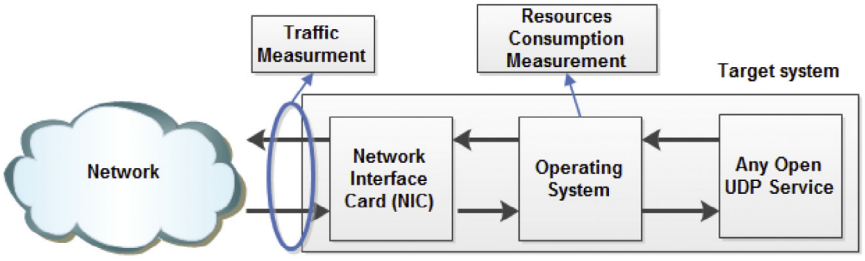

Fig. 11 – Target System Components.

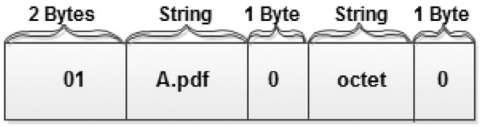

Fig. 12 – Format of the Attack Packet.

and extrinsic measurements are represented, as shown in Hussain et al. (2006). Testing environment is designed in such a way so that both amplifier and target computers will have two main measurement points. One measurement point will be used to capture all incoming and outgoing traffic and a second measurement point will be used to test the computer's resources. This is necessary in order to measure the intrinsic and extrinsic metrics:

- **Amplification factor** – Amplification factor can be measured in different ways, with different results. However, it still remains the key factor when assessing new amplification attacks. In this work, the amplification factor will be measured by using an equation used by Kambourakis et al. (2008) and a newly proposed equation. Nevertheless, both of these equations require measurement of several extrinsic variables. These are: the size of requests and responses, retransmissions, packet losses, and overall incoming and outgoing traffic. Hence this metric is an umbrella term to incorporate all these measurements.
- **Request/Response delay** – This is defined as a delay between the (legitimate) request and reply. Mirkovic et al. (2009) show that this metric could measure the effect of the Denial of Service attack in latency-critical applications (Mirkovic et al., 2009). However, they criticise that this metric is ineffective for "one-way traffic" and other non latency-critical ampplications. Hence, metric for these applications is presented in a paragraph below. It also appears that this extrinsic metric is widely used when evaluating Denial of Service attacks, for instance in Schuba et al. (1996). Thus, this work measures request/response delays to both amplifier and target. The results of this metric will be composed of minimum, maximum and average values so that they can be compared to other research papers and used in future work.
- **Total throughput** – defined as amount of traffic flowing to and from the measured machine at the unit of time. In comparison to Goodput, Throughput takes into consideration re-transmitted bytes (Kuzmanovic and Knightly, 2006). Throughput is an intrinsic, widely used metric when measuring DoS implications and is considered to be an important metric (Mirkovic et al., 2009). Throughput metric should not be used to evaluate DoS attacks against jitter-sensitive applications or loss-sensitive software; these applications can be assessed by request/response delay described above.
- **Usage of Processor resources** – Processing consumption is considered to be an intrinsic metric and will be measured on both amplifier and target in scenarios before and during the attack. Processing consumption is defined as amount of processing power in % being taken from the Central Processing Unit (CPU) by the process at given time. The parameters of the CPU will be shown in the Implementation section. Due to the nature of the testing, the victim machine could run any UDP service in order to be vulnerable to this attack. This testing environment will measure the effects of the default UDP services running in the Windows XP SP3. It was determined that this operating system has the following services open by default: 123, 137, 138, 445, 500, and 1900. Hence it will be observed which of these UDP services consume most of the CPU processing power. The usage of CPU resources will be tested three times to ensure maximum accuracy. From these three measurements an average will be created and analysed.
- **Observation of different TFTP servers** – TFTP servers' implementations could vary. Different implementations of standard might produce different responses, which could result in different amplification factors or resource consumption. Hence, this work will observe the effect of different TFTP software on the traffic.

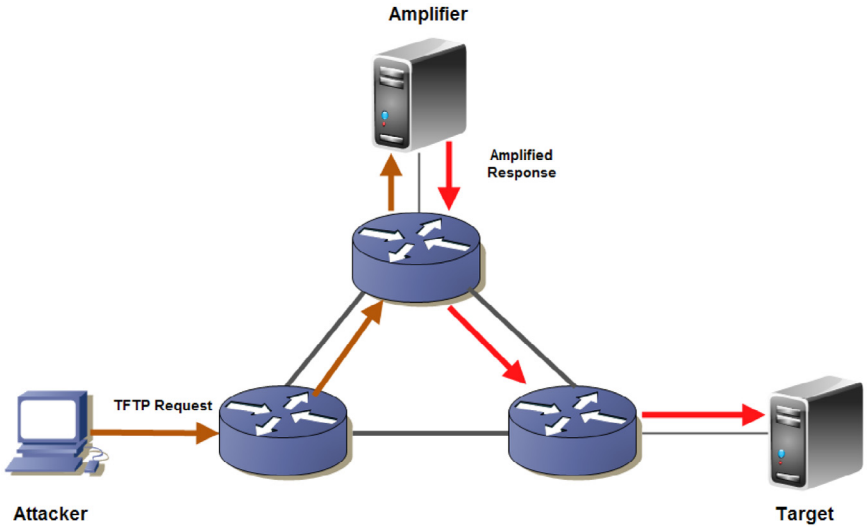

Fig. 13 – Testing Procedure.

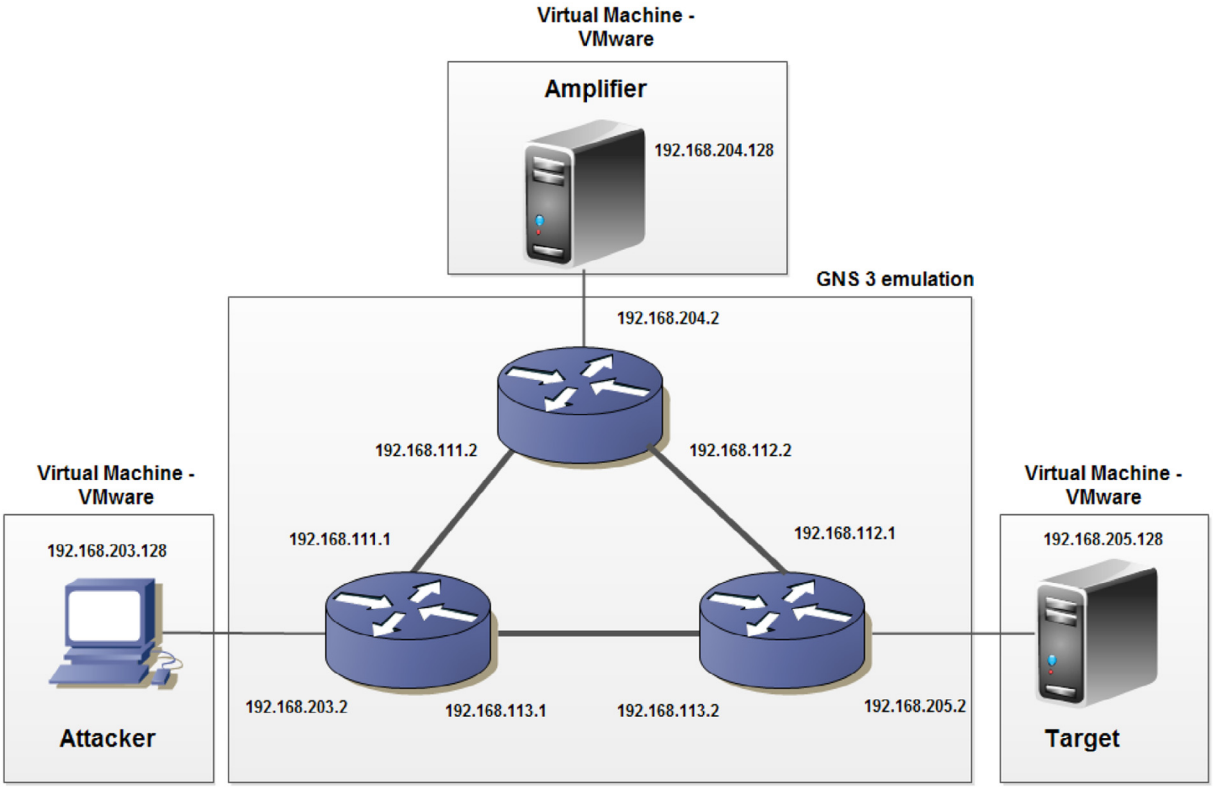

Fig. 14 – Implemented Topology.

4. Implementation

4.1. *Implementation of DoS environment*

The running environment will consist of three routers, one computer and two servers. GNS3 is a simulation software, which allows creation of variety of network topologies. Due to the lack of common framework for testing Denial of Service attacks previously discussed, this work will use the most commonly used testing approach. GNS3 simulation software will be used to simulate the proposed topology, in which three routers will be implemented. The used router image will be Cisco 2600. It will be configured to use EIGRP protocol to provide the most basic routing. The overall aim of the router configuration is to minimise its impact on the generated traffic in order to achieve the most accurate readings. The complete design of the created topology, including IP addresses, can be seen in Fig. 14. All the components present on this figure are run within a single machine.

4.1.1. *Virtual machines*

To allow more flexible simulation systems, the testing environment will be implemented in the VMware virtualization software. VMware is a software which allows several operating systems to be run at a same time in the same machine by using virtualization. VMware is a commonly used software in a DDoS simulations (Prolexic, 2013b or Stone-Gross et al., 2009). This work will use three virtual machines, each one connected to one router running in GNS3, as can be seen in Fig. 14.

4.1.2. *Computer hardware*

An important advantage of using the virtual machines is that there is a possibility of simulating almost any hardware. In order to achieve the most accurate and comparable results, it was decided that all involved parties will have the same hardware parameters. These parameters are shown in Table 1. The used operating system in both amplifier and target will be Windows XP SP3. The tests will be performed on clean installation of Windows, so that they are as accurate as possible. Amplifying machine has several TFTP servers installed in order to check the differences in implementation. Target machine will run default network services in the Windows operating system, which could be determined by running **`netstat -a -n -p udp`** command on the clean Windows installation.

For the testing purposes, attacker uses Kali Linux 5 R3 operating system, which is a Linux-based operating system, which is specifically adjusted to perform large variety of penetration testing techniques and contains several security tools (OffensiveSecurity, 2012).

Table 1 – Virtual computer parameters.

Component	Type
Processor (CPU)	AMD Athlon 64 X2 2.09 GHz
Memory (RAM)	768 MB
Hard disk (HDD)	Generic 15 GB SATA
NIC	100 Mbps

4.1.3. *Spoofed packet creation*

Scapy is a special network analysis tool written in Python (often referred to as Python extension), which allows creation of almost any network packets, without the necessity of manually calculating checksums and other packet control mechanisms (Biondi, 2010). The packet fields on almost any TCP/IP layer can be easily altered including the source IP address, port numbers which will be needed to direct DoS reflection packets to particular service.

The designed tests require the use of non-standard TFTP packet with spoofed IP address; this packet could not be created by using standard means. In the testing experiment, the attacking machine will create a loop in which multiple crafted packets are sent to the amplifying server. This appears to be realistic scenario, because most of the real-life attacks are generating large number of requests. In addition, this technique will show the potential of the amplified traffic and allow longer time to measure effects of this amplified flow to the targeted machine. The following Scapy commands is used to create the described packet:

```
a=IP(dst='192.168.204.128',src='192.168.205.128')/UDP(sport=445,dport=69)/TFTP()/TF
TP_RRQ(filename='A.pdf')


send(a, inter=0.3, loop=1, count=200)
```

The first line will create special TFTP packet with the required parameters. Note that the source port (sport) could change in order to direct attack against different service, as defined in previously discussed metric. The second line defines the parameters of the loop, where *a* is the previously defined packet to be sent, inter is interval between packets, loop is the starting number of the loop and count is number of times loop should be performed (number of sent packets).

4.1.4. *Tested TFTP servers*

There are many TFTP servers' implementations available for all major operating systems. This work will focus on the three widely used implementations:

- TFTPD32 – is a very popular, free, open-source TFTP server created by Philippe Jounin. It supports both IPv4 and IPv6 (Jounin, 2011). TFTPD32 has the ability to log the transmitted traffic in order to provide better accounting. It does not require installation and by default it also creates DHCP Server.
- SolarWinds TFTP Server – is a popular, free implementation of TFTP server. In addition, this implementation runs as a Windows service and appears to be the easiest to use. However, it does not offer any extensive configuration or logging options (SolarWinds, 2012).
- Open TFTP Server – is an open-source TFTP server designed for both Windows and Linux operating systems. This work will only test the Windows implementation. It also allows logging functions. In addition to tftpd32, it supports changing the Block Size of the packet (Dhir, 2013).

As it can be seen, different implementations could vary in several features. One of the key problems when assessing the different TFTP implementations is the possible ambiguity of the default timeout and retransmission values. This ambiguity is expected due to the fact that RFC 1350 – TFTP standard does define any exact retransmission values (Sollins, 1992). From the carried out research, it appears that there are no statistics focused on most commonly used TFTP server implementations. Hence, it is unknown which of these servers are most widely used. Nevertheless, Google search results indicate that SolarWinds TFTP Server appears to be the most popular implementation (Google, 2013).

4.2. *Measuring the metrics*

The evaluators used are:

- **Wireshark.** Wireshark is the well-known protocol analysis tool (Sharpe and Warnicke, 2013). It has been used very widely in research papers focused on DoS analysis (Bhople, 2012; Cowperthwaite and Somayaji, 2010; Takanen et al., 2008). Wireshark supports hundreds of protocols on different layers. It provides several plug-ins which allow detail traffic analysis. This work will use Wireshark in order to examine the transmitted traffic and to measure data flows in both amplifier and target. It will also be useful during the examination of packet sizes.
- **Amplification Factor.** Amplification factor is measured in two ways. In order to get all the necessary variables for the equations, several packet parameters would need to be determined. Size of both requests and replies will be determined by observing flowing traffic in Wireshark. Because of the fact that the target machine will not be able to capture TFTP request, Wireshark instances will be running all involved machines. In addition, when measuring amplification factor, retransmissions of the same packets will be observed as well as packet losses. This data will then be evaluated in order to determine the validity of newly proposed amplification factor equation. Furthermore, comparison of discovered amplification factors will be present.
- **Request/Response Delay.** Request response delay is an important factor when assessing the effects of Denial of Service

Table 2 – Results of measured responses.

TFTP server	Size of request	Size of response	Retransmission count
TFTPD32	56 bytes	558 bytes	6 responses + 1 error packet
SolarWinds TFTP server	56 bytes	558 bytes	6 responses
Open TFTP server	56 bytes	558 bytes	3 responses + 2 error packets

attacks. Hence, it is measured in both amplifier and the target. The request/response delay will be performed by using integrated Windows ping command. This command will generate 20 identical ICMP Echo requests to both attacker and target and measure the time of the responses. This will be done before, during and after the amplification attack taking place. As established in design section, these measurements will be performed 3 times in order to achieve maximum accuracy of the results. Obtained results will be evaluated and the effect of this attack on both amplifier and target will be evaluated.

- **Total Throughput.** Total throughput is measured in both amplifier and target by using performance monitor in Windows operating system. Data Collector set will be created with the "*bytes/sec*" counter set. This data collector will then be run for time period of 140 seconds. The time period of 140 seconds was chosen in order to monitor the normal operation of the service as well as allow enough time for completion of Scapy loop described above. Data will then be exported towards the coma separated text file (.csv) from which the precise values could be obtained.
- **Usage of Processor Resources.** CPU usage is measured by using performance monitor present in Windows systems. The measurement of the CPU usage will be part of a previously described data collector where "*%C1 Time*" counter will be selected. As described in previous section, this data collector will be run for the period of 140 seconds. Exported data will then by analysed to determine the results on each machine. Measurement of CPU usage could be quite difficult due to several reasons. Different CPU models would provide different CPU usage in the same situation. In addition, there could be pseudo-random fluctuations in CPU usage caused by background processes. In order to mitigate the effects of these, several precautions were taken. Firstly, all measurements are done on the same model of CPU to avoid incompatible results. Secondly, all measurements will be performed three times for each scenario/service in order to limit the fluctuations and to increase the accuracy.
- **Behaviour of different TFTP Servers.** As mentioned in previous sections, there are many TFTP server implementations. It is assumed that different TFTP implementations could have different timeout and retransmission behaviour. This is due to the fact that RFC for TFTP does not specify any required or specific values for these rates. In order to determine these values, responses from several TFTP servers will be observed by using Wireshark.

5. Evaluation

This chapter will present evaluation of the obtained experimental test results and analyse their significance. The results were obtained from the testing environment designed in Chapter 3 and implemented in Chapter 4. This section will contain several sub-sections each focused on the evaluation and critical analysis of different measurement results.

5.1. *Results for amplification factor*

An amplification factor is the main metric when measuring Denial of Service amplification attacks. Amplification factor was calculated by using the two previously described approaches. Both approaches required analysis of the transmitted traffic in Wireshark and then observed the size of the request and compared it with the response. The second approach required observing the total traffic in retransmissions as well as packet losses.

The responses shown in Table 2 are the averages from three performed tests. As it can be seen in the same table, responses generated by all tested TFTP implementations are much larger than requests. The uniform request created by an attacking system had a size of 56 bytes, whereas all responses had a size of 558 bytes which is almost 10 times more. This shows that amplification of the traffic indeed occurs in TFTP. The main reason for the present amplification is that the TFTP protocol is designed to be stateless and provides no source IP address verification which would be ideal for the attacking system. In addition, no advanced logging mechanisms are defined on standard and it is completely up to the TFTP server's implementation to provide these features. Moreover, it is shown that several retransmissions of the same packet are present in each tested server. These retransmissions could change the amount of traffic coming to the target, which would effectively change the amplification factor.

As shown in Section 2, many research papers create their own methods on how to calculate amplification factor. Cowperthwaite and Somayaji (2010) use logical definitions of the amplification attack. Amplification factor is often defined as the total amount of traffic received by the target divided by total amount of traffic sent by attacker. This work will refer to this high-level approach as Standard Amplification Factor (variable Aa). In addition, this work calculated the results for Kambourakis et al.'s (2008) equation and the newly proposed equation described above. Table 3 shows the results of calculations for the described amplification factors in the current testing environment.

As can be seen in Table 3, results calculated from Kambourakis et al.'s (2008) equation (variable A) showed that the amplification factor during the TFTP amplification attack was 9.96, which means that according to this equation the malicious traffic was amplified over 9 times. This equation did not take into account retransmission or packet losses in the network. Therefore, all three implementations of TFTP achieved the same implementations according to this equation. When

Table 3 – Achieved amplification factor.

Implementation	Kambourakis et al.'s equation	Proposed equation amplification factor
TFTPD32	$A = \frac{558}{56} = 9.96$	$A = \frac{558 \times 6}{56} = 59.78$
SolarWinds	$A = \frac{558}{56} = 9.96$	$A = \frac{558 \times 6}{56} = 59.78$
OpenTFTP	$A = \frac{558}{56} = 9.96$	$A = \frac{558 \times 3}{56} = 29.89$

comparing these values, it can be seen that Proposed amplification factor and the one calculated by using Kambourakis et al.'s equation vary significantly. This shows that the two different methods of calculating amplification factor can completely differ in the result values. Additionally, the calculated amplification factor by this equation does not correspond to the real amounts of transmitted traffic. The newly proposed equation (variable *An*) takes into consideration both retransmissions and packet losses. This makes this equation the closest to the actual amount of transmitted traffic. By using this equation, it was shown that TFTP implementations could vary significantly in amplification factors. The biggest amplification factors were achieved by Tftpd32 and SolarWinds TFTP servers, whereas the lowest was achieved by OpenTFTP server. The highest amplification was over 59.78 which corresponds to almost 60 times more traffic generated by the amplifier than sent by attacking system.

Table 4 – Amplification factors across various DoS techniques.

Attack type	Amplification factor
Smurf attack (Kumar, 2007)	255 (could be larger; depending on used network)
DNS amplification (Ye and Ye, 2013)	100 (up to)
TFTP amplification	60
NTP amplification (Prolexic, 2013b)	43
CHARGEN amplification (Prolexic, 2013b)	17
SNMP amplification (Prolexic, 2013b)	5 to 7 (depending on request)

5.1.1. *Comparison to other amplification factors*

When compared to other amplification attacks, for instance NTP (Prolexic, 2013b) or DNS amplification (Ye and Ye, 2013), the TFTP amplification factor appears to be quite high. Additionally, what is making TFTP amplification attack powerful and distinguishable from other attacks is the number of retransmissions performed by the amplifiers. Even though it was mentioned that amplification factor could be calculated in several different ways, values shown in Table 4 are created by using Standard Amplification Factor in order to provide comparable rates. Fig. 15 shows the graphical representation of Table 4. As can been seen from these, TFTP amplification appears to be one of the highest from all reviewed amplification factors.

5.2. *Results for request/response delay*

Request and Response delay is an important metric when measuring the effect of particular Denial of Service on the latency-sensitive applications (Mirkovic et al., 2009). It is important to realise that both minimum and maximum latency values are important for these applications. The request/response delay was measured by generating ICMP messages from the independent machine on the network to target. These messages were generated while no attack was occurring as well as during the attack. There were 20 identical ICMP Echo Request messages generated for each test attempt. The results are visible in Fig. 16. For most accurate results, three measurements were taken when no attack was occurring (blue line) and three measurements during the attack (red line).

Fig. 16 shows several different measurements, each having minimum, maximum and average delay values shown. Minimum delay is defined as the smallest value from the 20 ICMP responses. Maximum is defined as the largest value from these and average is the average of all 20 ICMP responses. Due to the fact that testing environment contains only one at-

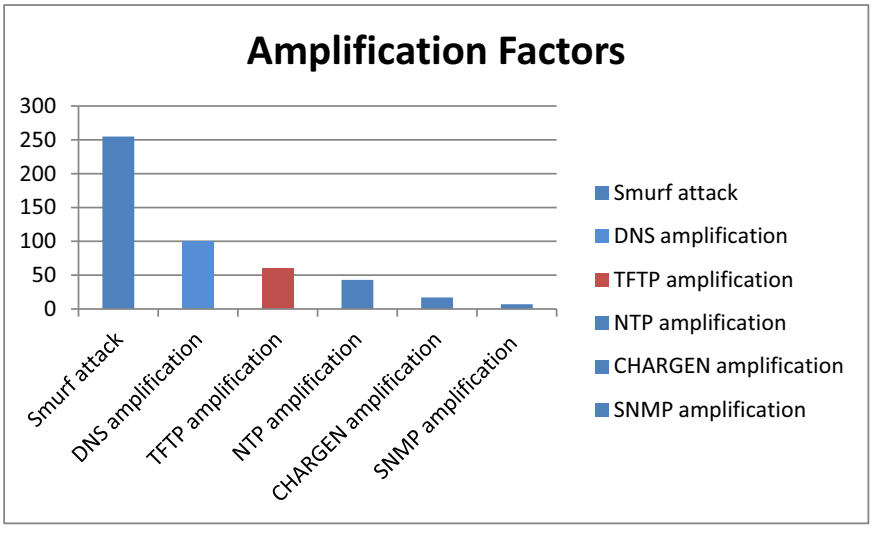

Fig. 15 – Comparison to other Amplification Factors.

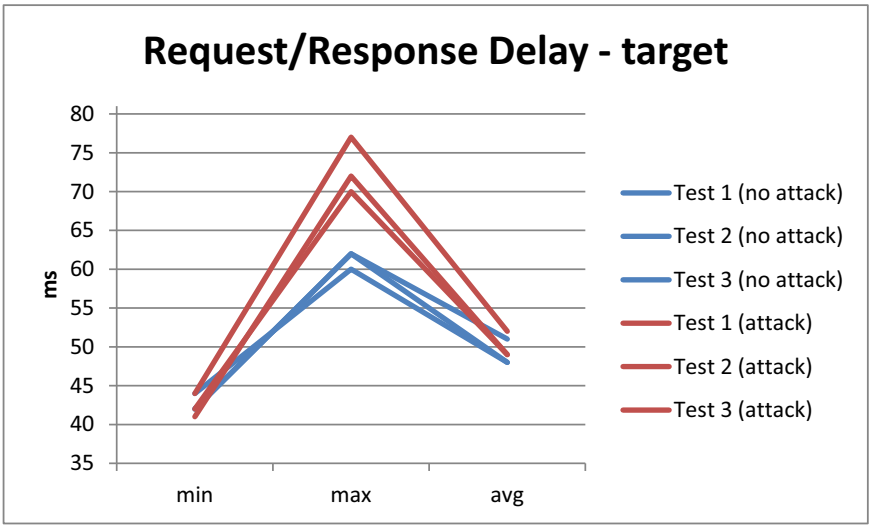

Fig. 16 – Request/Replay Delay on Target.

tacker and one amplifier, the obtained results were not as significant as they would be if the tested attack was distributed and performed by thousands of machines.

As it can be seen in Fig. 16, the minimum delay appears to not be affected by this attack. However, the maximum delay was significantly affected by the performed attack. The maximum delay increased from value 62 when no attack was occurring to value of 77 during the attack. This, if done in distributed manner, could have considerable consequences on latency-critical applications. Average values of delay were slightly higher during the attacks in comparison to normal network operation. All of these data would change if many more attackers and amplifiers were present; this would have much larger effect as it can be seen in many cases of large Distributed Denial of Service attacks.

The graph in Fig. 1 shows the maximum latency before, during and after the attack. The significant effect of this attack on the target's maximum latency is clearly visible. The maximum latency increased from 60 ms to 77 ms.

5.3. *Results for total throughput*

Total throughput was identified as an important metric to measure effects of this attack on the targeted service. Many Denial of Service attacks are focused on exhausting the throughput on the target machine (Guo et al., 2006). The total throughput was measured by using performance manager tool running on the target machine. The added counter was *Bytes Received/sec*. Due to the fact that the testing only simulated one attacking system, the transmitted data rates were lower than in real-world situation when the attack is distributed over many machines. Nevertheless, the measurements for total throughput confirm the accuracy of the newly proposed amplification factor equation.

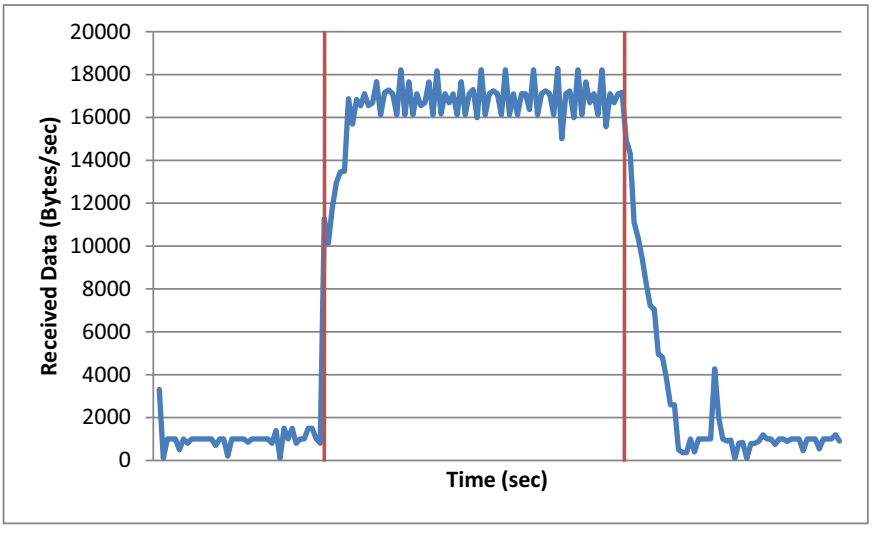

Fig. 17 – Total Throughput before, during and after Attack.

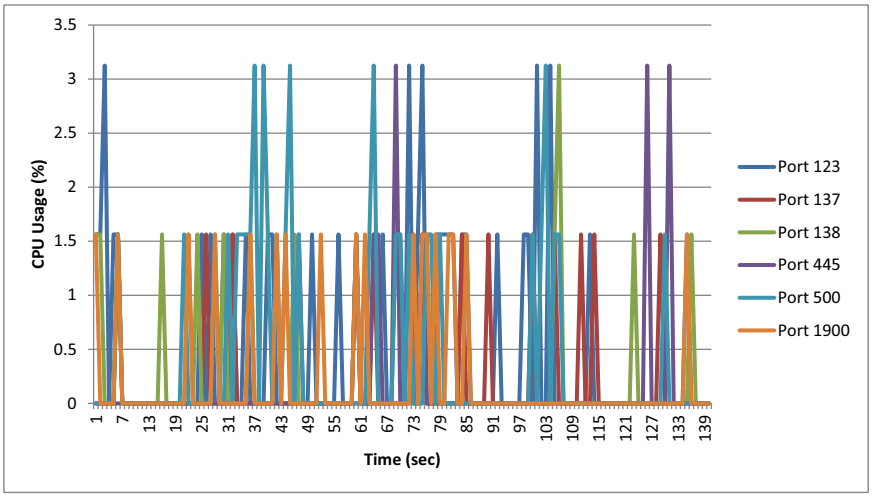

Fig. 18 – CPU Usage for Different Services during the Attack.

The obtained results are the average of three performed tests, as described in Design and Implementation section. As it can be seen in Fig. 17, the received data on the target before the attack were approximately 1000 bytes per second. This includes the broadcasts, routing updates and other protocol messages. When the amplification attack started, the amount of received data increased rapidly leading up to 18 200 bytes per second. This is over 18 times higher than the average traffic flow before the attack, using only one attacker machine and one amplifier. Hence, this shows that if this attack would be distributed throughout many amplifiers and attackers, its effects could have severe impact on the target throughput.

5.4. *Usage of CPU resources*

Processor (CPU) consumption and other computer resources are often important metrics when evaluating Denial of Service attacks. It appears that the evaluation of the CPU usage would be quite difficult due to several reasons. Firstly, the tested attack is only performed from one computer; hence it might not have visible/significant effect on the target. Secondly, usage of the CPU would change with the different CPU model. Additionally, CPU resources could be taken in pseudo-random manner by the programs running in the background. Therefore, this testing method can only show the approximate effects of this attack on the processor. However, several precautions were taken in order to make the measurements as accurate as possible. These are described in the Implementation section above. The measurement of the CPU usage was done in the integrated *perfmon* software tool, where *Processor Time* counter was selected.

Fig. 18 shows the CPU usage in percentage at the particular time during the attack. This figure shows the average of the obtained results during three performed tests. As it can be seen in Fig. 18, the tested attack lasted for 140 seconds. Attacks against several UDP services and their effects on the CPU usage are also clearly visible. It is notable that most of the tested services have taken similar amounts of CPU resources, although in slightly different times. This might be caused by different operation of these services and techniques they use to handle large amounts of invalid traffic.

The highest CPU consumption during the attack was distinguished on the port 500, which is used by IP sec (tunnelling) services (IANA, 2013). Second highest usage was also notable on service 445, which is used by Windows sharing and Active Directory services (IANA, 2013). Again, the effects on the CPU usage were only marginal; however, if this attack would be distributed, it is highly probable that the CPU consumption would be increased rapidly.

5.5. *Measured variations in different TFTP implementations*

As mentioned previously, there are several TFTP servers and it is unclear how they vary in implementation of TFTP standard and non-standard functions. There have been three tested TFTP implementations: TFTPD32, SolarWinds TFTP Server and Open TFTP server, each using default configuration. Two main metrics were observed – retransmission rates and timeout values. Retransmission rate determines how many times packets will be retransmitted if there is no response from the destination. Timeout value determines how long would the response be waited for. This means that after the message is sent from the TFTP server, the software will wait some time (timeout value) to receive any valid response. If no valid response is received, the message is retransmitted. The maximum time of retransmission is determined by the retransmission rate value.

The following variations were determined:

- TFTPD32 is a popular TFTP implementation. Testing discovered that default settings provide the retransmission count of six, each with the size of 558 bytes. After these six retransmissions, there is an error message generated with the size of 60 bytes. It was also discovered that in the default settings, the timeout rate for this TFTP implementation is variable. This means that first two retransmissions have timeout set to one second; the rest of the retransmissions are performed after 3 seconds.

- SolarWinds TFTP Server is a free implementation of TFTP, FTP and SFTP/SCP servers. The testing was focused only on the TFTP part of this software. It was discovered that SolarWinds TFTP Server has a default retransmission rate of 6, each with the size of 558 bytes. If none of these retransmissions get the response, no error message is generated. Timeout of this TFTP implementation was the highest from all tested – 16 seconds. Hence, one specially adjusted TFTP request could create a packet flow which would last for over 96 seconds (6×16).
- Open TFTP Server is an open-source implementation of the TFTP server. During the testing it was discovered that, in default settings, specially crafted request could produce three retransmissions, each with the size of 558 bytes. In addition to this, if there is no response after these three retransmissions, the error message is generated with the size of 60 bytes. The timeout of this TFTP implementation was measured to be three seconds. This means that after the message is sent from the TFTP server, the software will wait three seconds to receive any valid response. If no valid response is received, the message is retransmitted – up to three times.

5.6. *Evaluation of limitations of the proposed attack*

This section will evaluate the discovered limitations of this protocol when used in DoS amplification attacks. The presented method has one major limitation. In order for TFTP amplification attack to be successful, the exact filename of the stored file on amplifying TFTP server has to be known. Because TFTP standard does not provide any directory listing, the file names would have to be guessed manually. This could be a non-trivial issue because the file names could be completely random. One of the possible solutions to this problem is to use the brute-force technique, often called fuzzing, in which a large amount of file names are tested and the correct ones are marked (Takanen et al., 2008). There are many tools which allows brute-forcing TFTP names; for instance, the tool described in Gauci (2011). However, trying a large number of different filenames would appear very suspicious and could be easily detected by Intrusion Detection Systems.

Another solution is to create a list of most commonly used file names in TFTP. Few word-lists of these common file names already exists; for instance, the list present in *tftp–enum* nmap scan (Rudakov, 2009). This list could be further improved by using the following approach. It is well-known that Cisco devices use TFTP servers very often. Hence, apart from the file names already present in word-lists, the products which are the most sold could be determined, for instance those shown in RouterSwitch (2012). Then appropriate file names for these products could be found in several public lists, for instance the list present at Certs4u.info (2012), and added to the list. In this way, a word-list of commonly used words will be created with reasonable length. This would be less suspicious than the brute-force attack and provide a much faster results.

Another key limitation of the presented TFTP amplification attack is that vulnerable servers have to be identified beforehand. One of the important questions that had to be answered during the writing of this work was how many TFTP servers are accessible on the Internet. There have been some concerns that the TFTP is not widely used and hence it will be unrealistic to use this service for a real-world attack. However, from the current Internet scans available on InternetCensus (2012), it was determined that there are over 599 600 TFTP servers openly accessible on the Internet. Because this novel attack is taking advantage of the protocol flaws present in the standard, it is assumed that the vast majority of these open servers will be vulnerable to be used as amplifiers.

5.7. *Countermeasures*

The previous sections have shown how the TFTP amplification attack could be performed and what effects it can have on a targeted machine. In addition, several general DDoS detection and DDoS mitigation techniques were described. This section will describe what specific countermeasures could be applied on the amplifier to limit the participation in this attack. Despite the fact that the tested novel amplification attack is using protocol flaw in the TFTP standard, there are several methods which could be used on the TFTP servers to prevent participation in the amplification attack. These countermeasures were determined during the testing and they appear to be quite effective in the current testing environment:

- Make TFTP server inaccessible from the Internet unless necessary. Hiding the TFTP server behind a firewall significantly limits the options of the attacker. Even though this countermeasure is well known and has been used for some time (Ranum, 1992), it appears that a large number of networks do not utilise this countermeasure due to the configuration errors (Wool, 2009).
- Intrusion Detection System should be present to check the incoming and outgoing traffic to the TFTP server. If there are many requests for the same server and the same file name, access to the server should be temporarily blocked in order to avoid participation in DDoS attack. This is a common approach when detecting attacks by using Intrusion Detection System (Roesch and Green, 2012). It is unknown whether the current intrusion detection systems contain rules to check for these; if not, new rules would have to be created.
- Throughout the observation it was shown that some TFTP implementations have retransmission rate set to 5, which could provide attackers with five-time higher amplification. Hence, wherever possible, the retransmission rate should be set to 1. This could cause some issues especially in non-reliable networks, therefore compromise should be found between retransmission rates and error-correction.

In addition, it was shown that error messages are typically larger than the requests. This is mostly due to the fact that an error message contains custom-defined strings (Sollins, 1992), which could be quite large. Hence, the size of the error messages could be decreased by using shorter strings. For instance, “File Inaccessible” could be sent as “FI”, which would limit the size of the response and minimise the amplification rates. A detail log should be present in TFTP software in order to allow accounting, identification of attackers and exact time

periods of attacks. Additional countermeasure could be to have certain threshold of requests per second present in TFTP server, in order to avoid large amplification attacks.

6. Conclusions

The performed tests showed that TFTP is vulnerable of being used in amplification attacks with high amplification factor. This attack has been tested to have very high amplification factor of 60, which is the third highest from the reviewed literature. Due to the gaps in standards, lack of TFTP-based security mechanisms and the fact that this attack uses the protocol behaviour defined in the standard, vast majority of all open internet-based TFTP servers appear to be vulnerable. This could lead to over 599 600 publicly open servers which could be used for amplification attacks. Therefore, TFTP amplification attack appears to be a truly global security issue. To minimise the risk of this attack, a variety of countermeasures was shown which would effectively limit the severity of this attack.

Several variations in the TFTP implementations were measured. It was shown that all tested TFTP servers had different behaviour to the attack packet. This could be an issue when evaluating real-world amplification attacks, due to the fact that different TFTP implementations could produce different amplification factors. An important part of evaluation is a discussion of the limitations of novel TFTP amplification attack. Two key limitations were discovered and critically evaluated. Numerous solutions have been found for these limitations, which could be applied in future work.

It was shown that there is a lack of the commonly accepted definition of amplification factor (Abramov and Herzberg, 2013; Kambourakis et al., 2008; Kumar, 2007). Therefore, several amplification factors' equations were reviewed and critiqued. Several limitations were discovered; for instance, none of the reviewed equations had taken into account retransmissions or packet losses. Hence, this work is proposing a new equation to calculate the amplification factor. This equation is described, analysed and its accuracy evaluated. From the performed tests, it appears that proposed equation is more accurate when evaluating the TFTP amplification attack than the equation proposed by Kambourakis et al. (2008).

6.1. *Future work*

Several limitations were identified throughout this work. The designed testing environment was sufficient to test the presented theory; however, in order to simulate a real-world scenario, more machines would be necessary. It would be valuable to create the testing environment with many more computers and to observe the effects of the proposed attack on all involved amplifiers and targets.

It was shown that the attacker could create some form of a word-list in order to guess the name of the file present on the TFTP server. Although some of these word-lists exist, it would be beneficial to create new improved list according to the methods mentioned in the same section. An observation of the accuracy of this wordlist would be a part of the new testing methods. Even though a variety of countermeasures were shown, it would be advantageous to test their effectiveness on the larger system. In addition, testing the effectiveness of traditional DDoS countermeasures on this attack would also be beneficial. Measured results indicate that the proposed equation for amplification factor appears to be correct. Nevertheless, it was discovered that it does not provide fully accurate results. This is due to the fact that this equation assumes all retransmissions to be the same, which was not the case. Therefore, it would be beneficial if this limitation would be alleviated, which would improve overall accuracy of the equation.

REFERENCES

Abramov R, Herzberg A. TCP Ack storm DoS attacks. Comput Secur 2013;33:12–27. doi:10.1016/j.cose.2012.09.005.

Adib M. LOIC – low orbit ion cannon. <https://play.google.com/store/apps/details?id=genius.mohammad.loic&hl=en>; 2013 [accessed Nov 2015].

Akella A, Bharambe A, Reiter M, Seshan S. Detecting DDoS attacks on ISP networks, 3 pp. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.61.2113&rep=rep1&type=pdf>; 2003 [accessed Nov 2015].

Alomari E, Manickam S, Gupta BB, Karuppayah S, Alfaris R. Botnet-based distributed denial of service (DDoS) attacks on web servers: classification and art. Int J Comput Appl 2012;49(7):24–32.

Alturki M, Meseguer J, Gunter CA. Probabilistic modeling and analysis of DoS protection for the ASV protocol, 15 pp. <http://seclab.web.cs.illinois.edu/wp-content/uploads/2011/03/AlTurkiMG08.pdf>; 2008 [accessed Nov 2015].

Arora K, Kumar K, Sachdeva M. Impact analysis of recent DDoS attacks. Int J Comp Sci Eng 2011;3(2):877–84. <http://www.enggjournals.com/ijcse/doc/IJCSE11-03-02-093.pdf>.

Arukonda S, Sinha S. The innocent perpetrators: reflectors and reflection attacks. Adv Comput Sci 2015;4.

Baker F. RFC 1812 – requirements for IP version 4 routers. IEEE Standard. <http://tools.ietf.org/html/rfc1812#section-5.3.5.2>; 1995 [accessed Nov 2015].

Barnett R. HOIC DDoS analysis and detection. <http://blog.spiderlabs.com/2012/01/hoic-ddos-analysis-and-detection.html>; 2012 [accessed Nov 2015].

Bhople S. Server based DoS vulnerabilities in SSL/TLS protocols (Master thesis). <http://alexandria.tue.nl/extra1/afstversl/wsk-i/bhople2012.pdf>; 2012 [accessed Nov 2015].

Binsalleeh H, Ormerod T, Boukhtouta A, Sinha P, Youssef A, Debbabi M, et al. On the analysis of the Zeus botnet crimeware toolkit. In: Eighth annual international conference on privacy security and trust (PST). Ottawa, ON: 2009. p. 31–8 <http://www.ncfta.ca/papers/On_the_Analysis_of_the_Zeus_Botnet_Crimeware.pdf>.

Biondi P. Scapy documentation. <http://www.secdev.org/projects/scapy/doc/introduction.html#about-scapy>; 2010 [accessed Nov 2015].

Bogdanoski M, Shuminoski T, Risteski A. Analysis of the SYN flood DoS attack. Int J Comput Netw Inf Secur 2013;5(8):1–11. doi:10.5815/ijcnis.2013.08.01.

Busschers R. Effectiveness of defense methods against DDoS attacks by anonymous. <http://referaat.cs.utwente.nl/conference/16/paper/7312/effectiveness-of-defense-methods-against-ddos-attacks-by-anonymous.pdf>; 2010 [accessed Nov 2015].

Case J, Fedor M, Schoffstall M, Davin J. RFC 1157 – a simple network management protocol. IEEE Standard. <http://tools.ietf.org/pdf/rfc1157.pdf>; 1990 [accessed Nov 2015].

Certs4u.info. List of IOS images. <http://certs4u.info/ciscoios/>; 2012 [accessed Nov 2015].
CERT. Denial-of-service attack via ping. CA-1996-26. <http://www.cert.org/advisories/CA-1996-26.html>; 1996 [accessed Nov 2015].
CERT. Smurf IP denial-of-service attacks. CA-1998-01. <http://www.cert.org/advisories/CA-1998-01.html>; 1998 [accessed Nov 2015].
Constantin L. Cyber–thieves use DDoS to cover up wire transfer fraud. Computer World UK. <http://www.computerworlduk.com/news/security/3323120/cyber-thieves-use-ddos-to-cover-up-wire-transfer-fraud/>; 2011 [accessed Nov 2015].
Constantin L. Possibly related DDoS attacks cause DNS hosting outages. PC World. <http://www.pcworld.com/article/2040766/possibly-related-ddos-attacks-cause-dns-hosting-outages.html>; 2013 [accessed Nov 2015].
Cowperthwaite A, Somayaji A. The futility of DNSSec. In: Proceedings of 5th annual symposium on information assurance. 2010. p. 2–8 <http://people.scs.carleton.ca/~soma/pubs/acowpert-asia-2010.pdf?q=~soma/pubs/acowpert-asia-2010.pdf>.
Criscuolo PJ. Distributed denial of service. <http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA396999>; 2012 [accessed Nov 2015].
Czyz J, et al. Taming the 800 pound gorilla: the rise and decline of NTP DDoS attacks. In: Proceedings of the 2014 Conference on Internet Measurement Conference. ACM; 2014.
Deshpande T, Katsaros P, Basagiannis S, Smolka SA. Formal analysis of the DNS bandwidth amplification attack and its countermeasures using probabilistic model checking. In: IEEE 13th international symposium on high–assurance systems engineering (HASE). Boca Raton, FL: 2002. p. 360–7 <http://delab.csd.auth.gr/~katsaros/hase2011.pdf>.
Dhir A. Open TFTP server. <http://sourceforge.net/projects/tftp-server/>; 2013 [accessed Nov 2015].
Dittrich D. So you want to take over a botnet. In: 5th USENIX conference on large–scale exploits and emergent threats. 2012.
Douligeris C, Mitrokotsa A. Ddos attacks. In: 3rd IEEE conference on signal processing and information technology. 2003. p. 190–3 <http://jmiller.uaa.alaska.edu/cse465-fall2011/papers/douligeris2003.pdf>.
Durno S. Botnet analysis using command and control channels (December), 11 pp. <http://people.scs.carleton.ca/~paulv/durno-botnetcc-5407F-project2.pdf>; 2011 [accessed Nov 2015].
Feinstein L, Schnackenberg D, Balupari R, Kindred D. Statistical approaches to DDoS attack detection and response. In: Proceedings DARPA information survivability conference and exposition. IEEE; 2003. p. 303–14 doi:10.1109/DISCEX.2003.1194894.
Garg A, Reddy ALN. Mitigation of DoS attacks through QoS regulation. Microprocess Microsyst 2004;28(10):521–30. doi:10.1016/j.micpro.2004.08.007.
Garg D. DDOS mitigation techniques – a survey, p. 319–26. <http://ijcns.uacee.org/vol1iss1/files/V1-I1-62.pdf>; 2011 [accessed Nov 2015].
Gauci S. TFTP security scanning tools. <http://code.google.com/p/tftptheft/>; 2011 [accessed Nov 2015].
Geva M, Herzberg A, Gev Y. Bandwidth distributed denial of service: attacks and defenses. IEEE Secur Priv 2013;12(1):54–61. <http://doi.ieeecomputersociety.org/10.1109/MSP.2013.55>.
Gilad Y, Herzberg A. TCP injections for fun and clogging. <http://arxiv.org/pdf/1208.2357>; 2012 [accessed Nov 2015].
GitHub. LOIC source code. <https://github.com/NewEraCracker/LOIC>; 2013 [accessed Nov 2015].
Gligor VD. A note on denial-of-service in operating systems. IEEE Trans Softw Eng 1984;10(3):320–4.
Google. TFTP server search results – Google. <https://www.google.co.uk/?gws_rd=cr#bav=on.2,or.r_qf.&fp=832cd348f6dd6de4&q=tftp+server>; 2013 [accessed Nov 2015].
Guo F, Chen J, Chiueh T. Spoof detection for preventing DoS attacks against DNS servers. In: 26th IEEE international conference on distributed computing systems (ICDCS'06). IEEE; 2006. p. 37 doi:10.1109/ICDCS.2006.78.
Hampson NCN. Hacktivism: a new breed of protest in a networked world, p. 511–43. <http://digitalmediafys.pbworks.com/w/file/fetch/60358233/HampsonN2012Hacktivism.pdf>; 2010 [accessed Nov 2015].
Hussain A, Schwab S, Thomas R, Fahmy S, Mirkovic J. DDoS experiment methodology. In: Proceedings of the DETER community workshop on cyber security experimentation. 2006.
IANA. Service name and transport protocol port number registry. <http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>; 2013 [accessed Nov 2015].
ICANN. SSAC advisory SAC008 DNS Distributed Denial of Service (DDoS) attacks, pp. 1–16. <http://www.icann.org/en/groups/ssac/dns-ddos-advisory-31mar06-en.pdf>; 2006 [accessed Nov 2015].
Imperva. Monthly trend report. <https://www.imperva.com/docs/HII_Denial_of_Service_Attacks-Trends_Techniques_and_Technologies.pdf>; 2012 [accessed Nov 2015].
Ingle A, Awade M. Intrusion detection for ICMP – flood attack. Int J Comput Sci Inf Technol 2013;1(1):1–4.
InternetCensus. Services probe. <http://internetcensus2012.bitbucket.org/serviceprobe_overview.html>; 2012 [accessed Nov 2015].
James JI. Legal protest and distributed denial of service, pp. 5–7. <http://digitalfire.ucd.ie/wp-content/uploads/2013/03/Legal-Protest-and-Distributed-Denial-of-Service.pdf>; 2013 [accessed Nov 2015].
Jin C, Wang H, Shin KG. Hop–count filtering: an effective defense against spoofed traffic. <http://www.eecs.umich.edu/techreports/cse/2003/CSE-TR-473-03.pdf>; 2003 [accessed Nov 2015].
Jin S, Yeung DS. A covariance analysis model for DDoS attack detection. In: 2004 IEEE international conference on communications, vol. 4. IEEE; 2004. p. 1882–6.
Jounin P. Tftpd32. <http://tftpd32.jounin.net/>; 2011 [accessed Nov 2015].
Kambourakis G, Moschos T, Geneiatakis D, Gritzalis S. Detecting DNS amplification attacks. In: Critical information infrastructures security, vol. 5141. Springer; 2008. p. 185–93 <http://www.icsd.aegean.gr/infosec_base/papers/CRITIS_CR.pdf>.
Kambourakis G, Moschos T, Geneiatakis D, Gritzalis S. A Fair Solution to DNS Amplification Attacks, (October 2002), 2013. Retrieved from http://www.edi–info.ir/files/A–Fair–Solution–to–DNS–Amplification–Attacks.pdf.
Khanna S, Venkatesh SS, Fatemieh O, Khan F, Gunter CA. Adaptive selective verification: an efficient adaptive countermeasure to Thwart DoS attacks. IEEE ACM Trans Netw 2012;20(3):715–28. <http://seclab.illinois.edu/wp-content/uploads/2012/07/KhannaVFKG12.pdf>.
Kumar S. Smurf–based Distributed Denial of Service (DDoS) Attack Amplification in Internet. In: Second International Conference on Internet Monitoring and Protection. 2007 Retrieved from: <http://cs.uno.edu/~dbilar/11CSCI6621–NetworkSecurity/papers/Kumar(2007)Smurf–based Distributed Denial of Service (DDoS).pdf>.
Kumar VA, Sisalem D. TCP based denial-of-service attacks to edge network: analysis and detection. In: Intelligent information technology, vol. 3356. Lecture notes in computer

science. 2004. p. 214–23 <http://link.springer.com/chapter/10.1007/978-3-540-30561-3_23#page-2>.
Kuzmanovic A, Knightly EW. Low–rate TCP–targeted denial of service attacks and counter strategies. IEEE ACM Trans Netw 2006;14(4):683–96. doi:10.1109/TNET.2006.880180.
Kührer M, et al. Exit from hell? Reducing the impact of amplification DDoS attacks. USENIX Security Symposium. 2014.
Lan K, Hussain A, Dutta D. Effect of malicious traffic on the network. In: Proceedings of PAM. 2003.
Lau F, Rubin SH, Smith MH, Trajkovic L. Distributed denial of service attacks. In: SMC 2000 conference proceedings. 2000 IEEE international conference on systems, man and cybernetics: cybernetics evolving to systems, humans, organizations, and their complex interactions, vol. 3. 2000. p. 2275–80 doi:10.1109/ICSMC.2000.886455 Cat. No.00CH37166.
Lee N. Counterterrorism and cybersecurity. Springer New York; 2013. p. 119–42 doi:10.1007/978-1-4614-7205-6.
Lippmann RP, Fried DJ, Graf I, Haines JW, Kendall KR, Mcclung D, et al. Evaluating intrusion detection systems: the 1998 DARPA off–line intrusion detection evaluation*. In: DARPA information survivability conference and exposition. 2000. p. 12–26.
Liu Q, Zhang Y. TFTP vulnerability finding technique based on fuzzing. Comput Commun 2008;31(14):3420–6. doi:10.1016/j.comcom.2008.05.041.
MacFarland DC, Shue CA, Kalafut AJ. Characterizing optimal DNS amplification attacks and effective mitigation. In: Passive and active measurement. 2015. p. 15–27.
Mahajan R, Wetherall D, Anderson T. Understanding BGP misconfiguration. Comput Commun Rev 2002;32(4):3–16. doi:10.1145/964725.633027.
Mansfield G, Ohta K, Takei Y, Kato N, Nemoto Y. Towards trapping wily intruders in the large, vol. 34. Elsevier; 2000. p. 659–70 <http://www.thefengs.com/wuchang/work/courses/cse5xx_OGI/cse581_winter2002/papers/mansfield00wily_hacker.pdf>.
Menzies T. The raw and the uncooked: the Windows XP raw sockets saga, final words (Hopefully). <http://www.sans.org/reading_room/whitepapers/windows/raw-uncooked-windows-xp-raw-sockets-saga-final-words-hopefully_289>; 2002 [accessed Nov 2015].
Mirkovic J, Reiher P. A taxonomy of DDoS attack and DDoS defense mechanisms. Comput Commun Rev 2004;34(2): 39–54. <http://www.eecis.udel.edu/~sunshine/publications/ccr.pdf>.
Mirkovic J, Robinson M, Reiher P. Alliance formation for DDoS defense. In: Proceedings of the 2003 workshop on new security paradigms. 2004. p. 11–18 <http://dl.acm.org/citation.cfm?id=986658>.
Mirkovic J, Arikan E, Wei S, Thomas R, Fahmy S, Reiher P. Benchmarks for DDoS defense evaluation. In: Proceedings of IEEE Military communications conference, MILCOM 2006. 2006. p. 1–10.
Mirkovic J, Reiher P, Fahmy S, Thomas RK. How to test DoS defenses. In: Proceedings of conference for homeland security CATCH '09. Cybersecurity applications & technology. Washington, DC: 2009. p. 103–17 <http://ftp.cs.purdue.edu/homes/fahmy/papers/catch.pdf>.
Montoro R. LOIC DDoS analysis and detection. <http://blog.spiderlabs.com/2011/01/loic-ddos-analysis-and-detection.html>; 2011 [accessed Nov 2015].
Nguyen H, Choi Y. Proactive detection of DDoS attacks utilizing k–NN classifier in an anti–DDos framework. <http://pdf.aminer.org/000/339/184/svm_based_packet_marking_technique_for_traceback_on_malicious_ddos.pdf>; 2010 [accessed Nov 2015].
Nordstr O, Dovrolis C. Beware of BGP attacks. Comput Commun Rev 2004;34(2):1–8. <http://www.cs.princeton.edu/courses/archive/fall06/cos561/papers/nordstrom04.pdf>.
OffensiveSecurity. BackTrack Linux 2011 hompage. <http://www.backtrack-linux.org/about/>; 2012 [accessed Nov 2015].
Park K, Lee H. On the effectiveness of probabilistic packet marking for IP traceback under denial of service attack. In: Proceedings of IEEE INFOCOM 2001. Twentieth annual joint conference of the IEEE computer and communications societies. 2000. p. 338–47 <http://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=2490&context=cstech&sei-redir=1&referer=http%3A%2F%2Fscholar.google.co.uk%2Fscholar%3Fhl%3Den%26as_sdt%3D0%2C5%26q%3Dprobabilistic%2Btraceback%2Bddos#search=%22probabilistictracebackddos%22>.
Paxson V. An analysis of using reflectors for distributed denial-of-service attacks. Comput Commun Rev 2001;31(3):38–47. <http://dl.acm.org/citation.cfm?id=505664>.
Peng T, Leckie C, Ramamohanarao K. Protection from Distributed Denial of Service Attack Using History–based IP Filtering. In: IEEE international conference oncommunications, 2003. 2003. p. 482–6.
Peng T, Leckie C, Ramamohanarao K. Survey of Network–based Defense Mechanisms Countering the DoS and DDoS Problems. ACM Transactions on Computational Logic 2006;2(3). Retrieved from: <http://ww2.cs.mu.oz.au/~tpeng/mudguard/research/ACM_Computing_Survey_Final.pdf>.
Pras A, Sperotto A, Moura GCM, Drago I, Barbosa R, Sadre R, et al. Attacks by "Anonymous" WikiLeaks proponents not anonymous, 10 pp. <http://doc.utwente.nl/75331/1/2010-12-CTIT-TR.pdf>; 2010 [accessed Nov 2015].
Prentow TS, Krarup MV. MITM attacks on SSL/TLS related to renegotiation. <http://www.daimi.au.dk/~ivan/reports2009/MITMattacksSSL.pdf>; 2009 [accessed Nov 2015].
Prolexic. Prolexic quarterly global DDoS attack report, p. 20. <http://www.prolexic.com/kcresources/attack-report/attack_report_q113_english-version/Prolexic_Quarterly_Global_DDoS_Attack_Report_Q113_041613.pdf>; 2013a [accessed Nov 2015].
Prolexic. An analysis of DrDoS SNMP/NTP/CHARGEN reflection attacks part II of the DrDoS white paper series. <http://ictc.aeoi.org.ir/sites/default/files/An_Analysis_of_DrDoS_SNMP.pdf>; 2013b [accessed Nov 2015].
Ranum MJ. A network firewall, 10 pp. 1992.
Rastegari S, Saripan MI, Rasid MFA. Detection of Denial of Service attacks against domain name system using neural networks. Int J Comput Sci 2009;6(1):23–7. <http://arxiv.org/pdf/0912.1815.pdf>.
Roesch M, Green C. Snort users manual 2.9.3. 2012.
RouterSwitch. Top 20 best–selling Cisco products. <http://www.router-switch.com/promotion/2013/top-best-selling-cisco-products-list.html>; 2012 [accessed Nov 2015].
Rudakov A. TFTP–enum plugin. <http://nmap.org/nsedoc/scripts/tftp-enum.html>; 2009 [accessed Nov 2015].
Sairam AS, Subramaniam LA, Barua G. Defeating reflector based Denial-of-Service attacks using single packet filters. In: Proceedings of the 5th International ICST Conference on Communications and Networking in China. IEEE; 2010. p. 1–5 doi:10.4108/chinacom.2010.86.
Saltzer J, Schroeder M. The protection of information in computer systems. Proc IEEE 1975;63(9):1278–308. <http://www.acsac.org/secshelf/papers/protection_information.pdf>.
Sauter M. "LOIC will tear us apart": the impact of tool design and media portrayals in the success of activist DDOS attacks. Am Behav Sci 2013;57(7):983–1007. doi:10.1177/0002764213479370.
Savage S, Cardwell N, Wetherall D, Anderson T. TCP congestion control with a misbehaving receiver. Comput Commun Rev 1999;29(5):71–8. <http://ccr.sigcomm.org/archive/1999/oct99/savage.pdf>.

Schuba CL, Huhn MG, Spafford EH. Analysis of a denial of service attack on TCP. Proc IEEE Symp Secur Priv 1996;208–23. <http://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=2326&context=cstech&sei-redir=1&referer=http%3A%2F%2Fscholar.google.co.uk%2Fscholar%3Fq%3DSYNkill%2Bschuba%26btnG%3DSubmit%26hl%3Den%26as_sdt%3D0%252C5#search=%22SYNkillschuba%22>.
Schultz. A smorgasbord of denial of service, 25 Oct 2013. <http://blogs.cisco.com/security/a-smorgasbord-of-denial-of-service>; 2013 [accessed Nov 2015].
Sharpe R, Warnicke E. Wireshark user's guide. <http://www.wireshark.org/docs/wsug_html_chunked/>; 2013 [accessed Nov 2015].
Sherwood R, Bhattacharjee B, Braud R. Misbehaving TCP receivers can cause internet–wide congestion collapse technical report: UMD-CS-TR-4737. In: Proceeding of the 12th ACM Conference on Computer and Communications security. 2005. p. 383–92 <http://drum.lib.umd.edu/bitstream/1903/3019/1/optack-extended.pdf>.
Singh A, Singh B, Joseph H. Vulnerability analysis for FTP and TFTP. In: Vulnerability analysis and defense for the internet. 2008. p. 71–7.
Snoeren AC, Partridge C, Sanchez LA, Jones CE, Tchakountio F, Kent ST, et al. Hash–based IP traceback. Comput Commun Rev 2001;31(4):3–14. <http://www.cs.cmu.edu/~srini/15-744/papers/p1-snoeren.pdf>.
SolarWinds. FREE TFTP server & SFTP/SCP server. <http://www.solarwinds.com/products/freetools/free_TFTP_server.aspx>; 2012 [accessed Nov 2015].
Sollins K. The TFTP protocol (revision 2). IEEE Standard. <http://tools.ietf.org/pdf/rfc1350.pdf>; 1992 [accessed Nov 2015].
Stone-Gross B, Cova M, Cavallaro L, Gilbert B, Szydlowski M, Kemmerer R, et al. Your botnet is my botnet: analysis of a botnet takeover. In: 16th ACM Conference on Computer and Communications Security. New York, NY: ACM; 2009. p. 635–47 <https://iseclab.org/papers/yourbotnet.pdf>.
Takanen A, DeMott J, Miller C. Fuzzing for software security testing and quality assurance. <http://www.artechhouse.com/uploads/public/documents/chapters/Takanen-214_CH01.pdf>; 2008 [accessed Nov 2015].
The Economist. Cyber–attack in the Czech Republic, 13 March. <http://www.economist.com/blogs/easternapproaches/2013/03/cyber-attack-czech-republic>; 2013 [accessed Nov 2015].
Tsunoda H, Ohta K, Yamamoto A, Ansari N, Waizumi Y, Nemoto Y. Detecting DRDoS attacks by a simple response packet confirmation mechanism. Comput Commun 2008;31(14):3299–306. doi:10.1016/j.comcom.2008.05.033.
Tung L. Five percent of Web traffic caused by DDoS attacks. <http://www.blockdos.net/five_percent_of_web_traffic_caused_by_ddos_attacks.html>; 2010 (retrieved 0.4.05.13.).
Walfish M, Vutukuru M, Balakrishnan H, Karger D, Shenker S. DDoS defense by offense. ACM Trans Comput Syst 2013;28(1):2. <http://dspace.mit.edu/openaccess-disseminate/1721.1/72325>.
Wang H, Zhang D, Shin KG. Detecting SYN flooding attacks. In: INFOCOM 2002. Twenty-first Annual Joint Conference of the IEEE Computer and Communications Societies. New York, NY: 2002 <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1019404>.
Wang H, Jin C, Shin KG. Defense against spoofed IP traffic using hop–count filtering. IEEE ACM Trans Netw 2007;15(1):40–53. doi:10.1109/TNET.2006.890133.
WebSense. 2013 – threat report, p. 27. <http://www.websense.com/assets/reports/websense-2013-threat-report.pdf>; 2013 [accessed Nov 2015].
Wool A. Firewall configuration errors revisited, 17 pp. 2009.
Yau DKY, Lui JCS, Liang F, Yam Y. Defending against distributed denial-of-service router throttles. IEEE ACM Trans Netw 2005;13(1):29–42. <ftp://mail.im.tku.edu.tw/助教/bearhero/TON_ddos.pdf>.
Ye X, Ye Y. A practical mechanism to counteract DNS amplification DDoS attacks. J Comput Inf Syst 2013;9(1):265–72. <http://www.jofcis.com/publishedpapers/2013_9_1_265_272.pdf>.
Zhang M, Zhang W, Fan K. Application layer DDoS detection model. In: Communications and information processing. 2012. p. 37–45 <http://link.springer.com/chapter/10.1007%2F978-3-642-31968-6_5>.
Zoller T. TLS/SSLv3 renegotiation vulnerability explained. <http://www.g-sec.lu/practicaltls.pdf>; 2011 [accessed Nov 2015].

Bill Buchanan is a Professor in the School of Computing at Edinburgh Napier University, and a Fellow of the BCS and the IET. He currently leads the Centre for Distributed Computing, Networks, and Security, and works in the areas of security, Cloud Security, Web-based infrastructures, e-Crime, cryptography, triage, intrusion detection systems, digital forensics, mobile computing, agent-based systems, and security risk. Bill has one of the most extensive academic sites in the world, and is involved in many areas of novel research and teaching in computing. He has published over 27 academic books, and over 200 academic research papers, along with several awards for excellence in knowledge transfer, and for teaching, such as winning at the I ♥ my Tutor Awards (Student voted), Edinburgh Napier University, 2011 and 2014, and has supervised many award winning student projects.

Boris is a security researcher and a systems engineer working for a global security company. His work experience includes working in both the academic and commercial sphere. In his current role, he has been instrumental in increasing the security of a large network infrastructure deployment. He specializes in large systems security from both defensive and offensive perspective as well as large scale server maintenance. He has been twice selected for Google summer stipend programme where he worked on the implementation of statistical and mathematical algorithms. Boris was awarded with best Master's project and outstanding academic achievements. He has developed a new scalable mechanism to update firmwares on Dell servers. Boris holds first class degrees in Computer Networks (Mobile Computing) and Advance Security & Digital Forensics.

Rich is a Lecturer in the School of Computing at Edinburgh Napier University. His teaching role has involved lecturing across a range of subject areas including Computer and Network Security, Digital Forensics, and Software Development at both undergraduate and postgraduate level.

He has had a key role in the creation and development of the MSc in Security and Digital Forensics at Edinburgh Napier over the past five years, specifically the creation of the Network Security, e-Security, and Penetration Testing modules with input from industry partners, all using cutting edge methods and technologies. He led the project to create custom laboratories for teaching Security and Digital Forensics at Edinburgh Napier, the setting up of an Academy for EnCase Forensics, and was heavily involved in the development of a cloud-based virtualisation platform for the teaching and training of Security to both on-campus and distance learning students. Rich leads the undergraduate Security and Forensics Programme and has developed and teaches on many of the modules in the Security and Forensics group. Specialist topics include: Computer and Network Security, Network Forensics, Security Testing, Wireless Security, Databases, Cloud, and Software Development for Security and Forensics.