

HAKING

PRACTICAL PROTECTION

IT SECURITY MAGAZINE

Vol.9 No.03
Issue 03/2014(72) ISSN: 1733-7186

DDOS ATTACKS AND PREVENTION TUTORIALS

LAYER 7: APPLICATION LEVEL DDOS

**SUBVERTING BIND'S SRTT ALGORITHM
DERANDOMIZING NS SELECTION**

CLOUDBASED DDOS PROTECTION SERVICES





cigital
SecureAssist™



Find and Fix Security Defects During Development

Plug-in for Eclipse and Visual Studio identifies common security vulnerabilities and provides remediation guidance

Expert validated

Based on Cigital's experience in thousands of code reviews

Contextual

Guidance and examples specific to the language

Actionable

Code examples explain the right way and place to fix defects

Customizable

Incorporate organizational standards into guidance



Free 30-day trial: www.cigital.com/hakin9

Securing Assets Across Europe

Europe's leading independent, interdisciplinary security conference and exhibition

Over the past decade, Information Security Solutions Europe (ISSE) has built an unrivalled reputation for its world-class, interdisciplinary approach and independent perspective on the e-security market.

This year, ISSE will take place on 14th & 15th October in Brussels. Regularly attracting over 300 professionals including government, commercial end-users and industry experts who will come together for a unique all-encompassing opportunity to learn, share and discuss the latest developments in e-security and identity management.

Programme Topic Areas

- **Trust Services, eID and Cloud Security**
European trust services and eidentity regulation, governance rules, standardization, interoperability of services and applications, architectures in the cloud, governance, risks, migration issues
- **BYOD and Mobile Security**
Processes and technologies for managing BYOD programs, smartphone/tablet security, mobile malware, application threats
- **Cybersecurity, Cybercrime, Critical Infrastructures**
Attacks & countermeasures against industrial Infrastructures; CERT/CSIRT – European & global developments, resilience of networks & services, surveillance techniques & analytics
- **Security Management, CISO Inside**
CISOs featuring the latest trends and issues in information security, risk mitigation, compliance & governance; policy, planning and emerging areas of enterprise security architecture
- **Privacy, Data Protection, Human Factors**
Issues in big data & cloud, privacy enhancing technologies, insider threats, social networking/engineering and security awareness programs
- **Regulation & Policies**
Governmental cybersecurity strategies, authentication, authorization & accounting, governance, risk & compliance

In partnership with



DDoS Attacks and Prevention Tutorials

Copyright © 2014 Hakin9 Media Sp. z o.o. SK

Table of Contents

DDoS and The Internet

by Antonio Ieranò

08

There is always a lot to say about security and The Internet. A different form of an attack or a threat comes out to the public attention and is overexposed by media and vendor marketing. One truth we should always remember is that there are a lot of different attacks and, from time to time, one or another rises up or peaks due to several circumstances: political, environmental, technological or economical. There will be always a bunch of different technologies misused to perform the attack on the net...

Cloud-Based DDoS Protection Services

by Azi Ronen

15

In recent years Distributed Denial of Service (DDoS) attacks have become a mainstream threat to businesses, governmental agencies and critical infrastructure worldwide. DDoS attacks have grown in complexity, volume and sophistication. In a recent survey 65 percent of IT security practitioners reported experiencing an average of three DDoS attacks in the past 12 months...

Choosing a DDoS Protection Service

by Michael Lemire

19

In the early days of the Internet, DOS attacks were simpler. Protocol manipulation attacks such as Ping flood, TCP Syn flood and the so called Smurf Attack were designed to overwhelm victim hosts TCP stacks and bandwidth in order to prevent them from serving legitimate requests. Network vendors fought back and began to provide basic DOS protection against these most common protocol manipulation attacks. Border routers (routers which sit between your infrastructure and the Internet) provide access control lists (ACLs) which can be configured to drop unneeded protocols. Packet inspection firewall vendors added protocol anomaly protection capabilities. Configuring your router and firewall's basic DOS protection against these types of protocol anomalies can be considered baseline protection...

Subverting BIND's SRTT Algorithm Derandomizing NS Selection

by Nakibly Gabi, Roe Hay, Jonathan Kalechstein

21

We begin by describing the basics of the DNS protocol. We continue with a survey of known attacks on DNS, and nalize with a genuine, deterministic attack against BIND's SRTT (Smoothed Round Trip Time) algorithm. Our method enables derandomization of the target name server thus reduces the expected time of DNS cache poisoning attacks...

Layer 7: Application Level DDoS

43

Typically, a Denial of Service (DoS) condition occurs when a server or network resource is unable to service legitimate requests made to it, and therefore unable to perform a function it was designed to. DoS attacks have been around for some time, with the earliest attacks being dated to the first half of 1970's. This type of attack started out as an avenue for hackers to establish status in underground communities. However, these have evolved into far more sophisticated and dangerous forms that are directed at specific targets for a number of reasons, not excluding cyber-terrorism, corporate rivalry, hacktivism and even exhortation...

Tackling Layer 7 DDoS Attacks

by Ratan Jyoti

49

Distributed Denial of Service (DDoS) attack is a strenuous challenge where the spurious or fake packets are sent to the victim in abnormally large number. DDoS attempts to block important services running on victim's server by flooding the victim's server with packets. The difference with DoS is that DDoS is that the attacks do not originate from a single host or network but from multiple hosts or networks which might have already been compromised. One of the main challenges here is to find the location of attackers and then block traffic at points near to the source of the attacks. Layer 7 DDoS attacks target the application layer at web or mail servers (eg. HTTP(S), SMTP, FTP etc) such that the service can be denied in effective way to bring web server to lock up or crash. Since they operate at the application protocol level which is OSI Layer 7, this attack is known as Layer 7 DDoS attack...

DDoS Attacks and Defense

by Rodrigo Salvalagio, Eder Plansky Silva

53

Denial of Service Attacks exists in decades, but frequency, extension and sophistication are evolving faster than companies can absorb them. This article shows both sides: how attackers are orchestrating and how companies are protecting themselves...

DDoS Attack You Could Be Attacked Right Now. Are you Prepared?

by Abdy Martinez

64

A DDoS attack could happen at any moment. Even if you have a powerful and well-configured firewall, updated anti-virus and anti-spam or good security practices, depending on the mode of DDoS attack, you will be affected. Trust me! So, are you ready to confront a DDoS attack? No? Please, before you start reading this article, ask your ISP a quote for DDoS protection service...

Protection Against DDoS on the Cloud

by Ahmed Fawzy

68

Simply the denial of service (DoS) attempts to deny legitimate users to the cloud service, in the (DDoS) the same matter happens but the attack lunched through thousands of zombies or fake packets may led to SLA violation, loses in revenue, lost productivity...

A Simple SYN. Distributed Denial of Service (DDoS)

by Donald Gooden

71

DDOS: as defined by Wikipedia: is an attempt to make a machine or network resource unavailable to its intended users. This method is one of the oldest ways to hit a system (...) this method works on so many types of systems (...) web pages, apps, internet connections, smart phones, dumb phones, old phones, with sooooo many different ways to make this happen (...) DDOS the distributed part is the piece that makes the difference when it comes to the networking aspect of things...distributed... distributed ...a denial of service...I'm unable to get to my online banking page, my email isn't working today, I can't log into my system from home, from my cell phone, from my desk, from my...from my...from my...these are single system service denials... just single... Wikipedia: as clarification, DDOS attacks are sent by two or more person, or bots. Denial of Service (DOS) attacks are sent by one person or system...

Dear Hakin9 Readers!

Let us present our latest issue entitled DDoS Attacks and Protection. Inside, you will find a few interesting tutorials that will help you develop your skills. Our experts prepared 10 articles in which they aim to familiarize you with various attacks and defence techniques.

We hope you enjoy the issue.

Krzysztof Samborski
and Hakin9 team



Editor in Chief: Ewa Dudzic
ewa.dudzic.@hakin9.org

Managing Editor: Krzysztof Samborski
krzysztof.samborski@hakin9.org

Editorial Advisory Board: David Kosorok, Matias N. Sliafertas, Gyndine, Gilles Lami, Amit Chugh, Sandesh Kumar, Trish Hullings

Special thanks to our Beta testers and Proofreaders who helped us with this issue. Our magazine would not exist without your assistance and expertise.

Publisher: Paweł Marciniak

CEO: Ewa Dudzic
ewa.dudzic.@hakin9.org

Marketing Director: Krzysztof Samborski
krzysztof.samborski@hakin9.org

Art. Director: Ireneusz Pogroszewski
ireneusz.pogroszewski@hakin9.org
DTP: Ireneusz Pogroszewski

Publisher: Hakin9 Media sp. z o.o. SK
02-676 Warszawa, ul. Postępu 17D
NIP 95123253396
www.hakin9.org/en

Whilst every effort has been made to ensure the highest quality of the magazine, the editors make no warranty, expressed or implied, concerning the results of the content's usage. All trademarks presented in the magazine were used for informative purposes only.

All rights to trademarks presented in the magazine are reserved by the companies which own them.

DISCLAIMER!

The techniques described in our magazine may be used in private, local networks only. The editors hold no responsibility for the misuse of the techniques presented or any data loss.



[GEEKED AT BIRTH]



You can talk the talk.
Can you walk the walk?

[IT'S IN YOUR DNA]

LEARN:

Advancing Computer Science
Artificial Life Programming
Digital Media
Digital Video
Enterprise Software Development
Game Art and Animation
Game Design
Game Programming
Human-Computer Interaction
Network Engineering
Network Security
Open Source Technologies
Robotics and Embedded Systems
Serious Game and Simulation
Strategic Technology Development
Technology Forensics
Technology Product Design
Technology Studies
Virtual Modeling and Design
Web and Social Media Technologies

www.uat.edu > 877.UAT.GEEK

Please see www.uat.edu/fastfacts for the latest information about degree program performance, placement and costs.

DDoS and The Internet

by Antonio Ieranò

Around security and Internet there is always a lot of talk, from time to time different kind of attack or threats comes out to the public attention and are overexposed by media and vendor marketing.

One truth we should always remember is that there are a lot of different attacks and, from time to time, one or another rises or peak due to several circumstances: political, environmental, technological or economical, there will always be a turnover of different technologies misused to perform an attack on the net.

One classic form of attack that has been on the news for a while, but then actually never stopped to exist, is the so called Denial of Service.

What is a denial of service?

From Wikipedia:

In computing, a denial-of-service (DoS) or distributed denial-of-service (DDoS) attack is an attempt to make a machine or network resource unavailable to its intended users. Although the means to carry out, motives for, and targets of a DoS attack may vary, it generally consists of efforts to temporarily or indefinitely interrupt or suspend services of a host connected to the Internet. As a clarification, DDoS (Distributed Denial of Service) attacks are sent by two or more persons, or bots. (See botnet) DoS (Denial of Service) attacks are sent by one person or system.

In other words, a Denial of Service is a kind of attack that purpose is to stop, slow-down or somehow damage a service provided by someone in the network.

Denial-of-service attacks are considered violations of the Internet Architecture Board's Internet proper use policy, and violate the acceptable use policies of virtually all Internet service providers. They also commonly constitute violations of the laws of individual nations.

A DDoS (distributed denial of service) or a Dos (denial of service) are not related to specific technologies or process. Any hacking technique could be used to DDoS or dos a target, and sometimes we can have the same result using simple legal technique, or just because of misconfigurations.

To have a better understanding of what is a ddos attack we should first of all start from the very basic:

A dos attack aim is to damage (stop, slowdown..) a service

So as a first instance we should have a service running somewhere and someone else willing to attack it.

I will not take in account configuration errors or hardware crash since we are trying to understand the voluntary will to stop someone else service.

The situation is more or less the following:

Services running

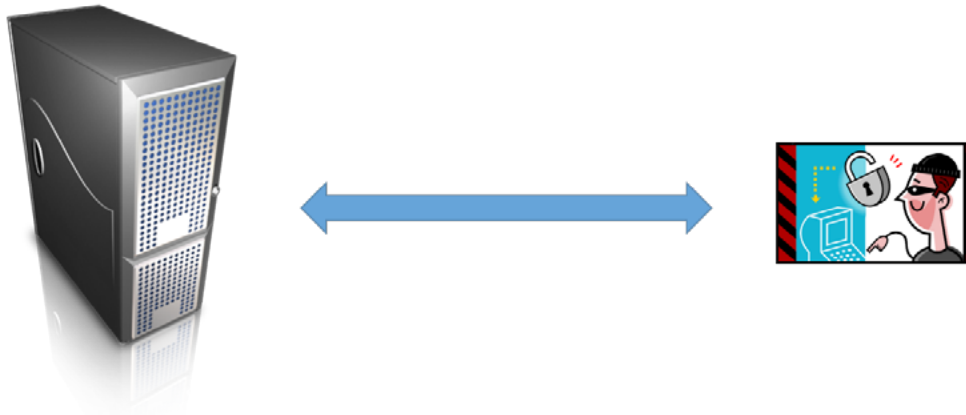


Figure 1. Service running and the hacker

We have a service running somewhere and a hacker trying to stop (or....) the service.

In order to be able to perform this operation the hacker can target different areas of the process that allow the service to run:

Services running exposures

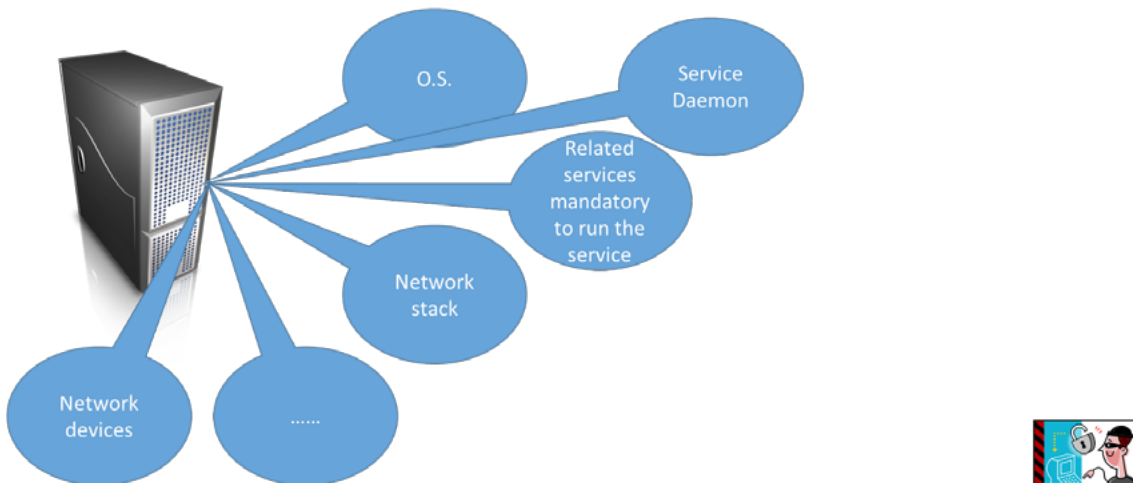


Figure 2. Service running exposures

Some of those attacking surface are common to any service running others can be service specific. So we will always have an Operating System running while a Database running is a need only for specific services.

- Every attacking area can be target to perform a Dos attack, so, as an example to stop a ecommerce service we could run different operations:
- We could try to hack the OS, and, as an example, escalate the administration rights in order to stop the service itself

- We could use a vulnerability related to the service itself to make it crash with, again a simple and understandable example, a memory buffer overflow.
- We could otherwise target a related service that is essential to the main objective function, think of targeting the database that is holding the information running on a wordpress site.
- But we can simply try to saturate the IP bandwidth of the server running the service in order to stop it,
- Or we can try to modify the network path of the service itself

And so on.

The result will always be the same; the service will stop or slow down.

In terms of what I can do to obtain the service denial I can:

- Try to saturate the resources related to that service
- Try to operate on the configuration parameters of something related to the service.

Resource saturation (Starvation Attacks)

When I try to perform a resource saturation, I need to exceed the computational or network resources of a specific sub target.

This kind of attack is quite common, and usually require that more point of attack works together in order to saturate the target providing a number of excessive request and or specifically crafted ones that overload the processing capability.

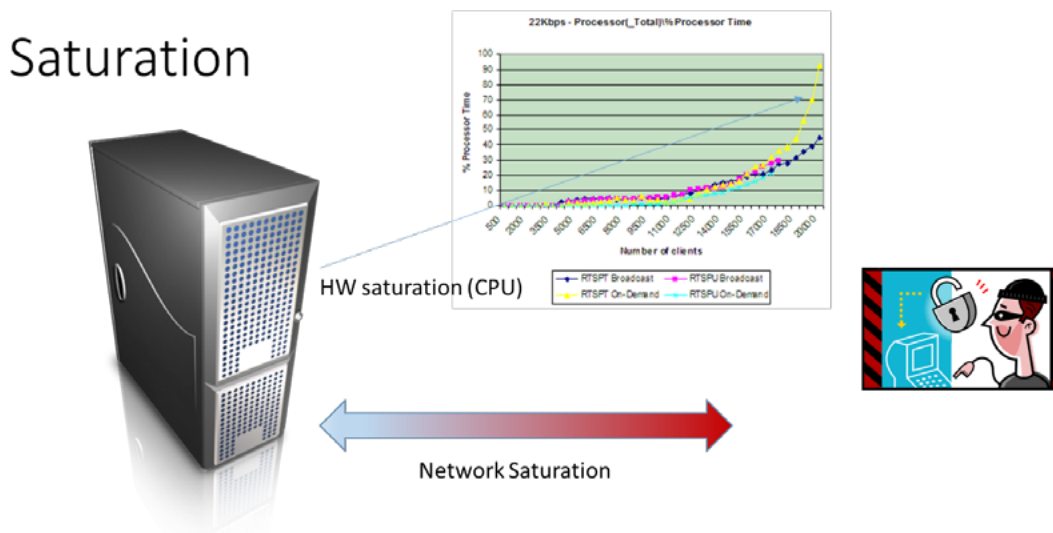


Figure 3. Resource Saturation

Normal target for that kind of operation are CPU, Disk and Network.

It is interesting to notice that a resource saturation can be effective even if performed on the target not directly related to the service we want to stop.

This means that the exposed area for a Dos is extremely Wide.

We can start considering what happen when we want to saturate the HW resources: as disk I/O or CPU.

One simple way to obtain this kind of result is just overloading the service with requests. Any request that need an answer will use CPU and disk resources, and since resources are limited the amount of instances that can be processed have a physical limit.

If we are able to exceed this limit we will be able to perform a system stop, at least till the system will be able to answer to all requests in the queue.

This simple DDoS attack have a great advantage, since it can use legit requests in order to obtain the needed hack.

This is typically the case of DDoS (Distributed denial of service) which structure is, at its basic, very simple:

An attacker somehow takes control of a set of computer that are used to perform the attack. This is usually the botnet environment.

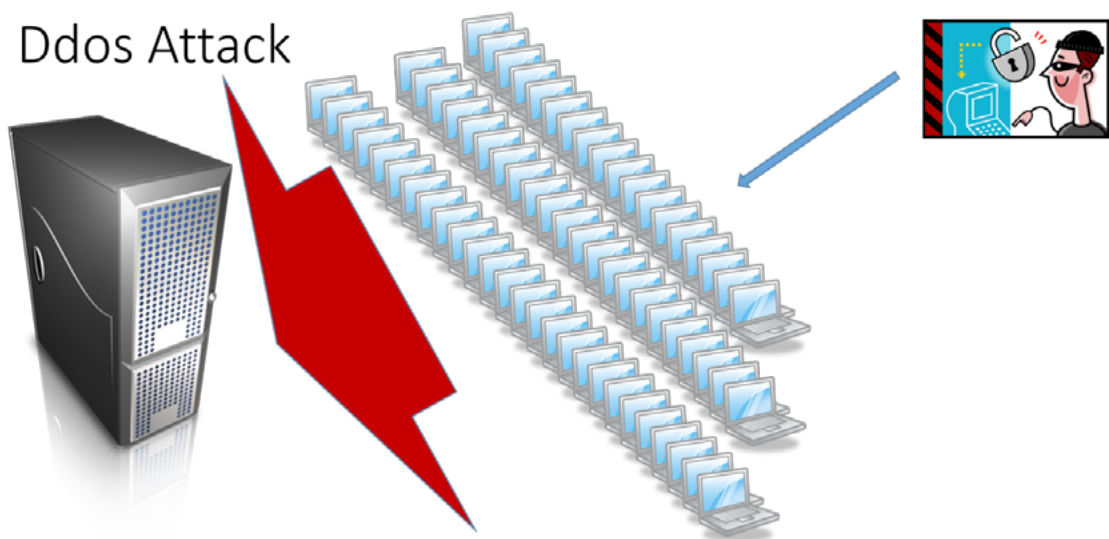


Figure 4. DDoS attack, the attacker use a multitude of computer to target the service

We should be aware that is particularly difficult to understand if we are in the presence of a DDoS attack or simply we have reached the limit of our configuration since this attack can be performed simply using a standard service request, the same request used by a usual user of the service itself.

We can understand we are in the presence of an attack when:

- It comes in a specific timeframe not related with normal operation
- It is a one-time event
- The request comes from unusual geographical location
- There are not systems that are exceeding resource consumption due to some scheduled or exceptional activity

In other words, we should be aware of all the resources and their usual baseline, the origin of the traffic we receive and the kind of resources used by the surrounding services.

A DDoS can be performed not only by Botnet, but is also a typical activism [1] manifestation, several campaigns or Anonymous and the other groups are able to move a great number of members that can just start visiting a specific site all together blocking it.

It is easier to understand we are in presence of a DDoS when the kind of traffic is unusual or forged/manipulated. As an example, an excess of ping can be a clear symptom of an incumbent DDoS attack.

In literature, we can find thousands of different kinds of attack that can provoke a DoS, most of them are network related, but the always present SQL injection is another classic example.

Typical Network attack include:

- Internet Control Message Protocol (ICMP) flood
- (S)SYN flood
- Teardrop attacks
- Peer-to-peer attacks
- Nuke

But we can find the same kind of attack (flood, spoofing) ad any layer and even against the service itself.

Think of a service that have to process your request, if your request is complicated enough could bring it to halt or slow so from extremely complex regular expressions, to handling http request for an enormous amount of time anything ca be used.

Sometimes a simple SQL injection request or a simple buffer overflow are enough to consume all disk I/O or CPU resource

A particularly easy way to perform a DDoS attack is, last but not least, not to attack the service itself but to prevent users to reach the service.

In this situation, the attacker wants to isolate the server providing the service targeting a key element in the chain that is used to reach the service itself.

Besides the obvious target of router or switch [2], a classical victim is the Name Server Structure. Attacking a DNS is quite easy and most effective due to the fact that most of the DNS running on the internet are poorly protected, heavily exposed and bad managed. This is a classical “Achilles’ Heel” ☺.

Configuration attack

Another way to perform a Dos Attack is to conveniently hack the host and escalate credentials in order to take control of the system. Most of the time it is of no need to be the administrator to perform those kinds of attack, some sort of a power user are enough to modify any specific flag that can cause the damage. Of course, that kind of attack requires a specific set of hacking knowledge in order to penetrate the platform and escalate the credentials conveniently.

Bug, Backdoors, security holes everything can be used.

When we think about ddos related to configuration modification we should extend our analysis to all the network surrounding.

Taking control of a router or a switch can be extremely useful to perform a DDoS attack, modifying Routing Map or QoS configuration, altering ACL are all techniques that can be used to obtain the result.

Why, Who, What and When?

Why?

One question can rise, why anyone should perform a DDoS attack?

There are several reasons to perform a DDoS or a DoS attack, some are evident some others can be more tricky.

The most evident is that someone wants to stop a specific service, this can be done, at least, for 3 reasons:

- Political or Activist reasons, to have visibility or to demonstrate a specific idea.
- Retaliation, blocking a service and asking money to restore the functionality.
- Negative marketing, to give a bad feeling of the service provided (they do not run...)

But a DDoS attack can be part of something more complex, as an APT. typically in those situations the DoS attack is used to

- Mislead attention to something that is not the real target
- Cover track of activities
- As a needed part of the APT because the reaction at the attack will let the attacker to penetrate the system.

Both be the final activity or part of a more complex attack the DoS result effective targeting a quite various set of areas. To give an indication on what could be the target for a DoS to a service is a hard exercise that requires a complete analysis of the network and all the interaction between the service involved and the surrounding. Not all the attacks need to be performed in the perimeter where the service resides, in a DNS attack the target could be the provider or even the root for that specific domain. As a result, a single technology that can help us against DoS and DDoS does not exist, but a set of technologies and procedures that runs from firewalls and IPS to dedicated DDoS technologies (that usually target a portion of the network exposure area) and may be some reputation services to analyze the origin of the request.

Who?

We sometimes misunderstand the extension and the depth of the DDoS phenomenon. One of the main reasons is that there is not a correct perception of what a DDoS can do and who can cause it.

Due to the extreme variety of DDoS technicality and effect there are plenty of subjects that can perform a DDoS attack including people without specific knowledge, since DDoS tools are available on the internet, and some (think of the Low Orbit Ion Cannon) of public domain and easily available.

In the end any traffic generator or stress test tool can be used to perform a basic DDoS attack, but there are also sales kits on the internet (the dark one, of course) and even "DDoS as a Service."

So it is not only a matter of hacker or activism (we all heard about Joker, Lulz, Anonymous, Iranian cyber army), but also criminality and even governments use this kind of techniques, the Stuxnet affair was related to government (Israel? USA? Both?) vs. government (Iran) using a DDoS weaponized software (SCADA controller) through malware (Stuxnet).

What?

Anything can be the target of a DDoS, anything that provides a service on a network is a good subject, so don't think there are areas that are "secure" or "safe". The question is if this service is meaningful for someone that can be targeted or wants to target someone else?

Even training could be a good reason, or just a demonstration to rise the level of awareness or just create a white rumor on the background to cover real intention.

When?

Any time is good, the evolving political, economic and technological situation create moment by moment, a world of “good” reason for a DDoS. So Do not ask yourself “if” but “when”

References

[1] Richard Stallman has stated that DoS is a form of ‘Internet Street Protests’

[2] May be someone remember the old classical Broadcast Storms, not always related to misconfiguration of switches...

About the Author



Antonio Ieranò is an IT professional, marketing specialist, and tech evangelist with over 16 years of experience serving as a community liaison, subject matter expert, and high-profile trainer for key technologies and solutions. Mr Ieranò's experience includes acting as the public face of Cisco security technologies; leading pan-European technical teams in development of new Cisco security products; and serving as a key public speaker and trainer on behalf of new high-tech products. His expertise spans IT development and implementation, marketing strategy, legal issues, and budget / financial management.

advertisement

An advertisement for CompuSleuth. The background is a dark blue space with a glowing globe of binary code (0s and 1s) on the left. A bright blue beam of light emanates from the globe towards a computer monitor on the right. The monitor shows a dark screen with some faint, illegible text. The overall aesthetic is futuristic and tech-oriented.

COMPUSLEUTH
Discovering Data One Byte at a Time

www.CompuSleuth.com
1-614-898-7500

Cloud-Based DDoS Protection Services

by Azi Ronen, Chief Strategy Officer, SecurityDAM

In recent years Distributed Denial of Service (DDoS) attacks have become a mainstream threat to businesses, governmental agencies and critical infrastructure worldwide. DDoS attacks have grown in complexity, volume and sophistication. In a recent survey 65 percent of IT security practitioners reported experiencing an average of three DDoS attacks in the past 12 months [1].

With an average downtime of 54 minutes per attack and the cost amounting to as much as \$100,000 per minute – it would have been expected that organizations put into practice preventative measures to protect their networks and business. However, this is far from being the case.

Many organizations do not employ any DDoS protection at all. Others rely on ISP solutions or use on-premises equipment, which at best can deflect a single type of attack. However, such solutions fail to provide adequate protection against multi-level attacks, and they lack the expertise to handle new types of attacks.

To ensure business continuity and provide solid DDoS protection, a different, multi-layer approach is needed – and such approach presents a new service opportunity for providers of managed security services (MSSPs).

Distributed Denial of Service attacks can be broadly categorized into two types:

- Volumetric attacks flood the victim with high volume of packets or IP flows, consuming network equipment and bandwidth resources. Some examples include SYN flood attacks (high packet-per-second attacks), large UDP packet floods (bandwidth attacks), and ICMP floods.
- Application attacks, also known as “low and slow” attacks, directly attack applications, servers of specific services, exploiting implementation weaknesses and design flaws. Some examples include HTTP Get or Post flood attacks, DNS flood attacks and SSL flood attacks.

Radware Security Survey:

Which services or network elements are (or have been the bottleneck) of DoS?

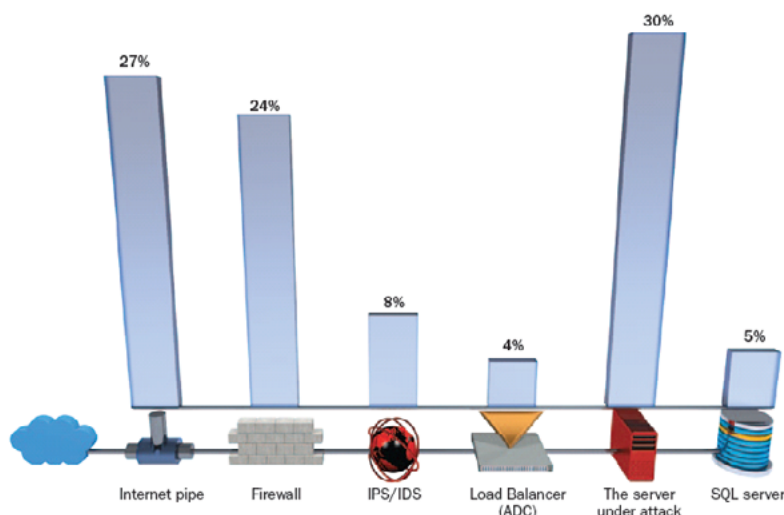


Figure 1. Radware Security Survey

As can be seen in Figure 1, 27% of the DDoS attack saturate the victim internet link, using a volumetric attack. Such attacks cannot be mitigated by any local device.

The Hybrid Approach for DDoS prevention Services

In order to get a full protection against all types of DDoS attacks, a multi-layer solution is required. The solution is composed of the following main components:

- CPE (Customer-premises equipment) is a detection and signaling device placed at the edge of the customer's data center. Constantly monitoring network traffic, the CPE learns the traffic patterns to establish a normal behavior baseline. It detects anomalies and DDoS attacks early on, mitigates application attacks (1 in Figure 2 below) and alerts the MSSP when the attacks are too large and saturate the enterprise access link.
- Scrubbing Centers, a cloud-based facility, manned by an emergency response team to ensure the fastest analysis and resolution of new attack types. When the network is under volumetric DDOS attack, traffic is redirected (2 in Figure 2 below) to the scrubbing center for attack mitigation. After filtering, clean traffic is passed back to its original destination using GRE tunnels (3 in Figure 2 below). Attack data is collected and stored, enabling real-time monitoring and historical reporting.
- A Customer's Portal, usually a web-based portal that provides real-time insight into events, attack characteristics, post-attack reports and statistics to the customers of the service.

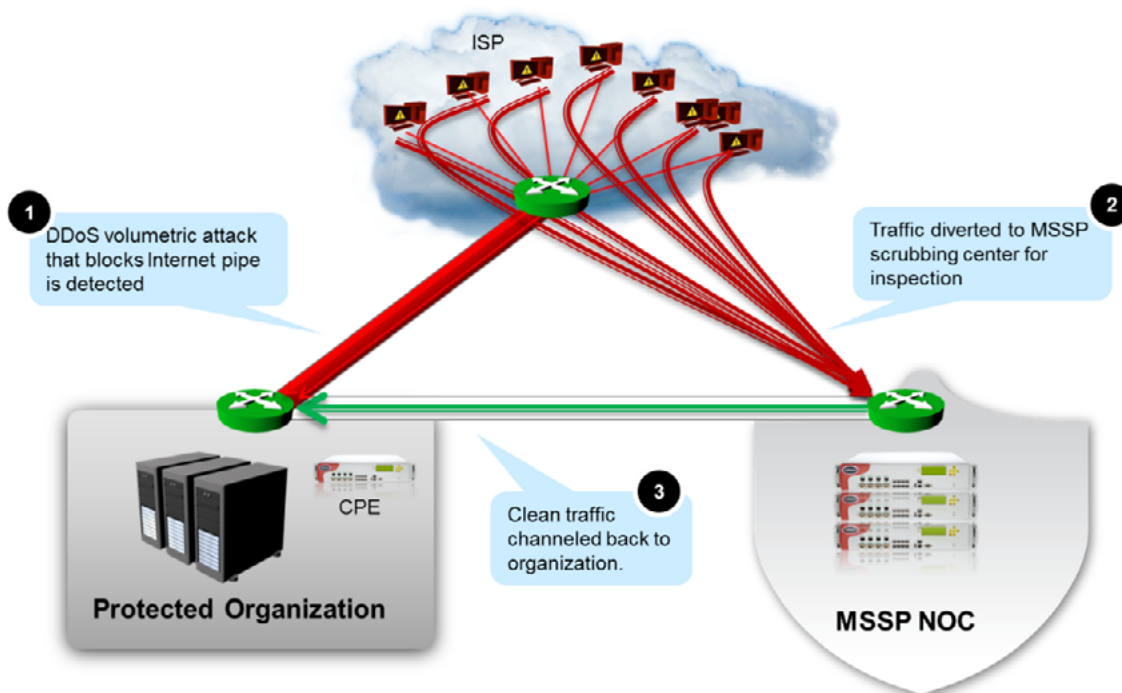


Figure 2. Network under volumetric DDOS attack

The Business Opportunity for MSSPs

Providing cloud-based DDoS protection services provides a unique business opportunity to MSSPs. Local solutions deployed by enterprises at the data center cannot handle volumetric attacks and require the use of on-demand, cloud service that will be able to mitigate high-volume attacks that sometimes reach a volume higher than 100 GBPS.

A recent report by Infonetics Research [2] concludes that the “global cloud and CPE managed security service market grew another 12% in 2012, to \$13 billion. While the majority of security service revenue in 2012 came from CPE-based services, by 2017 CPE revenue is expected to dip to 50% of total revenue. Infonetics forecasts sales of cloud-based security services to grow 69% over the next 5 years. “Other security services,” of which hosted DDoS services are a major and growing contributor, are anticipated to comprise over 20% of cloud-based security service revenue by 2017.”

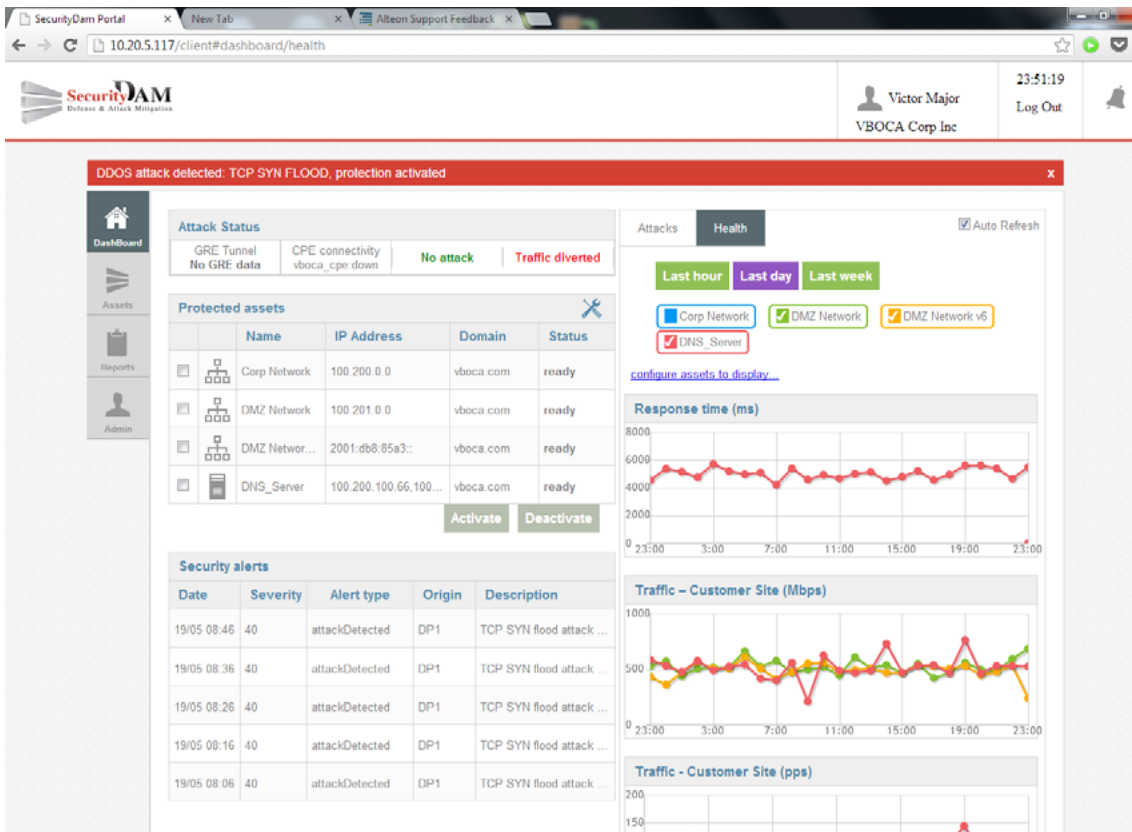


Figure 3. Customers' Portal – the Dashboard

It is the right time for MSSPs to join the growing business of DDoS prevention services. SecurityDAM was established to enable the service infrastructure for MSSPs in the shortest time to market, based on state-of-the-art DDoS prevention and service management technology, and team of security experts providing the best service to customers.

References

- [1] The research for Cyber Security on the Offense: A Study of IT Security Experts (http://security.radware.com/uploadedFiles/Resources_and_Content/Attack_Tools/CyberSecurityontheOffense.pdf), November 2012, by the Ponemon Institute and Radware.
- [2] Managed security services top \$13 billion in 2012 (<http://www.infonetics.com/pr/2013/2H12-Cloud-and-CPE-Managed-Security-Services-Market-Highlights.asp>); strong growth ahead for cloud security, April 2013, By Infonetics Research

About SecurityDAM

SecurityDAM provides world-class MSSP cloud-based solutions mitigating Distributed Denial of Service (DDoS) attacks on enterprise networks. Founded by a team of security experts, SecurityDAM is a member of the RAD group. For more information, see www.securitydam.com.

About the Author



Azi Ronen has more than 20 years of marketing and product management experience in computer networking and fixed and mobile telecommunications, with strong technical background.

Prior to SecurityDAM, Mr. Ronen served as Executive Vice President of Marketing (6 years) and Corporate Development (4 years) at Allot Communications, a global leader in Broadband Traffic Management. This follows a career as VP Marketing of Vocaltec (VoIP space) and VP R&D/Product management of RADLINX, member of the RAD group. Azi has a B.sc degree, Cum Laude, in Computer Sciences, from the Technion, Israel.

UPDATE
NOW WITH
STIG
AUDITING

“ IN SOME CASES
nipper studio
HAS VIRTUALLY
REMOVED
the **NEED FOR** a
MANUAL AUDIT ”
CISCO SYSTEMS INC.

Titania's award winning Nipper Studio configuration auditing tool is helping security consultants and end-user organizations worldwide improve their network security. Its reports are more detailed than those typically produced by scanners, enabling you to maintain a higher level of vulnerability analysis in the intervals between penetration tests.

Now used in over 45 countries, Nipper Studio provides a thorough, fast & cost effective way to securely audit over 100 different types of network device. The NSA, FBI, DoD & U.S. Treasury already use it, so why not try it for free at www.titania.com



www.titania.com

Choosing a DDoS Protection Service

by Michael Lemire

A Denial of Service (DDoS) attack is an attack which is designed to make a service unavailable. A Distributed Denial of Service (DDoS) attack is a DOS attack amplified by the use of dozens, hundreds or thousands of hosts attacking the same system.

In the early days of the Internet, DOS attacks were simpler. Protocol manipulation attacks such as Ping flood, TCP Syn flood and the so called Smurf Attack were designed to overwhelm victim hosts TCP stacks and bandwidth in order to prevent them from serving legitimate requests. Network vendors fought back and began to provide basic DOS protection against these most common protocol manipulation attacks. Border routers (routers which sit between your infrastructure and the Internet) provide access control lists (ACLs) which can be configured to drop unneeded protocols. Packet inspection firewall vendors added protocol anomaly protection capabilities. Configuring your router and firewall's basic DOS protection against these types of protocol anomalies can be considered baseline protection.

In the late 90's DDoS attacks continued to be developed and volumetric bandwidth attacks appeared. These type of DDoS attack are designed to send an overwhelming amount of seemingly legitimate TCP, UDP or ICMP requests to the victim by hundreds or even thousands of hosts, typically controlled by the attacker via a botnet of computers infected with malware.

In addition to these layer 3 network DDoS attacks; layer 7 application DDoS attacks are also becoming more common. Like volumetric bandwidth attacks, application layer attacks typically send seemingly legitimate HTTP GET or POST requests to a web server. One common example is an attacker sends continuous and repetitive requests for web pages which don't exist on the web server, forcing the web server to generate 404 errors. If enough such requests for non-existent web pages are sent to the victim, the web server processing capability is completely overwhelmed generating 404 errors that it is unable to process legitimate page requests.

Both network and application layer DDoS attacks are becoming more prevalent and effective as the availability of prepackaged tools such as Low Orbit Ion Cannon, Dirt Jumper as well as botnet for hire services have enabled DDoS attacks to become more readily available to attackers. Additionally the rise of hacktivism organizations such as Anonymous have facilitated more coordinated attacks. By all accounts the risk of being a victim of a DDoS attack are increasing and the size and complexity of DDoS attacks continue to grow.

Choosing a DDoS protection service requires both a risk and cost analysis. If you are running a web property which is high profile, provides a service which end users depend on or if downtime will negatively impact your business financially not only should you harden your network devices and servers against basic DOS attacks but you must have contingency plans in place to respond to a large scale DDoS attack.

Types of DDoS prevention services

Fundamentally, DDoS prevention and mitigation services are designed to filter (scrub) DOS attack traffic and allow legitimate traffic before it reaches your network and server infrastructure. It will be important to conduct a cost benefit analysis when choosing a DDoS protection service. Based on the risk to your organization of a DDoS attack, the potential negative impact of a DDoS attack and your budget you may choose to implement DDoS protection on an as needed basis (during an attack) or as a permanent protection.

Cloud-based DDoS protection services are designed to be the front door to your web property. The term cloud-based signifies the vendor's network inspection and scrubbing infrastructure is remote to your data center. The vendor's infrastructure may be in a single or multi-data center topology or, in the case of DDoS protection provided by CDN (content distribution network) vendors like Akamai, Cloudflare, Incapsula and others may be distributed across dozens of data centers globally.

Implementing a cloud-based DDoS protection service is relatively straightforward and is accomplished with DNS. Upon signing up for the service the vendor will provide you with your new IP address; you change the DNS A and/or CNAME records for your web site to the IP address given to you by the vendor. The vendor's inspection and scrubbing infrastructure drops the DDoS traffic and sends the legitimate traffic to your original IP address with the original headers and source IP of your users intact. If your site utilizes SSL/TLS the vendor will need to decrypt the traffic to inspect it so they will need your SSL certificate and its private key to install on their side.

In cases where you are responsible for multiple web sites hosted on several or a subnet of IP addresses in a data center it may make more sense to look at DDoS protection which enables you to redirect an entire subnet to the DDoS protection service infrastructure via BGP. Prolexic (now owned by Akamai) is an example vendor in this space. These types of DDoS protection services may be implemented in an always on or in an as needed / during an attack situation.

In addition to layer 3 network DDoS protection capabilities, inquire with your vendor if they provide layer 7 protection capabilities. Layer 7 protection typically employs the concept of rate limiting – if a client is sending a large amount of HTTP GET or PUT requests to your web server which reach a threshold which is indicative of a DDoS attack – the DDoS protection service should drop that traffic before it ever reaches your web server.

Summary

As DDoS attacks increase in size and complexity it is increasingly important that system and network administrators beef up their defenses. While system hardening and perimeter protection should be implemented, these measures won't keep your site online if you become a target of a large scale DDoS attack. For this reason a DDoS protection vendor should be engaged. Whether you choose to engage before or during a DDoS attack is based on your own risk assessment and budget.

About the Author



Mike is the Information Security Officer for Pearson Higher Education, the world's leading education company. Previous to Pearson Mike created the Information Security and Compliance program at 2 fast growing startups – Acquia and RiskMetrics. Mike has held technical and management positions at MSCI, JPMorgan, Moore Capital Management and Time, Inc. Mike earned his B.S. from New York Institute of Technology and has attended postgraduate education at Columbia and Boston University. Mike was certified as a CISSP in 2006 and CCSK in 2013. Twitter: @mike_lemire

Subverting BIND's SRTT Algorithm

Derandomizing NS Selection

by **Roe Hay, Jonathan Kalechstein and Gabi Nakibly, Ph.D.**

We begin by describing the basics of the DNS protocol. We continue with a survey of known attacks on DNS, and finalize with a genuine, deterministic attack against BIND's SRTT (Smoothed Round Trip Time) algorithm. Our method enables derandomization of the target name server thus reduces the expected time of DNS cache poisoning attacks.

The catalyst for creating the domain system was the rapid growth of the Internet. Before DNS, host name to address mappings were transferred in a single, hosts file, using FTP. This scheme couldn't handle the growth of the number of hosts. In addition, the network population moved from timeshared hosts to local networks of workstations, and an hierarchical administration was needed. Moreover, the internet applications were getting more and more sophisticated so a general name service was needed.

The DNS design goals

There are several design goals:

1. The primary goal is a namespace that will refer to resources. For generalization purposes, names don't have to include identifiers, addresses, routes etc.
2. The rapid changes in size of the database, and the high frequency of updates requires maintaining the database in a distributed manner with local caching. Fetching a consistent copy of the database should be avoided. The same principles should hold for namespaces structure as well.
3. The application should be useful, and not restricted to just one application. Names can be used to retrieve host addresses, mailbox data and to be determined information. All data associated with a name is tagged with a type.
4. Allow the ability to use the same namespace in different protocol families or management.
5. Name server transactions needs to be independent of the system that carries them.
6. Different types of computers (personal, large timeshared hosts) should be able to use the system.

The elements of the DNS

- Domain name space, Resource Records define a tree structure associated with data. Each node and leaf of the tree contains a set of information (resources) which can be extracted using query operations.
- Name servers are programs which hold partial information regarding the domain namespace. In general, a name server has complete information about a subset of the domain space, and has pointers to name servers which can lead to information about other parts of the tree. A name server is called AUTHORITY for a part of a tree, if it has complete information about that part. The authoritative information is formed by units called ZONES, which can be automatically replicated to redundant name servers.
- Resolvers are programs that receive client requests, and forward them to name servers, and vice versa. They must be able to access at least one name server. They answer queries by either directly using the name server information, or indirectly by using referrals to other name servers. The resolver can be implemented as a system routine.

The DNS elements [Moc87a]

Domain name space and resource records

Name space specifications and terminology The domain namespace is in the form of a tree. Each tree node is linked to a resource set (which can be null). Each node has a case-insensitive label, which is up to 63 octets. Brother nodes should not have the same label. In addition, the empty label is reserved to the tree's root. The domain name is the concatenation of labels from the node to the root. The total number of octets that represent a domain name is limited to 255 (this also includes the label lengths). A domain is defined to be a subdomain of another domain if the latter is a suffix of it.

In order to make the DNS applicable, conversion rules must be made from object names (hosts, IP addresses, mailboxes) and domain names. For example, hosts are simply a subset of the text representation for domain names while IP addresses are mapped into the IN-ADDR.ARPA domain.

Resource Records. A domain name which identifies a node, has a set of resource records. Each resource record has several attributes, including:

1. *Type*: A 16 bit value that specifies the type of the resource record.
 - `A` A host address
 - `CNAME` Canonical name of an alias
 - `HINFO` CPU and OS used by a host
 - `MX` Mail exchange for the domain
 - `NS` The authoritative name server for the domain
 - `PTR` A pointer to another part of the domain name space
 - `SOA` The start of a zone of authority
2. *Class*: A 16 bit value that specifies the protocol family or instance of a protocol
 - `IN` The internet system
 - `CH` The Chaos system
3. *TTL*: A 32 bit value that specifies the time to live of the RR. It specifies how long the RR can be cached before it should be deprecated by the resolver. A zero TTL prohibits caching.
4. *RDATA*: A type and sometimes class dependent data
 - `A` For the IN class: a 32 bit address. For the CH class, a domain following by 16 bit Chaos address
 - `CNAME` a domain name
 - `MX` A16 bit preference value, followed by a host name. (d) NS a host name
 - `PTR` a domain name
 - `SOA` a few fields

Canonical names and aliases. The domain name system provides a mechanism that multiple names can identify the same resource. This is done using the CNAME RR, which contains the canonical name in the RDATA section of the RR. If the CNAME RR is present at a node, no other data should be present.

This is done in order to insure that data for the canonical name and its aliases would not be different, and also insures that cached CNAME can be used, with no need to check with the authoritative server for other RR types. If the name server cannot find a requested RR (which is not a CNAME) for a given domain name, it checks if the domain names contain a CNAME. If so, it includes the CNAME in the response and restarts the query with the domain name specified by the CNAME. Domain names in resource records which point at another name, should always point at the primary name and not an alias. For robustness, CNAME chains should be followed, and loops should be signaled as an error.

Queries and responses. Queries are sent to the name server in order to provoke a response. In the internet, they are carried in UDP or TCP. The response includes an answer, a referral, or an error indication. Queries are usually not generated directly by the user, but rather using a resolver. There is a standard specification for the DNS queries and responses format. The message format contains a header with a number of fields which are always present, and sections which include query parameters and RRs. The header contains a four bit opcode field which differentiate between different queries.

The four sections are:

1. *Question.* Query name and parameters
2. *Answer.* RRs which directly answer the query
3. *Authority.* RRs which describe other authoritative servers. May optionally contain the SOA RR for the authoritative data in the Answer section
4. *Additional.* RRs which may be useful for the RRs contained in the other sections.

The defined and not obsolete queries are:

1. *Standard.* Specifies the target domain name (QNAME), type (QTYPE, 16 bits), class (QCLASS, 16 bits) and asks for matching RRs. QTYPE is a superset of the RR types, and also includes AXFR (Zone transfers), MAILB (all mail box related RRs), and '*' (wildcard, matches all RR types). QCLASS is a superset of the RR classes, and also includes '*' which matches all RR classes. Responses for the wildcard QCLASS can never be authoritative.
2. *Inverse.* This is an optional query type, and is used to map resource records into domain names. Since the domain system is organized using domain names, the completeness or uniqueness of inverse queries cannot be guaranteed. Inverse queries should not contain a proper TTL, thus should not be cached.

Name servers

Name servers are basically the place where the domain database is held. The database is divided into sections called zones, which are distributed among the name servers. The essential task of a name server is to answer queries using the zones' data. When a query is issued, the name servers will generate a response based on local information, which might include an answer, or a referral to another name server which may contain the relevant information. A given zone must be available on at least two servers, and is often available by more. The purpose of this is to ensure its reachability in case of a communication failure.

How the database is divided into zones. The domain database can be partitioned in two ways: by class and by cuts made in the namespace. The database for each class is handled separately from other classes. Since the namespaces are identical for all classes, the classes can be thought as an array of parallel trees. Within a class, cuts are made between adjacent nodes. Once all cuts are made, each group of connected nodes is a separate zone. The zone is defined authoritative for all nodes in the connected region. This means that there is no empty zone, and that all nodes in a zone are connected. Because of the tree structure, there is always the top node (closest to the root), which often identifies the zone. It would be possible (but not useful) to create a single zone, or a zone from each node, but usually an area is created where an organization wishes to take control of a sub-tree.

Technical considerations. The zone data has 4 major parts:

1. Authoritative data for all nodes within the zone.
2. Data that defines the top node of the zone.
3. Data that describes delegated sub-zones.
4. Data that allows access to name servers for sub-zones.

All of this data can be stored in RRs, so name servers can easily pass zones by sending RRs. The authoritative data of a zone is all RRs attached to its nodes. The RRs that describe the top node are very important, and are of two types: one that lists the servers for the zone, and a single RR that describes the zone management parameters. The RRs describing cuts at the bottom of the zone are called NS RRs. These RRs name the servers for the sub-zone. These RRs are not part of the authoritative data. Since not all NS RRs hold the address for the sub-zone, zones also keep glue RRs which keep the address of a sub-zone name server, in order to support transition between parent zones and their sub-zones during resolution.

Queries and responses. As explained before, the main activity of name servers is to answer queries. The message format is described below. There are two types of modes in which the name server replies:

1. *Iterative mode*, in which the server answers using only local information. The response should be either an answer, an error, or a referrals to another name server.
2. *Recursive mode*, in which the name server's response is always an error or an answer, but never a referral. The use of recursion mode can happen if and only if both the client and the server agrees to use it. The use is negotiated using two bits in query and response messages.

The algorithm. The algorithm used by the name server depends on the operating system and data structures used to store RRs. The following algorithm assumes the RRs are organized in several tree structures, one per zone, and one for the cache.

1. Set/clear the value of recursion if recursion mode takes place. If recursive service is available go to 5, else go to 2.
2. Search the available zones for the zone which is the nearest ancestor to QNAME. If such a zone is found, go to step 3, otherwise step 4.
3. Start comparing label after label, and finish when:
 - If the QNAME is a full match, we found the node. If the data at the node is a CNAME, and QTYPE doesn't match CNAME, copy the CNAME RR into the answer section of the response, change QNAME to the canonical name in the CNAME RR, and go back to step 1.
 - If a match would take us out of the authoritative data, we have a referral. This happens when we encounter a node with NS RRs marking cuts along the bottom of a zone. Copy the NS RRs for the sub-zone into the authority section of the reply. Put whatever addresses are available into the additional section, using glue RRs if the addresses are not available from authoritative data or the cache. Go to step 4.
 - If at some label, a match is impossible (i.e., the corresponding label does not exist), look to see if the "*" label exists. If the "*" label does not exist, check whether the name we are looking for is the original QNAME in the query, or a name we have followed due to a CNAME. If the name is original, set an authoritative name error in the response and exit. Otherwise just exit. If the "*" label does exist, match RRs at that node against QTYPE. If any match, copy them into the answer section, but set the owner of the RR to be QNAME, and not the node with the "*" label. Go to step 6.
4. Start matching down in the cache. If QNAME is found in the cache, copy all RRs attached to it that match QTYPE into the answer section. If there was no delegation from authoritative data, look for the best one from the cache, and put it in the authority section. Go to step 6.

5. Using the local resolver to answer the query. Store the results, including any intermediate CNAMEs, in the answer section of the response.
6. Using local data only, attempt to add other RRs which may be useful to the additional section of the query. Exit.

Wildcards. Special treatment is given to RRs with owner name starting with label “*”. Such RRs are called wildcards. When appropriate conditions are met, the name server creates RRs with owner name = QNAME. The format of wildcard RRs is not different than any other RR. The owner name of a wildcard is of the form “*.<domain>”, when domain doesn’t have more “*” symbols. The wildcards refer to children of the domain, but not to the domain itself. Wildcard RRs do not apply for the following cases: When the query name is in another zone, or when the query name or a name between the wildcard domain and the query name is known to exist.

Zone maintenance and transfers. As changes happen, it is necessary to spread the information throughout the name servers. This can be done using the zone transfer part of the DNS protocol. When a change occurs, the name server which is the master of the zone, loads the new zone so that other servers of the zone can periodically check it for updates. The other servers can know a change happened since the serial field of the SOA RR advances every time a change is made to the zone. The periodic check by the secondary servers is controlled by 3 parameters: REFRESH, RETRY and EXPIRE. When a new zone is loaded, the server waits REFRESH seconds until the next check. In general, new checks are attempted every REFRESH seconds. When an update is required, the server requests a zone transfer using AXFR request which is usually replied by a series of messages. The messages are basically RRs that allow the receiver to construct a copy of the zone. Servers usually do the last steps against the master, but can use other secondary servers in case that server has better performance or when the master is unavailable.

Resolvers

1. Client-Resolver interface

- (a) *Typical functions.* There are three typical functions: i. Host name to host address resolution, i.e. resolving a hostname into one or more addresses. ii. Host address to host name translation iii. General look up. The resolver may return to the client, one of the RRs with the requested data, a name error if the domain name does not exist, or a data not found error, if the domain name exists but the type does not.
- (b) *Aliases.* During the resolution process, the resolver may find that the QNAME is an alias. If that’s the case, it should restart the resolution with the new name, unless the query type is CNAME. Chains of aliases should be handled correctly and not return an error. Loops should be caught and errors be returned to the client.
- (c) *Temporary failures.* In case of an error which is not a DNS one, e.g. a link failure, the resolver should not signal a name or data not present error to the application.

2. Resolver internals

- (a) *Stub resolvers.* These kinds of resolvers simply pass the query to another name server which supports recursive queries.
- (b) *Resources.* A resolver may also have shared access to zone files of a local name server, in order to reduce the resolutions latency. However, it must not let cached information override zone data.

3. Algorithm. The outline of the resolution algorithm is as follows:

- (a) If the answer is in local information (cache, zone files), return it to the client.
- (b) Find the best servers to query from the SLIST (data structure that contains a list of name servers) (c) Send the queries to the servers, waiting for one response.
- (d) Analyze the response:

- i. In case of an answer or a name error, cache the answer, and return it to the client. ii. In case of a better delegate, cache that information and go to step (b).
- iii. In case of a CNAME that is not an answer itself, change the query to the one found in the CNAME RR, and go to step (a).
- iv. In case of a server failure, delete the server from the SLIST, and go back to step (b).

Step (a) consults the local cache unless specified otherwise by the client. If it has access to the name server's zone files, it should have preference over cached data. Step (b) chooses a locally available NS RR which is closest to the query name. If the A record for the domain name specified by the NS RR is not available, the resolver has multiple choices. For example, it can start a parallel resolution. In case the search for NS RRs returns an empty set, then the list is initialized from a preconfigured list of servers. The list is composed of several root servers (so the resolver has access to the whole namespace) and local servers (in case the network becomes partitioned from the internet). Step (c) sends queries to the servers, until a response is received. The idea is to cycle around the servers, with a timeout. Step (d) involves analyzing the response. The resolver should verify the ID field in the response, matches the request's. In case of a delegation, the resolver should check if the delegation is closer to the query, than the ones found in the SLIST. Otherwise, this indicates a bogus request and should be avoided. The delegation RRs and any address RRs should be cached.

Protocol details [Moc87b]

Introduction

Overview. From the user point of view, domain names are arguments given to the resolver whose job is to return information like the host address. The entire process of distributing data between name servers is transparent to the user. From the resolver point of view, the database is distributed among name servers, and it knows at least one. When a query is received from the user, the resolver asks for the information from the name server which in turn answers, sends a failure message, or gives a referral to another name server. The name servers are responsible for the data and were described above. The name server handles queries from resolvers and repeatedly checks for zone updates, contacting the master name server of a zone.

Character case. From the time [Moc87b] was written, comparisons between strings is case-insensitive, but this was left open for possible future expansions.

Size limits. A few objects in DNS have size limits, while some of them can be changed:

Labels: 63 octets or less.

Names: 255 octets or less.

TTL: positive values of a signed 32 bit number.

UDP messages: 512 octets or less.

Domain name space and RR definition

Name space definitions. A domain name is a sequence of labels. Every domain ends with a null label (the root). Each label is a couple of length and value. The length is limited to 63 octets, so the two most significant bytes of each length value are zero. The total length of a domain name is limited to 255 octets. Labels can contain any 8 bit values, but [Moc87b] recommends using the grammar:

```
<domain> : : = <subdomain> | " "  
<subdomain> : : = <label> | <subdomain> "." <label>  
<label> : : = <letter> [ [ <ldh-str> ] <let-dig> ]  
<ldh-str> : : = <let-dig-hyp> | <let-dig-hyp> <ldh-str>  
<let-dig-hyp> : : = <let-dig> | "--"
```

<let-dig> : : = <letter> | <digit>

<letter> : : = any one of the 52 alphabetic characters A through Z in upper case and a through z in lower case

<digit> : : = any one of the ten digits 0 through 9

Domain names are compared in a case-insensitive manner. Non-alphabetic codes match exactly.

RR definitions

- Format. All RRs contain a header with the following attributes (in the following order):
 - Name (16 bit) The name of the node in the domain namespace.
 - Type (16 bit) The RR type code.
 - Class (16 bit) The RR class code.
 - TTL (32 bit) Signed integer specifies the lifetime of the record. Zero means no caching.
 - RDLenght (16 bit) Specifies the length of the RData field.
 - RData (Variable) A string of octets.

| QType values | QClass values |
|---|---------------------------------|
| A (1) A host address | IN (1) The internet |
| NS (2) An authoritative name server | CH (3) The CHAOS class |
| CNAME (5) The canonical name | HS (4) Hesiod |
| SOA (6) The source of zone authority. | * (255) A request for any class |
| MB (7) A mailbox domain name (experimental) | |
| MG (8) A mail group member (experimental) | |
| MR (9) A mail rename domain name (experimental) | |
| NULL (10) a null resource record (experimental) | |
| WKS (11) A well known service description | |
| PTR (12) A domain name pointer | |
| HINFO (13) Host information | |
| MINFO (14) Mailbox or mail list information | |
| MX (15) Mail exchange | |
| TXT (16) Text strings | |
| AXFR (252) A request for zone transfer | |
| MAILB (253) A request for mailbox records | |
| * (255) A request for all records | |

- Standard RDATA formats. The following definitions can be used in all classes. The NS, SOA, CNAME and PTR have the same format in all classes. A character string consists of a length octet followed by binary information. A domain name is as described above.
 - CNAME CNAME: A domain name
 - HINFO CPU: A character string. OS: A character string
 - MB MADNAME: A domain name that specifies the mailbox
 - MG GNAME: A domain name that specifies the mailbox which is a member of the mail group specified by the domain name

* **MINFO RMAILBX**: A domain name that specifies the mailbox that is responsible for the mailing list or mailbox. If it is the root, the owner of the domain name should be used. **EMAILBX**: A domain name that specifies the mailbox which should receive error messages. If it specifies the root, errors should be returned to the sender.

- **MR: NEWNAME**: A domain name which specifies that mailbox which is the new name of mail- box.
- **MX: PREFERENCE**: 16 bit integer that specifies the preference of this RR. Lower values have higher preference. **EXCHANGE**: A domain name that specifies the host that is responsible for the owner of the RR record
- **NULL**: Anything up to 65535 octets or less.
- **NS: NSDNAME**: A domain name that specifies the host authoritative of the domain. The named host should have a zone starting at the owner name of the specified class.
- **PTR: PTRDNAME**: A domain name that points to another domain name in the namespace.
- **SOA: MNAME**: A domain name that is the primary source of data for the zone. **RNAME**: A domain name that specifies the mailbox of the owner of the zone. **SERIAL**: Unsigned 32 bit version number of the zone. It wraps around and should be compared using sequence space arithmetic. **REFRESH**: a 32 bit time interval before the zone should be refreshed. **RETRY**: a 32 bit time interval that should pass before a failed refresh retires. **EXPIRE**: a 32 bit that specifies an upper limit on the time before the zone is no longer authoritative. **MINIMUM**: 32 bit minimum TTL that any RR of this zone should hold.
- **TXT: TXT-DATA**: One or more character strings.

The following definitions are used by the Internet class only:

- **A: ADDRESS**: A 32-bit internet address
- **WKS: ADDRESS**: A 32 bit internet address. **PROTOCOL**: An 8 bit IP protocol **BITMAP**: A variable 8 bit aligned bit map, which specifies the ports of the protocol.

The IN-ADDR.ARPA domain. The IN-ADDR.ARPA domain is a special domain whose goal is to provide host address to host name mapping, and to enable queries to locate all gateways of a specified network. The same functions provided by this domain can be done using inverse queries, however, since this domain name space is structured according to address, the functionality can be achieved without an exhaustive search. Domain names under the IN-ADDR.ARPA domain have up to 4 labels where each of which represents one octet of an internet address.

Messages

Format. All communications in the domain protocol are carried in a uniform format called message. All messages contain the following sections (in the following order):

- **Header**: Always present, specifies which of the following sections is present, and specifies whether the message is a query, response, etc.
- **Question**: This section contains fields that describe the question to a name server, using QTYPE, QCLASS and QNAME.
- **Answer**: Contains RRs that answer the question.
- **Authority**: Contains RRs that point to an authoritative name server.
- **Additional**: More RRs that relate to the query (not necessarily answer related).

Header section format. The header holds the following fields:

- ID: A 16 bit identifier. it is copied to the reply, and helps match a reply to a request.
- QR: 1 bit. 0: the message is a query 1: the message is a response.
- OPCODE: 4 bits. 0: a standard query (QUERY) 1: an inverse query (IQUERY) 2: a server status update (STATUS) 3-15: reserved for future use. These bits are set by the initiator and are copied to the response.
- AA: Authoritative answer. Valid in responses, and indicates that the responding name server is an authority for the domain name in question.
- TC: Means that the message was truncated due to a length greater than possible on the channel.
- RD: A bit that is set if the initiator wishes the name server to pursue the query recursively.
- RA: Means (at response) that recursive query support is available.
- Z: Reserved for future use and must always be set to 0.
- RCODE 4 bits which are relevant to responses. Their interpretation:
 - 0 No error
 - 1 Format error: a syntax error occurred and therefore the name server was unable to interpret the message.
 - 2 Server Failure: error because of the name server (not format).
 - 3 Name Error: the domain specified in the query doesn't exist (meaningful only from an authoritative name server).
 - 4 Not implemented: the type of query isn't supported by the name server.
 - 5 Refused: means that the name server refuses to execute the query request because of its policy.
 - 6-15 reserved for future use.
- QDCOUNT: An unsigned 16 bit integer indicating the number of entries for the question section.
- ANCOUNT: An unsigned 16 bit integer indicating the number of RRs in the answer question.
- NSCOUNT: An unsigned 16 bit integer indicating the number of name server resource records in the authority records section.
- ARCOUNT: An unsigned 16 bit integer indicating the number of resource records in the additional records section.

Question section format. The question section carries QDCOUNT entries in the following order:

- QNAME: This is a domain name, built as a sequence of labels, each consists a length octet. The last label is the NULL label. No padding is used.
- QTYPE: Two octets specifying the type of question (from TYPE field, see RR structure).
- QCLASS: Two octets specifying class.

Resource record format. Described above.

Message compression. In order to reduce messages size, repetitions of the same domain name in a message can be eliminated. To do so, an entire domain name or a part of it is replaced by a pointer to another occurrence of the domain name. The pointer is 16 bits starting with two ones, which help distinguish it from a label. The next 14 bits are an offset to the location of the pointed object from the beginning of the message. Thus the domain name can be represented as one of the following:

- A pointer.
- A sequence of labels.
- A sequence of labels ending with a pointer.

Pointers can only be used for cases where the format isn't specific for a class, otherwise name servers and resolvers would have to know the format of all RRs they handle. Implementations can use or not use pointers at their own will, however, they are required to understand incoming messages using pointers.

Transport. DNS messages can be transported via virtual circuits (TCP) or datagrams (UDP), while datagrams are preferred.

UDP usage. UDP messages are sent via port 53, and are restricted to 512 bytes (otherwise the message is truncated). It is known, of course, that the UDP service isn't as reliable as TCP, and using UDP in zone transfer is not acceptable. However, it is the preferred method for queries. That being said, a retransmission strategy is required and is recommended as follows:

- The client should try other servers and server addresses before repeating a query to a specific address of a server.
- The retransmission policy should be based on statistics (based on past events). The minimum retransmission interval should be between 2-5 seconds.

TCP usage. TCP messages are sent via port 53. The message is prefixed with a two byte length field which gives the message length, used by the low level processing to assemble a complete message. The following policies are recommended:

- Other activities should be allowed waiting for a TCP message.
- Several connections should be supported.
- The client should interminate the connection, so connection should not be closed until all client requests have been satisfied.
- If a server must close a dormant connection to clear resources, it should wait for at least 2 minutes in which the connection wasn't used.

Name server implementation

Architecture. The name server must support concurrent activities. It should not block UDP requests while it waits for TCP data. It should not block requests when it handles recurring requests. It's up to the implementation whether or not to serialize requests from a single client. The implementation is free to use any data structure, although a recommended structure consists of three parts:

1. Catalog. The list of available zones and a pointer to each zone data structure.
2. Individual zone data structures.
3. and Cache.

Time. Both the TTL found in RRs and the timing data for refreshing activities are a 32 bit timer in seconds unit.

Standard query processing. The general algorithm is defined above. If the QCLASS is a wildcard one, or it covers several classes, the response must not be authoritative unless the server is certain that it contains all classes. RRs which are inserted in the additional section with duplicate RRs in the authority or answer sections, can be omitted from the additional sections. The MINIMUM value in SOA is a lower boundary of the TTL of any data served by the zone.

Inverse query processing. Supporting inverse queries is optional. In case the server receives an inverse query but it does not support one, it should return a “Not implemented” error. Inverse queries contain an empty question section, with a single RR in the answer section. Responses for inverse queries contain the all names that the server is aware of which map to the answer, in the question section. Since the server does not have a full view of the namespace, these responses are by definition partial. Comparisons for inverse queries should be case insensitive, when possible. If the response is not empty, the owner name and TTL of the RR in the answer section are modified to match the first RR in the questions section. RRs which are found in inverse queries cannot be cached using the standard mechanism because the server has partial information.

Zone refresh and reload processing. If the name server cannot parse the zone file, it should answer the requests as it does not own the zone. If a new zone version is available during AXFR, the server should continue sending the old version. In case it is unable to complete the transfer, it should reset the connection.

Resolver implementation

Transforming a user request into a query. The resolver first translates a user request, dependent on the local operating system, into a search specification for RRs. Allowing the resolver to work efficiently, it must be able to handle multiple requests, so each pending request is represented as a state block which usually contains:

- A timestamp indicating the time the request began.
- Parameters to limit the work the resolver performs on a single request. The resolver should have a global per request counter which decrements every time a work is done on the request. Once the counter reaches 0, the request will be terminated.
- The SLIST data structure which watches the state of a request in case it waits for foreign responses.

Sending the queries. Each request is represented by some state information. What is most desirable is that the resolver maximizes the probability that a request is answered, minimizes the time it takes to answer a query, and minimizes excessive transmissions. The resolver chooses the name server to query using the SLIST. The list contains all NS RRs which correspond to the ancestor zone known by the resolver. The server has a set of default servers it will approach if SLIST is empty. It will then add the default to SLIST, and add any relevant history information to the addresses in SLIST; A partial ranking of the available name server addresses that exist. Each time an address is chosen; the state should be altered to prevent its selection again until all other addresses have been tried. The timeout for each transmission should be 50-100% greater than the average predicted value to allow for variance in response.

Processing responses. The first step is to parse the response:

- Check if the header is reasonable, else throw the message.
- Check if the sections of the message RRs are correctly formatted.
- Check the TTL. If the TTL is very long, throw the message or lower it to a reasonable amount of time (optional).

After the first step, the resolver needs to check whether the response matches the resolver request. This is done by matching the ID field, and verifying that the question section match with the desired information.

Using the cache. It would make sense to keep all data received by the resolver in the cache, because it could help save time in future clients' requests. There are a few exceptions:

- When several RRs are available from the same owner, the cache must either cache them all or none. If the resolver receives a truncated message with a possible subset of RR, then it shouldn't cache the subset.
- If the result of caching is that it receives preference over authoritative data, then caching shouldn't happen.
- Inverse query result shouldn't be cached.
- The result of a standard query when the QNAME contains "*" should not be cached.
- RRs of questionable reliability should not be cached.

Known attacks

Cache Poisoning Attacks

DNS Cache poisoning enables attackers to inject forged RRs into the target resolver's cache. This may enable further attacks, with an impact on both integrity and confidentiality. For example, integrity can be broken if the target cache is poisoned with an A RR of an insecure update server, that now points to an attacker controlled IP, the attacker may takeover any host which uses the fake cache entry. A canonical example for impact on confidentiality is where that attacker poisons a cache with an A RR of some web server which has private user data, so it would point to an attacker's controlled IP. When the user browsed to that web server, he would connect the attacker's web server instead of the genuine one. Both examples can be mitigated with server-side authentication methods (such as SSL certificates). It is important to note that both normal resolvers and stub resolvers are an attractive target, although an attack on the former has a greater impact. *Man-in-the-Middle* attacks are ones where the attacker can modify and/or view traffic between the client and the server. *Rogue name servers* are attacks where a malicious name server manages to poison the cache of the client with non-authoritative data. *Blind (off-path) poisoning* is a type of attack where the attacker does not see the traffic between the client and the server, but manages to poison the cache of the client nevertheless.

Man-in-the-middle

The attack is possible because the original DNS specification does not verify message authentication, so the attacker can stand between the resolver and the name server, which means he can see and change queries or responses. The attacker usually races the name server and impersonates it. It sends a spoofed answer to a query. The resolver can't tell if the answer is really from the correct server, and the attack becomes a success once the resolver stores the answer in its cache. The resolver is expecting to see an answer with the same query ID, port and query name and the attacker is disclosed all this information. The attacker can also set the TTL field to a high value, so the record will stay in the resolver cache for a long time.

Rogue name servers

Caching of Out-of-Bailiwick data [Berb] The original resolution algorithm described in [Moc87a] allowed any name server to poison the cache of the client. All a malicious name server had to do is to provide referrals with glue information, and that would be cached by the client, even if the glue information was out of bailiwick. For example, suppose a name server authoritative for the domain name `foo.com` had been asked to resolve `bar.foo.com` then a legitimate action would be to delegate the resolution to another name server, which is authoritative for the `bar.foo.com` zone. This is done by inserting the responsible NS RR to the authority section of the reply. DNS also allows to provide glue information in the additional section of the reply, i.e the A RR of the responsible name server. Both the NS and A RRs are not authoritative data of `foo.com`, so caching that information should be done with special care, otherwise a malicious name server could provide an NS and A records of a target domain name of its choice, which would be cached by the client. Unfortunately, in the past, many name servers did not verify that the domain name is in bailiwick of the contacted name server, so they were vulnerable to this trivial poisoning. The notion of 'Bailiwick' is not well-defined, each implementation has its own guidelines. For example, the BIND v9.4.1 name server [bin], according to [SS10] distinguishes between referrals and answered queries. If the query is answered, i.e. the response contains at least one record in its answer section, then the record is cached if it matches the entry

in the query section, while NS records in the Authority section are only cached if they are a sub-domain or match the contacted name server domain name. If the response is a referral, the NS record(s) found in the Authority section are cached if and only if the query name is a sub-domain of the NS record(s) domain name, and the NS record(s) domain name is a sub-domain of the contacted name server domain name. For a referral, A records found in the additional records are cached if and only if they are a sub-domain of the contacted name-server.

Out-of-Bailiwick dependencies [Berb]. The security issue here is that resolution of DNS domains sometimes rely on out of bailiwick servers, which in many cases are unreliable. This is best explained by an example. If a name server asks for the address of `www.w3.org`, it can either contact the root, the org or the `w3.org` servers. One of `w3.org` DNS name servers is `w3csun1.cis.rl.ac.uk`. This server can define the address of `w3.org`, and update other servers to prevent contradicting information. Now looking at `w3csun1.cis.rl.ac.uk`, one of `ac.uk` DNS servers is `ns.eu.net`, which can define the address of `w3csun1.cis.rl.ac.uk`, and transitively define the address of `w3.org`. This idea goes on, until the resolution depends on obscure servers like `beer.pilsnet.sunet.se`, which runs an old version of BIND, which allows everyone to take over the machine.

Blind poisoning

Vulnerability to Birthday attacks [cer02]. Some of the DNS implementations are exposed to birthday attacks. In these implementations multiple requests for the same RR will generate multiple queries for these requests. Applying this technique works a lot better than brute-force approach achieving DNS spoofing. What happens is that the attacker generates many RR queries to the resolver, forcing the resolver to open several queries for that RR. Then the attacker sends several spoofed responses from the name servers to the resolver. With a right amount of queries and responses a spoofed response will be accepted with high probability. The attack was found to be effective against caching name servers that provide recursive services. A table with the common cases follows:

| Guess | Open requests | # of packets for 0.5 success |
|-----------|---------------|------------------------------|
| TXID | 1 | 32.7k |
| TXID | 4 | 10.4k |
| TXID | 200 | 427 |
| TXID | unlimited | 426 |
| TXID+port | 1 | 2.1 billion |
| TXID+port | 4 | 683 million |
| TXID+port | 200 | 15 million |
| TXID+port | unlimited | 109k |

Insufficient TXID randomization [cer08]. The DNS header contains a 16bit attribute dubbed TXID. According to the original DNS specifications [Moc87a, Moc87b], the TXID of a UDP DNS response must match the request's, otherwise the response shall not be accepted by the client. In order to make the forged response be accepted by the client, it must arrive before the legitimate one does. If the TXID value is not chosen uniformly, then the probability per attempt increases, which may lead to a feasible attack. In fact, in the past, many implementations were found vulnerable.

Insufficient source port randomization: Dan Kaminsky's attack [Kam08, SS10]. Considering the TXID distributes uniformly over 16 bits, the success probability for a single blind attempt is $p = 2^{-16}$. Naturally the number of attempts till success, N , distributes geometrically with p . Thus expected number of attempts for success is $\mathbb{E}(N) = \frac{1}{p} = 65536$. This number of attempts is feasible for attack. Moreover, the attacker can increase p if he manages to inject more than one forged response before the legitimate one arrives. This attack has a major caveat, as a failed attempt with an A RR would be cached by the server, effectively increase the expected attack time to an infeasible value, if the TTL is high. In order to avoid caching, the target per attempt is chosen to be unique, a sub-domain of the target domain. Suppose the attacker wants to attack `www.foo.com`, then the chosen domain names will be `1.foo.com`, `2.foo.com`, etc. In order to actually affect `www.foo.com` the attacker would reply with a referral. Under its authority section would be an NS record: `.foo.com IN NS .www.foo.com` and under the additional section would be a glue A record: `.www.foo.com IN A 6.6.6.6`. This kind of answer does not break the bailiwick policy (at least of BIND, see above), so both records are allowed to be cached by the server. [Elz97] introduces the notion of ranking. If a data in a received RR set has a higher

ranking than the cached one, it shall replace it. BIND implements a similar ranking system, with a few changes. For example, NS RRs found in referrals always overwrite the cached data, so this makes it possible for an attacker to overwrite cached NS RRs.

Correlation between the TXID and the source port [RH12]. After [Kam08], many implementations introduced source port randomization. As explained above, ideally it adds more bits of entropy, in order to increase the attack expected time. However, it is not sufficient to just randomize both the source port and the TXID, since both values can be correlated. For example, [RH12] used the same randomization algorithm on both values. Its random value is the least significant 16 bits of the current time in microseconds, xored with the current time in seconds xored with the PID (process ID). Since there are two successive calls to this algorithm, the two yielded pseudo-random values are very much correlated to each other, so the number of random bits is not much above 16, which leads to a feasible attack expected time.

Defeating source port randomization using Port Exhaustion [RH11]. In an ideal world, source port randomization adds 16 bits of entropy to the 16 bit of randomized TXID bits. Thus the yielded attack expected time is infeasible. However, UDP source ports are a system resource. The DNS resolver cannot guarantee that all the 65536 UDP ports are available. Therefore, if the attacker manages to drastically decrease the number of available ports, source port randomization can be defeated. [RH11] describes both a local and a remote attack. The local attack targets multiuser environments, such as Microsoft Windows Server. A non-privileged user is allowed to bind on many UDP ports, thus he can cancel the source port randomization implemented by the local DNS (stub) resolver. Since the local DNS cache is shared amongst all system users, a successful poisoning attack affects everyone and may lead to privilege escalation and information disclosure. The remote attack targets Java Applets. A user is lured to browse to a malicious website. Once he does, an applet is automatically downloaded and executed. Before the patch, Java Applet code was not restricted on the amount of UDP ports it could bind on, thus it could defeat the source port randomization.

Defeating source port randomization, the NAT case [AH12]. As described above, source port randomization protects against cache poisoning, by increasing the entropy in DNS request by a factor of $O(2^{16})$. This attack increases the likelihood of guessing the source port allocated by the NAT device. There are 3 main port allocating mechanisms in popular NAT and FW devices:

1. Random pick of an available port until all ports are exhausted.
2. Per-destination sequential allocation, which means that for the first time a package is delivered to a specific destination. The port is picked randomly, but the following packages to that destination will use the consecutive port.
3. Port-preserving allocation, which means that for the first time a package is delivered to a specific destination. The port is picked randomly, but the following packages to that destination will use the same port.

In this survey, we will describe the attack against mechanism 1. At first we describe a few assumptions:

- The NAT device translates the following tuple: $(S_{IP}, S_{port}, D_{IP}, D_{port}) \rightarrow (NAT_{IP}, NAT_{port}, D_{IP}, D_{port})$, and the mapping is denoted by the function f .
- NAT maintains the mapping between (S_{IP}, S_{port}) to the external port NAT_{port} for T seconds counting from the last packet.
- The NAT device maps randomly.
- NAT drops packets from $(S_{IP}, S_{port}) \rightarrow (D_{IP}, D_{port})$ when all external ports allocated for (D_{IP}, D_{port}) are exhausted.
- There is a zombie under the attacker control who contacts the attacker and receives a signal to do the following attack:

1. The zombie sends UDP packets to the name server of the attacked domain which FQDN is $ns.v.com$ $\forall p \in \{available\ ports\}$. The NAT computes f and sends all the messages with the new $f(p)$ and also saves the mapping $(p, f(p))$ for the next T seconds. Since the packets are not legitimate DNS packets, they are all ignored by the name server and get no response.
2. The attacker then sends a spoofed message from the name server (from the same UDP port) through the NAT to a specific port 666 of the zombie. The NAT computes $f^{-1}(666)$ and returns the message to the zombie who now knows the mapping of $(666, f(666))$.
3. The zombie waits until almost T seconds pass, and then sends additional empty UDP messages $\forall p \in \{available_ports\} \setminus \{f^{-1}(666)\}$. Now all the ports got new mapping except the mapping for port 666 which timed out, so the only available external port to send messages to the name server in the same port is 666.
4. The zombie sends a single query to the resolver, asking for $r.v.com$ where r is random. This ensures no caching will be made by the resolver and makes sure a DNS query for this FQDN will be made. The resolver sends a query from a random port p using a random identifier (TXID) i .
5. The attacker now sends forged responses $\forall i \in 2^{16}$ possible values and if one of the responses matches the resolver; accepts the following records: $[r.v.com\ A\ 6.6.6.6]$ and $[v.com\ NS\ r.v.com]$. From now on the resolver considers $6.6.6.6$ as a valid IP for the authoritative DNS server of $ns.v.com$. The resolver also forwards the A RR to the zombie who understands the process succeeded and notifies the attacker.
6. The resolver receives a legitimate ‘non-existing domain’ (NXDOMAIN) response from the name server. If the attack succeeded, this response would be ignored, since the query was not pending any more. Otherwise, the resolver forwards the NXDOMAIN response to the zombie, who will inform the attacker. They will repeat the attack from step 1 (as soon as the ports expire on the NAT).

Defeating IP address randomization [AH12]. Both the IP source address and IP destination address can be used for extra bits of randomization, in order to protect against blind cache poisoning attacks. Selection of random source IPs is scarce so this attack focuses on random destination addresses. The idea is that several name servers can be authoritative for the same domain name, and the resolver simply randomly chooses which one to query. If the queried server is unresponsive, it is not queried for several minutes, so the entropy is effectively reduced. The attack shows how to lure the resolver to think that the target is unresponsive. It exploits IP fragmentation. If the DNS response is larger than the MTU, it must be fragmented, into $\lceil \frac{ResponseSize}{MTU} \rceil$ fragments. The attacker races on the second fragment, making the reassembled DNS response invalid, thus making the resolver neglect the target name server for some period of time. In order for the attack to succeed, the attacker must guess the IP ID field (16 bits) correctly. A typical server usually contains a buffer in which it stores received fragments (since they can arrive out of order). If the buffer size is bigger or is equal to $2^{16} + 2$, then the attacker can win with a success probability of 1. If the IP ID field is globally incremented, then the attacker can deduce the next IP ID field by querying the server. If the IP ID field is per-destination incremented, and the buffer size is n , then the attacker can place $n-2$ values with identical distance, and make $\lceil \frac{2^{16}}{n-2} \rceil$ queries. The success probability would be 1. If there is a load balancing device, then each server behind it can choose its own strategy. Let’s assert that there are k servers which are chosen at round-robin. The attacker can either divide the buffer into k logical buffers each of size $\lfloor \frac{n-2}{k} \rfloor$. The other option is one buffer and make $O(k)$ queries per attempt in order to insure that the target server is reached.

Defeating DNS Query randomization [AH12]. *Case toggling* is a defense against DNS poisoning, in which the cases of letters of the domain name are randomly switched, and they are validated in the response. The problem is that the distribution of domain queries with 20 characters is not an indication to the number of characters the attacker will try to poison. e.g. Kaminsky-style attacks.

Random prefix is another defense mechanism in which prefixes are added to DNS queries allowing to apply the 0x20 on more letters and harden the guessing of the query. DNS queries are composed of sub domains, each is limited by 63 bytes, and the total number of characters is limited to 255 bytes. A naive implementation of this defense can be easily circumvented. An attacker who wishes to poison $.com$ for example can easily send a query of maximal size that will not allow adding any more letters to the beginning. The query will also consist of numbers, to circumvent the 0x20 protection.

DNS Updates attacks

Weak authentication: IP-based [SY11]. DNS Updates are carried over UDP, thus if the authentication is based on the source IP, it can be easily defeated by IP spoofing.

Lack of authorization [SY11]. Authenticating a user is not always enough. If the update requests are not authorized, an authenticated user can change all RRs in the zone. This opens a door to attacks like redirection, DoS and more. While compromising a server is hard, compromising a client is a much easier task. Many of the clients don't use any security measures at all. So an attacker can compromise an authenticated client, calculate the required MAC, and send false update messages. Due to these problems restricted access should be given. In BIND for example, the node is restricted to update specific records based on the key used for authentication.

Other poisoning attacks

Poisoning via AXFR [Bera]. This weakness will be described by an example. Given an ISP which is a third party DNS server for .edu and princeton.edu. The data of the zone from the princeton.edu. AXFR server is:

```
haven.princeton.edu NS serv1.net.yale.edu
serv1.net.yale.edu A 128.112.128.15
```

Now, we assume the victim's cache has the following legitimate NS record: `yale.edu NS serv1.net.yale.edu`. If the user asks for `www.haven.princeton.edu`, the cache will approach the root server, see that ISP is a .edu server and receive the same information as mentioned above from the zone AXFR server. The cache trusts and saves `serv1.net.yale.edu` information (because the ISP is a .edu server). If the user asks for `www.yale.edu`, from the information above it will be forwarded to 128.112.128.15 to get the address of `www.yale.edu`, so basically Princeton is now in control of Yale web server.

DoS Attacks

Spoofing attacks [HB08, FG06]

Amplification attack using recursive requests [usc06, RV06, Pis06, Pri12]. Name servers which accept recursive requests can be forced into participating in a DDoS attack. The attacker holds a name server which serves a very large TXT record (4000 bytes) with a high TTL. The DNS query request is approximately 60 bytes, so the amplification factor is about 1: 70. The attacker controls a botnet that issues DNS queries asking for the large TXT record to the open name servers with a spoofed source IP that points to the target of the attack. If the record is not cached, it will be requested from the attacker's name-server. Afterwards, the response is replied to the victim. Since the large response causes IP fragmentation, it adds an extra overhead at the victim's machine, and in addition, only one fragment contains the UDP header, so it cannot be easily identified as a DNS DoS attack.

DoS by poisoning

Invalid mapping [And98] A possible poisoning attack is injecting a false A RR. For example, if a client wishes to browse to `www.google.com`, and he receives a false A RR which contains an IP with no name server behind, then the client would not be able to access the website.

Negative caching [And98]. Caching of non-existing domains (NXDOMAIN) was introduced in [And98]. This feature allows an attacker, during a successful DNS poisoning attack, to inject a NXDOMAIN entry into the target cache, which will be alive according to the minimum of TTL of the SOA record, and the MINIMUM field of the SOA record. The attack has a greater psychological effect than non-responsive servers, since non-existing domain is usually not a temporary condition. In addition, some applications also behave differently. For example, an SMTP server will automatically bounce a mail message sent to a non-existing domain, but may retry if the target server is non-responsive. Thus, if it retries after the invalid A RR record is removed from the cache, the mail message will be sent successfully.

Information Disclosure

Zone disclosure via AXFR [I73]

While this protocol is useful in transferring zone information between name servers, it is exposed to information gathering for purposes of mass mailing, distributed DoS attacks etc. The problem is when name servers do not use restrictions on AXFR queries. This data can be used to find mail relays, proxies, hosts with specific OS and hundreds of thousands of records with meaningful host names. It is also possible to perform DoS attack on a name server sending many AXFR queries. The best way to handle this is to hold an access list to prevent unauthorized access.

Query accepting DNS servers [I73]

A name server which openly accepts queries can be used in order to leak sensitive information, such as zone data (e.g. private IP addresses) or sensitive cached data. Name servers are advised to restrict access to their cache and zone data.

DNS related attacks

DNS Rebinding [CJ07]

This attack targets web browsers. In modern browsers, one domain cannot access data of another domain, this is dubbed as the Same-Origin Policy. For example, JavaScript code of `foo.com` cannot read responses of `bar.com`. The attacker controls a domain, e.g. `attacker.com`. The user is lured to browse to the attacker's domain. The first query returns the attacker's controlled web server IP (A RR), so the attacker can download malicious JavaScript. According to the Same-Origin policy, this JavaScript is restricted to `attacker.com`, so it has no effect on the user's privacy. However, if the returned A RR has a low TTL, then it will be cached out, and another network access to `attacker.com` (for example, accessing a remote file) will result in another DNS query. This time, the attacker can return a different IP and that breaks the Same-Origin Policy. For example, the attacker may return an internal IP address, in order to leak sensitive information from the victim's network. The attacker can also create raw sockets, so the victim becomes a proxy. For example, the victim can be used as a spam relay. Some browsers implement DNS Pinning as a defense mechanism. It basically makes the resolver ignore the TTL value, and cache the responses for a fixed duration. However, DNS Pinning has been found ineffective in certain circumstances by [CJ07].

Phishing

The method of Phishing is creating a replica of a legitimate web page to hook users and trick them into submitting personal information. This is often done by links to these websites carried in e-mails, or instant messaging forwarding to these websites.

Old school. A popular Phishing attack vector is registering a domain name which is similar to the name the attacker wants to impersonate. For example, if the target is `www.paypal.com`, the attacker may register `www.paypal.com` in order to attract naive users.

Internationalized domain names DNS only supports ASCII domain names. Names with non-ASCII characters are encoded using Punycode [Cos03] into ASCII. This opens that gate to phishing attacks, because some Cyrillic letters look identical to Latin ones. For example, the Unicode letter U + 0430, the Cyrillic *a*, can look identical to U + 0061, the Latin *a*. The attacker can exploit this behavior. For example, if he wanted to attack `www.bankofamerica.com`, he could register the same name, but replace one of the Latin *a*'s with their Cyrillic counterparts.

New attack: Subverting BIND's SRTT algorithm

The SRTT algorithm

BIND maintains a dynamic list of candidate name servers for resolving a particular query. The chosen server is the one with the lowest SRTT (Smoothed Round Trip Time). The SRTT value for each name server is stored in a global cache indexed by the name server's IP and calculated as follows:

1. Init. When a candidate is not cached and added, it receives a very low value: $SRTT_{init} = 31 \wedge R \mu sec$ where R is a 32bit unsigned value returned by the `isc_random_get` routine. If we assert that the 5 least significant bits of R distribute uniformly, then $SRTT_{init} \sim Uni(0, 31)$.
2. Update. Whenever a response is received, the SRTT becomes a weighted sum of the old SRTT and new RTT value: $SRTT_{update} = 0.7 \cdot SRTT_{old} + 0.3 \cdot RTT_{new}$.
3. Decay. In order to avoid starvation, for each request the resolver produces, unused candidates are multiplied with a decay factor (0.98).
4. Error. In case of an unanswered request, or a network error (e.g. an ICMP "Destination Port Unreachable" message is received), the candidate name server is punished with 200 ms (with a max value of 1 sec).

Vulnerability

Our attack forces the algorithm to lower the SRTT of any name server we choose. The attack achieves the same result as the probabilistic method depicted in [Pet09]. However, our attack does not require spoofing nor guessing. It is deterministic and always succeeds. Moreover, the target name server is unaware of the attack.

First variant

Prerequisites. The attacker owns a domain name served by NS A_1 . The attacker is required to have another name server, A_2 , with a low latency from the target resolver, R . Specifically, if the attacker tries to lower the SRTT of name server V on R , it requires that A_2 has a lower SRTT on resolver R than V 's SRTT (denoted as $SRTT_{original}$).

Attack flow.

1. The attacker inserts A_2 to R 's SRTT cache. For example this can be done by querying R for some domain A_2 is authoritative of, say `a2.foo`.
2. R will eventually contact A_2 and return the answer to the attacker.
3. The attacker queries R for a domain name that A_1 is authoritative of, say `a1.foo`.
4. R eventually contacts A_1 which replies with a delegation that contains:
 - (a) A fresh list of non-open name servers (i.e. external clients do not have access to its local cache), C_1, \dots, C_n .
 - (b) name server A_2 .
 - (c) name server V .
5. R will first query each non-open name server, and they will refuse.
6. R queries A_2 which returns a valid answer.
7. R returns a valid answer to the attacker.

When the resolver receives the delegation (4), it creates a new SRTT entry for C_1, \dots, C_n with a low value ($SRTT_{init}$), thus they are queried before A_2 and V . After n queries to the non-open name servers, according to the SRTT algorithm, both A_2 's SRTT and V 's SRTT will be 0.98^n of their original value before the attack (n decay operations). As of the requirement that A_2 's SRTT is lower than V 's SRTT, A_2 's SRTT will also be lower than V 's SRTT after n decay operations, thus V is not queried and the iterative resolution ends by querying A_2 , thus V 's SRTT is now a low bogus value ($0.98^{n+1} \cdot SRTT_{original}$).

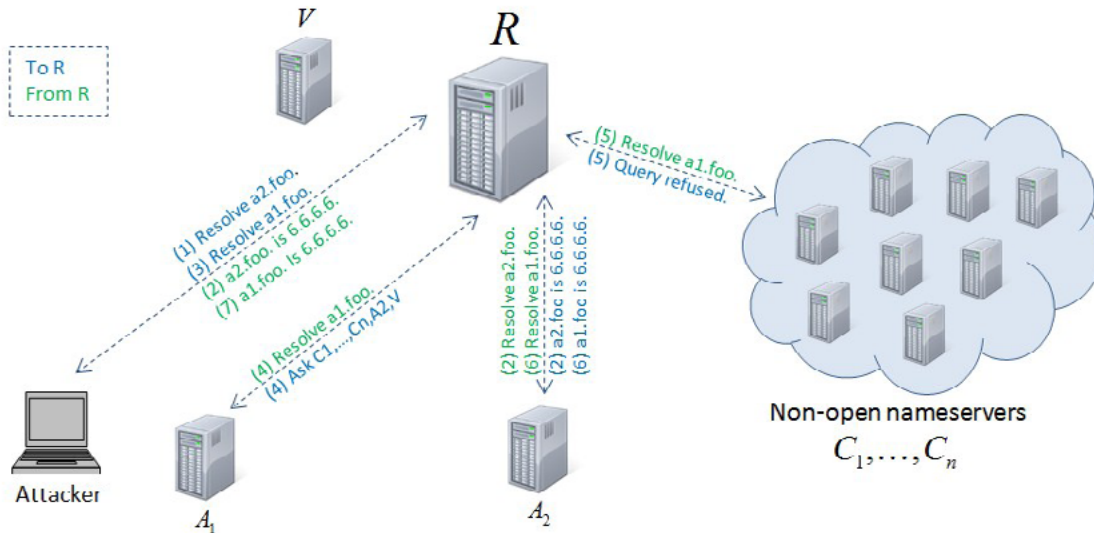


Figure 1.

Second variant

Prerequisites. Here the attacker is not required to have a low-latency name server, however m attempts require m servers $A_{2,1}, \dots, A_{2,m}$ controlled by the attacker (otherwise the attacker has to wait for the SRTT entry to expire from the cache).

Attack flow (i^{th} attempt).

1. The attacker queries R for a domain name that A_1 is authoritative of, say $a1.foo$.
2. R eventually contacts A_1 which replies with a delegation that contains:
 - (a) A fresh list of non-open name servers, C_1, \dots, C_n .
 - (b) name server $A_{2,i}$.
 - (c) name server V .
3. R will first query each non-open name server, and they will refuse.
4. R queries $A_{2,i}$ which returns a valid answer.
5. R returns a valid answer to the attacker.

When the resolver receives the delegation, it creates a new SRTT entry for $A_{2,i}$ and for each non-open name server, with SRTT_{init}. Thus $A_{2,i}$'s SRTT is lower than V 's SRTT. Therefore the order of queries is as depicted above. According to the SRTT algorithm, V 's SRTT is multiplied with 0.98^T where T is a uniform random variable (since $A_{2,i}$'s SRTT can be lower than one of the non-open name servers) with an average of $\frac{n}{2} + 1$.

Impact

This attack is good for derandomization (of the selected name server) which reduces the attack time of blind (off-path) DNS cache poisoning. In addition, the attacker can be a MiTM in one path but not in another, so this attack can also enable on-path poisoning. Moreover, it may assist in DoS. Since the update operation of the SRTT algorithm takes into account the previously defined SRTT, the recovery is not instant, thus subsequent requests to V will not cause it to immediately revert the SRTT to the genuine value.

Evaluation

This first variant was evaluated against BIND 9.8.1-P1 with the following setup:

| Role | IP |
|--------------------------------------|--------------------|
| Victim V | 192.168.19.202 |
| Resolver R | 192.168.42.100 |
| Attacker A1 | 192.168.19.250 |
| Attacker A2 | 192.168.42.251 |
| Non-caching NSs C_1, \dots, C_{30} | 192.168.19.170-199 |

- BIND's SRTT cache before step (3) of the attack:

```
; Unassociated entries
;
; 192.168.19.250 [srtt 106324] [flags 00002000] [ttl 1780]
; 192.168.19.202 [srtt 76745] [flags 00002000] [ttl 1771]
; 192.168.19.50 [srtt 126469] [flags 00002000] [ttl 1780]
; 192.168.42.251 [srtt 329] [flags 00002000] [ttl 1780]
; 192.168.19.201 [srtt 50637] [flags 00002000] [ttl 1771]
;
```

- BIND's SRTT cache after step (7) of the attack:

```
; Unassociated entries
;
; 192.168.19.197 [srtt 73101] [flags 00002000] [ttl 1787]
; 192.168.19.174 [srtt 82638] [flags 00002000] [ttl 1793]
; 192.168.19.195 [srtt 83952] [flags 00002000] [ttl 1789]
; 192.168.19.172 [srtt 74040] [flags 00002000] [ttl 1792]
; 192.168.19.193 [srtt 84831] [flags 00002000] [ttl 1788]
; 192.168.19.170 [srtt 83988] [flags 00002000] [ttl 1792]
; 192.168.19.191 [srtt 79689] [flags 00002000] [ttl 1790]
; 192.168.19.189 [srtt 84951] [flags 00002000] [ttl 1793]
; 192.168.19.187 [srtt 75852] [flags 00002000] [ttl 1787]
; 192.168.19.185 [srtt 76194] [flags 00002000] [ttl 1790]
; 192.168.19.183 [srtt 71946] [flags 00002000] [ttl 1788]
; 192.168.19.250 [srtt 120966] [flags 00002000] [ttl 1786]
; 192.168.19.181 [srtt 88161] [flags 00002000] [ttl 1790]
; 192.168.19.202 [srtt 41014] [flags 00002000] [ttl 1794]
; 192.168.19.179 [srtt 73713] [flags 00002000] [ttl 1790]
; 192.168.19.177 [srtt 84861] [flags 00002000] [ttl 1787]
; 192.168.19.198 [srtt 76785] [flags 00002000] [ttl 1791]
; 192.168.19.175 [srtt 74013] [flags 00002000] [ttl 1792]
; 192.168.19.196 [srtt 72825] [flags 00002000] [ttl 1791]
; 192.168.19.173 [srtt 78900] [flags 00002000] [ttl 1786]
; 192.168.19.194 [srtt 78915] [flags 00002000] [ttl 1789]
; 192.168.19.171 [srtt 85113] [flags 00002000] [ttl 1789]
```



```
; 192.168.19.192 [srtt 73800] [flags 00002000] [ttl 1789]
; 192.168.19.190 [srtt 80958] [flags 00002000] [ttl 1793]
; 192.168.19.188 [srtt 67695] [flags 00002000] [ttl 1786]
; 192.168.19.50 [srtt 126469] [flags 00002000] [ttl 1761]
; 192.168.19.186 [srtt 74088] [flags 00002000] [ttl 1794]
; 192.168.19.184 [srtt 73941] [flags 00002000] [ttl 1791]
; 192.168.19.182 [srtt 85155] [flags 00002000] [ttl 1788]
; 192.168.19.180 [srtt 78822] [flags 00002000] [ttl 1794]
; 192.168.42.251 [srtt 397] [flags 00002000] [ttl 1794]
; 192.168.19.201 [srtt 50637] [flags 00002000] [ttl 1752]
; 192.168.19.178 [srtt 77985] [flags 00002000] [ttl 1791]
; 192.168.19.199 [srtt 71937] [flags 00002000] [ttl 1792]
```

```
; 192.168.19.176 [srtt 86967] [flags 00002000] [ttl 1788]
;
```

The SRTT of V before the attack is 76745. The SRTT of V after the attack is 41014 which is approximately $0.98^{31} \cdot 76745$ as expected.

References

- [AH12] Haya Shulman Amir Herzberg. Security of Patched DNS, 2012. <http://u.cs.biu.ac.il/~herzbea/security/12-04-derandomisation.pdf>.
- [And98] M. Andrews. RFC 2308: Negative Caching of DNS Queries (DNS NCACHE), 1998. <http://www.ietf.org/rfc/rfc2308.txt>.
- [Bera] D.J. Bernstein. How the AXFR protocol works. <http://cr.yp.to/djbdns/axfr-notes.html>.
- [Berb] D.J. Bernstein. Notes on the Domain Name System. <http://cr.yp.to/djbdns/notes.html>.
- [bin] Bind. <https://www.isc.org/software/bind>.
- [cer02] Various DNS service implementations generate multiple simultaneous queries for the same resource record, 2002. <http://www.kb.cert.org/vuls/id/457875>.
- [cer08] Multiple DNS implementations vulnerable to cache poisoning, 2008. <http://www.kb.cert.org/vuls/id/800113>.
- [CJ07] Andrew Bortz Weidong Shao Dan Boneh Collin Jackson, Adam Barth. Protecting Browsers from DNS Rebinding Attacks, 2007. <http://crypto.stanford.edu/dns/dns-rebinding.pdf>.
- [Cos03] A. Costello. Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA), 2003. <http://tools.ietf.org/rfc/rfc3492.txt>.
- [Elz97] R. Elz. RFC 2181: Clarifications to the DNS Specification, 1997. <http://www.ietf.org/rfc/rfc2181.txt>.
- [FG06] Tzi-cker Chiueh Fanglu Guo, Jiawu Chen. Spoof Detection for Preventing DoS Attacks against DNS Servers, 2006. <http://www.ecsl.cs.sunysb.edu/tr/TR187.pdf>.
- [HB08] Paul Francis Hitesh Ballani. Mitigating DNS DoS Attacks, 2008. <http://research.microsoft.com/en-us/um/people/hiballan/pubs/ccs08-staledns.pdf>.
- [I73] I733. BIND 9 DNS Security. http://www.nsa.gov/ia/_files/vtechrep/I733-004R-2010.pdf.
- [Kam08] Dan Kaminsky. Black Ops 2008: It's The End Of The Cache As We Know It, 2008. http://s3.amazonaws.com/dmk/DMK_BO2K8.ppt.
- [Moc87a] P. Mockapetris. RFC 1034: Domain Names – Concepts and Facilities, 1987. <http://www.ietf.org/rfc/rfc1034.txt>.
- [Moc87b] P. Mockapetris. RFC 1035: Domain Names – Implementation and Specification, 1987. <http://www.ietf.org/rfc/rfc1035.txt>.
- [Pet09] Emanuel Petr. An Analysis of the DNS cache poisoning attack, 2009. <https://labs.nic.cz/files/labs/DNS-cache-poisoning-attack-analysis.pdf>.
- [Pis06] David M. Piscitello. Anatomy of a DNS DDoS Amplification Attack, 2006. <http://www.corecom.com/external/livesecurity/dnsamplification.htm>.
- [Pri12] Matthew Prince. Deep Inside a DNS Amplification DDoS Attack, 2012. <http://blog.cloudflare.com/deep-inside-a-dns-amplification-ddos-attack>.
- [RH11] Yair Amit Roei Hay. DNS Poisoning via Port Exhaustion, 2011. <http://bit.ly/q31wSq>.
- [RH12] Roi Saltzman Roei Hay. Weak Randomness in Android's DNS resolver, 2012. <http://bit.ly/MkteBx>.
- [RV06] Gadi Evron Randal Vaughn. DNS Amplification Attacks, 2006. <http://www.isotf.org/news/DNS-Amplification-Attacks.pdf>.
- [SS10] Vitaly Shmatikov Soel Son. The Hitchhiker's Guide to DNS Cache Poisoning, 2010. http://www.cs.utexas.edu/~shmat/shmat_securecomm10.pdf.

- [SY11] Amir Qayyum Sadaf Yasmin, Muhammad Yousaf. Security Issues Related with DNS Dynamic Updates for Mobile Nodes: A Survey, 2011. <http://corenet.org.pk/data/Security%20Issues%20Related%20with%20DNS%20Dynamic%20Updates%20for%20Mobi-20110112-005730.pdf>.
- [usc06] The Continuing Denial of Service Threat Posed by DNS Recursion, 2006. http://www.us-cert.gov/reading_room/DNS-recursion033006.pdf.

About the Authors

Nakibly Gabi



I have more than a decade of experience in networking technologies and security. Since 2006 I am with the National Research & Simulation Center where I lead state-of-the-art network security research projects. Since 2008 I also serve as an adjunct lecturer and researcher at the Technion. In the summer of 2012 I was a visiting scholar at Stanford University's security lab. I am a recipient of the Katzir Fellowship (2007-2012). I hold a B.Sc. in Information Systems Engineering and B.Sc. in Industrial Engineering and Management from the Technion (received in 1999, summa cum laude). In 2008 I finished the direct track to PhD in Computer Science at the Technion. My thesis supervisor was Prof. Reuven Cohen.

Jonathan Kalechstain



I am a CS (Computer Sciences) Masters student at Tel Aviv University, after graduating from Technion, Israel Institute of Technology in 2013. My main interests are: Formal methods, Graph Theory, SAT solvers and Computer Security. My Thesis is supervised by professor Nachum Dershowitch. It deals with algorithms to improve SAT solvers based on a smarter search. I have been working as a masters student in Intel since February 2013, at the formal verification team. I helped developing a very successful tool that finds speed failures using SAT solvers.

Roe Hay

Roe leads the Application Security Research Group at IBM. He has vast knowledge and experience in network and mobile security. Roe holds a B.Sc. degree from Technion, Israel Institute of Technology.

advertisement



better safe than sorry
www.demyo.com

Layer 7: Application Level DDoS

by Neha Malik

How DDoS can be devastating for applications that aren't looking.

Typically, a Denial of Service (DoS) condition occurs when a server or network resource is unable to service legitimate requests made to it, and, therefore, unable to perform a function that it was designed to. DoS attacks have been around for quite some time, with the earliest attacks being dated to the first half of 1970's. This type of attack started out as an avenue for hackers to establish status in underground communities. However, today these have evolved into far sophisticated and dangerous forms that are directed at specific targets for a number of reasons, not excluding cyber-terrorism, corporate rivalry, hacktivism and even exhortation.

When a targeted Denial of Service attack is carried out using a large number of usually unwitting devices and internet connections across the world, it becomes a Distributed Denial of Service attack (DDoS). The skeletal structure of DDoS botnets is usually a variation of this:

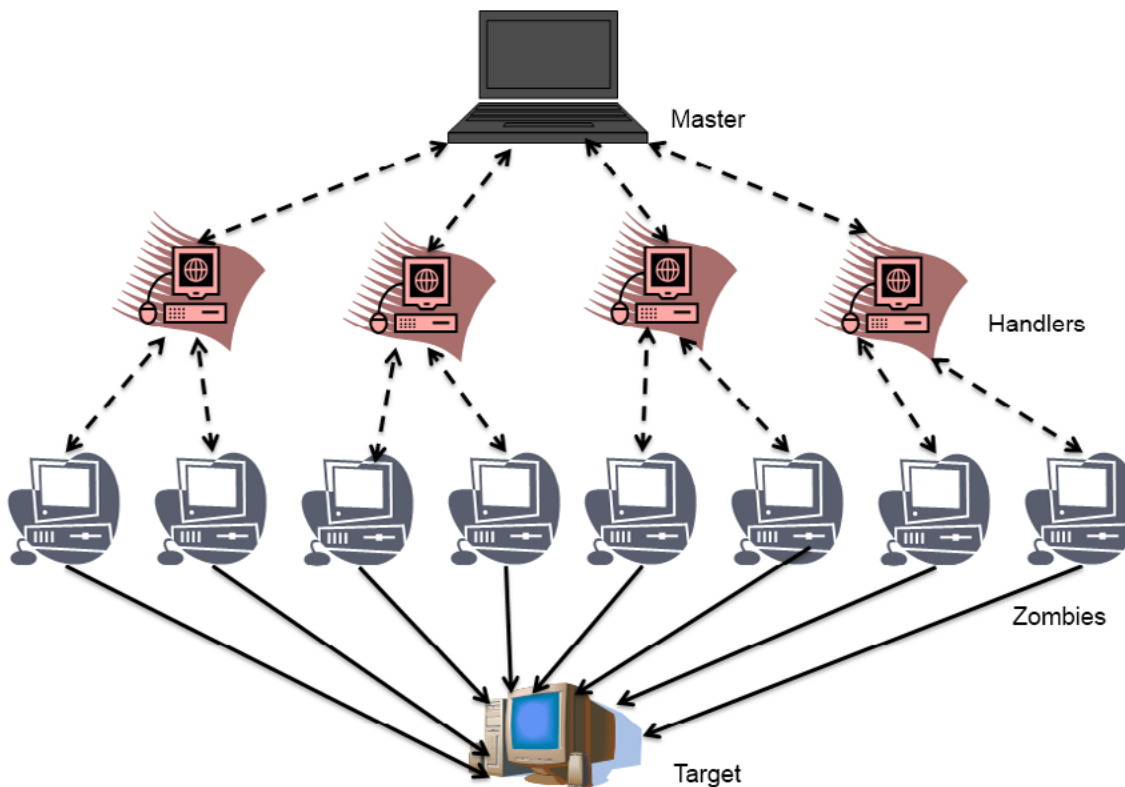


Figure 1. Skeletal DDoS Anatomy

Distributed Denial of Service often thought of to be an attack concerned with Layer 3 or Layer 4 of the OSI Model. While DDoS defenses are getting better and more intelligent, attackers have managed to stay a step ahead all steps of the timeline. According to NSFOCUS, one major DDoS news event happened every 2 days, and one common DDoS attack happened every two minutes!

Despite this, attackers have not yet exploited the full range of vulnerabilities present in many online services for carrying out DDoS attacks. This is especially true of the Application and data processing layers. By nature, the application layer is more generic than the network layer due to a wide variety of applications. However, the methods to implement these applications are similar, leaving the application layer open to a large array of attacks, including unsophisticated ones. This explains these attacks are rapidly becoming the weapon of choice during recent years. According to a DDoS attack statistics analysis report by Prolexic, application level DDoS attacks consisted of 23.24% of total attacks in Q4 of 2013 alone. A number of Gartner reports show a disturbing deviation towards them as well.

Getting Popular

The motivation behind attackers going increasingly for application DDoS is simple:

Detectability

Layer 7 attacks are more difficult to detect than standard network-level DDoS attacks. The idea behind this assault is not the *appearance* of the data packets but the *intention*. Packets performing Layer 7 attacks look the same as any other legitimate packet to a firewall or IDS.

Efficiency

Unlike network attacks, application DDoS does not generate traffic spikes and alert detection mechanisms. Layer 7 DDoS is meaner and leaner in terms of lower consumption of bandwidth and requirement of intermediary resources.

Traceability

Application DDoS attacks use HTTP and HTTPS traffic. Traffic of malicious origin can be disguised via largely available proxy servers without much effort. Many proxy servers are notorious for not maintaining history or logs, making the attack much harder to trace back to its source.

Attack Anatomy

There are a number of attack vectors that have been used for exploitation of applications so far. Some of the most widely known are summarized below:

HTTP GET Flood Attack

This attack is carried out by bombarding the target server with a series of legitimate HTTP GET requests. This means that at this stage, the TCP three-way handshake has been completed and a valid connection has been established, deceiving Layer 4 detection devices. The idea behind the attack is to send a large number GET requests that are intended to exhaust server resources. Hence, attack vectors are made up resource-intensive requests like demands for large files or objects.

The logs for this type of attack look like any other request to the application:

```
80.93.170.6 - - [15/Apr/2014:19:40:08 -0530] "GET /?436873463892 HTTP/1.1" 200 440 "www.imdb.com/  
title/tt0298482" ""  
179.10.10.92 - - [15/Apr/2014:19:40:08 -0530] "GET /?44328956742393 HTTP/1.1" 200 440 "www.  
youtube.com/playlist?list=PLF7A3E4527BCC3B56" ""
```

Particularly vulnerable are areas of search functionality within applications, especially those allow searches with wildcard characters, as these query databases and may lead to a database-level crash.

Some of the well-known tools for this type of attack include LOIC (Low Orbit Ion Canon), XOIC and HULK (HTTP Unbearable Load King).

HTTP POST Attack

This attack is executed via seemingly innocent requests with the *Content-Length* header manipulated to reflect a large value, e.g. 1000000. This tells the server how much content it should wait for before it

considers the request to be completed. Next, data is sent character by character over a very long period of time. The web server is forced to keep the connection open during this duration, affecting and/or denying legitimate user connections. This is especially fatal in a DDoS situation and may lead to the server crashing. It takes 20,000 such connections for an IIS server to be DDoS'ed!

Commonly used tools for this attack scenario are RUDY(R-U-Dead-Yet) and Tor's Hammer. OWASP has also come up with OWASP DoS HTTP POST tool.

HTTP Slow Read Attack

This attack works in a reverse way of HTTP POST attack. Instead of sending the server small requests, the approach taken is to reduce the client receive window to a very small size. Hence, server connections are compelled to stay open as the client reads responses indefinitely. This method bypasses server policies that filter slow-deciding customers. This attack proved to be successful when the following two conditions are satisfied:

The response size is large. This is easily satisfied with many web pages reaching sizes of up to 1MB.

Server send buffer size is known or can be estimated and the client receive buffer size is accordingly made small. The default value of send buffers is usually between 65Kb and 128Kb.

SlowHTTPTest can be used to carry out or test for HTTP Slow Read attack.

Slowloris Attack

Slowloris holds connections open by sending partial HTTP requests, particularly at the header. It works by sending incomplete header information distributed over long periods of time, thus holding up the server. Web servers look for a double carriage return to understand the end of a HTTP header. However, Slowloris continues sending information without providing the header's end.

The Slowloris tool is capable of modifying sent headers depending upon the target host configuration. For high traffic websites, the attacker may have to wait for all sockets to become available in order to consume the web server resources.

NTP Amplification Attack

2013 was the year of DNS Amplification. However, 2014 seems to be the year of NTP Amplification, with a reported rise of 371% in the first quarter. For a DNS Amplification attack, the amplification factor (ratio of response to request) is 8X. For an NTP attack on a busy server, it can reach 206X! An attacker, armed with the list of open NTP servers available on the internet, sends an NTP *monlist* (or `MON_GETLIST`) command with the source IP spoofed to be the target's IP address. The result is a very large response split over multiple packets directed to the target, leading to a Denial of Service condition.

Monlist modules can be found in Nmap as well as Metasploit.

SNMP Amplification Attack

Unlike other DDoS attacks, SNMP allows attackers to take over network devices as well and use them as bots in attacking other targets. To execute this attack, the attacker needs a list of exploitable SNMP hosts as well as community strings. This can be obtained by port-scanning IP addresses or obtaining the SNMP host list through private sources. In the next step, the attack sends a SNMP `BulkGetRequest` command to the SNMP Management Information Base (MIB), which returns amplified content. This attack request expectedly made with the source IP spoofed, leading to the target being overwhelmed with SNMP responses. `Snmpbulkwalk` is one of the tools used for this purpose.

So far, known instances of SNMP Amplification are comparatively lesser in number. This could be accounted to the lower visibility of SNMP servers over the Internet and the additional password requirements. However, the theoretical amplification factor of SNMP attacks has been found to be 650X – It is better to be safe than sorry here!

SMTP DDoS Attacks

SMTP DDoS can happen through several attack vectors. The first way is when thousands of emails are sent using computers across the web to one single SMTP server.

The second way is through backscattering attacks, where the attacker forces the SMTP server to generate a large number of non-delivery reports. Since non-delivery reports often include the full body of the original message along with attachments, the multiplicative force of this affect creates a DoS condition.

PyLoris tool is known to be used for execution of protocol-based, and specifically SMTP, DDoS attacks.

Application Logic Attacks

Apart from the various widespread attacks that are known to be launched against Layer 7, there is also a category of attacks that is seemingly overlooked and is not well-defined as of yet. These include application business logic and implementation logic flaws. Possible attack vectors can include the following:

- Exploitation of hidden bottlenecks in the application architecture. For example, applications that implement large client-facing tiers but have a small resource farm at the back-end to handle client requests.
- Applications implementing poor data validation techniques and thus, being DoS'ed by common injection flaws.
- Automated submission of data through Dictionary attacks for logins, overloading application functions, deliberately invoking race conditions or attacking multiple entities.
- User data manipulation leading to unexpected server errors or opening up known exploitable vulnerabilities.
- For applications locking out accounts permanently on entering invalid credentials, intentional account lockout of all accounts, including self, resulting in Denial of Service.
- Creation of multiple fake users for applications that do not require manual intervention at registration and thereby, starving application resources.
- Exhaustion of application session resources by creating an excessive number of active connections.

Shield of Defense

What we have seen previously are few known attacks that have been carried out against applications for creating DoS conditions. The actual number and type of assaults is continually growing and changing. Hence, it is imperative that organizations also evolve to keep up with the attackers. The best defense mechanisms are always first proactive, then reactive. Therefore, security measures for protection start at Design. In addition to specific precautions that are required for unique attacks, the following broad behaviors are a must for having a sound Defense in Depth structure.

Analysis of Logical Flaws

As the threat landscape for application attacks in general and DDoS in particular comes in focus, the golden rule of Input Validation becomes even more important for web application design. From an attacker's perspective, the first line of thought is almost always what an application user is allowed to

do. It is imperative that user input and actions not be implicitly trusted or assumed, and be thoroughly validated before consumption. Validation should be done at every application layer for the relevant layer, as input that is harmless for one layer may be intended for another. Secure Code Reviews and Penetration Tests hold special importance in this defense path.

Use of Anti-DDoS Tools and Testing

Many large vendors in the market have come up with DDoS solutions customized to include application protection. Some examples are solutions provided by Rackspace, F5, Juniper and others, including Akamai's KONA Site Defender, F5 BIG-IP Application Security Manage (ASM), Sucuri WAF, Cisco Traffic Anomaly Detector XT and Cisco Guard XT. Applications in question also need to undertake simulated attacks on a regular basis to understand their reaction to the same. While regular Penetration Testing is carried out for most Internet-facing application, it is crucial for these tests include examination for Denial of Service and Brute Force attacks.

Active Monitoring and Update

Finally, live monitoring mechanisms need to be in place to look out for unexpected application behavior. While a portion of this might be taken over by anti-DDoS solutions in place, it is necessary to have alert processes in place specifically for application attacks, along with SLA's defined for code changes by developers to stop attacks in question for good. Apart from these, regular and fast updates of vulnerable software as patches are released, is imperative. Other techniques like Blackholing requests can also be used. However, Blackholing discards legitimate traffic as well and may not be the best solution in an Application DDoS scenario.

It is important to understand that there is no such thing as "100% security" (Remember Titanic?). A combination of the above measures along with keeping updated on latest attack techniques can provide effective protection against Application DDoS in the long run.

On the Web

- <http://www.slideshare.net/prolexic7885/prolexic-d-do-s-attack-report-q4-2013-ddos-attack-trends-and-statistics> – DDoS attack trends report for Q4 2013
- <http://en.nsfocus.com/SecurityReport/2013%20NSFOCUS%20Mid-Year%20DDoS%20Threat%20Report.pdf> – 2013 Mid Year DDoS Threat Report
- https://media.blackhat.com/bh-dc-11/Brennan/BlackHat_DC_2011_Brennan_Denial_Service-Slides.pdf – Black Hat Presentation on DoS attacks
- <http://blog.cloudflare.com/understanding-and-mitigating-ntp-based-ddos-attacks> – Article on NTP Amplification attacks

About the Author



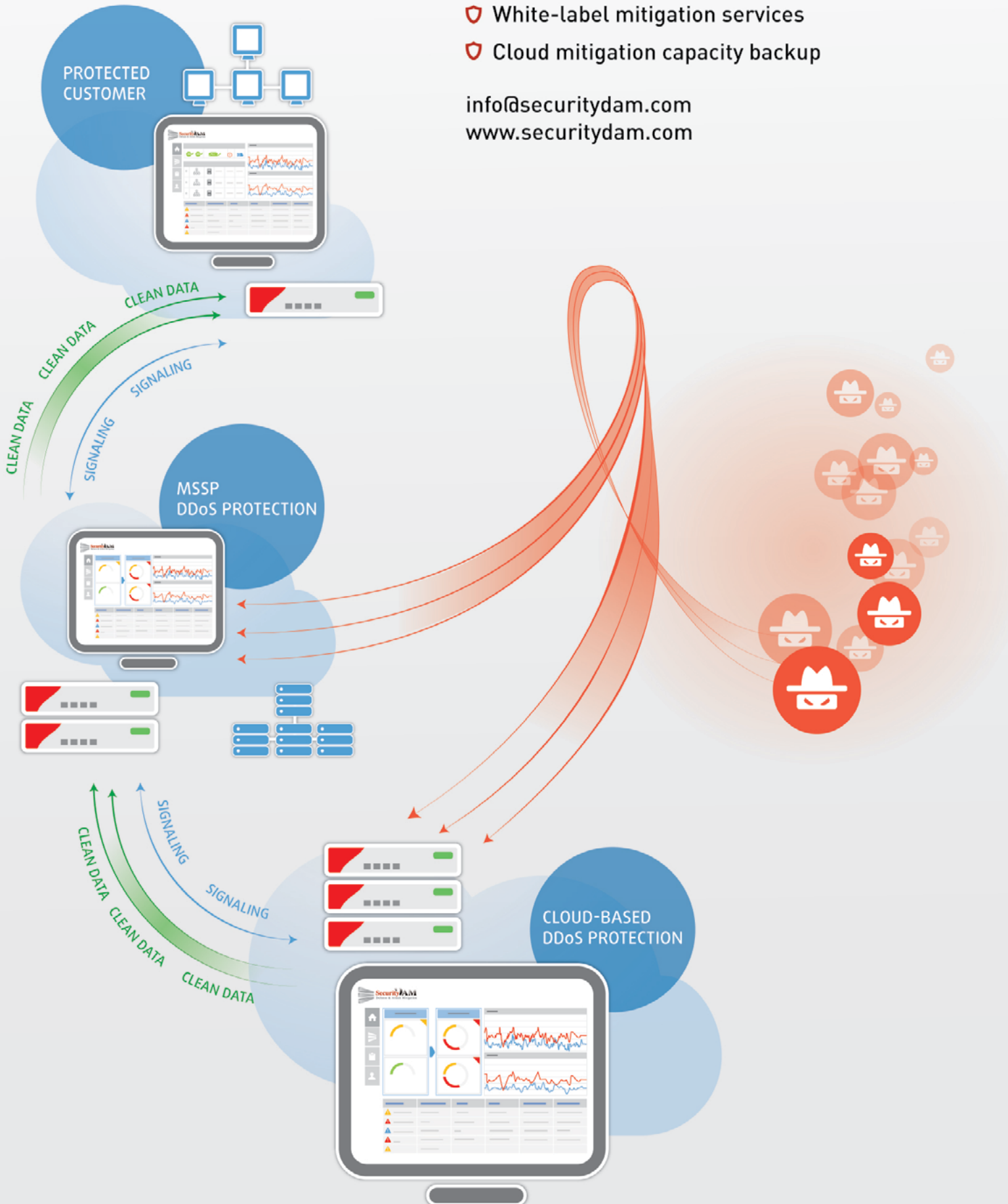
The author has over 5 years of experience in Information Security and currently works at KPMG Advisory Services. She was previously a Developer and Security Technologist at Microsoft Corporation.

HYBRID DDoS PROTECTION SERVICES for MSSPs and ENTERPRISES

Leverage SecurityDAM infrastructure and expertise to offer end-to-end DDoS Protection Services:

- 🛡️ Independent stand-alone systems
- 🛡️ White-label mitigation services
- 🛡️ Cloud mitigation capacity backup

info@securitydam.com
www.securitydam.com



Tackling Layer 7 DDoS Attacks

by Ratan Jyoti

Distributed Denial of Service (DDoS) attack is a strewn challenge where the spurious or fake packets are sent to the victim in abnormally large numbers. DDoS attempts to block important services running on victim's server by flooding the victim's server with packets. The difference with DoS and DDoS is that the attacks do not originate from a single host or network but from multiple hosts or networks which might have already been compromised. One of the main challenges here is to find the location of attackers and then block traffic at points near to the source of attacks. Layer 7 DDoS attacks target the application layer at web or mail servers (e.g. HTTP(S), SMTP, FTP, etc.) such that the service can be denied in an effective way to cause the web server to lock up or crash. Since they operate at the application protocol level which is OSI Layer 7, this attack is known as a Layer 7 DDoS attack.

Following are the major Layer 7 DDoS attack types

Table 1. Layer 7 DDoS attack types

| Types | Attack sends/target | Effect of attack is to |
|-----------------------------|--|--|
| Request-Flooding Attacks | high numbers of legitimate application layer requests | beat its session resources |
| Repeated One-Shot Attacks | high numbers of workload requests to TCP sessions | bring down the service |
| Asymmetric Attacks | "high-workload" requests in normal numbers to consume large server resources e.g. processor, disk space, memory etc. | in order to degrade the service or bring it down |
| Application-Exploit Attacks | vulnerabilities in applications | allow the attacker to gain control of the application. |

Layer 7 DDoS – Traffic Types

To be successful DDoS attackers need to change or redefine the traffic types and headers. Some of the common traffic types which are manipulated are as follows:

Table 2. Traffic types and headers

| GET Traffic | POST Traffic |
|-------------------|--------------------|
| HTTPS GET Flood | HTTP POST Flood |
| HTTPS GET Request | HTTP POST Request |
| HTTP GET Flood | HTTPS Post Flood |
| HTTP GET Request | HTTPS POST Request |

Layer 7 DDoS – Mitigation

A Layer 7 DDoS attack overloads application server and is complex, stealthy, and hard to detect because they are very similar to legitimate web traffic. Even simpler Layer 7 attacks e.g. targeting login pages with random user credentials, or repetitive random searches can significantly overload CPUs, memory and databases. Attackers can randomize or repetitively change the signatures of a Layer 7 attack, making it very difficult to detect and alleviate. These attacks can be mitigated by monitoring visitor behaviour, blocking known malicious bots, and by challenging and authenticating the visitor requests. Some of them are discussed below:

IP Filtering Database

IP address databases may be created to store authenticated and legitimate IP addresses of frequent users. IP addresses may be treated as authenticated only if it has appeared in the network for reasonably longer time after a successful TCP handshake. IP Filtering may be activated only when there is a very high network or server utilization which leads to dropping of packets. Once IP filtering is activated it starts discarding packets whose IP address does not appear in the IP addresses database. Hashing techniques may also be used to make an efficient IP addresses database. Here the challenge is to record IP addresses in a database and to remove expired IPs. The logic for addition and deletion may be built depending upon the type of network, its traffic and the application.

Trust building

Distinction between legitimate and malicious traffic senders may be made by establishing trust between the sender and the application/network. Based on the past recorded behaviour of the sender, the trust level between sender and the application/network may be established. Nature of historical traffic may be used to determine the trust level which, in turn, may be used for establishing trust level. Positive trust level may be medium at first, which after reasonable time may be treated as high, if traffic appears to be legitimate for a longer time. A sample trust building framework may be as follows:

Table 3. A sample trust building framework

| Trust | Positive Trust – interim | Positive Trust – continuing | Negative Trust |
|-----------------|--|--|---|
| Trust Level | Medium | High | Low |
| Characteristics | Normal traffic recorded during the session flooding for a shorter time interval. | Normal traffic recorded during the session flooding for a reasonable longer time historically. | Abnormal high traffic during the session flooding has been observed historically. |

A customized trust building logic may be built depending upon the type of application/network after analyzing historical data.

Challenge Response Tests- Source Verification

Challenge-response tests can sometimes be very effective in tackling Layer 7 DDoS attacks by identifying zombies or spoofed hosts. This challenge response system determines the type of browser and TCP/IP structure of the source. Some common effective challenge responses for detecting Layer DDoS attack can be:

JavaScript Challenge

In JavaScript challenge, a code of JavaScript set in the HTML is sent to the source for authentication. The source with a complete JavaScript engine can respond to the challenge correctly. Most of the times the hacking DDoS tools lack the complete browser setup to attack and can fail in this test.

HTTP 302 Redirect Challenge

In this case also the full fledged browser will respond successfully to HTTP 302 redirects. The following is an example of a HTTP 302 redirect.

| Request | Response |
|--------------------------|--------------------|
| GET /index.html HTTP/1.1 | HTTP/1.1 302 Found |

Request from non compliant source/browser may be blocked.

HTTP Cookie Challenge

In this type of challenge, the source browser's cookie handling and responding is tested. A source that fails to respond may be blocked.

CAPTCHA Challenge

It is a proven approach to determine if the request is coming from a human or if it is automated. Successful responses to CAPTCHA may place the source in the whitelist or as a permitted source. Others may be blocked.

TCP SYN Challenge

By this challenge, the TCP stack is validated through the response received from the client. Sending back a RST packet in response to the first SYN and again sending a SYN-ACK with an incorrect sequence number and evaluating response from the source can be useful in determining the spoof.

Application layer DDoS attacks are the most challenging among network security threats. Application layer DDoS attacks restricts resources, reduces revenue, and is the cause of customer disappointment among other losses. These are the most complex to resolve particularly, when the target is a Web server. Modern day firewalls with flow inspection and deep inspection can be useful in preventing and reducing the Layer 7 DDoS attacks by limiting the session and challenging the requests.

About the Author



Ratan Jyoti is currently working as a Chief Manager (Information Security) at Vijaya Bank. He has more than 12 years' experience in Information Technology and Information Security in Banking Industry. He is a Certified Information Systems Security Professional and Certified Information Systems Auditor. His major areas of expertise includes Information Risk Management, IS Audit, Cyber forensics, Information Security Management System and Compliance. He is a regular contributor to reputed International. Journals and Magazines in the area of Information Security.



**A Cyber criminal can target and breach
your organization's perimeter in less than
a second from **anywhere** in the world ...**

Are You Prepared?

ANRC delivers advanced cyber security training, consulting, and development services that provide our customers with peace of mind in an often confusing cyber security environment. ANRC's advanced security training program utilizes an intensive hands-on laboratory method of training taught by subject matter experts to provide Information Security professionals with the knowledge and skills necessary to defend against today's cyber-attacks and tomorrow's emerging threats.

ANRC's consulting and development services leverage team member knowledge and experience gained in the trenches while securing critical networks in the U.S. Department of Defense and large U.S. corporations. ANRC tailors these services to deliver computer security solutions specific to the needs of the customer's operational environment. Our approach emphasizes a close relationship with our clients as an integral part of our service. We believe we're all in the security battle together, and we view our customers as key members of our team in the fight.

TRAINING :: CONSULTING :: SOLUTIONS www.anrc-services.com

DDoS Attacks and Defense

by **Rodrigo Salvalagio, Eder Plansky Silva**

Denial of Service Attacks exist in decades, but frequency, extension and sophistication are evolving faster than companies can absorb them. This article shows both sides: how attackers are orchestrating and how companies are protecting themselves.

Denial-of-service attacks are one of the most common attacks performed by non-skilled attackers. Not because it is easy to do, but because it is easy to reproduce with the many tools available and easier for customers and partners to notice the service outage.

And because of that, this type of attack is always the center of discussion and source of headaches.

In a brief definition, a denial-of-service attack is when computational resources or services are unavailable for legitimate users due to a vast amount of requests sent to the target, or to messages crafted to exploit vulnerabilities to an application that possess a security flaw.

While reasonably easy for attackers to launch, DDoS attacks are a complex and tough threat for companies and organizations to prevent and mitigate. Some of the reasons for such a complexity is the fact that DDoS encompasses a huge number of compromised machines (or agents) acting on behalf of the attacker, and also that the traffic sent by these machines is quite similar to those sent by legitimate users.

Types of DDoS

There are 3 categories of DDoS: Attacks by Packet Volume, Resource Starvation and Application DDoS. The next table describes the differences between them:

Table 1. Types of DDoS Attacks

| Category | Description |
|---------------------|--|
| Packet Volume | This type of DDoS is based in volume of network packets sent to the victim, causing a flood in the bandwidth. Basically, there are so many packets flowing thru the link that it denies all legitimate packets to reach the servers. |
| Resource Starvation | This attack occurs when certain types of packets are sent to the victim that forces use of computational resources to its limit, resulting in a denial of service condition. |
| Application | An application level DDoS or DoS is when a specially crafted command or network packet causes the server to hang. An example of this: Buffer Overflows, fork bombs or XDoS that is based on XML files. |

The next session of this article will show readers the impact and characteristics of each type. Statistics and print-screens will help us understand why these attacks are so effective against infrastructure and application.

Packet volume

This type of attack is easy to understand, the bigger, the better. Attackers will use tools to generate traffic, often with a spoofed source, that will consume the bandwidth available.

It's measured in packets per second (pps) or bits per second (bps) and will flood the network entirely. To cause this congestion, the packets must flow from the attackers machine to the destination server, but following basic network rules, for example: The packet must contain a source and destination address, must contain a destination port, the CRC checksum must be valid and type of protocol must be filled.

If a packet is not formed or crafted with these particularities, then it is prone to be discarded by the router before it has even left the attackers infrastructure.

A UDP packet has fewer rules to follow and it has fewer packet fields to be filled, but due to its size, a single computer can generate a lot of them. Another particularity of a UDP packet is that it is not connection-oriented. In other words: Once a packet is sent there's no guarantee that the packet can reach its destination or the origin receives a response back.

But the most important thing to notice here is that both types of packets (UDP and TCP), if crafted correctly, will be transferred using the victim's link. This condition will affect every real client that tries to connect with the victim with latency. For example: If it is a retail store and clients try to reach their webpage, each access will become slower and slower until the packets take too much time to travel the network and it is discarded.

The next diagram will show a simple packet flood using 3 computers and one switch: The first one (Attacker) will send a lot of packets, the second one (Victim) will host a simple resource and the third one (Customer) will try to reach the victim's resource.

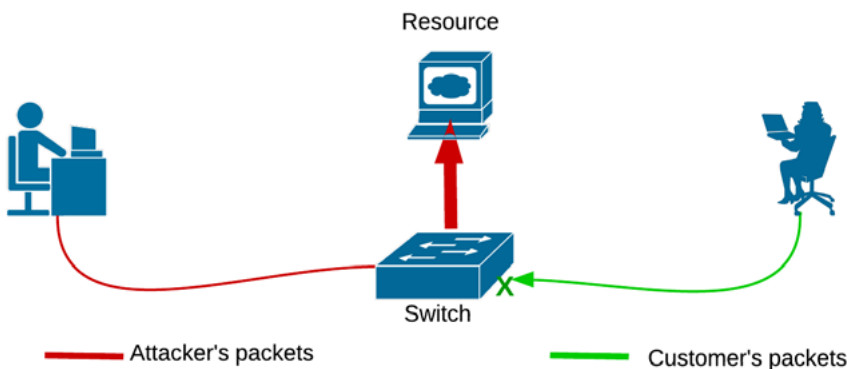


Figure 1. Packet flood using 3 computers and one switch

The next images show us a basic Synflood taking place at the attackers' workstation. As we can see in Figure 2, 23227508 packets were shot by hping3 command, even without a response back, the victim machine were affected by this traffic and the network starts to become unstable (Figure 3)

```

bash
salvabook:~ rsalvalagio$ sudo hping3 10.1.1.4 --rand-source --flood -S -L 0 -p 445
Password:
HPING 10.1.1.4 (en0 10.1.1.4): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 10.1.1.4 hping statistic ---
23227508 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms

```

Figure 2. Synflood at the attackers' workstation

```

bash
64 bytes from 10.1.1.4: icmp_seq=464 ttl=128 time=0.482 ms
64 bytes from 10.1.1.4: icmp_seq=465 ttl=128 time=0.484 ms
64 bytes from 10.1.1.4: icmp_seq=466 ttl=128 time=0.488 ms
Request timeout for icmp_seq 467
Request timeout for icmp_seq 468
Request timeout for icmp_seq 469
64 bytes from 10.1.1.4: icmp_seq=470 ttl=128 time=533.850 ms
64 bytes from 10.1.1.4: icmp_seq=471 ttl=128 time=322.729 ms
Request timeout for icmp_seq 472
64 bytes from 10.1.1.4: icmp_seq=473 ttl=128 time=496.202 ms
Request timeout for icmp_seq 474
64 bytes from 10.1.1.4: icmp_seq=475 ttl=128 time=409.178 ms
Request timeout for icmp_seq 476
64 bytes from 10.1.1.4: icmp_seq=477 ttl=128 time=323.077 ms
64 bytes from 10.1.1.4: icmp_seq=478 ttl=128 time=835.547 ms
Request timeout for icmp_seq 479
64 bytes from 10.1.1.4: icmp_seq=480 ttl=128 time=549.095 ms
64 bytes from 10.1.1.4: icmp_seq=481 ttl=128 time=333.688 ms
Request timeout for icmp_seq 482
64 bytes from 10.1.1.4: icmp_seq=483 ttl=128 time=521.635 ms
64 bytes from 10.1.1.4: icmp_seq=484 ttl=128 time=322.924 ms
64 bytes from 10.1.1.4: icmp_seq=485 ttl=128 time=561.325 ms
64 bytes from 10.1.1.4: icmp_seq=486 ttl=128 time=323.063 ms
64 bytes from 10.1.1.4: icmp_seq=487 ttl=128 time=370.267 ms

```

Figure 3. Latency time

As we can see, our switch tried to handle the attacker's request, but after a while it became unresponsive and latency takes over avoiding other hosts to communicate in the same VLAN.

What is happening here is that our switch has no throughput to send all packets to its destination and additionally can't take all responses back to the sender. This was made in a laboratory environment, but the same applies to the internet environment. It is possible to notice that a single computer can generate a lot of traffic and this can be amplified by thousands if we use a botnet. Another point we may draw attention to is the latency time increasing in Figure 3.

Resource starvation

In the field of resource starvation attacks, denial of service happens by flooding data to the target in order to drain the computational resources provisioned by the system. For example, a huge amount of crafted messages may require very long processing and exhaust CPU cycles, it may take up network bandwidth, or even consume memory resources that are allocated for those messages sent to the target system. By consuming the target's resources, access to legitimate users will afterwards cease.

An example of memory starvation happens during a TCP SYN flood attack, which takes advantage of the TCP session establishment. A TCP communication starts by a process known as a three-way handshake, as illustrated in the figure below:

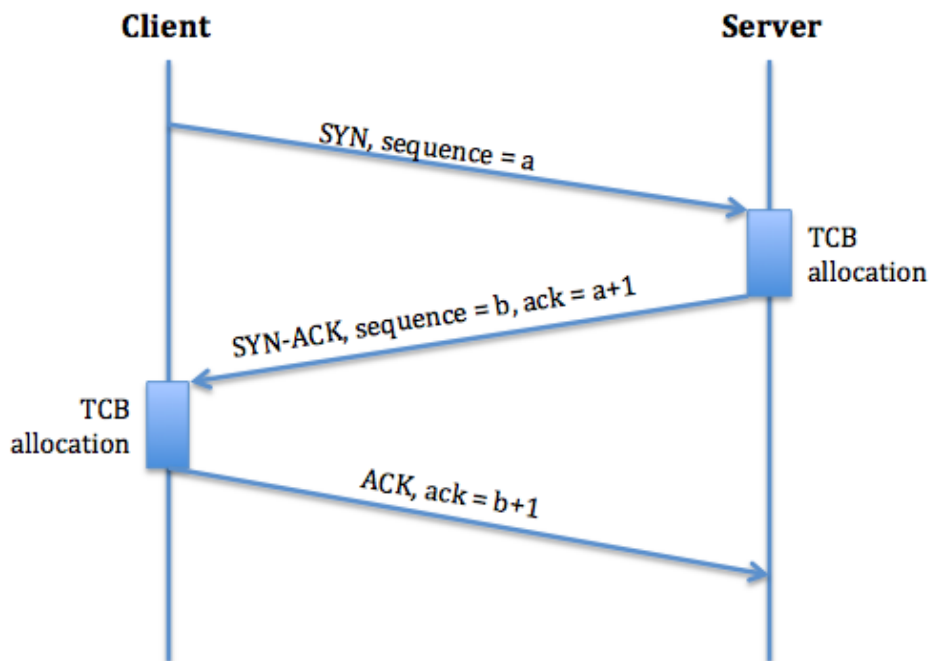


Figure 4. Memory starvation

As soon as the SYN packet is received, the server responds with a SYN-ACK message and then allocates the TCB (Transmission Control Block), which is simply a memory table used by the server to control the communication with several clients requesting services. After sending the SYN-ACK, the TCP communication is half-open and it keeps this way while the server waits for the third piece of the three-way handshake, which is the ACK sent by the client.

The trick here is the following: attackers will keep on sending massive SYN messages to the server as they were legitimate clients, however, they will never send the final ACK to complete the TCP session. On the other hand, the server will keep its TCB allocated for these half-open communications and will possibly have its memory overwhelmed. As a result, the server will deny further access to any legitimate client due to a lack of resources.

The next print-screen shows us a server in normal condition with low CPU use (Figure 5), and Figure 6 presents the server under a Synflood attack with half-open connections:



Figure 5. No DDoS running

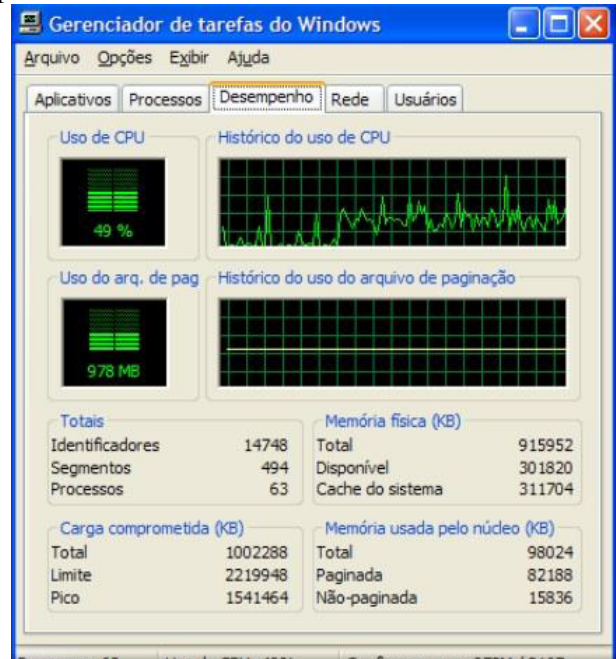


Figure 6. With Syn half open

These first 2 print screens were taken without the use of a firewall. The next image shows us the results with Windows firewall turned on:

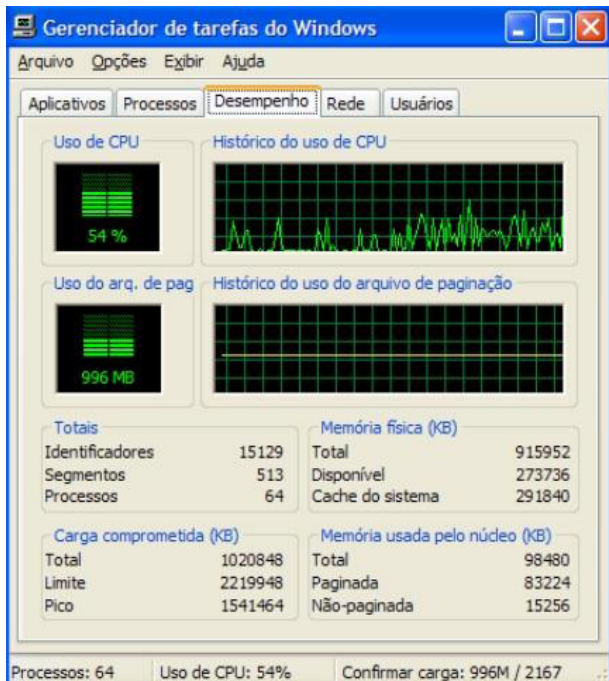


Figure 7. DDoS protected by firewall.

The firewall is not much use in this case because just to analyze the packet and match against a rule takes some resources (CPU cycles) to complete. That is the reason why DDoS attacks are so powerful, it abuses the design of the TCP protocol.

Application

As opposed to packet volume attacks, application attacks target functions inside the applications, so we may say that it's disguised to look like legitimate traffic, except it targets specific application packets. The attack on the application layer can disrupt services such as the retrieval of information or search function.

When a company is targeted in such an attack, we may infer that the main objective is to prejudice and prevent the company from obtaining money for its services. Application layer attacks are especially made to obtain advantage on website functions or user input to drain system resources.

For example: an attacker can craft a HTTP POST command, (instead of GET) requesting an image in a particular web site that increases the use of resources, such as CPU cycles, memory and disk I/O.

Let's go a little further in this topic and analyze the chain of events: using this simple attack described above, where just a JPG image is constantly downloaded by a botnet.

- A 3-way handshake is established with the client, memory and port are now reserved to complete the communication.
- Beside that, the 3-way handshake just crossed a router, a firewall, an IPS and a WAF (web application firewall), all that sending logs to a SIEM solution.
- I'm assuming here, that the router just routed the packets and nothing else, no logs, no ACL, no complex checking. But, a firewall has a lot of tasks to perform: it must check if the packet is RFC compliant, must assure that source has rights to access the destination and it must maintain audit logs, causing an intense disk I/O.
- The same occurs with the IPS, it must check against several rules and take a decision whether the communication is valid or not. And again, it must maintain audit logs and restorability.
- Now the full picture: all of these security devices are trying to make their best to protect the environment, but there are so many requests and so much I/O to write logs, that it is not uncommon when security infrastructure becomes unavailable before the target system.

In most cases, this type of attack is made using a botnet and a command and control server (C&C). Where infected workstations and cellphones can receive commands and actions to perform.

Another hypothetical situation: let's take a News company or a Portal ISP with this set of mature infrastructure and architecture. (Let's make this harder to attackers and we will presume that our News Company or ISP has a security mindset with proper defense mechanisms against DDoS).

Here is the scenario: Four routers in the cluster, holding 2 internet links from different service providers, they also bought a very expensive DDoS solution that cleans unwanted traffic and injects only valid packets to the internal network. Behind that 4 more firewalls in an active-active configuration. After that, 2 IPS in cluster, but using a fail open configuration (where a packet can flow in if it fails to check against IPS rules). In the front end, there are 8 Apache servers in a DMZ that consults 2 Oracle databases to provide content to end users and mount the home page based on device, whether a mobile device or a workstation.

In the home page there is a search function that filters user input and protects the database against SQL Injection techniques, also, this feature protects against cross site scripting and other HTTP attacks.

To provide fast access to search the content, developers made a cache function, to prevent each search request to be carried to the database. This solution improved the site's overall performance immensely.

It may appear that it is very hard to cause a denial of service condition, (indeed it is), but the cache function needs to be re-indexed from time to time when a great volume of searches is made, so customers always have the freshest content in a timely fashion. The act of rebuilding the index forces the database to read all lines of the new table and generates a new table with a new index. As we may see, this is very resource intensive and may cause the index to break.

So, a talented attacker can craft special commands to send random searches so many times, causing database slowness that may cause the website to turn unresponsive.

The impact of application flood attacks is much more severe than network flood attacks. It is much easier to detect and block a network flood attack, which is about sending a large volume of UDP and TCP floods, typically spoofed, rather than an application flood attack where the attackers are using real IP addresses from real machines and running complete application transactions, but the users are not real.

Common tools available

The next table consolidates the most common tools used by attackers to generate DDoS attacks. However, the reader must notice that this list is only the most prominent tools available, we can continue page after page mentioning a myriad of tools, but they are all birds of a feather.

Table 2. Types of DDoS Attacks

| | | | | |
|--------------|-------------|----------------|---------|---------|
| Trin00 | TFN & TFN2K | Trinity | Omega | LOIC |
| Stacheldraht | Shaft | Kaiten | Slammer | Phatbot |
| Plague | Knight | Xdcc | Mstream | XOIC |
| hping3 | DDoSIM | R-U-DeadYet | PyLoris | Davoset |
| GoldenEye | PowerDDoSer | itsoknoproblem | Stuxnet | HOIC |

This table represents a small portion of available tools an attacker can use and it is easy to download from the internet. All tools mentioned in this article were targeted by researchers around the world in complete scrutiny. Evaluating their methodology, exploited vulnerabilities, source-code, commands and internal commands.

So I strongly recommend reading each analysis performed by researchers in order to fully understand the characteristics and modus-operandi.

The article's next session will explain ways to detect a Denial-of-Service attack. It may seem strange because it's easy to check if your system is unavailable. You can check it by yourself, your customers will reach call center lines telling you, You have a problem, I can't reach your main site or I can't send any e-mails to you. However, discovering an infrastructure problem and a DDoS attack is not an easy task.

How to detect and respond to a DDoS attack?

Attackers are using sophisticated spoofing techniques and common protocols, such as HTTP, HTTPS, DNS, SMTP and etc., making DDoS attacks even more stealthy and disruptive. These attacks, which use legitimate application protocols and services, are very difficult to identify and contain; employing packet-filtering or rate-limiting measures simply completes the attacker's task by shutting everything down, causing the denial of legitimate users.

Let's drill down our current IT security arsenal against DDoS.

Firewall

They are too deep into our infrastructure and when a DDoS reaches a firewall it's already too late. I know firewalls play a critical role in IT security scenario, but it does not have qualities to impede the most sophisticated denial of service.

They are intended to filter non-compliant packets, but there's a lack of anomaly detection counter measures. The lack of anomaly-detection capabilities means that firewalls can't recognize when permitted protocols are being used as an attack vector.

IDS/IPS

Although IDSs provide excellent application layer attack-detection capabilities, they have a weakness: they cannot detect DDoS attacks using valid packets, such as Application layer DDoS. Although they offer some anomaly-based capabilities, which are required to detect such attacks, they require extensive manual tuning by experts and do not identify the specific attack flows.

Routers

Again, it can protect us from simpler DDoS based on protocols, like ping attacks, non-used protocols, etc., but attackers are evolving and are using permitted and allowed protocols.

We can use Unicast Reverse Path Forwarding (URPF) to stop spoofed attacks on the outbound side, which is generally ineffective against today's DDoS attacks because it is intended to block traffic that does not belong to the subnet. However, because attackers can spoof source IP addresses from the same subnet they are sitting behind, such a strategy can be easily defeated. Additionally, for effectiveness, it would have to be implemented in front of every router exposed on the Internet.

SIEM

During an attack, all sources described above generate a lot of logs, when I mean a lot, it is really a lot. Being sufficient to fill an entire NAS solutions in a matter of hours. That being said, we can actually detect a DDoS, but our SIEM will be out of use since it will last only a few hours and maybe you can't even log in.

WAF

Web application firewalls play an important role here, but it's not bullet proof nor silver bullet against DDoS attacks. They have some sort of anomaly detection and threshold limits, but they can't tell precisely real users from bots. When certain thresholds are reached, it will start denying services to legitimate users, accomplish attacker's first intention. WAF certainly has its importance, and can protect legacy systems without the need to develop new code, but for valid traffic, we still need better tools. And lastly,

Manually

Well, it isn't effective because in a DDoS, with multiple spoofed addresses, because there's no way you can accomplish blocking every single IP address. And maybe the firewall turns unresponsive before you can even login.

There is hope, but first we need a new mindset that understands that a DDoS is another category of attack that needs special tools, designed process, integration with development teams and proper configuration in overall infrastructure.

This new approach must be able not only to detect but to clean and tell apart good traffic from bad traffic, and this is only possible by specific solutions. Unfortunately, at the time of this article there are not open-source tools capable of delivering protection in production environments. Paid solutions do a really great job, but the two major players in the field of DDoS protection, charges customers a lot.

When we have money as a part of security field, the subject changes a little, since we will have to do some cost-benefit math, the protection strategy should not cost more than 6% of total revenue or profit brought by the business we are trying to protect.

How can you defend your company?

This may sound like a cliché, but do patch your systems in order to fix security flaws and protect them from many types of denial of service attacks. Though the most common DDoS attacks involve sending massive amount of malicious network packets, a host possessing a security bug may also shutdown with just a few specially crafted packets aimed to exploit a specific flaw.

Other forms of DDoS attacks that take advantage of normal system operations can be defended and protected as well, such as:

Protecting the host: SYN Flood

Recap the TCP three-way handshake and SYN flood discussed before: in order to establish and control TCP connections, the server maintains a TCB table with all the TCP sessions, and for each one of them, memory is allocated by the server. By flooding the system with SYN packets, an attacker could force the target to allocate memory into the TCB table for every single connection half-open by those SYN messages, which could drain all the memory available by the target.

The TCP protocol can be fortified against this type of attacks by using a technique called syncookies (for those familiar with HTTP cookies, this is a similar approach). In summary, during the establishment of a TCP session, the server will perform the three-way handshake slightly different. Upon the receipt of a SYN packet, the server will set a cookie for that connection in order to keep track of it, however, it will not allocate memory for that TCP session until it is fully established. This will protect the target against memory starvation caused by an excessive amount of half-open TCP connections.

In practice, the following are examples of how to configure syncookies in your operating system:

Linux

```
"echo 1 > /proc/sys/net/ipv4/tcp_syncookies"
```

Windows

The following versions of Microsoft Windows have Syn attack protection enabled by default: Vista, 7, 2008, 2008 R2, 8, 2012, and 2012 R2.

For other versions, try the following:

In the System Registry, set the following parameter to 1 or 2: HKLM->System->CurrentControlSet->Services->Tcpip->SynAttackProtect.

Note: for Windows Vista SP2 and Windows 2008 SP2, the following configuration may also be necessary (according to kb 969710):

Set the value data to 1 in for the entry: HKEY_LOCAL_MACHINE->SYSTEM->CurrentControlSet->Services->Tcpip->Parameters->EnableConnectionRateLimiting

FreeBSD

Syn protection is configured by default. In case you want to manually set it, the command is: `sysctl -w net.inet.tcp.syncookies=1.`

Protecting the network: IP spoofing

This is a limited approach and will basically protect your network from spoofed IP addresses that do not belong to legitimate clients.

During a DDoS attack using IP spoofing, attackers use source IP addresses that change randomly and that do not belong to them. In fact, these random source IP addresses may even appear to the target as one belonging to the destination's own network address space.

Approaches for protecting your infrastructure from this scenario include:

- **Ingress filtering:** using your Internet border network devices, apply routes, ACL or rules to drop any source address entering your network that (1) belongs to your own network, (2) that is reserved to any special purposes (such as classes A and E), (3) that is a private address (10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16).
- **Egress filtering:** as good practice, block any traffic leaving your network with a source address that does not belong to a host inside your company.

Fortifying your IT infrastructure

- **Implement layers for critical and noncritical hosts:** By separating critical from non-critical servers, you can deploy better and more tailored defenses against DDoS attacks and other sorts of threats, as well as it can simplify your IT infrastructure, that way enhancing the capabilities of your technical and non-technical defense arsenal. For instance, by placing services into different groups, you can more easily police communication flowing among those groups with a firewall, an IDS/IPS, or another technical security control.
- **Split services within hosts:** again, another approach that may sound like a security cliché, but that it is still neglected by many organizations. For instance, having a DNS server running only domain name services will simplify operations monitoring and anomaly detection. Conversely, if a DNS server is also providing a web server and is affected by a denial of service attack crafted to shutdown DNS services, this attack may also halt your web server and any other services that may be running on this single server.
- **Reduce the attack surface:** the less you show, the less you are exposed. There are a few and simple measures you can use to reduce the attack surface of your IT infrastructure, for instance:
 - Drop ICMP replies to the external host since ICMP messages may reveal route information about your internal network.
 - Segregate your DNS servers into external and internal zones. Clients out of your organization do not need to know the names you use to access your intranet, for instance. Moreover, your internal DNS server may be preserved if a DDoS (or even other attacks) compromises your external DNS servers.
 - And at last, but not least, assess your external network and analyze any open port or services publicly available. Do they really need to be there?

Summary

DDoS attacks will grow in size and complexity, challenging IT Security consultants and specialists, it poses a major threat, as well as a political activism tool (hacktivism) to internet operation, and we do not foresee any special solution that will solve or eradicate this menace.

The actual protocol based on ipv4 has design flaws, as the internet itself due to its packet-switching based infrastructure, and that impedes a complete solution. Moreover, all countermeasures available in the market are based on routing and traffic analysis, although far from perfect.

A structured planning is required to assemble hardened infrastructure, a hardened application and a strong methodology to deal with such attacks. The major challenge is to transform a total outage into a small annoyance.

We strongly recommend that denial-of-service should be incorporated in the company's risk management program, providing IT Security staff means to dilute these threats.

Glossary

Botnet: a network of infected machines (or simply bot) which are controlled by master servers, which in turn are controlled by an attacker. These bots are usually the agents who start DDoS attacks by flooding a target with network packets.

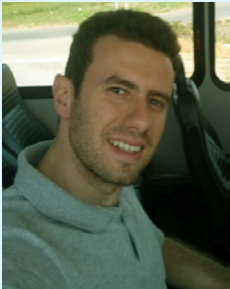
IP Spoofing: the forgery of an IP source address in an IP packet. Since the IPv4 does not force the source IP address in its packet, this field can be easily replaced by an attacker. The goal here is to trick the destination host into replying to a request to another address which is not the original sender.

SIEM: Security Information and Event Management – provides event consolidation and correlation from multiple sources in a real-time basis.

WAF: Web Application firewall – works at the application layer and intermediates the communication between clients and web application. It is able to detect and block web attacks as well as to apply virtual patching for known vulnerabilities in web servers and web applications.

About the Authors

Rodrigo Salvalgio



Leverages 10 years of experience in Pentest, Vulnerability Assessment and Incident Response in telco, banking and services organizations. With intensive hands-on approach to Information Security with innovative leadership in incident response and data center networks, can passionately deliver services to mitigate IT Risks.

Eder Plansky



Has been working with infosec for over 10 years, with experience in large telco, financial, and government organizations. He acted several years within a CSIRT team and has run complex projects aimed to mitigate security risks. He also holds CISSP and ISO 27002 certifications.

Developing for Amazon Web Services? Attend Cloud DevCon!



June 23-25, 2014

San Francisco

Hyatt Regency Burlingame

www.CloudDevCon.net



Attend Cloud DevCon to get practical training in AWS technologies

- Develop and deploy applications to Amazon's cloud
- Master AWS services such as Management Console, Elastic Beanstalk, OpsWorks, CloudFormation and more!
- Learn how to integrate technologies and languages to leverage the cost savings of cloud computing with the systems you already have
- Take your AWS knowledge to the next level – choose from **more than 55 tutorials and classes**, and put together your own custom program!
- Improve your own skills and your marketability as an AWS expert
- Discover HOW to better leverage AWS to help your organization today

Register Early
and SAVE!

A BZ Media Event

CloudDevCon



DDoS Attack

You Could Be Attacked Right Now. Are you Prepared?

by **Abdy Martínez**

A DDoS attack could happen at any moment. Even if you have a powerful and well-configured firewall, updated anti-virus and anti-spam, or good security practices, depending on the mode of DDoS attack, you will be affected. Trust me! So, are you ready to confront a DDoS attack? No? Please, before you start reading this article, ask your ISP a quote for DDoS protection service.... Hurry up!

In this article, you will learn what a DDoS is, how it works, types of attacks, and an amazing tool that permits you to test this attack in your devices (be careful). Enjoy the reading!

What is DoS? Yes, with just one “D”

Wise Wikipedia defines DoS (Denial of Service) as “an attempt to make a machine or network resource unavailable to its intended users.” Basically, it is to deny the victim access to a particular resource attempting: to “flood” a network, to disrupt connections between two devices, to prevent a particular (resource, asset, device, what?) from accessing a service or to disrupt service to a specific system.

This attack has a huge impact in the organization because depending on the target of your enterprise, this attack can effectively disable your organization. The commercial cost of such attacks can be tremendous.

Caution: It is “DoS” not “DOS,” be careful when you type it. DOS (all capital letters) refers to Disk Operating System.

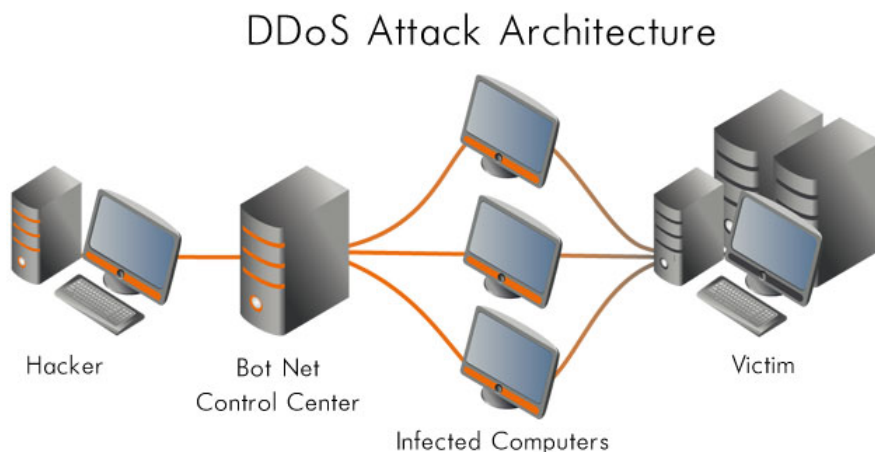


Figure 1. DDoS Attack Architecture

Adding a “D” for Distributed

DDoS refers to Distributed Denial of Service. A DDoS attack is performed by a dedicated group of attackers voluntarily using their own machines or our machines, because they can hijack machines to use for the attack. Our computers could be part of a DDoS without our knowledge.

An attacker could take control of your machine, forcing it to send massive amounts of data, spam, requests or packets to a defined target. The attack is “distributed” because the attacker is not performing a one-and-one attack; it is using multiple computers to launch DoS attack to a specific target.

Virus-infected computers used in a DDoS are called zombies (no, headshots don’t work for these cases). A large group of zombie computers is called a botnet (robot network).

You might not notice any difference, or you might notice your computer is not as fast as it used to be. That is because it may be busy participating in a DDoS attack at the same time you are working on it.

Types of Attacks

There are many types of DDoS attacks, depending on the targets: firewalls, IPS, routers, switches, servers, websites, mobile devices, even data center devices and sensors.

We can summarize them in three basic types of attack:

- physical destruction or alteration of network components
- consumption of scarce, limited, or non-renewable resources
- destruction or alteration of configuration information

Let’s review three of my favorite attack methods (remember, that the options to perform a DDoS are unlimited, your imagination is the limit):

- *Layer 7 DDoS attack*: this type of attack is one of the most difficult attacks to mitigate against because they mimic human behavior as they interact with the user interface. A sophisticated Layer 7 DDoS attack may target specific areas of an application, making it even more difficult to separate from normal traffic. The hacker group Anonymous is famous for performing Layer 7 DDoS attacks.
- *Ping of Death attack*: it works by generating and sending certain kinds of network messages that are technically unsupported but known to cause problems for systems that receive them. This attack may crash or “hang” computers.
- *TCP SYN attack*: this attack exploits “TCP three-way handshake” by having an attacking source host generate TCP SYN packets with random source addresses toward a victim host. The victim destination host sends a SYN ACK back to the random source address and adds an entry to the connection queue. Since the SYN ACK is destined for an incorrect or non-existent host, the last part of the “three-way handshake” is never completed and the entry remains in the connection queue until a timer expires, typically for about one minute. By generating phony TCP SYN packets from random IP addresses at a rapid rate, it is possible to fill up the connection queue and deny TCP services to legitimate users.



Figure 2. Hactivist collective Anonymous

Symptoms of a DDoS attack

The following symptoms could indicate a DoS or DDoS attack:

- unusually slow network performance
- unavailability of a particular website
- dramatic increase in the amount of spam you receive in your account

Even if you do correctly identify a DoS or DDoS attack, it is unlikely that you will be able to determine the actual target or source of the attack.

If you notice that you cannot access your own files or reach any external websites from your work computer, contact your network administrators. This may indicate that your computer or your organization's network is being attacked.

But don't be paranoid. Not all disruptions to service are the result of a denial-of-service attack. There may be technical problems with a particular network, or system administrators may be performing maintenance.

ISP... your strategic ally

In my experience, the most effective method to protect against the impact of DDoS attacks is to stop them before they even reach your company's network and systems. For that reason, it is so important to partner with your ISP to block the attack at the gateway (at your ISP's network), far away from your network premises.

Many ISPs offer a "clean pipes" SLA that commits to a guaranteed bandwidth of legitimate traffic. At this moment, where security should be the primary objective in your design and contracting requirements, the availability and pricing of "clean pipes" services should be one of the criteria evaluated when selecting an ISP.

If your ISP can not bring you "clean pipe" services, you should consider changing it (I am just joking... or not?). Also, you could consider obtaining this service through cloud providers that offer subscription services that scrub traffic before it enters the network.

Also, you can consider a hybrid scheme that combines on-premise and cloud mitigation into an integrated and comprehensive DDoS protection. The unique architecture provides full protection from multi-vector DDoS attacks directed at any layer: network, server, or application.

Remember to ask for a written SLA from your ISPs (local or cloud) that clearly outline their permitted responses in the face of a DDoS attack.

Perform a DDoS by yourself with LOIC

Low Orbit Ion Cannon (LOIC) is an open source network stress testing and denial of service attack application. LOIC was originally developed by Praetox Technologies, but was later released into the public domain and now is hosted on several open source platforms.

It is used to generate a massive amount of network traffic (illegitimate TCP, UDP, or HTTP packets) in order to utilize network or application resources.

Cooperation is the key! One computer running LOIC cannot generate enough requests at once to overwhelm your target. For that reason, the IRC-based "Hive Mind" mode enables a LOIC user to connect his copy of LOIC to an IRC channel in order to receive a target and other attack parameters via an IRC topic message.

Be aware, that LOIC does not make any attempt to spoof its users' IP addresses, and most volunteers running LOIC are unaware of this lack of anonymity.

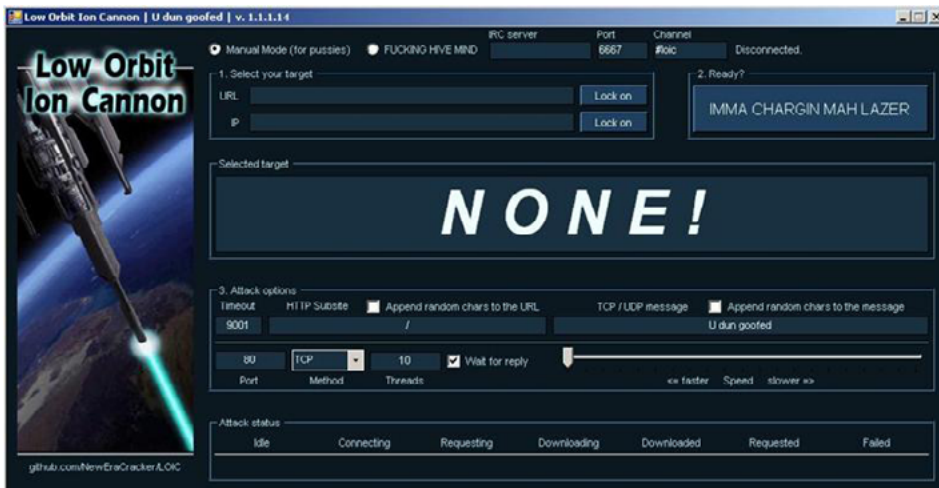


Figure 3. Low Orbit Ion Cannon (LOIC)

This effective tool is easy to use, just need three steps:

Step 1: Run the tool (yes, this is quite important).

Step 2: Enter the URL of the website or the IP address and click on “Lock On.” Then, select attack method (TCP, UDP or HTTP).

Step 3: Change other parameters per your choice. Now click on the button labeled as “IMMA CHARGIN MAH LAZER.”

With these three basic steps, you will mount an attack on the target that you select. I invite you to test it. Enjoy it, remember your imagination is the limit.

Conclusions

Call your ISP! What are you waiting for?

References

- http://en.wikipedia.org/wiki/Denial-of-service_attack
- http://www.cert.org/historical/tech_tips/denial_of_service.cfm
- <http://www.us-cert.gov/ncas/tips/ST04-015>
- <http://www.cisco.com/c/en/us/support/docs/ip/ip-multicast/14760-4.html>
- http://www.radware.com/Resources/ddos_attacks.aspx?terms=ddos
- <http://www.elithecomputerguy.com/2013/01/21/ddos-attacks-using-low-orbit-ion-cannon-loic/>
- <http://resources.infosecinstitute.com/loic-dos-attacking-tool/>

About the Author



Abdy Martínez, Network & Support Engineer at Mossack Fonseca & Co., is specialized in Network / Information Security and Forensics. CCNA Security, CompTIA Security+ (2011 objectives) and CCDA certified.

Protection Against DDoS on the Cloud

by Ahmed Fawzy

Simply the denial of service (Dos) is an attempt to deny legitimate users to use the cloud service, in the (DDoS) the same matter happened but the attack was launched through thousands of zombies or fake packets and may have led to SLA violation, loses in revenue, lost productivity ...etc.

Between 99.0% and 100% of the cloud is SLA, which means in the case of a DDoS attack this may cause unavailability of the service. This scenario will be very bad for the cloud provider because if the cloud goes down the SLA will be violated.

How do DDoS attacks work?

There are many types of DDoS attacks which may be implemented against your cloud like: Bandwidth Attack, DNS Flood, Amplification attack, Smurf Attack, Ping of death, Reflective attack and DDoS on the applications. Most of the attacks happened in two ways:

Bandwidth attacks: by consuming network resources and bandwidth using huge amount of packets that reserve space in the memory.

Applications DDoS attacks: by exploiting vulnerability in the cloud application to perform DDoS attack on the cloud., For example, the application allows users to input without validation if the attacker wrote small code to fill this application form with valid random data generated from random IP's. This will lead to DDoS on the cloud.

How to protect your cloud against DDoS?

There are many ways you should follow to get the maximum security for your cloud against DDoS; first thinking should go to being proactive in dealing with this type of incident by preparing assumptions for the incident by answering these questions:

- How to detect the attack?
- How to detect the attack type?
- How we can deal with each attack?
- What should happen before the attack?
- What should happen during the attack?
- What should happen after the attack?

In other words you should fully imagine each type of DDoS attack by creating a plan for every type.

Now we can speak about the ways of protection that the cloud (Provider or Owner) should take to avoid or mitigate the attack:

1. *Use DDoS mitigation solution:* between more than 10 DDoS mitigation solutions you can select based on the features you get like: Real-time Monitoring, Compliances, determining network attacks, Automatic Whitelisting, IP Blocking, work against application layer attacks ..etc. There are many providers that work with clouds like: CloudFlare Enterprise, Incapsula DDoS Protection, Prolexic DDoS Mitigation Services, Verisign DDoS Protection Services ..etc.

2. *Cloud IPS (Intrusion Prevention System)*: if the attack has a known signature it will make the IPS mission easy to stop the attack.
3. *Stateful Firewall*: well configured firewall can stop the attack especially if it was something like a SYN flood because there are some smart firewalls which have features related to detecting SYN flood attacks. *Black holing*: by redirecting all malicious traffic to a “black hole” non-existent server; for example.
4. *ISP support*: the ISP at the edge routers can terminate the attack by redirecting all traffic to the black hole, so coordinating with the ISP is very useful and important before the incident occurs and this is the benefit of planning for the all of the scenarios.
5. *Cloud IDS (Intrusion Detection System)*: using IDS works on the signature and the behavior of the attacks implemented against the cloud. This will be useful during the incident to determine the attack type.
6. *Upstream Filtering*: this is implemented by ISP using ACL for the client router. In this case, the Access list will be applied in the outbound direction on the ISP’s edge router connected with the customer’s router.
7. *Your cloud should adhere to external penetration Tests*: Pen testing implemented by experts in the cloud will discover the gaps between the current situation and the optimum situation you should have at your cloud. Pen testing may also guide you to many problems you can’t see because external eyes can detect more than the inside ones.
8. *Load balancer and redundant service*: by implement the load balancing concept using hardware or software you partially mitigate the risk of any denial of service attack in addition to achieve the maximum performance your client dream with, so when design cloud you should consider the load balancing and making your service redundant
9. *Simply disable the ping on your firewalls and routers*: disabling ping on the firewalls and routers stop mitigate some attacks like land attack, reflective attack and ping of death, it is cheap and effective control you should use if ping is not needed in your environment


Conclusion

The DDoS attack sometimes may be non-stoppable especially if its launched by a botnet of zombies using smart techniques scattered around the world. Since some unready companies can’t differentiate between an attacker and the real user, it is possible these companies will block legitimate users from accessing the service. thus, if left open, all the traffic might smash the cloud infrastructure. So, we should invest more time and effort for the most appropriate solution to get the full protection against the DDoS attack.

About the Author



Ahmed is Information Security Section Head in Raya, Egypt has more than 11 years’ experience in security consultation, develop security policies, security auditing, penetration testing, vulnerability assessment, code reviewing, development, security training and writing exploits; Ahmed has many certifications like (CEH-CHFI-ECSA-ITIL-MCP-MCPD-MCSD-MCTS-MCT).



Bridging the gap between business & technology

Ranked among the largest minority-owned IT services firms in the U.S. having Global Delivery capabilities

Operations in the U.S., U.K., India, Singapore and Philippines with over 9000 professionals

Leading mid-tier IT vendor with end-to-end IT capabilities spanning Technology Consulting, Application Outsourcing and Infrastructure Services

Recognized on numerous occasions by GS 100, The International Association for Outsourcing Professionals, FinTech 100 & InformationWeek

Collabera
www.collabera.com

A Simple SYN. Distributed Denial of Service (DDoS)

by Donald R-B Gooden

DDoS: as defined by Wikipedia: is an attempt to make a machine or network resource unavailable to its intended users. This method is one of the oldest ways to hit a system...this method works on so many types of systems...web pages, apps, internet connections, smart phones, dumb phones, old phones, with sooooo many different ways to make this happen... DDoS the distributed part is the piece here that is the difference when it comes to the networking aspect of things...distributed.... distributed ...a denial of service...I'm unable to get to my online banking page, my email isn't working today, I can't log into my system from home, from my cell phone, from my desk, from my...from my...from my...these are single system service denials...just single... Wikipedia: as clarification, DDoS attacks are sent by two or more person, or bots. Denial of Service (DoS) attacks are sent by one person or system.

DDoS is distributed...one person many nodes as in a botnet or zombie army of machines...or a legion of people with a single machine, or multi-machines, or even multi-machines, multi legions multi bots, multi zombie, and multi geo graphic and multi military...so many ways, so many services, so many ways...

A simple SYN...

The networking aspect of the distributed ...the TCP/IP...the SYN...the hello to the network...to the machine...to the switch...the hub...well maybe not the dumb hub but the dumb phone that would ring the house to alert a call, the smart phone that connects to the app that connects to the network that connects to the email that connects to the memo to the manager that connects the manager to the customer or to the CEO or to the administrators that set the meeting. The networking aspect of the DDoS...the distributed botnet system of a legion of hacked up nodes that will SYN when its time to and the new era of mobile phones with hacked up apps that allow any kind of hat to launch on a network at will...take a look at the OWASP TOP 10 and look at all the ways that lack of system control leads to a DDoS attack...not just the fact that DDoS has been a way and means of proving that you have what it takes to Syn on the net but the ways that these types of things happen...A DDoS SYN launched at a network...not hard...really not hard...just look at search results on the net for testing TCP/IP connections, or packet generators (that will add the UDP section I'm leaving out of this...great to know same attacks bigger files...aww yeah...)...the SYN is just the first part of this...there are the rest of the testing tools that come along with these admin tools that set much more in motion...DDoS is as simple as ping -t to IP address from more than one person or more than one machine...as simple as just sending the hello...as simple as a web page request of http get...as simple as...

Wikipedia

Denial of service attacks are considered violations of the Internet Architecture Board's Internet proper use policy, and also violate the acceptable use policies of virtually all Internet service providers. They also commonly constitute violations of the laws of individual nations.

As simple as...a fake email with a link to download an app onto your home system, your work system, your smart phone, a hacked up node from just a click...your network is overseas, you have managed providers, you have analysts, you have, you have, you now have a hacked up node...these are the more simple as approaches to the DDoS SYN of the net...the others well more complex...the legion with multi locals and multi systems that can check out the edge routers on each one of your entry points into the network that knows that the dumb phone will ring at a desk and someone will answer or they can leave a voicemail...the voice mail "you just got hacked"...gotta be one of the more funnier ones on the list...I've gotten the "I think my network is hacked" call, I'm sure it's better than that call...that one was firewall management software

and module all in one setup...the box got hit and the attackers had some keys to play with...the guys were huddled around this one for sure...that kind of firewall hacked really...how can you tell...load this software, check this router, watch these machines for services, look at these logs...ftp running on the wrong port gave it away but you have to do all the steps ya know.

...to simple is as simple is...the botnets of the home machines that are for surfing the web or online music or videos or banking...pls leave the online banking part out when you are system talking, or come up with a joke about it some how...the old man if that guy had wanted money instead of just making a point...the legion of men and women in the industry for years have made this point over and over and over...separate things when it comes to systems and your money...your system and your taxes, your system and the documents that can have a boat in bay that belongs to a guy that has your name, your cash, and all of your fun...separate...separate it...the botnet that has your music collection from the pop charts not as bad as the one that has your tax returns, and retirement fund information...separate it...simple is as simple is...when a legion launches a botnet attack from an army of machines that will ping the network to death, or launch a few thousand half open handshakes to a web server, or hit the router just outside the network with SYNS or SSH, or even just connect to the web apps so they can launch a DDoS layer 7 application attack while the system is under heavy load from the botnet attack on the web box to get the system to run at a high level then crash it with a few hundred requests that look like they should be happening but are just taking place to have the administrators looking into high load on the processors, or a lot of requests to the SQL machine, and the one that actually hit the box is hidden in the log somewhere...great somewhere in the log, and now all the data from the box was just copied down via HTTPS: from the web server and that was encrypted, and now I have to go over the logs to find out what just happened because I'm not really sure, but I know that all the files were accessed... now we know we have been accessed, and have to call in the team...our team, the consultant team...the local FBI branch, who, when...you mean call them now...YES now...the simple SYN...just a simple SYN...

Now we have the FBI guys at the office, and the CEO is not happy, not happy at all...the system he ordered to be put in place so that the company would pass inspection has just been downed, or the files that were on the network are gone or not gone but accessed and copied off, and someone with access to the Department of Justice would like to know why...why in the world did all the files that had name, social security numbers, health care records, or all that and addresses, and date of birth just flew encrypted out of the network...yep the guys with access to putting people in the Federal prison system want to know why...why the time out settings were just a little too high for the system you have and that that was covered in the training manuals, nobody bothered to test the new system and because just installing that new hardware covered the checkbox in the inspection.

As simple as...

Reading and reading and reading and reading, and reading and more reading...The Sans Institute has some great documents on the subject (I read up Dr. Martin Reed's Denial of Service attacks and mitigation techniques: Real time implementation with detailed analysis before writing this...as well as some OWASP documents... The IEEE referenced in the Doctor's SANS document should most defiantly be on the list... the Infosecinstitute.com has a layer seven DDoS attacks article that is worth the read...

About the Author



Mr. Gooden is an IT Analyst / Project Manager with over thirteen years of experience in information technology (IT), including four years of cryptography, contingency theater support, telephone message switching, mobile communications, internetworking computer systems and computer/communications security for the United States Air Force. Donald also has five plus years of experience working with two different security solutions integrators as a security consultant / project manager on CheckPoint Firewall/VPN products, NetScreen firewalls, Nokia security, Rsa, Alladin and Citrix product line. He has over six years of dedicated experience in information security/project management and has expertise with: security policy and procedure development, security architecture planning, change management, audit programs, firewall installation, configuration and management. The bulk of those installations being meshed installation (VRRP) configuration. He is also an expert in the installation and configuration, and technical security documentation for Virtual Private Networks (VPN). Donald has successfully completed and received certifications as a CheckPoint Certified Security Engineer/Expert, Nortel Networks Enterprise Security Primer, and Citrix Administration.

Meet the Developer-Friendly Payment Solution



3 easy steps to optimized checkouts:

1

Create the checkout page

With Gate2Shop, you can optimize your payment pages by using ready-made templates or by customizing payment pages to your site look and feel.

2

Test and optimize

An effective payment page variant testing tool, A/B Testing helps you gain insight into user behaviour, increase payment conversion in the short and long term.

3

Accept payments worldwide

With dozens of alternative and local payment methods offered in multiple currencies, the personalized checkout allows you to reach users from all around the world.

✓ Easy integration ✓ Cross-platform ✓ Secure



Call for a free consultation: +44 20 3051 0330
www.g2s.com

 **Dr.WEB®**
since 1992



Dr.Web 9.0 for Windows — the rapid response anti-virus

1. Reliable protection against the threats of tomorrow
2. Reliable protection against data loss
3. Secure communication, data transfer and Internet search



© Doctor Web
2003 — 2013

www.drweb.com

Free 30-day trial: <https://download.drweb.com>

New features in Dr.Web 9.0 for Windows: <http://products.drweb.com/9>

FREE bonus — Dr.Web Mobile Security:
<https://download.drweb.com/android>

