Deloitte.

RootedCON

DoS
A practical approach

Gerardo García Peña

Index

Introduction

What is it?

- Destroying or flooding the victim's key assets, forbidding access to its resources.
- The attacks are usually executed from the Internet.
  - But this vector is only one amongst a lot.
  - There are a lot of techniques and vector attacks such as worms and trojans have demonstrated.
  - These factors makes this phenomena hard to understand.

Introduction

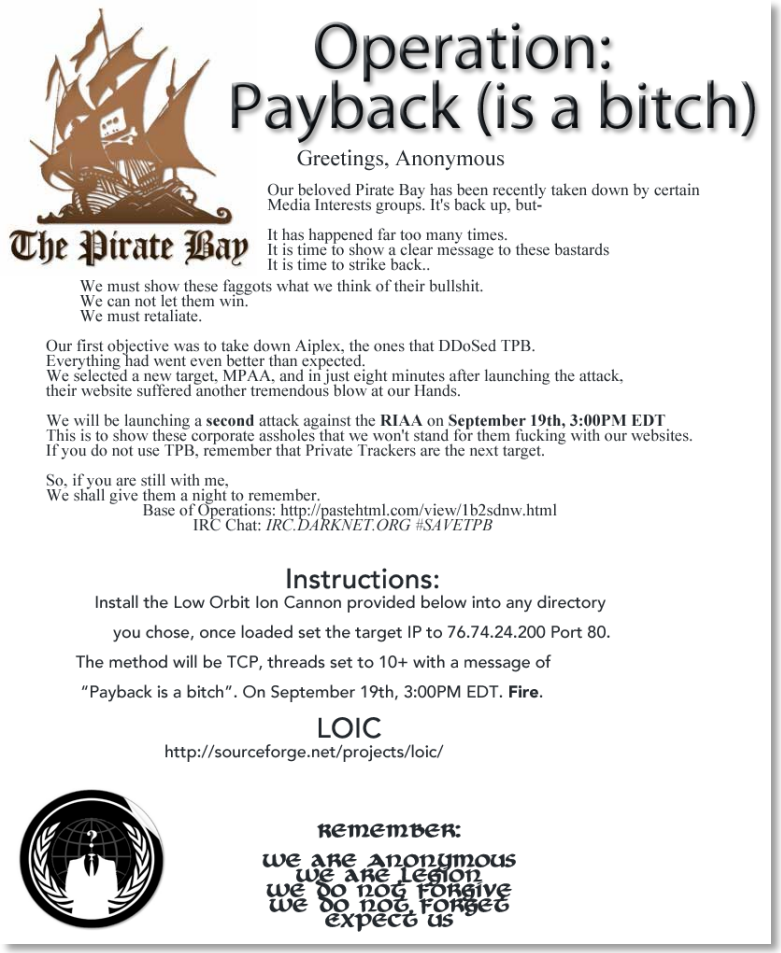
Why?

- Personal motivations or vengeance
- Prestige (in order to gain respect)
- Hurting a competitor
- Politics
- Indirect attacks (the victim is the one in need of the service, not the service provider)

Introduction

Risk types

- Financial loses
- Image damage
- Systems disruption
- Data integrity impact
- …



The Pirate Bay

Operation:
Payback (is a bitch)

Greetings, Anonymous

Our beloved Pirate Bay has been recently taken down by certain Media Interests groups. It's back up, but-

It has happened far too many times.
It is time to show a clear message to these bastards
It is time to strike back..

We must show these faggots what we think of their bullshit.
We can not let them win.
We must retaliate.

Our first objective was to take down Aiplex, the ones that DDoSed TPB.
Everything had went even better than expected.
We selected a new target, MPAA, and in just eight minutes after launching the attack, their website suffered another tremendous blow at our Hands.

We will be launching a **second** attack against the **RIAA** on **September 19th, 3:00PM EDT**
This is to show these corporate assholes that we won't stand for them fucking with our websites.
If you do not use TPB, remember that Private Trackers are the next target.

So, if you are still with me,
We shall give them a night to remember.
Base of Operations: http://pastehtml.com/view/1b2sdnw.html
IRC Chat: *IRC.DARKNET.ORG #SAVETPB*

Instructions:

Install the Low Orbit Ion Cannon provided below into any directory

you chose, once loaded set the target IP to 76.74.24.200 Port 80.

The method will be TCP, threads set to 10+ with a message of

"Payback is a bitch". On September 19th, 3:00PM EDT. **Fire**.

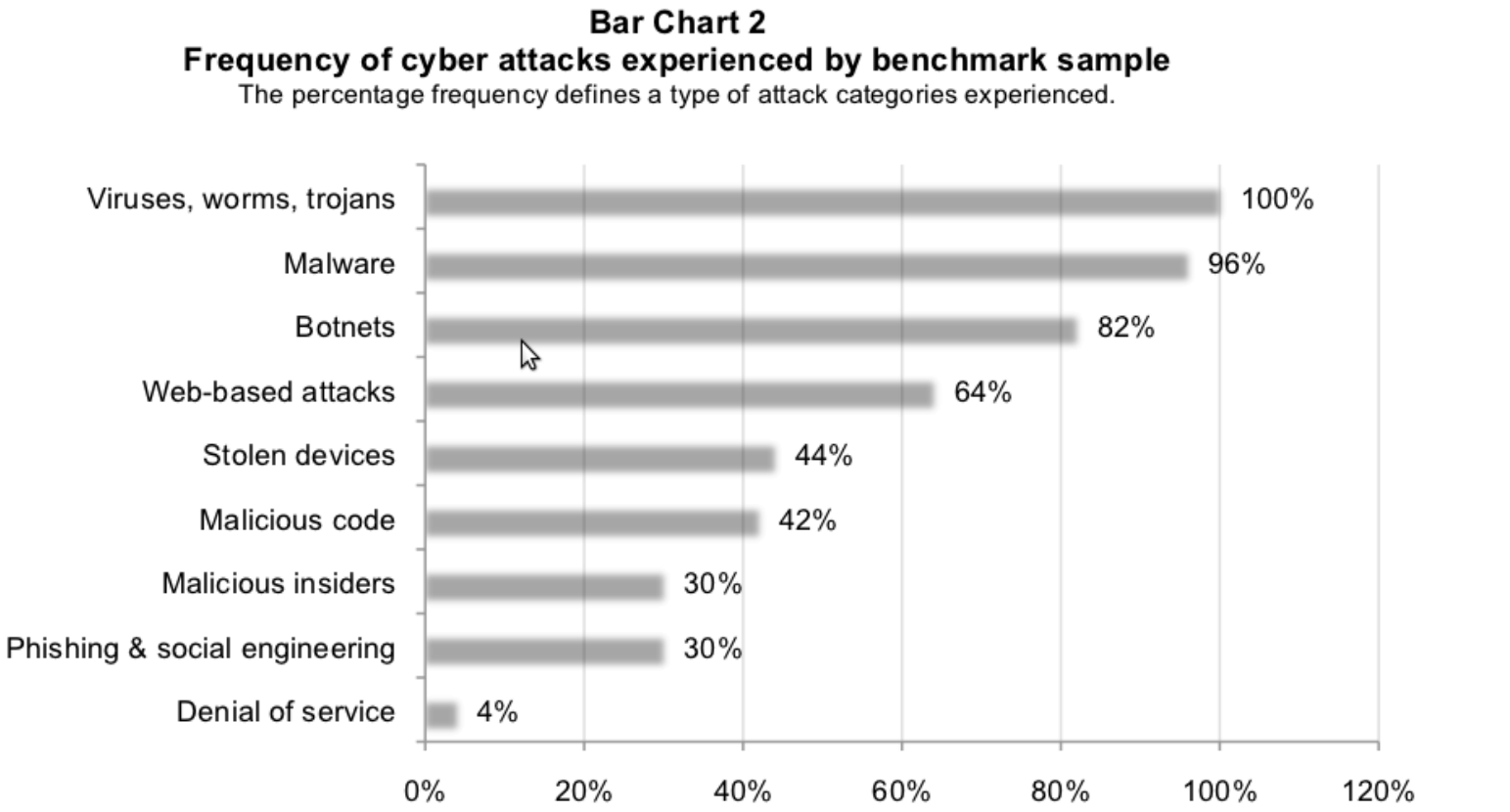
LOIC
http://sourceforge.net/projects/loic/

REMEMBER:
WE ARE ANONYMOUS
WE ARE LEGION
WE DO NOT FORGIVE
WE DO NOT FORGET
EXPECT US

Introduction

What is its probability and impact? (1/3)

- It is not a “common” risk,
- but it is “in crescendo”

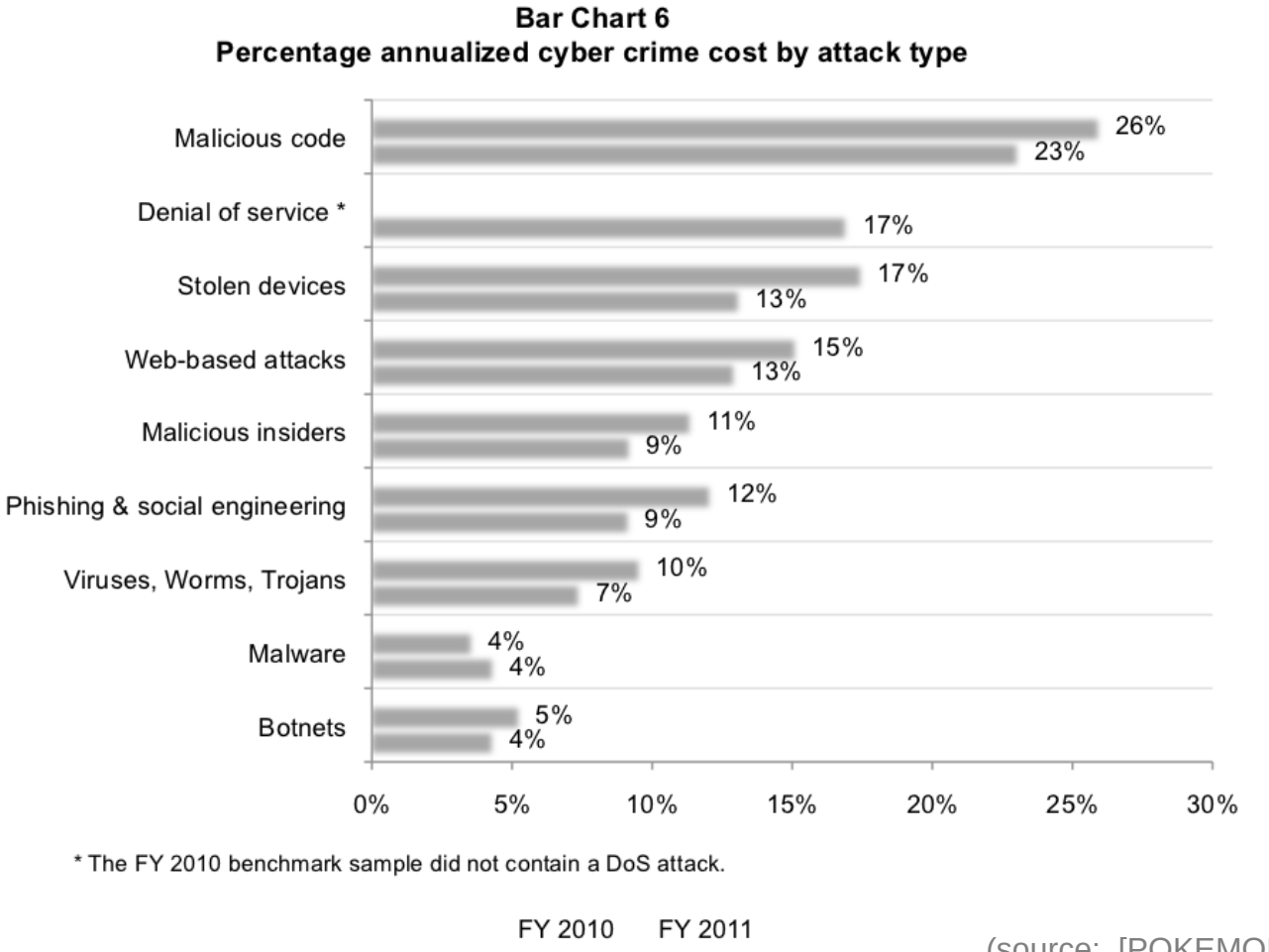


(source: [POKEMON])

Introduction

What is its probability and impact? (2/3)

- Despite not being very likely, its impact is important.

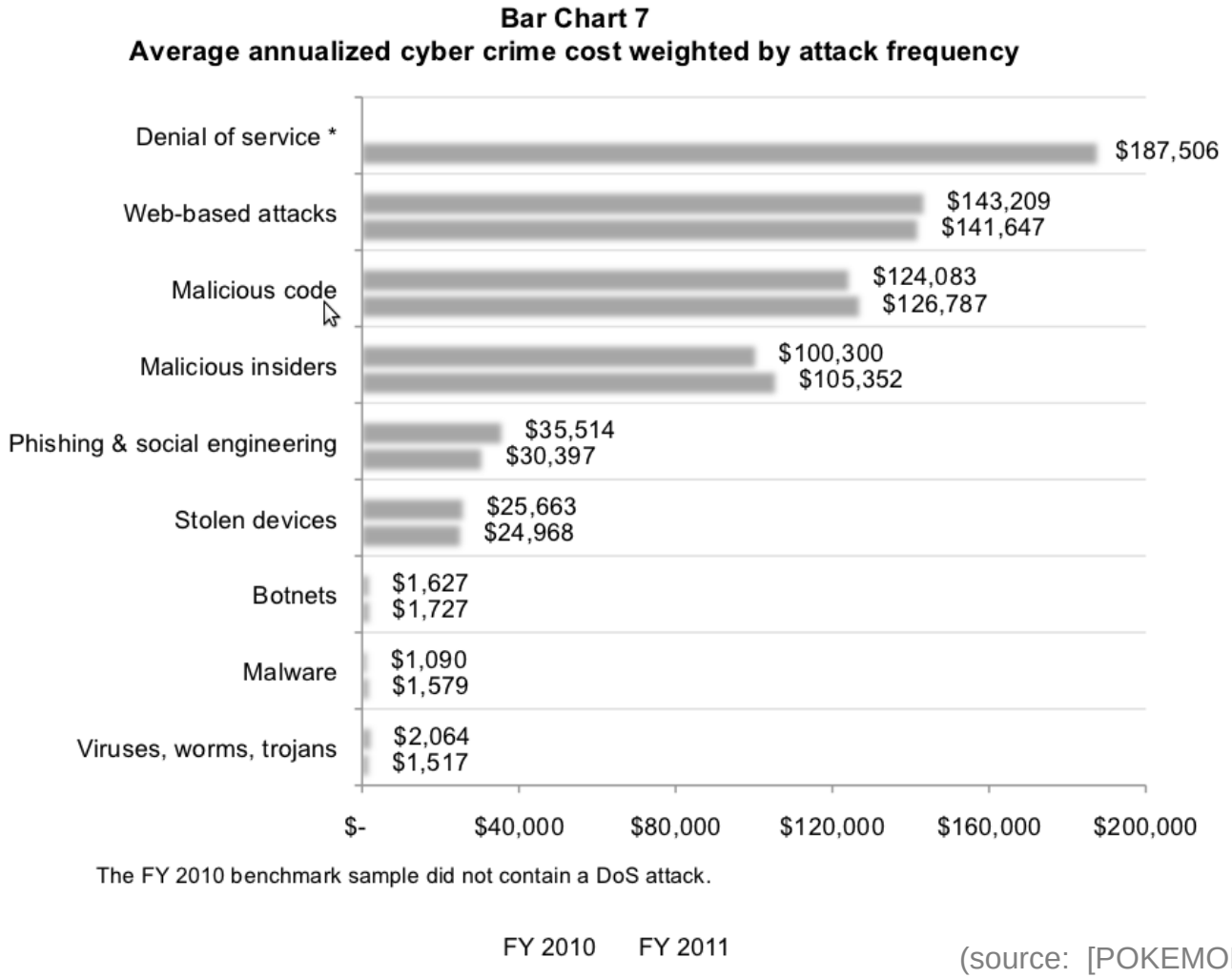


(source: [POKEMON])

Introduction

What is its probability and impact? (3/3)

- But if we calculate the average annualized threat costs weighted by their attack frequency…



(source: [POKEMON])

Introduction

Why does this happen?

- This risk can only be avoided by people; they don't like to confront it.
- It is difficult to test it because of its possible implications.
- The mitigation usually requires coordination between different teams and only in rare cases it can be mitigated on a single point.
  - Multidisciplinar
  - Extreme situations
- Because it is passed like a time bomb amongst theirselves.

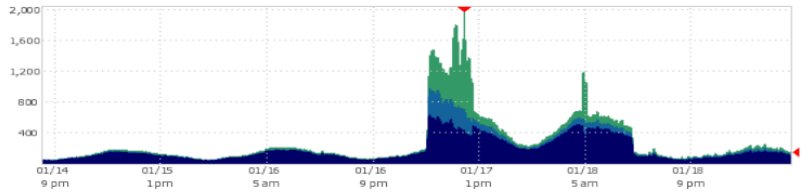
(Vote Cthulhu; Why choose the minor evil?)

Index

Analysis methodology

DoS management

- Preparation
  - Actuation plans,
  - Platform adaptation, and
  - Agreements with ISP and other providers
- Inspection
  - Attack seizing and identification
  - Attack monitoring
- Absorb
  - System tuning,
  - Service prioritization,
  - Assets protection
- Deflect
  - Block the attack,
  - Information delivering prioritization
  - Load or attack limitation



Before Attack

Agent	% Visitors
IE 8.0	21%
Firefox 3.6	16%
Feedfetcher	13%
Safari 533.19	9.7%
IE 7.0	8.8%
Other Browsers	32%

Location	Requests	% of Total
USA	66,795,330	36%
United Kingdom	16,879,792	9%
France	11,563,374	6%
Germany	11,344,749	6%
Canada	8,149,259	4%
All Other Countries	66,434,318	40%
Total	185,436,822	100%

During Attack

Agent	% Visitors
Opera	32%
Firefox 3.0	8.3%
Firefox 2.0	7.5%
IE 7.0	6.5%
Firefox 1.5	5.8%
Other Browsers	40%

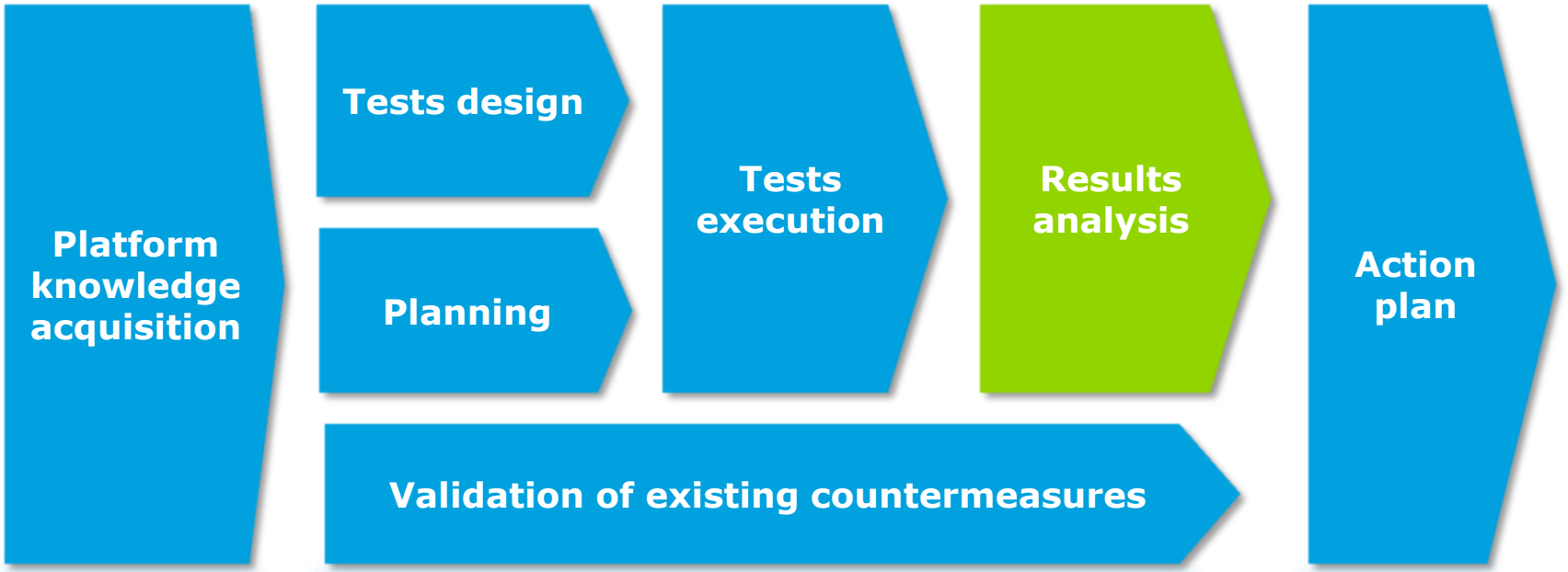
Location	Requests	% of Total
USA	75,068,890	27%
Brazil	26,586,001	10%
United Kingdom	14,759,973	4%
Canada	9.643.411	4%
Mexico	9,234,362	3%
All Other Countries	140,125,542	50%
Total	275,418,179	100%

(source: [AKADDOS])

Analysis methodology

Audit, evaluation and preparation methodology (1/2)

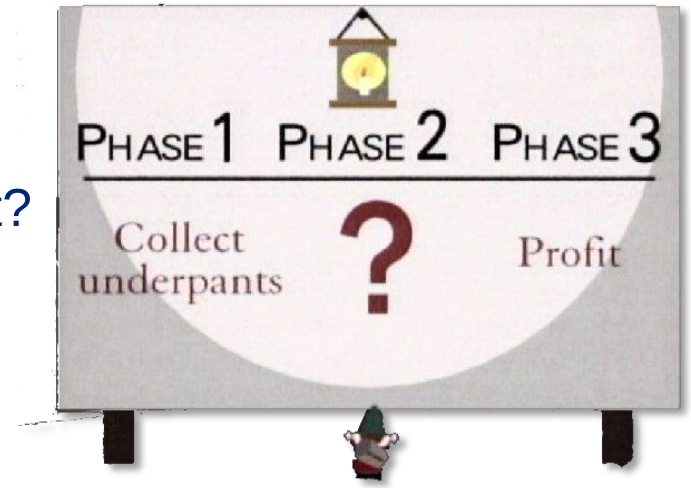
- These projects are commonly known as
  - Denial of Service
  - Performance evaluation
  - Stress tests
  - …
- And they always share the same look'n'feel



Analysis methodology

Audit, evaluation and preparation methodology (2/2)

- Knowledge acquisition
  - Network map analysis
  - Communication and protocols analysis (pcap analyzer)
  - Identification of services and assets chain
  - Monitoring systems identification
    - ❖ HP OpenView? SNMP? WEBEM/WMI? iftop?
- Tests design
  - Evaluate and analyze assets using a DoS taxonomy
  - Identify dependencies in the target system
  - Decide how to monitor systems and network performance
- Planning
  - How will we execute the tests?
  - Which tools will we use?
  - Do we need to develop new tools for this project?
- Execution
- Finally the existing countermeasures are validated and, if needed, we design or propose new ones.



Index

Our tools

Tools referenced in next slides (1/5)

- Chucuchu
  - A web application DoS tool
  - A “little” more powerful than LOIC
  - Designed for high performance
  - “easy configuration”
  - Highly customizable so it is easy to adapt it to any web based application
  - It is able to work with transactions, not only with simple requests and responses.
  - It is able to take decisions depending on the responses.
  - Multiprocess and distributed.
  - It saves a lot information like all conversations, timestamps, and any critical detail for a deeper analysis.

Our tools

Tools referenced in next slides (2/5)

- DoS:IS (free & open-source)
  - A low-level network DoS tool
  - This tools allows/provides
    - ✓ Mechanisms for working at a low level with several network protocols
    - ✓ Network simulation conducted by a simple description language that makes possible to forge data flows and communication scenarios in an easy manner
    - ✓ DoS based on protocol attacks, raw packet-forging, etc.
    - ✓ Modular and easily extensible.
    - ✓ Provides an API to ease the development of new attacks.
  - Very unstable, but it is an excellent repository of low-level network code
  - Somebody wants to contribute?

https://github.com/killabytenow/dosis

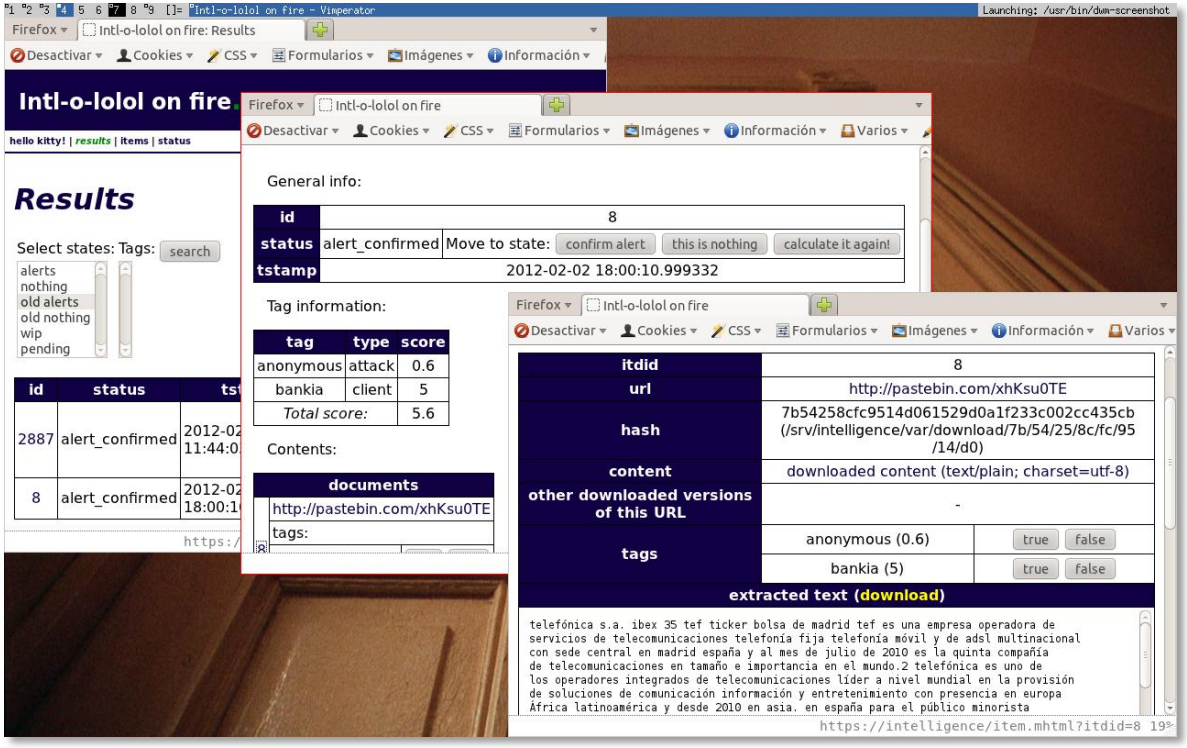
```
# configuration
? THOST="127.0.0.1"
? TPORT="80"
? SRT="5.0"
? RT="30.0"

# script
#0.0 ON 1 LISTEN DEBUG
0.0 ON 1 LISTEN
+.0 ON 2 SEND DEBUG
+.0 ON 3 TCP OPEN DST $THOST $TPORT PAYLOAD FILE("tcpopen.payload") DELAY 100 DEBUG
+.1 ON 4 TCP RAW DST $THOST $TPORT FLAGS S PERIODIC [ 0.2 ]
$SRT OFF 4
$RT OFF *
```

Our tools

Tools referenced in next slides (3/5)

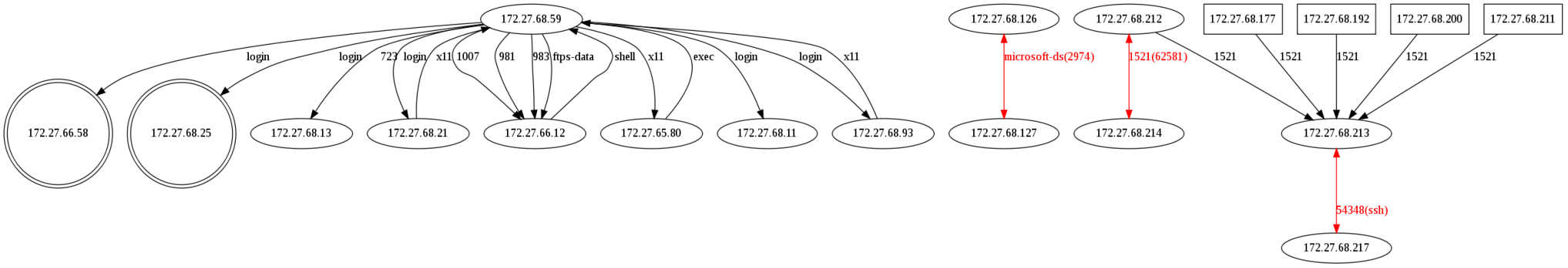
- Intelligence
  - Software for early risks detection in Internet.
  - It is an expert system that analyses a lot of information sources looking for:
    - ✓ Possible threats and attacks
    - ✓ Vulnerability disclosures
    - ✓ Information leaks
    - ✓ Interesting news



Our tools

Tools referenced in next slides (4/5)

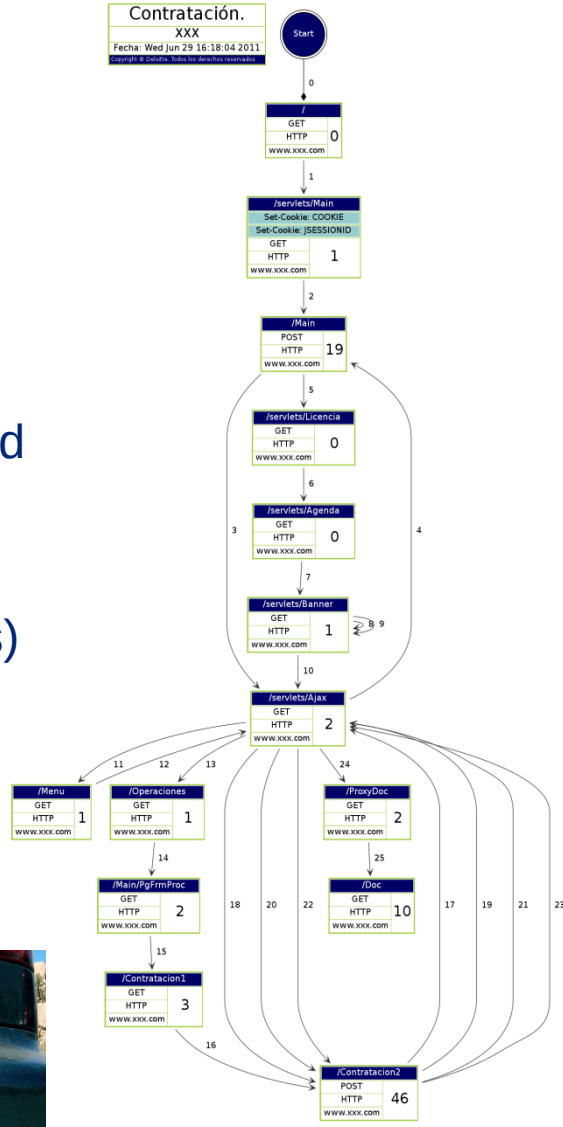
- PCAP analyzer
  - Designed for monitoring and systems analysis
  - Based on libpcap and detects dependencies and interactions among machines in a network.
  - It can be used both in real time and post-analysis
  - It builds networks maps where the following parameters are shown:
    - ✓ Dependencies
    - ✓ Relationships
    - ✓ Communication directions and their throughput
    - ✓ Network problems
  - Output can be rendered in several formats: graphical, HTML reports, Excel sheets or even dumped to a SQL database.



Our tools

Tools referenced in next slides (5/5)

- Atackw
  - Software for exhaustive analysis of web application transactions
  - Amongst its functionalities we use its profiling tools and features for
    - ✓ web transactions monitoring, and
    - ✓ detect heavy transactions/forms which could be exploited to trigger an application level DoS situation
- ANO.LOLCATHOST.ORG (scenery)
  - It is a “happy” pics repository published in the Internet (lolcats)
  - Developed under best SSDLC practices.
  - It is a test scenery for testing DoS attacks and practice with anti-DoS techniques.



Index

Taxonomy

The big unknown

- The first problem we have to face is to understand the problem.
- We cannot solve a problem without understanding it well,
- A DoS is a complex phenomena that can affect us in a lot of different ways and in many levels.
- But when we look for a classification we find a limited vision of it:
  - DoS
  - Distributed DoS
  - Botnets
  - And a little bit more…
- It is necessary to have a nice taxonomy.
- What advantages does a taxonomy have?
  - It helps to identify the assets that could be threatened by a DoS.
  - It shows how and on which levels we can mitigate it
  - It can reveal the different techniques an attacker could use.
- Everything aforementioned will help us to make a plan to manage it in a precise and detailed way.
  - Better than thinking on UDP, TCP and ICMP.

Taxonomy

Existing taxonomies (1/3)

- The most classical article is "*A Taxonomy of DDoS Attack and DDoS Defense Mechanisms*" [TAXDOS].
- It presents a detailed and a complete taxonomy of attacks and defenses.
- Example, the attack taxonomy:

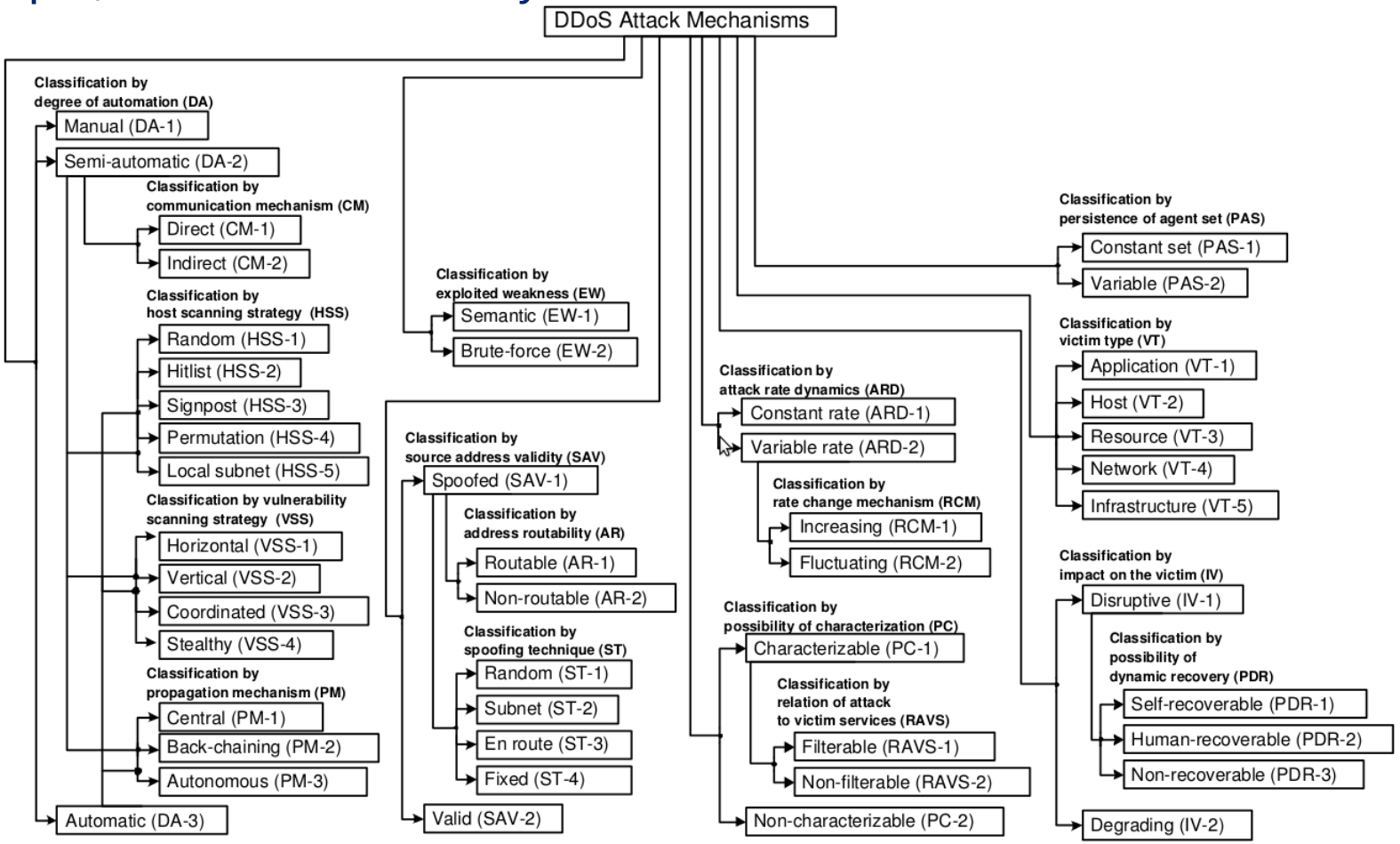
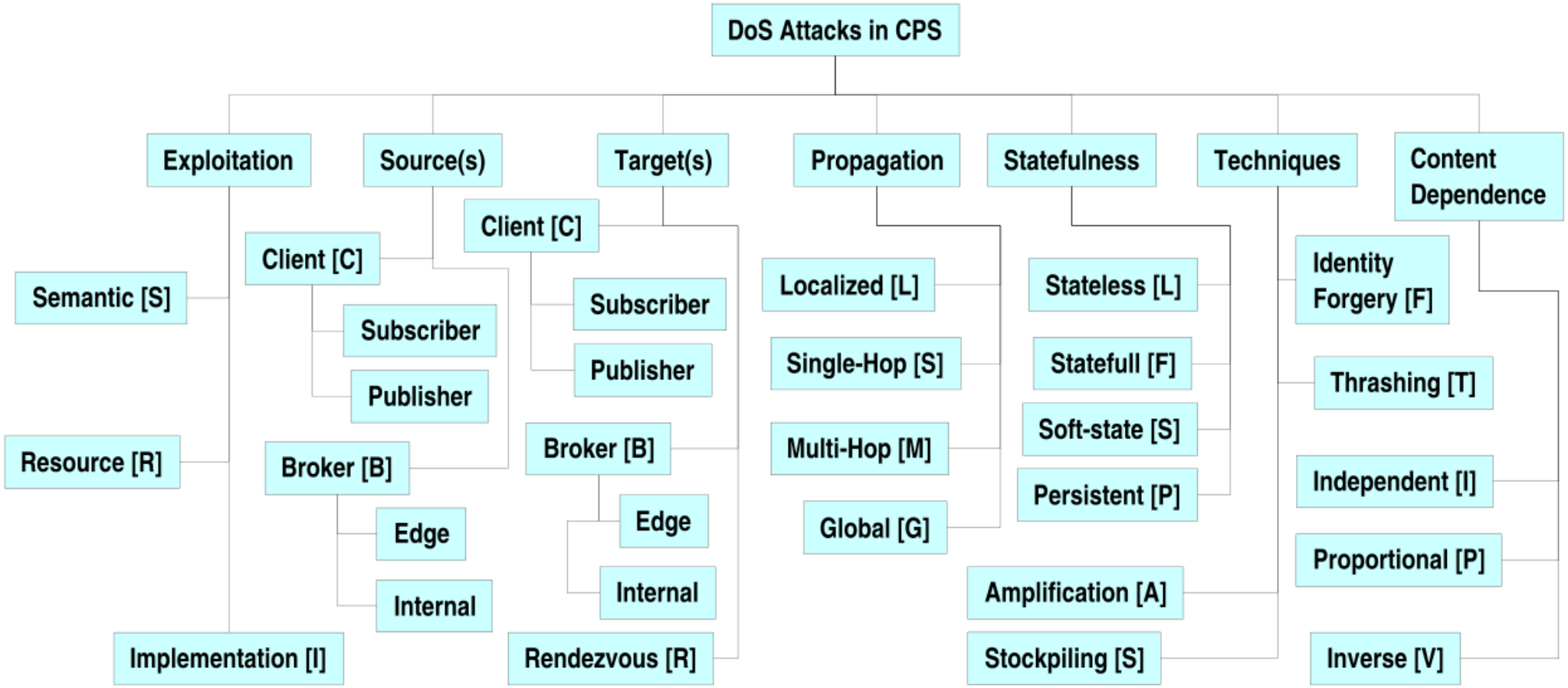


Figure 1: Taxonomy of DDoS Attack Mechanisms

Taxonomy

Existing taxonomies (2/3)

- Another article, "*A Taxonomy for Denial of Service Attacks in Content-based Publish/Subscribe Systems*" [TAXCPS] provides a taxonomy of DoS against CPS systems.



Taxonomy

Existing taxonomies (3/3)

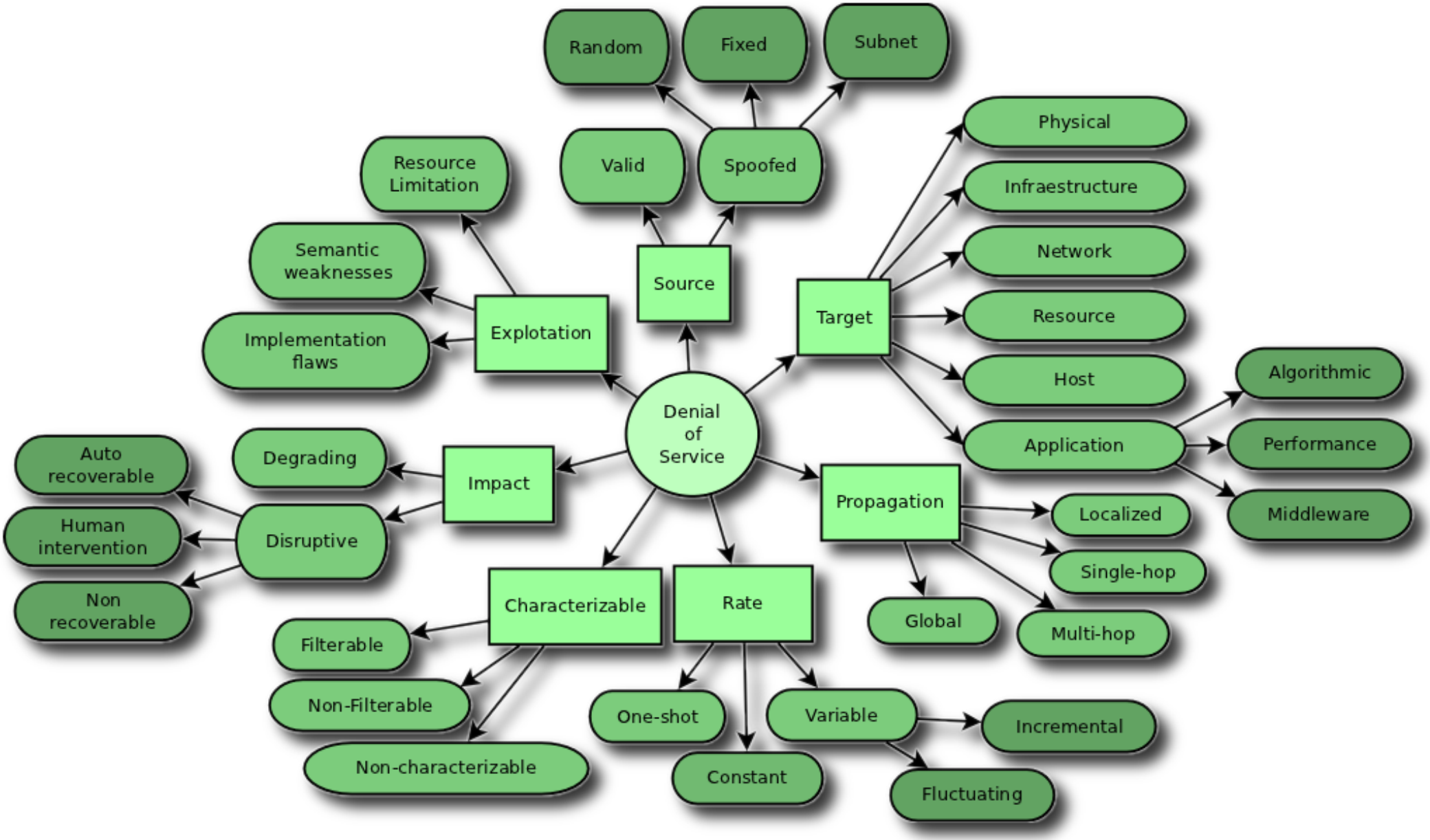
- The basic problem is that it does not exist (or we have not found it) a taxonomical classification comprehensive enough to cover all known DoS attacks.

- For this reason, we have used the existing taxonomical classifications for
  - Building a generic taxonomy to help us when executing our projects.
  - But being simple and clear enough
    - ✓ to see the forest behind the trees
    - ✓ help us to perform quick analysis on an existing platform to DoS risks exposition
    - ✓ evaluate the main mitigation techniques we can apply on it

- This approximation is essential for us because:
  - It helps on building the asset enumeration
  - See how these assets will be used during a DoS
  - Detect which attack vectors could be used
  - And how to mitigate them

Taxonomy

Example taxonomy



Taxonomy

Example taxonomy (1/5)

- Following is an example taxonomy that we could use during the execution of a DoS project:

	Classification		Description	Examples
Exploitation (mechanism)	Resource limitation (bruteforce)		Great quantity of connections/request that exhaust victim resources	• Deliberated attack • Linked in a mainstream portal • Successful business case
	Semantic weaknesses		Exploits a specific protocol or system characteristic	• An insecure protocol • A feature can be used to trigger a DoS situation
	Implementation flaws		Exploits flaws in protocol or software implementations	• A bad software implementation • An attacker exploits a service in a way that needs an excessive resource consumption. • A DoS countermeasure is subverted.
Source	Spoofed	Random	Packets seem to come from a random fake source.	• Network level DoS attack, probably of SYN flood, ICMP flood or UDP flood type.
		Subnet	Packets seem to come from specific networks	• Perhaps it is a “reflection attack” • It may even be a device incorrectly configured monitoring a network address.
		Fixed	Packets come from a fake address	• A DoS attack perhaps limited by some egress/ingress rules.
	Valid		We can identify the attack source	• Maybe it is a social attack (Anonymous) • Botnets • Slashdot effect

Taxonomy

Example taxonomy (2/5)

- Following is an example taxonomy that we could use during the execution of a DoS project:

	Classification	Description	Examples
Target	Physical	The aim is to permanently damage an asset in a manner it remains destroyed forever (i.e. bricked or physically broken).	• A bad firmware update • A non-verified update with corrupted parts • A self-destruction mechanism triggered remotely by an attacker. • Forcing a mechanical system.
	Infraestructure	An attack against an asset,device and/or protocol that brings down our service	• A load balancer without RAM • An ARP poisoning attack • Our ISP network falls
	Network	Attacks based on bandwidth exhaustion (enormeous quantitis of traffic/connections)	• Anonymous
	Resource	A direct or indirect attack against a critical resource on which our service depends.	• A DNS server on which depends a SCADA network. • A peak of user authentications saturate the enterprise auth. backend affecting all company services.
	Host	The server platform (OS) is directly attacked for bringing down the service.	• Sockstress attacks • Blaster

Taxonomy

Example taxonomy (3/5)

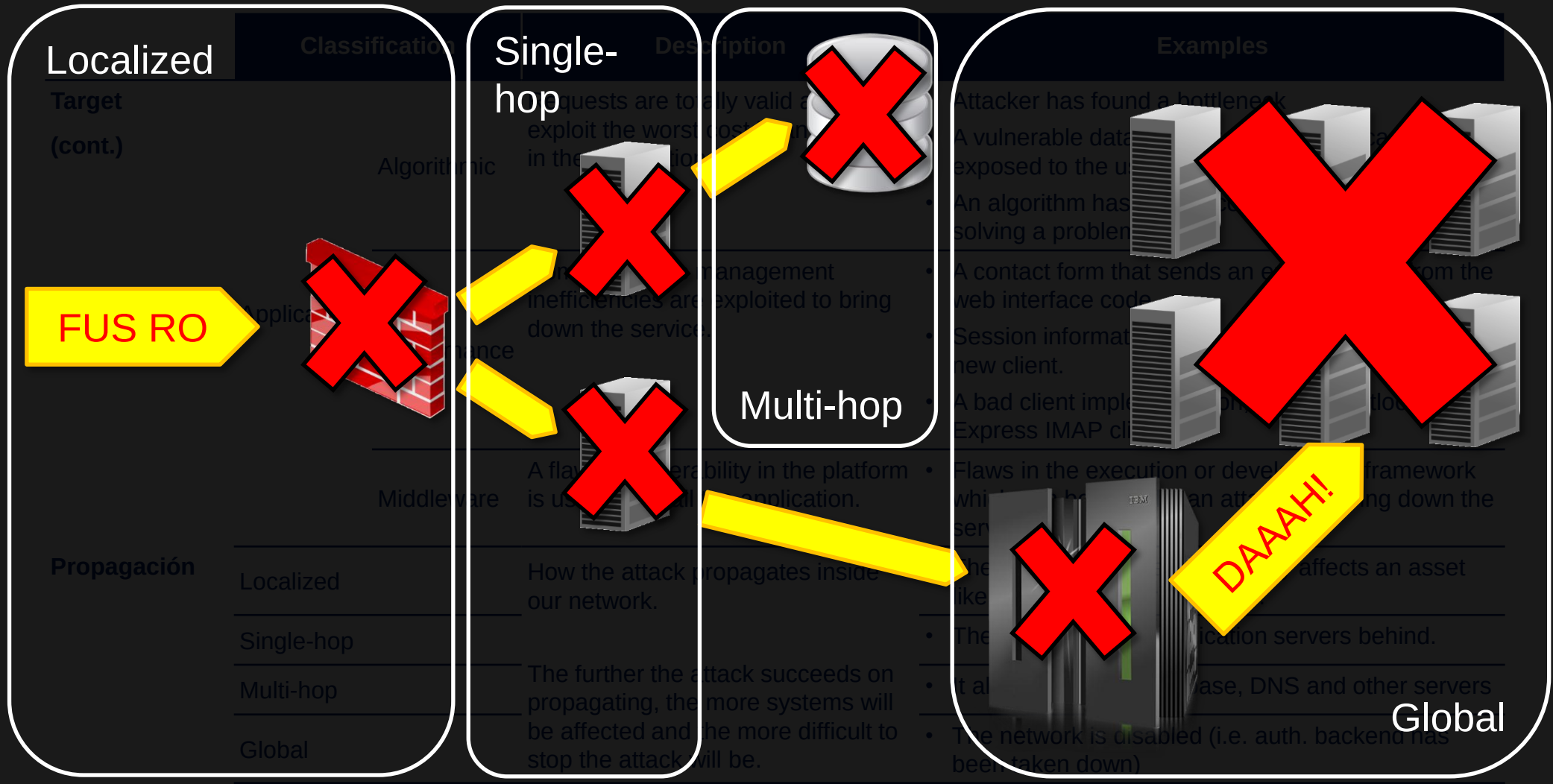
- Following is an example taxonomy that we could use during the execution of a DoS project:

	Classification		Description	Examples
Target (cont.)	Application	Algorithmic	Requests are totally valid and they exploit the worst cost in an algorithm in the application	• Attacker has found a bottleneck • A vulnerable data structure (worst cost case) is exposed to the user. • An algorithm has been incorrectly chosen for solving a problem.
		Performance	Some resource management inefficiencies are exploited to bring down the service.	• A contact form that sends an email directly from the web interface code. • Session information (heavy) is created for each new client. • A bad client implementation, like the Outlook Express IMAP client.
		Middleware	A flaw or vulnerability in the platform is used to derail the application.	• Flaws in the execution or development framework which can be used by an attacker to bring down the service.
Propagación	Localized		How the attack propagates inside our network. The further the attack succeeds on propagating, the more systems will be affected and the more difficult to stop the attack will be.	• The attack is focused and it only affects an asset like a web server or a firewall.
	Single-hop			• The attack hits the application servers behind.
	Multi-hop			• It also affects the database, DNS and other servers
	Global			• The network is disabled (i.e. auth. backend has been taken down)

Taxonomy

Example taxonomy (C/E) (Propagation diagram)

- Following is an example taxonomy that we could use during the execution of a DoS project:



Classification	Description	Examples
Target (cont.)		
Algorithmic	quests are to ly valid exploit the worst in the	Attacker has found a bottleneck A vulnerable data exposed to the u An algorithm has solving a problem
Application	anagement inefficiencies are exploited to bring down the service	A contact form that sends an e rom the web interface code Session informat new client. A bad client impl Express IMAP cl
Middleware	A fla erability in the platform is us pplication.	Flaws in the execution or devel framework whi an att ng down the ser
Propagación		
Localized	How the attack propagates inside our network.	he affects an asset like
Single-hop		The ication servers behind.
Multi-hop	The further the attack succeeds on propagating, the more systems will be affected and the more difficult to stop the attack will be.	t al ase, DNS and other servers
Global		The network is disabled (i.e. auth. backend has been taken down)

Taxonomy

Example taxonomy (4/5)

- Following is an example taxonomy that we could use during the execution of a DoS project:

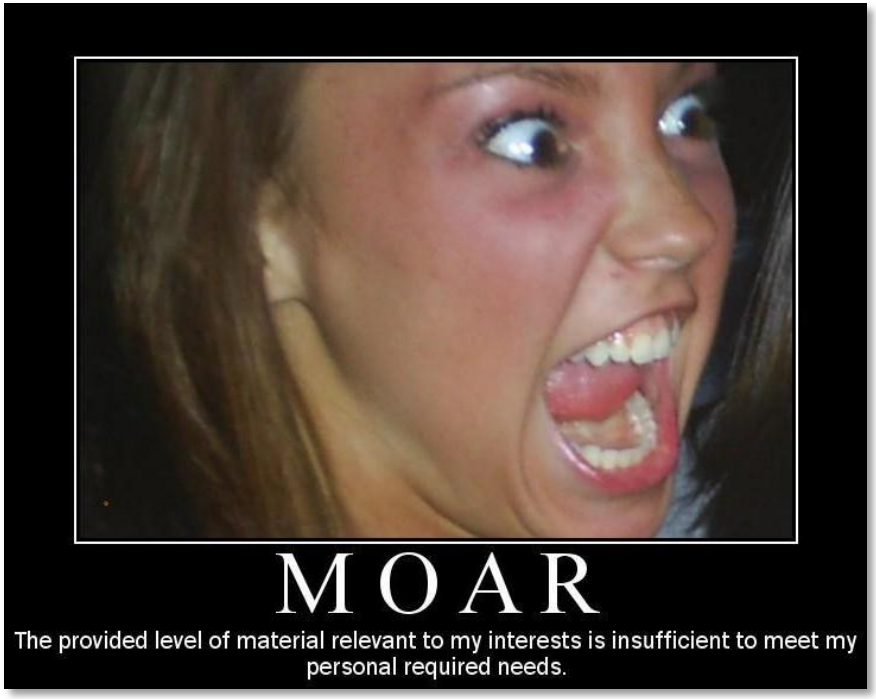
	Classification		Description	Examples
Rate	One-shot		Service disabled sending only some few bytes.	• Algorithmic complexity attack • A DoS exploit • Very bad coding practices in a public form
	Constant		A constant interaction flow	• Botnet
	Variable	Fluctuating	Quantity of traffic changes during time, so it is difficult to detect.	• Anonymous • An attack executed during rush hours
		Incremental	Bandiwdth usage grows as time pases.	• Anonymous • A worm
Characterizable	Filterable		The attack can be identified and filtered	• Quick Win!
	Non-filterable		The attack is more or less identifiable, but it cannot be filtered (it seems legit).	• LOIC
	Non-characterizable		It is impossible to distinguish between attack and legit traffic.	• A saturated network connection where users are using software that camouflages their connections inside a SSL layer.

Taxonomy

Example taxonomy (5/5)

- Following is an example taxonomy that we could use during the execution of a DoS project:

	Classification		Description	Examples
Impact	Disruptive	Auto recoverable	System auto recovers once the attack has finished	• We have watchdogs or systems able to detect services denegation and auto recover them
		Human intervention	Somebody have to press the button	• The system remains in an unstable state and it needs to be restarted by an human
		No recoverable	System is damaged and it cannot be recovered.	• Phlash Dance
	Service degradation		Service is degraded but the service is not damaged and it is working	• Once the attack is finished everything comes to normality without any fail or damage.

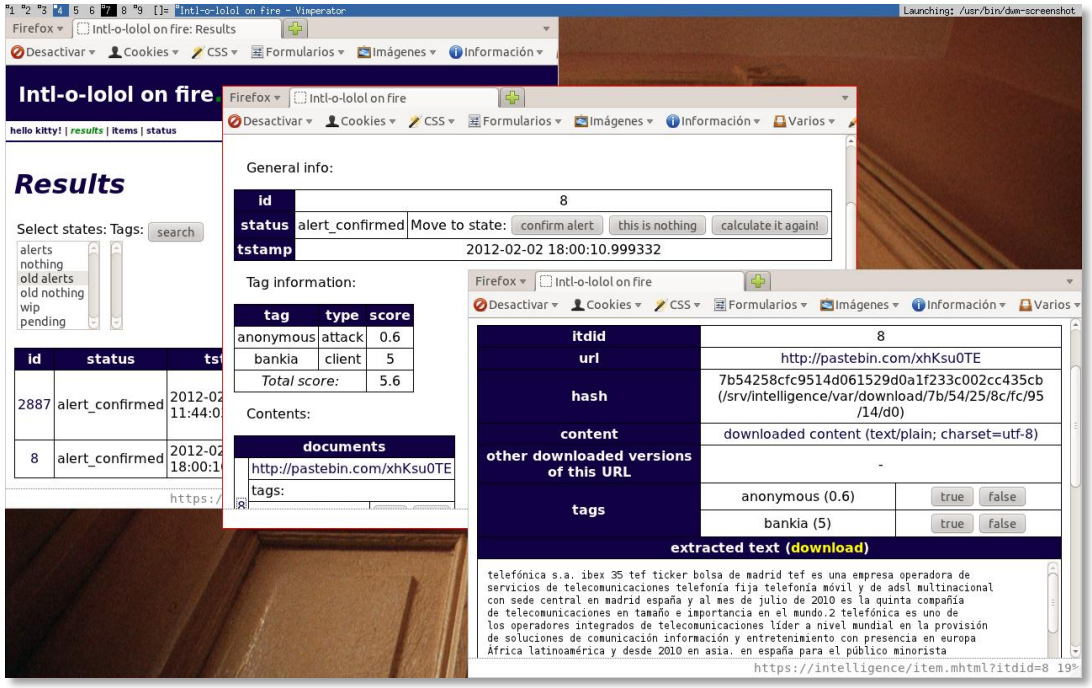


Index

Countermeasures

Attack detection, early detection

- Early detection is useful for
  - Warning our clients
  - Deactivating non-essential parts of our service
  - Prepare our anti-DoS countermeasures and mechanisms
  - Contract specialists or specialized services (like Akamai or Telefónica)
  - Focusing our efforts on protecting certain assets (specially with malware or a hacking attack)
  - In short, **take decisions before a potential disaster occurs**.
- Example: Deloitte Intelligence
  - Collect data from different sources
  - Uses AI for document tagging



Countermeasures

Attack detection, pattern detection

- Anomalies or inferences
  - Using an IPS/IDS
  - Light-weight detection [LWDETECT] based on BLINC [BLINC]
  - Makes the system able of
    - ❖ Reroute traffic to a null network (RTBH routing)
    - ❖ Reduce systems load
      - ➢ Change to static contents
      - ➢ Enable a third-party product
      - ➢ DOYS, challenges or even enable an Overlay Network
- Implement countermeasures in services at programming level
- Example at ano.lolcathost.org:
  - With mod_evasive a trigger is flagged indicating that we are under an attack
  - The AJAX interfaces automatically asks to the client. If he is not verified yet, a hash problem is presented (that will make him to use some CPU seconds).

    (The most difficult operations published to users in Ano are centralized through Ajax)
  - Once the client has solved the problem, it uses if to operate with Ano in a normal manner during a limited time.

Countermeasures

Attack detection, third party detection

- Outsourcing DoS managing or finding it out by someone else
  - Discovering if we are attacked by someone else – Academic
    - Backscatter analysis [CAIDA]
      - This technique works with IP-Spoofing attacks: nodes outside of our network are “splashed” by our replies, revealing a potential DoS attack in progress to an external observer.
  - Third party services like Akamai
    - Attack detection
    - Characterization
    - And they try to stop/manage it using their services.

Countermeasures

Factors affecting a countermeasure

	Preguntas
Effectiveness	How capable is a defense mechanism?
Reliability	Does it always mitigate a DoS attacks as well, or is it sometimes less effective?
	Is there a possibility for false positives?
Misusability	Can an attacker exploit a defense mechanism in an unexpected way as a tool for achieving a DoS condition?
Collateral damage	Does a defense mechanism cause any negative side effects, like performance problems or a requirement for extensive human intervention?
Proactivity	Can a defense mechanism prevent atttacks or does it only react to existing attacks?
Completeness	What kind of other defense mechanisms are required?
Reaction delay	How fast does a defense mechanism react to intrusions?
Ease of implementation	Is it feasible or possible to implement a defense mechanism?
	Does it involve other organizations?
	Implementation cost worth the benefit?
Ease of use	Is the human interface easy to use?
	Does a defense mechanism fit with an already existing security infraestructure?
Installation place	What is the optimal place to implement it?

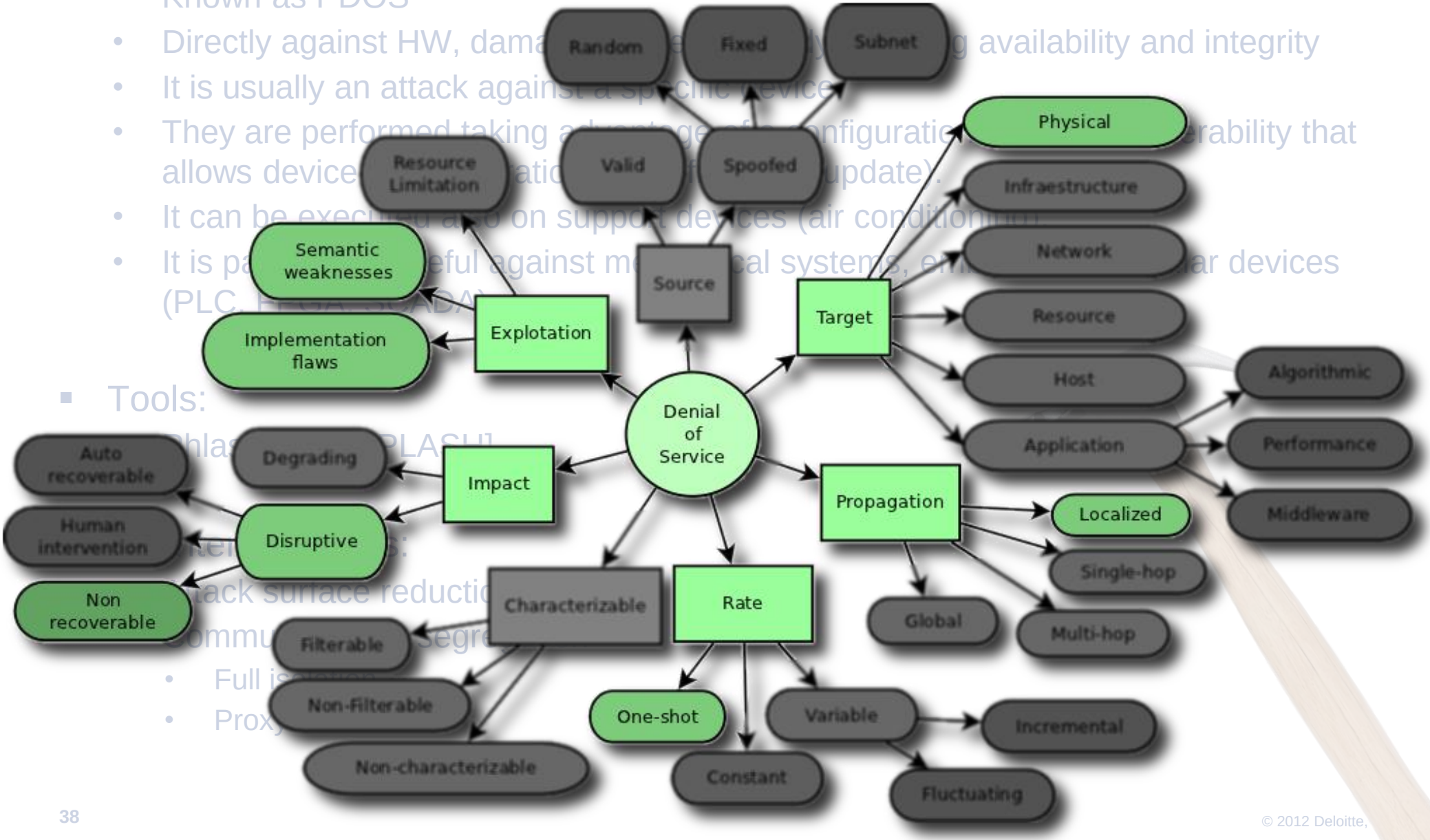
(fuente [MITDOS])

Index

Classification, techniques, tools and mitigation

Physical level

- Description
  - Known as PDOS
  - Directly against HW, dama… availability and integrity
  - It is usually an attack against a spe…
  - They are performed taking a… nfiguratio… erability that allows device… ati… update).
  - It can be executed also on suppo… devices (air conditioning)
  - It is pa… eful against m… cal systems, em… ar devices (PLC, FPGA, SCADA)
- Tools:
  - Phlas… PLASH]
- Mitigation:
  - Attack surface reductio…
  - Commu… Segre…
    - Full is…
    - Proxy…



Classification, techniques, tools and mitigation

Physical level

- Description
  - Known as PDOS
  - Directly against HW, damaging it permanently, harming availability and integrity
  - It is usually an attack against a specific device.
  - They are performed taking advantage of a configuration flaw or a vulnerability that allows device reconfiguration (i.e. a firmware update).
  - It can be executed also on support devices (air conditioning).
  - It is particularly useful against mechanical systems, embedded or similar devices (PLC, FPGA, SCADA)

- Tools:
  - PhlashDance [PLASH]

- Countermeasures:
  - Attack surface reduction
  - Communications segregation
    - Full isolation
    - Proxys

Classification, techniques, tools and mitigation

Infrastructure: semantic weaknesses

- A protocol or an algorithm can have design flaws not easily solved without breaking the compatibility
- Examples:
  - SSLv2
  - TCP/IP
  - ISAKMP aggr[illegible]de [illegible],
- This type[illegible]bilities are a [illegible]because
  - They a[illegible]rd to [illegible]stand)
  - It i[illegible]
  - Their impact is very high becaus[illegible]difficult to cha[illegible]
    the pr[illegible]
    nd t[illegible]are us[illegible]disrupt[illegible]
    s:
    [illegible]sis
    [illegible]ockstr[illegible]
  - ikescan
  - naphta
  - thc-ssl-do[illegible]

Classification, techniques, tools and mitigation

Infrastructure: semantic weaknesses

- A protocol or an algorithm can have design flaws not easily solved without breaking the compatibility
- Examples:
  - SSLv2
  - TCP/IP
  - ISAKMP aggressive mode [DOSPK], …
- This type of vulnerabilities are a trend because
  - They aren't useful for *script-kiddies* (hard to understand),
  - It is investigation, so it is cool,
  - Their impact is very high because it is too difficult to change the protocol,
  - And the countermeasures are usually disruptive..
- Tools:
  - dosis
  - sockstress
  - ikescan
  - naphta
  - thc-ssl-dos

Connection flood
Zero window connection
Small window
Segment hole
Req fin pause
Activate reno pressure
Stacheldraht

Classification, techniques, tools and mitigation

Infrastructure: semantic weaknesses

- Mitigation:
  - Establishing secondary communication channels for mission critical services
  - Deactivating protocol features
    - ✓ Disable SSLv2
    - ✓ Aggressive mode (ISAKMP)
  - A work plan that, once the attack is understood, enables RTBH [RFC5635]
  - TCP/IP tuning
    - ✓ Syncookies
    - ✓ Timeout 3way hs, timeout fin states, timeout unused connections, …
      - Example in Linux:

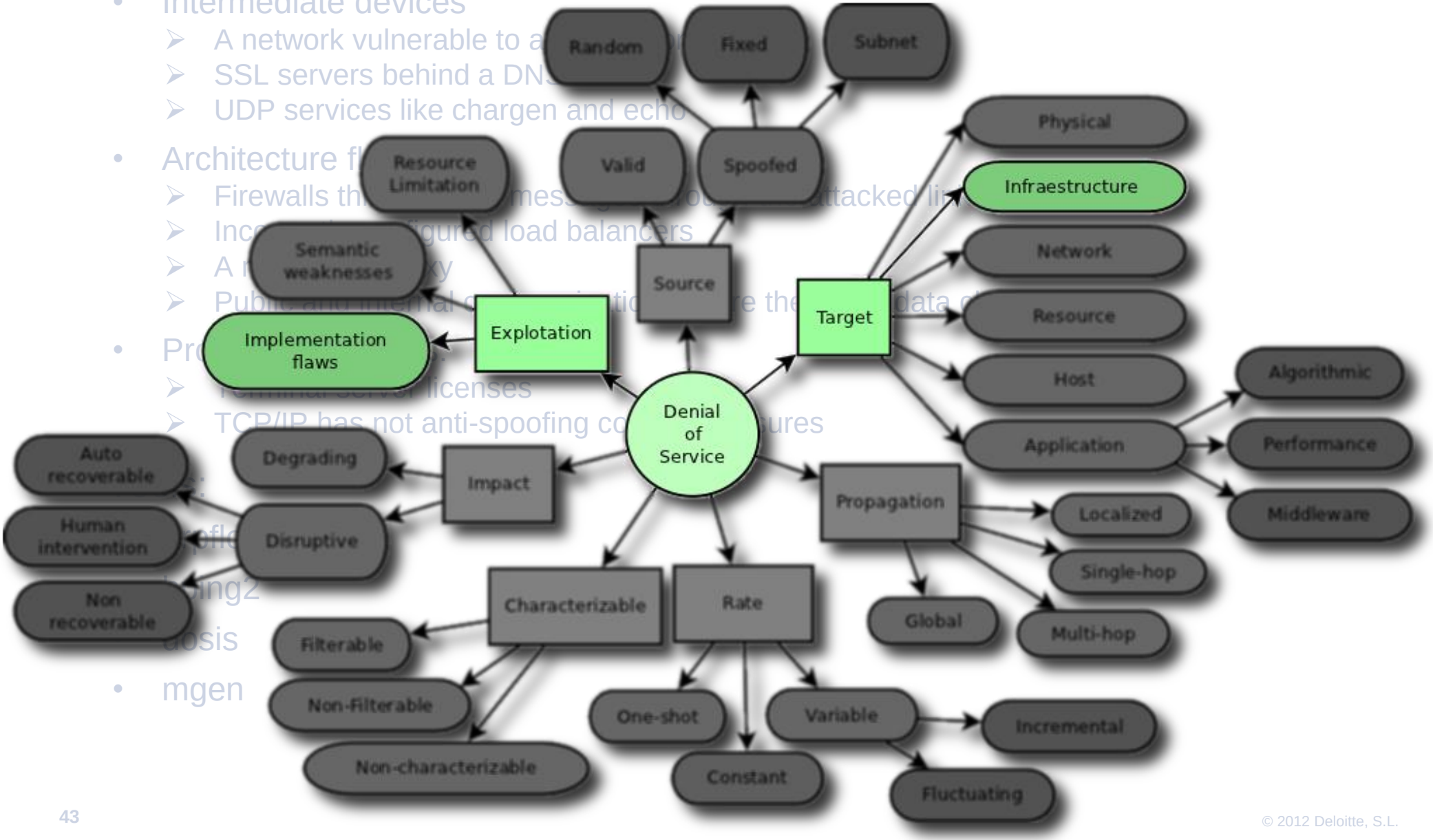
Linux TCP/IP stack parameter	Default	Example
/proc/sys/net/ipv4/tcp_keepalive_time	7200	30
/proc/sys/net/ipv4/tcp_keepalive_probes	9	2
/proc/sys/net/ipv4/tcp_max_ka_probes	5	100
/proc/sys/net/ipv4/tcp_syncookies	0	1

  - Update to a new protocol version (as soon as it exists)

Classification, techniques, tools and mitigation

Infrastructure: implementation flaws

- A lot of attack vectors:
  - Intermediate devices
    - A network vulnerable to a
    - SSL servers behind a DN
    - UDP services like chargen and echo
  - Architecture f
    - Firewalls th
    - Inc
    - A
    - Pub
  - Pr
    - licenses
    - TCP/IP has not anti-spoofing co
  - ing2
    - dosis
  - mgen



Classification, techniques, tools and mitigation

Infrastructure: implementation flaws

- A lot of attack vectors:
  - Intermediate devices
    - A network vulnerable to amplification attacks
    - SSL servers behind a DNS RR
    - UDP services like chargen and echo
  - Architecture flaws
    - Firewalls throw syslog messages through the attacked link
    - Incorrectly configured load balancers
    - A misplaced proxy
    - Public and internal communications share the same data channel
  - Protocol limitations:
    - Terminal server licenses
    - TCP/IP has not anti-spoofing countermeasures
- Tools:
  - arpflood
  - hping2
  - dosis
  - mgen

Classification, techniques, tools and mitigation

Infrastructure: implementation flaws

- Countermeasures:
  - Access control (if it is possible)
  - Disable features that allow to hit a limitation and trigger a DoS condition.
  - Install security appliances to limit the indiscriminate use or exploitation of a service.
  - Network redesign.
  - Mechanisms optimization:
    - ✓ Aggressive configuration in firewalls
    - ✓ Deactivate expensive algorithms (3DES in SSL)
    - ✓ Improve HTTP configuration
    - ✓ Set limits to neutralize the harmful effects of other limits

Classification, techniques, tools and mitigation

Infrastructure: bruteforce

- Description:
  - The stability of the service is threatened by a significant amount of traffic that is overloading a certain asse
  - This category includes Do ks on a forced use of the protocol.
  - Families: SYN MP
- Tools:
  - dosis
  - mgen
  - hpi
- Countermeasures:
  - The t is being able to critical-mass
  - orst our cture
  - The e must be balanced in all poi
  - Impro phic disposition, replicate assets and redesig he network.
  - nimize dependencie s.
  - Segrega is m separate layers.
  - Security es (Checkpoint)
  - Prepare ers for f se
  - Labrea

Classification, techniques, tools and mitigation

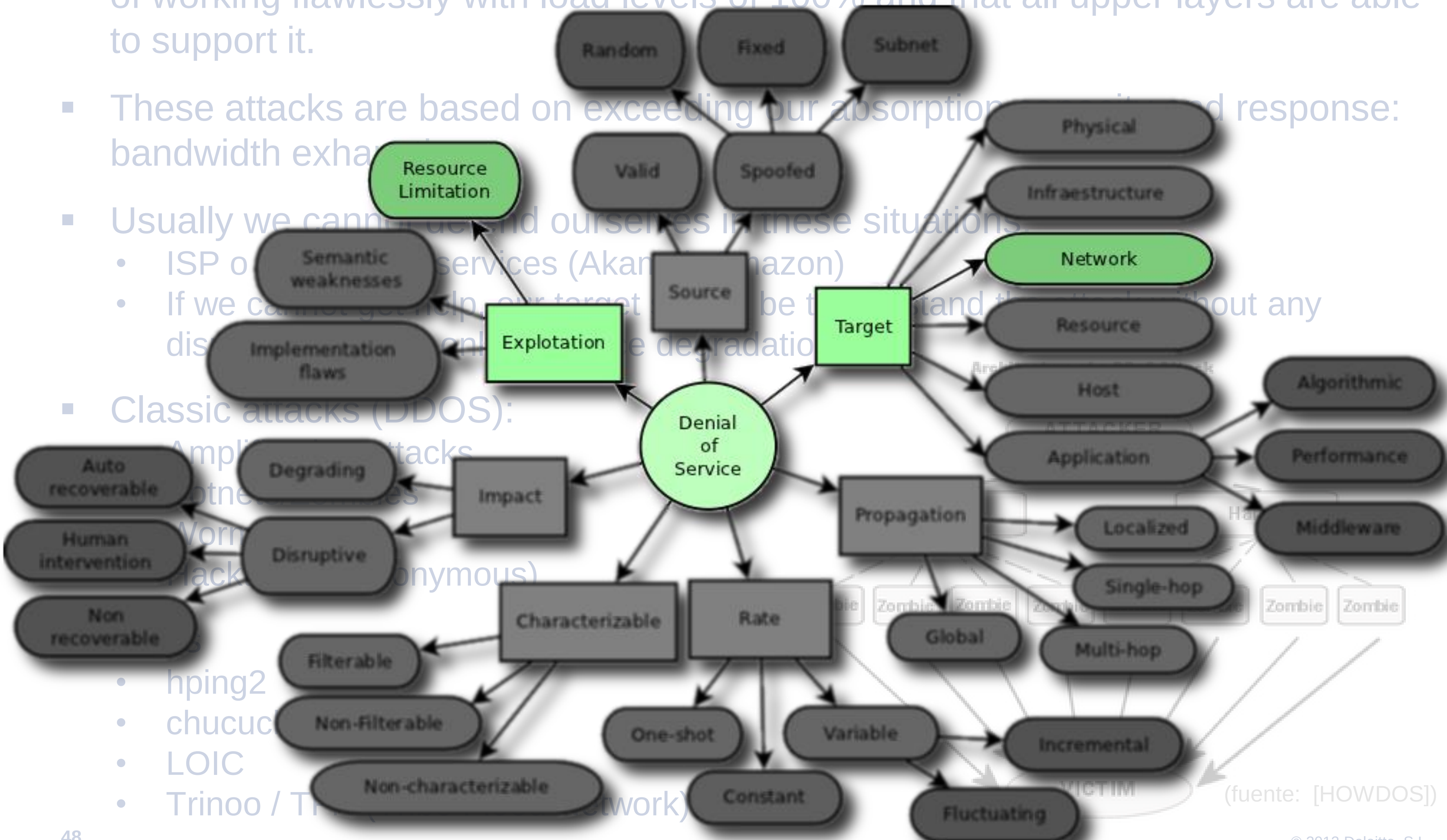
Infrastructure: bruteforce

- Description:
  - The stability of the service is threatened by a significant amount of traffic that is overloading a certain asset
  - This category includes DoS and DDoS attacks based on a forced use of the protocol.
  - Families: SYN, UDP, ICMP flood
- Tools:
  - dosis
  - mgen
  - hping2
- Countermeasures:
  - The target here is being able to handle a critical-mass traffic, or in other words the worst load that our infrastructure is supposed to support.
  - The performance must be balanced in all points of the communications chain.
  - Improve geographic disposition, replicate assets and redesign (again) the network.
  - Minimize dependencies and ping-pong effects.
  - Segregate protocols management in well separated layers.
  - Security appliances (Checkpoint)
  - Prepare the upper layers for facing these attacks.
  - Labrea

Classification, techniques, tools and mitigation

Network: bruteforce

- To achieve success at this level we have to assure that our system is capable of working flawlessly with load levels of 100% and that all upper layers are able to support it.
- These attacks are based on exceeding our absorption … response: bandwidth exha…
- Usually we cann… ourselves in these situations.
  - ISP o… services (Akam… azon)
  - If we ca… target … be t… and t… out any dis… degradatio…
- Classic attacks (DDOS):
  - Ampl… attacks
  - …otne…
  - Worm…
  - Hack… (Anonymous)
  - hping2
  - chucuc…
  - LOIC
  - Trinoo / T… network)

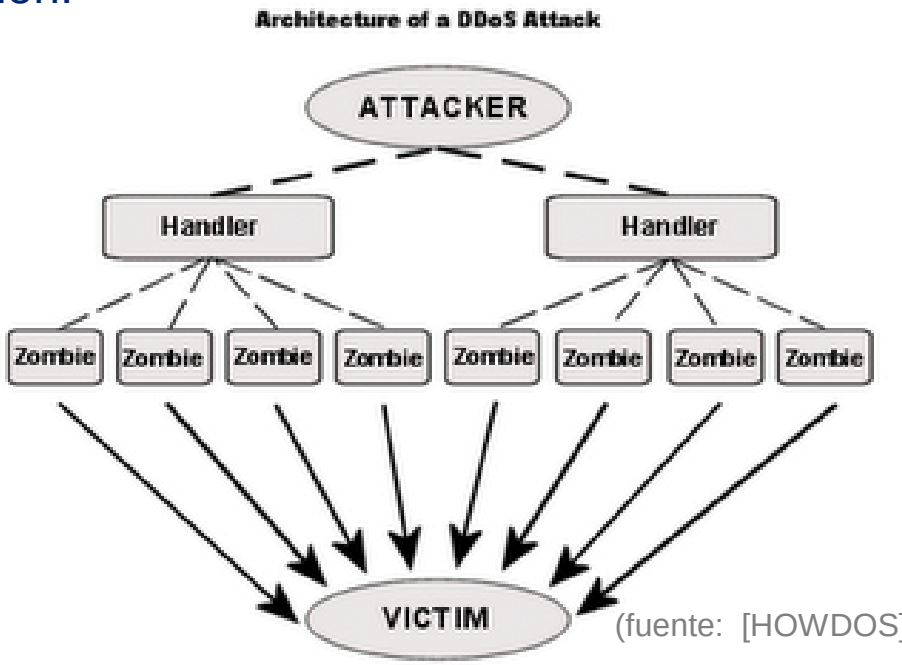


(fuente: [HOWDOS])

Classification, techniques, tools and mitigation

Network: bruteforce

- To achieve success at this level we have to assure that our system is capable of working flawlessly with load levels of 100% and that all upper layers are able to support it.
- These attacks are based on exceeding our absorption capacity and response: bandwidth exhaustion.
- Usually we cannot defend ourselves in these situations:
  - ISP o specialized services (Akamai, Amazon)
  - If we cannot get help, our target should be to withstand the attack without any disruptive effect – only a service degradation.
- Classic attacks (DDOS):
  - Amplification attacks
  - Botnets/Zombies
  - Worms
  - Hacktivism (Anonymous)
- Tools
  - hping2
  - chucuchu
  - LOIC
  - Trinoo / TFN (Tribe Flood Network)



(fuente: [HOWDOS])

Classification, techniques, tools and mitigation

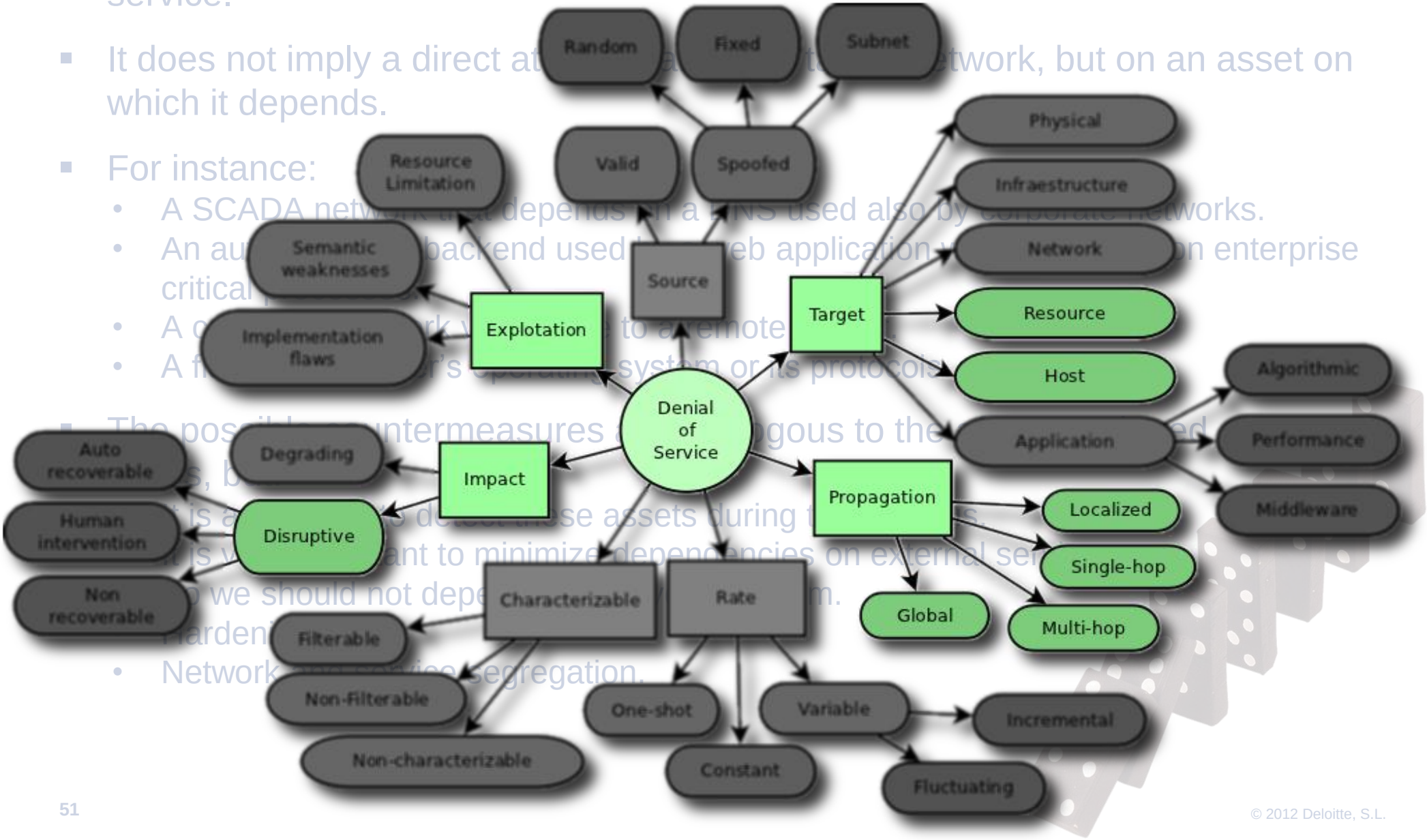
Red: bruteforce

- Countermeasures:
  - You must have a **plan**, clear and flexible enough and **persons** capable of executing it.
  - It is essential to have a strict configuration in your perimeter firewalls:
    - ✓ Smurf Amplifier Registry (http://smurf.powertech.no/)
    - ✓ Ingress & egress rules
    - ✓ Client bandwidth limits
    - ✓ Client connections limits
    - ✓ Restrictive policies (DROP)
    - ✓ Mechanisms for stopping scans and storm propagations (labrea)
  - The attack should be stopped in the nearest point to the attacker.
  - Overlay or distributed networks (like Akamai)
    - ✓ i.e. Akamai DDoS Defender
    - ✓ Cloud Computing
  - Offer service on different channels depending of the geographic origin (GeoDNS)
  - Implement RTBH Routing

Classification, techniques, tools and mitigation

Resources and Host

- These DoS attacks are focused against a resource or host that sustains the service.
- It does not imply a direct at… …etwork, but on an asset on which it depends.
- For instance:
  - A SCADA network … depends on a DNS used also by corporate networks.
  - An au… backend used … eb application … on enterprise critical …
  - A c… …rk … to a remote …
  - A f… …'s operating system or its protocols.
- The pos… …ountermeasures … …ogous to the …s, b… …
  - It is a… …s detect those assets during t… …
  - It is … …ant to minimize dependencies on external ser… …
  - So we should not depe… …m.
  - Harden…
  - Network … segregation.



Classification, techniques, tools and mitigation

Resources and Host

- These DoS attacks are focused against a resource or host that sustains the service.
- It does not imply a direct attack against the target network, but on an asset on which it depends.
- For instance:
  - A SCADA network that depends on a DNS used also by corporate networks.
  - An authentication backend used by a web application which depends on enterprise critical processes.
  - A computer network vulnerable to a remote exploit.
  - A flaw in the server's operating system or its protocols.
- The possible countermeasures are analogous to the aforementioned cases, but:
  - It is a key step to detect these assets during the analysis.
  - It is very important to minimize dependencies on external services.
  - So we should not depend excessively on them.
  - Hardening.
  - Network and service segregation.

Classification, techniques, tools and mitigation

Application: introduction

- This level requires a lot of work because it is not usually protected against DoS attacks
  - In spite of usually having the most critical impact
  - And being the easiest point to get a DoS condition
- The reason:
  - An inefficient or poor design
  - Bad coding practices
  - An incorrectly used or configured backend
  - An excessive use of dynamic contents

- At this level the results are overwhelming.
  - In several projects we have demonstrated that using a 1999 Spanish ADSL connection (256 kbps) an attacker could take down all the application servers.
  - In one case the network was designed to support more than 2 Gbps.
- For simplification we put here all the software stack over the OS:
  - Database
  - Web server
  - Applications server
  - Libraries and the application itself

Classification, techniques, tools and mitigation

Application: DoS vulnerabilities

- It is the most effective and devastating DoS attack.
- Especially because any vul[illegible]vert the server's operations could be used for triggering [illegible]
- The result ranges from a temporary DoS to an asset'[illegible]tion (and sometimes even [illegible]).
- There ar[illegible]ols to perform these attacks. We [illegible]n metasploit because [illegible]of use.
- Key c[illegible]s:
  - Attack surface minimization.
  - Impl[illegible]ess controls for [illegible]g that shou[illegible]
  - Protect complex p[illegible] using [illegible]urity fa[illegible]proxies or security appli[illegible]
  - Periodic vulnerability [illegible]
  - Protect the most impo[illegible]ltip[illegible]y
  - An effe[illegible]nstant patching and response protocol.
  - Pay att[illegible]s (cybe[illegible]nc[illegible]

Classification, techniques, tools and mitigation

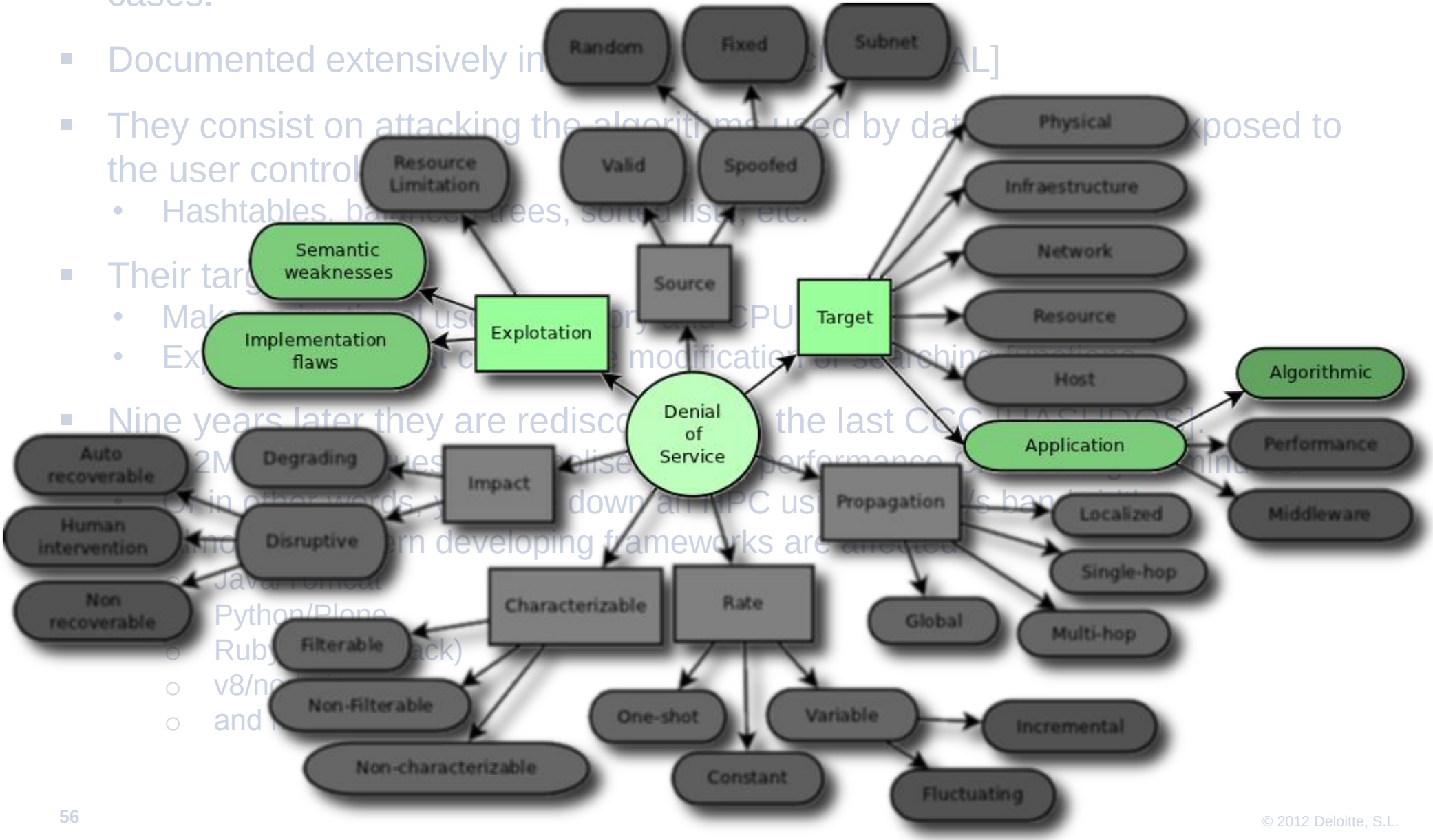
Application: DoS vulnerabilities

- It is the most effective and devastating DoS attack.
- Especially because any vulnerability that could subvert the server's operations could be used for triggering a PDoS condition.
- The result ranges from a temporary DoS to an asset's logic destruction (and sometimes even physical).
- There are a lot of tools to perform these attacks. We emphasize on metasploit because of its ease of use.
- Key countermeasures:
  - Attack surface minimization.
  - Implement access controls for everything that should not be public.
  - Protect complex protocols using security layers like proxies or security appliances.
  - Periodic vulnerability scans and pentests.
  - Protect the most important access with multiple security layers.
  - An effective, constant patching and response protocol.
  - Pay attention to news (cyber-intelligence).

Classification, techniques, tools and mitigation

Application: Algorithmic DoS (1/2)

- Attacks based on exploiting the computational costs of algorithms in their worst cases.
- Documented extensively in
- They consist on attacking the algorithms used by dat
  the user control
  - Hashtables, balanced trees, sorted lists, etc.
- Their targ
  - Mak
  - Ex
- Nine years later they are redisco
  - 2M
  - In other words, y
  - Java/Tomcat
  - Python/Plone
  - Ruby
  - v8/no
  - and



Classification, techniques, tools and mitigation

Application: Algorithmic DoS (1/2)

- Attacks based on exploiting the computational costs of algorithms in their worst cases.
- Documented extensively in 2003 in the article [DOSAL]
- They consist on attacking the algorithms used by data structures exposed to the user control:
  - Hashtables, balanced trees, sorted lists, etc.
- Their target is
  - Make an irrational use of memory and CPU,
  - Exploiting the worst case of the modification or searching functions.
- Nine years later they are rediscovered in the last CCC [HASHDOS]:
  - A 2Mb Java request monopolises a high performance CPU during 44 minutes.
  - Or in other words, you take down an HPC using a 6kbit/s bandwidth.
  - Almost all modern developing frameworks are affected:
    - Java/Tomcat
    - Python/Plone
    - Ruby/Cruby (Rack)
    - v8/node.js
    - and more…

Classification, techniques, tools and mitigation

Application: Algorithmic DoS (2/2)

- Mitigation:
  - All user input is evil. For this reason, he should not be able to choose it without restrictions, except under determined exceptions and only with LIMITS.
  - We should detect beforehand an input that could lead to a “worst case”, or at least implement restrictions to abort the request before triggering a DoS condition (timeouts, watchdogs, etc).
  - These limits are easily implemented in the attack presented at CCC [HASHDOS] using mod_security [MSHASHDOS]:
    - ✓ Limit POST size without attached files
    - ✓ Limit the quantity of input parameters
  - If we cannot establish limits, we should:
    - ✓ Introduce controls to avoid or limit the indiscriminate use of intensive operations
    - ✓ Result caching
    - ✓ Use execution queues – It is easier to establish starvation controls on a queue, and it is more difficult to take down an asynchronous presentation layer.
      - This technique is used by ano.lolcathost.org for image security analysis and processing

Classification, techniques, tools and mitigation

Application: locality and client state

- Because HTTP is a *stateless* protocol, there is and there will be a lot bibliography related to keeping information state during transactions..
- And not taking the proper c … nd how it should be kept could put the availability of our platform at stake.
- The state inform… uld … different
  - Its generation could be expensive
  - If it is … on an single DB could lead to a glo…
  - If it is localized on the … an h… shing … ers (ping-po… …ses.
  - A big … *working-set* could kn… o… lication.

- Inter… s:
  - … only keep the esse… rmation in your st… lation.
  - … contained in the main databa… should only be updated when necessary.
- … se tools or technique… clas… KI… igned and ciphered … s sake).
  - Don't a… a client s… vis… main page.
  - Make the st… difficult by using challenges (Cit… Protector).

Classification, techniques, tools and mitigation

Application: locality and client state

- Because HTTP is a *stateless* protocol, there is and there will be a lot bibliography related to keeping information state during transactions..
- And not taking the proper care to what information and how it should be kept could put the availability of our platform at stake.
- The state information could become a risk in different manners:
  - Its generation could be expensive.
  - If it is centralized on an single DB it could lead to a global overhead.
  - If it is localized on the server itself we can have trashing between servers (ping-pong) or internal remote accesses.
  - A big user *working-set* could kill our application.
- Countermeasures:
  - Only keep the essential information in your state information.
  - It should not be contained in the main database, and it should only be updated when necessary.
  - Use tools or techniques like VIEWSTATE or classic COOKIES (but signed and ciphered for God's sake).
  - Don't allow to create a client state only visiting the main page.
  - Make the state creation difficult by using **challenges** (Citrix Netscaler).

Classification, techniques, tools and mitigation

Application: dynamic contents excess

- Nowadays it seems a taboo to think about static contents.
- But they are the best choice [illegible] visit peak: they are very easy to serve.
- Additionally, people tend to put all page generation t[illegible]one server, causing a bottle
- An easy [illegible] strategy is to share the page gen[illegible]een several servers a[illegible]e page comp[illegible]n to the client:
  - Ad[illegible]
  - Ma[illegible] (static)
  - Static content (images, css, js[illegible]ved by others
- [illegible]nd [illegible]d[illegible]th be [illegible]d using techniques like Ajax
- It is always recommend[illegible]main page.
- [illegible]ost cu[illegible]attack[illegible]the
- Trick for ot[illegible] we detect an important increment of requests for one page we can [illegible] static vers[illegible]ent page.

Classification, techniques, tools and mitigation

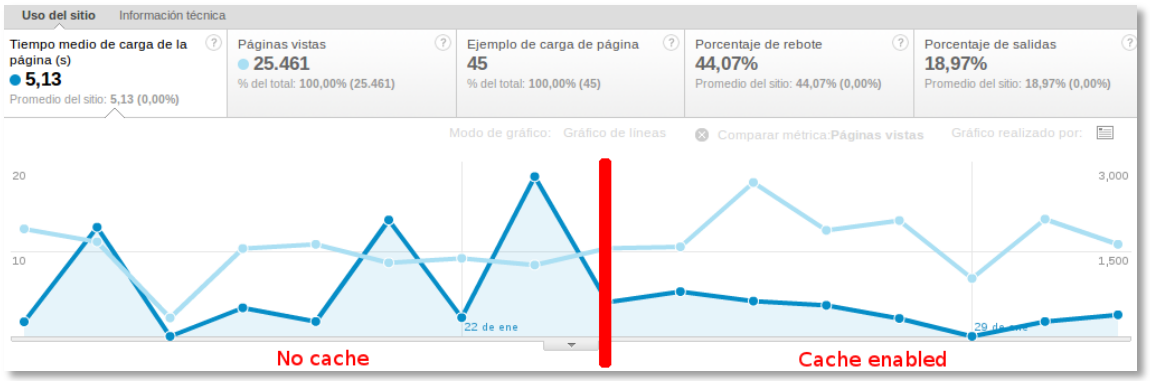
Application: dynamic contents excess

- Nowadays it seems a taboo to think about static contents.
- But they are the best choice when facing an important visit peak: they are very easy to serve.
- Additionally, people tend to put all page generation techniques on one server, causing a bottleneck..
- An easy and simple strategy is to share the page generation between several servers and leave the page composition to the client:
  - Ad servers
  - Main page server (static)
  - Static content (images, css, js, etc) served by others servers
  - And the other elements can be loaded under demand using techniques like Ajax
- It is always recommended to have a static main page.
  - Most currently attacks are focused only on the main page.
- Trick for other pages: if we detect an important increment of requests for one page we can offer a static version of that page.

Classification, techniques, tools and mitigation

Application: next step, caching

- Usually a 90% of a web page is always the same.
- We can implement or use a cache system to reduce the time used to render a page:
  - Cache-Cache (mod_perl/Mason HQ)
  - Varnish (web accelerator, generic)
- Tools like Varnish Cache are some of the basic pillars in a web acceleration environment (Akamai has something similar).
- They can be programmed for
  - Caching dynamic contents
  - Precaching dynamic contents that may be requested.
  - Decorate served pages to reduce the server load.
  - Or even program adaptive caching policies able to change on how to serve certain resources under certain workloads.



Classification, techniques, tools and mitigation

Application: don't put all your eggs in one basket

- To receive visits from **all over the world** is an inherent characteristic of the Internet.
- But we don't have to provic whole world.
- Maybe interestingly, we can,
  - Disperse the s og
    (Watch out: without centralizing everything in the same subsystem!)
  - Try to s as close as y n
    - ✓ T
    - ✓
    - ✓ And only fall where we ar ed

o makes possib apt o ce es to the most active
s o our planet.
And makes possible to most important
ts/cou atio e a est of the world.

Classification, techniques, tools and mitigation

Application: don't put all your eggs in one basket

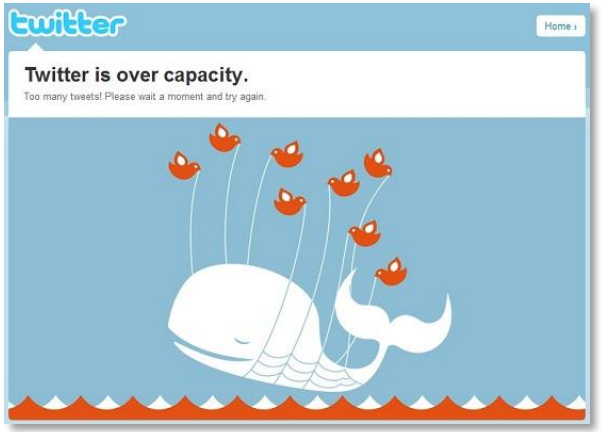
- To receive visits from **all over the world** is an inherent characteristic of the Internet.
- But we don't have to provide the same service to the whole world.
- Maybe interestingly, we can,
  - Disperse the service geographically

    (Watch out: without centralizing everything in the same subsystem!)
  - Try to serve users as close as you can
    - ✓ To reduce latencies
    - ✓ Serve quicker
    - ✓ And only fall where we are attacked
- It also makes possible to adapt or balance our services to the most active hours or parts of our planet.
- And makes possible to keep serving to our most important clients/countries/locations, even when we are attacked from the rest of the world.

Classification, techniques, tools and mitigation

Application: from lost to the river (Source: [ZENGLISH])

- In the worst case we can always run away or look for help before falling in a self-destruction.
- There are several approximations to this:
  - Denial You of Service [PWDOS]
    - ✓ The server detects the “most active” clients and it cuts them off the service.
  - Twitter is over capacity
    - ✓ Twitter stops before being saturated.
    - ✓ The service is not available, but it is elegant.



  - Show a queue
    - ✓ This mechanism is widely used by direct download pages to protect their bandwidth.
    - ✓ But it is useful if we signed up the supersales representative of the year and he gets a contract to sell tickets for the upcoming U2 or Madonna concert.
  - Sign up for a Cloud service that allows us to scale
    - ✓ But it is more difficult than it seems because for doing this type of actions we first have to adapt our software and resources to distributed environments.
    - ✓ This is not always possible because of time and resources.

Agenda

- Introduction
- Analysis methodology
- Our tools
- Taxonomy
- Countermeasures
- Classification, techniques, tools and mitigation
- **Bibliography**

Bibliography

I love reading ☺ (1/6)

[AKADDOS] ***Akamai DDoS Defender***,
http://www.akamai.com/html/solutions/ddos_defender.html

[AUTRES] ***Autonomic Response to Distributed Denial of Service Attacks***,
Dan Sterne, Kelly Djahandari, Brett Wilson, Bill Babson, Dan Schnackenberg, Harley Holliday, and Travis Reid.

[BIGDNS] ***GeoDNS BIND patch***
http://www.caraytech.com/geodns/

[BLINC] ***BLINC: Multilevel Traffic Classification in the Dark***,
Thomas Karagiannis, Konstantina Michalis Faloutsos, Papagiannaki

[CAIDA] ***Inferring Internet Denial-of-Service Activity***,
David Moore

[CACHE2] ***Cache-Cache***,
Jonathan Swartz
http://search.cpan.org/dist/Cache-Cache/

[DNSAMP] ***DNS Amplification Attacks***,
Randal Vaughn and Gadi Evron

[DOSAL] ***Denial of Service via Algorithmic Complexity Attacks***,
Scott A. Crosby & Dan S. Wallach

[DOSPK] ***Denial of service in public key protocols***,
Pasi Eronen

Bibliography

I love reading ☺ (2/6)

[DPROT] ***DOS and DDOS protection***,
http://www.opensourcerack.com/2010/10/14/dos-and-ddos-protection/

[GEODNS] ***GeoDNS—Geographically-aware, protocol-agnostic load balancing at the DNS level***,
John Hawley

[HASHDOS] ***Efficient Denial of Service Attacks on Web Application Platforms***,
Alexander “alech” Klink & Julian “zeri” Wälde

[HOWDOS] ***Como hacer un ataque DDOS***,
Williams Melgar
http://www.comunidadbloggers.com/2011/02/como-se-hacer-un-ataque-ddos.html

[LABREA] ***LaBrea Tarpit***,
http://labrea.sourceforge.net/

[LEMMA] ***Method of Mitigating DDoS Attacks by Randomly choosing and Dynamically Changing Routing Information***,
Samsom Lemma

[LWDETECT] ***Lightweight Detection of DoS Attacks***,
Sirikarn Pukkawanna, Vasaka Visoottiviseth, Panita Pongpaibool

[MAYDAY] ***Mayday: Distributed Filtering for Internet Services***,
David G. Andersen

Bibliography

I love reading ☺ (3/6)

[MGEN] ***Multi-Generator (MGEN)***,
Naval Research Laboratory (NRL)
http://cs.itd.nrl.navy.mil/work/mgen/

[MEVASIVE] ***mod_evasive***,
Jonathan Zdziarski
http://www.zdziarski.com/blog/?page_id=442

[MITDOS] ***Mitigating denial of service attacks: A tutorial***,
Jarmo Mölsä

[MSHASHDOS] ***ModSecurity Mitigations for ASP.NET HashTable DoS Vulnerability (CVE-2011-3414)***,
http://blog.spiderlabs.com/2012/01/modsecurity-mitigations-for-aspnet-hashtable-dos-vulnerability-cve-2011-3414.html

[NAPHST] ***The NAPTHA DoS vulnerabilities***,
Bob Keyes (SecuriTeam)
http://www.securiteam.com/securitynews/6B0031F0KA.html

[NISTIHG] ***Computer Security Incident Handling Guide***,
Karen Scarfone, Tim Grance & Kelly Masone

[OVERCAST] ***Overcast: Reliable Multicasting with an Overlay Network***,
John Jannotti, David K. Gifford, Kirk L. Johnson, M. Frans Kaashoek, James W. O'Toole Jr.

Bibliography

I love reading ☺ (4/6)

[OVERDOSE] ***OverDoSe: A Generic DDoS Protection Service Using an Overlay Network***,
Elaine Shi, Ion Stoica, David Andersen, Adrian Perrig

[PAXSON] ***An Analysis of Using Reflectors for Distributed DoS Attacks***,
Vern Paxson

[PHLASH] ***PhlashDance: Discovering permanent denial of service attacks against embedded systems***,
Rich Smith, HP Labs

[POKEMON] ***Second Annual Cost of Cyber Crime Study; Benchmark Study of U.S. Companies; August 2011***
Sponsored by ArcSight(an HP Company) Independently conducted by Ponemon Institute LLC

[PWDOS] ***Prevent DoS attacks in your web application***,
Omar AL Zabir
http://omaralzabir.com/prevent_denial_of_service__dos__attacks_in_your_web_application/

[RFC2827] ***RFC2827 Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing***,
Ferguson & Senie

Bibliography

I love reading ☺ (5/6)

[RFC5635] ***RFC 5635: Remote Triggered Black Hole Filtering with Unicast Reverse Path Forwarding (uRPF)***,
W. Kumari & D. McPherson

[SECAPCH] ***20 ways to Secure your Apache Configuration***,
Pete Freitag,
http://www.petefreitag.com/item/505.cfm

[SLOWRD] ***ModSecurity Advanced Topic of the Week: Mitigation of 'Slow Read' Denial of Service Attack***",
http://blog.spiderlabs.com/2012/01/modsecurity-advanced-topic-of-the-week-mitigation-of-slow-read-denial-of-service-attack.html

[SOSDOS] ***SOS: An Architecture For Mitigating DDoS Attacks***,
Angelos D. Keromytis, Vishal Misra, Dan Rubenstein

[SRAW] ***SOCK RAW demystified***,
http://sock-raw.org/papers/sock_raw

[TAXCPS] ***A Taxonomy for Denial of Service Attacks in Content-based Publish/Subscribe Systems***",
Alex Wun, Alex Cheung & Hans-Arno Jacobsen

[TAXDOS] ***A Taxonomy of DDoS Attack and DDoS Defense Mechanisms***,
Jelena Mirkovic, Peter Reiher

[TDOST] **Trends in Denial of Service Attack Technology**,
CERT® Coordination; Center, Kevin J. Houle, & George M. Weaver

Bibliography

I love reading ☺ (6/6)

[TFN] ***The 'Tribe Flood Network' distributed denial of service attack tool***, David Dittrich

[TFN2K] ***TFN2K - An Analysis***, Jason Barlow & Woody Thrower

[VARNISH] ***Varnish Cache***, http://www.varnish-cache.org/

[ZENGLISH] ***From lost to the river***, Guester and Colin, ISBN-13: 978-8484600008

This is the end my only friend

The snake is long, seven miles.

¿Any questions?

Gerardo García Peña <ggarciapena@deloitte.es>

Greetings

This presentation wouldn't have been possible without their help

- To **Emet-Jon Velasco**, for his help and support developing this presentation.
- To **Oriol Navarro** and **Alexis Porros** for reviewing my despicable English translation.
- To my **Deloitte team** for their human and professional excellence.
- To **Nopcode** and **Summer Camp Garrotxa** where DoS:IS was presented for the first time.
- To the **ano.lolcathost.org** team for giving me a testbed for my tests.
- To **Silvia** and **Sergi** for the most important support. ☺
- To **Federico Dios** and **Patrick Sullivan** from Akamai for their help and support.
- And to all the **bibliography authors**, the key for the contents of this presentation.
- And remember: the security is not always what it seems, like the legend of Beaver Dam.

Si desea información adicional, por favor, visite www.deloitte.es