



# Attacks on Large US Bank During Operation Ababil March 2013



## **Table of Contents**

Executive Summary
Background: Operation Ababil
Servers Enlisted to Launch the Attack
Attack Vectors
Variations of HTTP/s Floods
Clear Text HTTP's Flood
Search Page Request HTTP Attack5
Authentication Engine Flood5
Additional Notes on the HTTP/S Floods6
Summary6



## **Executive Summary**

Along with several major banks in the U.S., one large US Bank has been under attack since October 2012 as part of the infamous "Operation Ababil". The bank has been unable to mitigate the attacks for more than 5 months and has suffered from continual service interruptions on the banks online services.

On March 12th another massive attack period started, but this time Radware DefensePro was already deployed onsite and Radware's Emergency Response Team (ERT) was invoked. During the next few weeks the ERT worked closely with the bank to modify its system infrastructure and deploy Radware's AMS until it achieved a successful mitigation for all attacks.

The main attacks seen at the bank were volumetric UDP/ICMP floods coupled with HTTP/S application floods. The volumetric floods reached a peak of 16Gbps and were mitigated upstream by the ISP. However, the ISP was not able to mitigate the application level attacks and these were the attacks which took down the bank each time. Some of the Bots that participated in the attack were more advanced than we have previously seen and were able to follow 302 redirects and cookies. At some point, one Bot was even able to successfully pass the Java script challenge for the first time ever. This required the ERT and Radware R&D to come up with a very quick resolution for the advanced Bot.

## **Background: Operation Ababil**

On early September 2012, videos, about 14 minutes in length, that claimed to be "trailers" of a longer named "**Innocence of Muslims**" film were uploaded to YouTube. The film that claimed to contain offending content to the Muslim community has invoked demonstrations and violent protests in many Muslim countries which included among others an attack on U.S consulates and embassies.

On September 18th, 2012 a group called "Cyber fighters of Izz ad-din AI qassam" announced an upcoming cyber attack campaign and called for an attack on what they claim 'American and Zionist' targeted. The attack campaign was named "Operation Ababil" which was also the name of a failed Pakistani military operation that occurred in April, 1984.

The prolong ongoing attack campaign has set a goal to take down US based financial institutions web sites and online services.

#### **Servers Enlisted to Launch the Attack**

In recent months, Radware's ERT has witnessed what may be a new dramatic change in the DDoS landscape - the appearance of server-based botnets. Unlike the early days of single-server attacks, the new DDoS attacks employ multiple server machines, spread out geographically and organized in a powerful botnet. This new type of server-based DDoS architecture can be much more threatening than the common botnet attacks for several reasons:



Figure 1- Original Attack Campaign Announcement



- **Firepower** servers have a much larger upload bandwidth, which enables fewer machines to produce the same impact as many client Botnets. Consider that the average US home computer has 600 Kbps upload speed, whereas a typical hosted server has an upload speed of 1Mbps-100Mbps up to X150 times the bandwidth speed.
- **Reliability** servers provide a far more reliable environment compared to home PCs. Home PCs are frequently shut down or taken offline, so that attackers must enlist a much larger number of computers than those that will actually be used in the attack. Servers, on the other hand, are always online and available for an attack.
- **Command and Control** controlling a small number of highly available servers eliminates many of the challenges related to orchestrating thousands of unreliable botnet computers.

The attack on the bank continued the trend of attacks by servers and the attackers used the infamous Bot named 'itsoknoproblembro' aka `BroBot`. 'itsoknoproblembro' is a general purpose PHP script injected into the victim machine allowing the attacker to upload and execute arbitrary Perl scripts.

The 'itsoknoproblembro' script injects an encrypted payload, in order to bypass IPS and Malware gateways, into the index.php, allowing the attacker to upload new Perl scripts at any time. Initial server infection is usually done by using the well known RFI/LFI techniques.

By uploading Perl scripts that run different DoS flood vectors, the server might act as a Bot in a DDoS Botnet army. Although originally designed for general purpose, some variants of this tool were found in the wild, customized to act as a proprietary DDoS tool, implementing the flood vector logics inside.

This attack tool is hard to detect as the Bot dynamically updates its attacking nodes and characteristics. Each compromised server can produce high bandwidth and high rate of traffic utilizing multiple attack vectors in parallel.

## **Attack Vectors**

The main attacks seen at the bank were volumetric UDP/ICMP floods coupled with HTTP/S flood. The volumetric floods were mitigated upstream by the ISP's and reached levels greater than 16Gbps. The ISP's however could not mitigate application level floods and these were the attacks which took down the bank each time.

ERT observed the following types of attacks at the bank:

- 1. HTTP/S floods
- 2. TLS/SSL negotiation attacks
- 3. Server cracking attempts
- 4. Volumetric floods

## Variations of HTTP/s Floods

Multiple vectors of HTTP and HTTP's floods were observed. They were designed with saturating connection tables, HTTP's daemons, bypassing CDN's, bypassing a fixed signature by changing payloads, impact backend CPU and SSL/HTTP daemon resources, exhaust backend DB systems. Below are some of the attack types we have seen.

#### **Clear Text HTTP's Flood**

This flood was designed to saturate connection tables to port 443 and exploit any exception handling on the backend system. Essentially this flood sends clear text HTTP commands and headers to port 443. It is identifiable in the fact that it is sending HTTP clear text to HTTPs port, which is not supported in this particular environment or in most environments. Also the URL is dynamic so it would bypass CDN caching.



Image: Source     Definition     Protocol     Length     Info		
Topstream eq 14         Expression Clear Apply Save           Time         Source         Destination         Protocol Length Info		
Time Source Destination Protocol Length Info		
59 2013-03-22 21:34:30.718761     TCP     66 39155 > https [Ack] 5eq=500200520 Ack=33011315       57 2013-03-22 21:34:30.718724     TCP     66 39155 > https [StN, Ack] 5eq=500200520 Ack=33011315       57 2013-03-22 21:34:30.718328     TCP     74 39155 > https [StN, Ack] 5eq=500200520 Ack=3301       57 2013-03-22 21:34:30.718328     TCP     74 39155 > https [StN] 5eq=500200520 Ack=3301       57 2013-03-22 21:34:30.718328     TCP     74 39155 > https [StN] 5eq=500200520 Ack=300       50 2013-03-22 21:34:30.718328     TCP     66 https > 39155 [Ack] 5eq=3301131589 Ack=50020064       51 2013-03-22 21:34:30.037682     TCP     66 https > 39155 [Ack] 5eq=3301131589 Ack=50020064       58 2013-03-22 21:34:30.037682     TCP     54 https > 39155 [StN, Ack] 5eq=3301131589 Ack=50020064       58 2013-03-22 21:34:30.037682     TCP     76 https > 39155 [StN, Ack] 5eq=3301131589 Ack=50020064       58 2013-03-22 21:34:30.037682     TCP     76 https > 39155 [StN, Ack] 5eq=3301131589 Ack=50020064	9 win-14600 Len-0 TS 131589 win-14600 Len 00 MSS-1460 SACK_PES 9 win-63821 Len-0 TS 200550 win-63821 Len-0 TS 200550 win-60 Len-0	fotal Length Session ID Length Ref 52 60 471 52 52 52 52 52 52 64
Follow TCP Steam         -Stream Content         GET /963c335ec4 HTTP/1.1         Hots:         User-Agent: DOCOMO/2.0 SxH021 (compatible; Y!3-SRD/1.0; http://help.yahoo.co.jp/help/jp/search/indexing/indexing-27         Accept-Language: en-us, en:q=0.5         Accept-Chrocoling: deflate         Accept-Chrocoling: deflate         Accept-Chrocoling: deflate         CLIENT-DF:         Connection: Keep-Allve         Connection: Keep-Allve         Connection: Keep-Allve         End       Save & Brint         Maccel       ASCII         End       Save & Brint         ASCII       EBCDIC         Hot Dump       C Amys	(,hten))	

#### **Search Page Request HTTP Attack**

This HTTP flood was directed to port 80 and is accessible to public web clients, i.e. you do not need to be authenticated to run this search query. The attack was designed to bypass CDN, exhaust web daemon's connection tables as well as the backend system by querying the DB for each search query run. The search query is dynamic and therefore cannot be cached either by the default web cache or CDN services.

Time	Source	Destination	Protocol	Length Info	Total Length Session ID Length Referer
0242 2061-03-27 1	9:44:37.261086		TCP	62 4542 > http [SYN] Seq=313990226 Win=65535 Len=0 M	SS+1460 SACK_PERM+ 48
0243 2061-03-27 1	9:44:37.261088		TCP	60 http > 4542 [SYN, ACK] Seq=2258959042 Ack=3139902	27 Win=1400 Len=0 44
0967 2061-03-27 1	9:44:37.304638		TCP	60 4542 > http [ACK] Seq=313990227 Ack=2258959043 Wi	n+65535 Len+0 40
1206 2061-03-27 1	9:44:37.323603		HTTP	386 GET sarch/default.pageb92a2 HTTP/1.1	372
1207 2061-03-27 1	9:44:37.323606		HTTP	448 HTTP/1.0 200 OK	434
	Follow TCP Stream Stream Content GET 1_Uticoccol, coursel, Utfault, pag Most: waw.in User-Agent: msnPPocket-Products/1. Accept-incoding: deflate Accept-Encoding: deflate Accept-Encoding: deflate Connection: Keep-Alive Keep-Alive: 109 Cache-Control: no-cache WTTP/1.0 200 OK Expires: Sat, 6 May 1995 12:00:00 P3P: CP-MOIL ADD EXP SAL COM NAW Cache-Control: no-store, no-cache Pragma: no-cache Content-Length: 144 Connection: Close chtml>cbody>cscript>document.cook body>c/html> Entire conversation (726 bytes)	eb92a2 HTTP/1.1 0 (+http://search.msn.com/ms q=0.7,*;q=0.7 0 GT 0 GT 0 GT 0 GT 0 GT 0 GT 0 GT 0 GT	nPocket.htm k=0, pre-ch ff17de81; p	eck=0 ath=/';window.location.href=window.location.href; <th></th>	

#### **Authentication Engine Flood**

Authentication retries were a common attack vector seen. This authentication attempt is an HTTP request to the authentication engine. The idea here was to exhaust resources to the web server in terms of the authentication engine itself and the connection table. The engine itself wasn't really affected but the connection limit capacity of the HTTP daemon is. This URL is also not cached by any CDN service so the request goes through directly to the servers.



tcp.stream eq 783	Expression.	. Clear Apply Save					
Time	Source	Destination	Protocol	Length Info	Total Length Session ID Length	Refe	
23 2061-03-19 23:43	:49.843310		HTTP	454 GET /wwwhilemintb HTTP/1.1	440	htt	
24 2061-03-19 23:43	:49.843312	A REAL PROPERTY AND A REAL	HTTP	448 HTTP/1.0 200 OK	434		
58 2061-03-19 23:43	:49.750399		TCP	60 [TCP Dup ACK \$3\$7#1] VHSVC-2 > http [ACK] Seq=147956666 Ack=145	7971 40		
20 2001-03-19 23:43	49.843370		TCP	60 [TCP Dup ACK 5625#1] VMSVC-2 > http [ACK] Seq=147957066 ACK=143	7971 40		
80 2061-03-19 23:43	149.843433		TCP	60 http > vHsvc-2 [ACK] Seq=145/9/3190 ACK=14/95/06/ W1n=1400 Len-	0 40		
0/ 2001-03-19 23:43	10 912260		TCP	60 VH5VC-2 > http [ACK] Seq=14/930000 ACK=143/9/2/93 #10=1310/0 Lt	n=0 40	_	
29 2001-03-19 23:43	-49.043309		TCP	60 vesuel > here [ETH DEN ACV] Can_137057066 Arbu1357073100 vir	-12/ 40	_	
55 2061-03-19 23:43	-20 750324		TCP	78 URGUE 2 V HELP [F18, F58, HELP] SEQULATION HEARTAN 2751270 HIL 78 URGUE 3 V HELP [F18, F58, HELP] SEQULATION HEARTAN 25535 LAND RECULATION URG	0.67 64	_	
56 2061-03-19 23:43	-29.750325		TCP	78 vmsvc-2 > http (SVN) Sec_147956665 win=65535 i en=0 MSS_1460 wS	2 52 64		
	Steam Content           GET /auch           MOST           X-Purpose: preview           User-Agent: Mozilla/5.0 (Macintosh: Intel Mac OS X 10_6_8) Applewebkit/534.57.2 (XMTML, like Gecko) version/5.1.7 Safari/534.57.2 Accept: text/html,application/xml:q=0.9,*/*:q=0.8           Refere: In						
me 5623: 454 byte hernet II, Src: Ra jernet Protocol ve insmission Control pertext Transfer P	Accept-Encoding: gzlp, deflate Connection: keep-alive http://lo.200 ok Expires: Sat, 6 May 1995 12:00:00 GMT P39: CB-MOI ADD DEV PSAI COM NAV OUR C Cache-Control: no-store, no-cache, mus Pragma: no-cache Content-Length: 144	TRo STP IND DEM t-revalidate, post-che	ck=0, pre	-check-0			
	chtml>-chody>-sscript>-document.cookie='zzzzzzec71e417zzzzzzec71e417; path=/';window.location.href=window.location.href;-         Entire conversation (P94 bytes)         End       Save & Bint         ASCI       EBCDIC         Hex Dump       C Anzys   End       Save & Bint       ASCI       EBCDIC       Hex Dump       C Anzys       Raw						

## Additional Notes on the HTTP/S Floods

The floods targeting the bank were not a high rate in terms of bandwidth but caused significant damage to the backed and impacted service availability. The floods were mixed and there was never just a single HTTP/s flood happening at one time. All header values were dynamic and were constantly changed during the course of the attack.

#### Summary

During the entire Operation Ababil attack campaign, the attackers demonstrated a high level of sophistication and skills that took down the online services of some of the world's largest banks. The attack on this bank also revealed the dynamic of the attackers and their ability to change their attack tools and patterns during the attack campaign. The attackers even managed to modify their Brobot to successfully respond to a java script challenge, and to appear as a legitimate user at some point during the campaign. The fact that the bank uses CDN services made the detection and attack mitigation even more complex as the CDN masks a lot of the required information, which results in longer response time and adds more complexity.

As demonstrated in this attack case, organizations should deploy a DDoS mitigation solution on premise and combine with 24x7 support of security experts to successfully fight today's cyber security threats.

© 2013 Radware, Ltd. All Rights Reserved. Radware and all other Radware product and service names are registered trademarks of Radware in the U.S. and other countries. All other trademarks and names are the property of their respective owners.

SVC-ERT-US-Bank-Attack-Report-DS-01-2013/05

