Imperva's Web Application Attack Report

Edition #2 - January 2012

Table of Contents

① Abstract

As a part of its ongoing Hacker Intelligence Initiative, Imperva's Application Defense Center (ADC) observed and categorized attacks across 40 applications, monitoring millions of individual attacks targeted at web applications over a period of six months, June-November 2011. The analysis shows:

- **Hackers continue to increase the scale of their attacks**: In our last report, we explained that websites are probed about once every two minutes, or 27 times per hour. Over the past six months, the number of probes has dropped to 18. Though a drop, this change does not mean hackers are any less persistent. In fact, when applications are attacked, hacker firepower actually saw a 30% increase. In July, we reported that applications experience about 25,000 attacks per hour. In the last six months, this has increased to nearly 38,000 attacks – or ten per second.
- **Hackers exploit five common application vulnerabilities**: We have identified and investigated malicious traffic containing the following technical attacks: *Remote File Inclusion (RFI), SQL Injection (SQLi), Local File Inclusion (LFI), Cross Site Scripting (XSS) and Directory Traversal (DT). Cross Site Scripting and Directory Traversal* are the most prevalent classical attack types.
- **Hackers are relying on business logic attacks due to their ability to evade detection**: We also investigated two types of Business Logic attacks: *Email Extraction and Comment Spamming (EmExt and ComSpm, respectively, in following Figures and Tables)*. Comment Spamming injects malicious links into comment fields to defraud consumers and alter search engine results. Email Extraction simply catalogs email addresses for building spam lists. These Business Logic attacks accounted for 14% of the analyzed malicious traffic. Email Extraction traffic was more prevalent than Comment Spamming. A full anatomy of BLAs is described in this report.
- **The geographic origin of Business Logic attacks were**:
  - *Email extraction was dominated by hosts based in African countries.*
  - An unusual portion of the *Comment-spamming* activity was observed from *eastern-European countries.*

At the end of our report, we provide a list of technical recommendations for security teams.

② Executive Summary

At the Gartner Cyber Security Conference keynote in June 2011, New York Times technology columnist David Pogue called 2011 "the year of the hacker." Indeed, 2011 witnessed several major high-profile breaches across government and private enterprises along with the growth in hacktivism, nation-sponsored hacking as well as "old fashioned" hacking for profit. Imperva's web application attack report (WAAR) is designed to help security teams as well as nontechnical business managers understand cyber security risks and to identify mitigation steps.

Our key findings this time:

› **Hackers continue to increase the scale of their attacks**: In our last report, we explained that websites are probed about once every two minutes, or 27 times per hour. Over the past six months, the number of probes has dropped to 18. Though a drop, this change does not mean hackers are any less persistent. In fact, when applications are attacked, hacker firepower actually saw a 30% increase. In July, we reported that applications experience about 25,000 attacks per hour. In the last six months, this has increased to nearly 38,000 attacks – or ten per second.

Attack automation is attractive for several reasons:

- Automatic tools enable an attacker to attack more applications and exploit more vulnerabilities then any manual method possibly could.
- The automatic tools that are available online save the attacker the trouble of studying attack methods and coming up with exploits to applications' vulnerabilities. An attacker can just pick a set of automatic attack tools from the ones that are freely available online, install them, point them at lucrative targets and reap the results.
- The tools use resources, like compromised servers that are employed as attack platforms, more efficiently.

In fact, there were two prominent examples of automated attacks this year:

- FBPwn: This tool, developed in Egypt, automatically creates copies of real Facebook user accounts, friends users and then extracts personal data from those who accept the request. Considering Facebook's rapid growth, this example highlights how hackers must scale in parallel.[1]
- Refref: This tool, developed by hacktivist group Anonymous, automates SQL injection attacks with the purpose of shutting down the application. This attack, known as a denial of service (DOS), has traditionally been accomplished in the past by bot armies at some expense to the hacker. Refref allows a single hacker to bring applications down by themselves without relying on 3rd parties.[2]

› **Hackers exploit five common application vulnerabilities**: We have identified and investigated malicious traffic containing the following technical attacks: *Remote File Inclusion (RFI), SQL Injection (SQLi), Local File Inclusion (LFI), Cross Site Scripting (XSS) and Directory Traversal (DT). Cross Site Scripting and Directory Traversal* are the most prevalent classical attack types. Why are these vulnerabilities targeted? Hackers prefer the path of least resistance and application vulnerabilities offer an rich target. As a Forrester survey recently noted, "Most security organizations continue to focus inappropriate attention on network vulnerabilities and reactive network security tools rather than on proactive application security practices."[3]

[1] http://blog.imperva.com/2011/09/the-automation-of-social-engineering.html
[2] http://www.theregister.co.uk/2011/08/04/anon_develops_loic_ddos_alternative/
[3] Forrester Research, Forrsights: The Evolution Of IT Security, 2010 To 2011, February 2011.

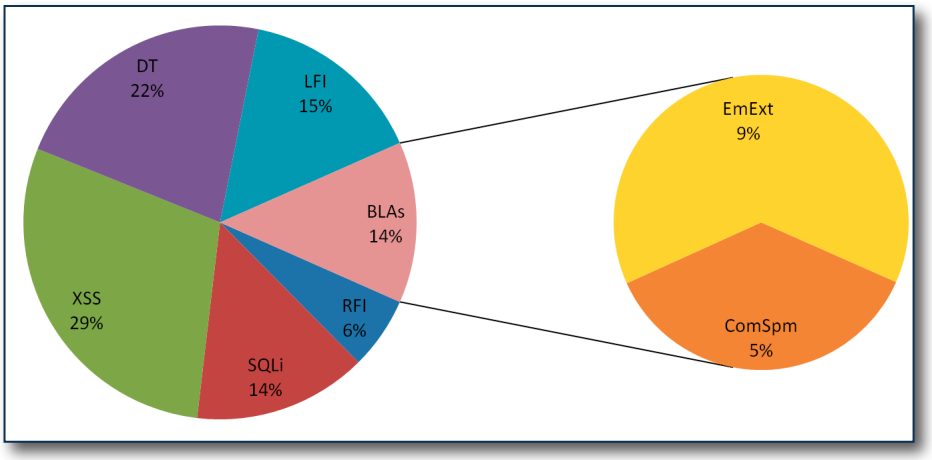

Figure 1: Relative volume of malicious traffic

- **Hackers are relying on business logic attacks due to their ability to evade detection**: We also investigated two types of Business Logic (BLA) attacks: *Email Extraction and Comment Spamming (EmExt and ComSpm, respectively)*. These Business Logic attacks accounted for 14% of the analyzed malicious traffic. Email Extraction traffic was more prevalent than Comment Spamming. What makes BLAs attractive for hacking? BLAs follow a legitimate flow of interaction of a user with the application. This interaction is guided by an understanding of how specific sequences of operations affect the application's functionality. Therefore, the abuser can lead the application to reveal private information for harvesting, allocate her a disproportionate amount of shared resources, skew information shared with other users, etc. – often bypassing security controls. The motivation for BLAs is that the attacker can convert these effects to monetary gains. In fact, we detail the anatomy of BLAs from start to finish.
- **The geographic origin of business logic attacks**:
  - *Email extraction* was *dominated by hosts based in African countries.* We believe the spam traditions, originating with the Nigerian 419 scams, continue today in African nations.
  - An unusual portion of the *Comment-spamming* activity was observed from *eastern-European countries.* As Brian Krebs noted, Eastern Europe, particularly Russia, have flourishing industries selling fake medicines and other scams.[4]

[4] http://krebsonsecurity.com/2011/02/pharma-wars/

③ Analysis Methodology

This security summary report is based on observing and analyzing Internet traffic to 40 web applications during the past 6 months (June-November 2011). We extracted from the traffic security attacks on these applications, categorized them according to the attack method, and identified patterns and trends within these attacks.

Monitoring of the web applications deployed at these sites over a period of several months was accomplished using automatic tools. The logged security events were analyzed using Imperva's special-purpose software and knowledge base. This analysis used several measures, including matching to known attack signatures, comparison to black lists of malicious hosts, and calculation of statistical properties of malicious traffic. Imperva's security experts performed additional detailed analysis of the important events and patterns.

We would like to emphasize the *changes made in our analysis methodology* relative to our previous Web Attack Analysis Report[5]. These changes had enhanced the collected data and had affected the reported results compared to those reported for November 2010-May 2011:

- We have added 10 web applications to the observed applications set, so our data is more varied.
- We have categorized the observed malicious traffic into 7 types, compared to 4 types in the previous report. The new categories include business logic attacks, which were not analyzed in the previous period's report.
- We have improved and refined our automatic web attack detection and analysis tools, in accordance with new attacks methods and newly published applications' vulnerabilities that were reported during the last months.

Therefore, this report represents a better understanding of current attacks employed against web applications in the wild.

The insights from the analysis led us to conclusions about the threatmap faced by web applications in today's World Wide Web. We also came up with recommendations for thwarting the prevalent threats and improving web security.

[5] See: http://www.imperva.com/download.asp?id=114

④ Analysis Results

This report summarizes our analysis of attacks on web applications. The malicious traffic we observed can be broadly described as either Technical attacks or Business Logic attacks (BLA). Technical attacks seek *security weaknesses* in an application and use *exploits* which are tailored to these vulnerabilities, for purposes like controlling the application's server, modifying the application's data or harming the application's users. BLAs mimic an innocent application user and through a *flow of legitimate operations* force the application to behave in a manner that is profitable to the attacker.

The technical attack types that were frequently observed in the data are: SQL Injection, Local File Inclusion, Remote File Inclusion, Directory Traversal and Cross Site Scripting. Of the Business Logic attacks, Email Extraction and Comment Spamming were the most visible.

We begin our analysis by describing phenomena and trends that are common to the entire observed malicious traffic. We dedicate a section to define each attack type and describe specific phenomena associated with it in sections 4.3-4.4. Section 4.2 delves into the use of automation in all attack types.

4.1 Overview

4.1.1 - Attacks Categories

The relative volume of traffic associated each attack type during June-November 2011 is shown in Figure 2. We have identified and investigated malicious traffic containing the following technical attacks: *Remote File Inclusion (RFI), SQL Injection (SQLi), Local File Inclusion (LFI), Cross Site Scripting (XSS) and Directory Traversal (DT). Cross Site Scripting and Directory Traversal* are the most prevalent classical attack types.

We also investigated two types of Business Logic attacks: *Email Extraction and Comment Spamming (EmExt and ComSpm, respectively, in following Figures and Tables)*. These Business Logic attacks accounted for 14% of the analyzed malicious traffic. Email Extraction traffic was more prevalent than Comment Spamming.

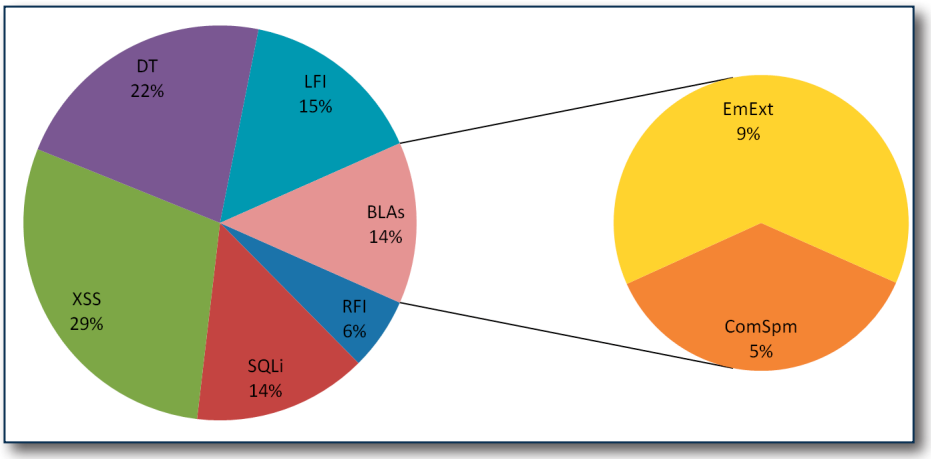

Figure 2: Relative volume of malicious traffic

Table 1 and Table 2 summarize the statistical attributes of the occurrence rate of each attack category:

- All attacks showed large variance in occurrence rate. This is true in particular for SQLi.
- The attack traffic has short periods of activity peaks punctuated with long periods of low activity. We use the median of the rate of attacks as an estimate of the activity without the peaks, since it is less sensitive to outliers. The average attack rate is 2-3 times higher than the commonly-observed rate.
- As further proof that most of the malicious activity of each category occurs within short bursts, and to get a feel for the shape of the asymmetric distribution of attack rates over time, we calculated the 1st and 3rd quartiles. These are the occurrence rates under which the lowest 25% or 75% of the hourly (or daily) occurrence rates were observed. 75% of the time the attack rate does not exceed 2-3 times the commonly-observed rate indicated by the median.

  In our last report, we explained that websites are probed about once every two minutes, or 27 times per hour. Over the past six months, the number of probes has dropped to 18. Though a drop, this change does not mean hackers are any less persistent. In fact, when applications are attacked, hacker firepower actually saw a 30% increase. In July, we reported that applications experience about 25,000 attacks per hour. In the last six months, this has increased to nearly 38,000 attacks – or ten per second.

	Maximum	Median	1st Quartile	3rd Quartile	Average	Standard Deviation
SQLi	13684	8	4	22	51	308
RFI	1033	8	4	19	33	84
LFI	2436	21	10	45	54	126
DT	5573	33	14	70	73	247
XSS	5612	67	12	129	99	269
ComSpm	1977	11	4	25	18	38
EmExt	7615	10	5	17	28	213

Table 1: Hourly attack activity statistics

	Maximum	Median	1st Quartile	3rd Quartile	Average	Standard Deviation
SQLi	28121	610	327	1321	1112	2255
RFI	5627	240	130	562	467	640
LFI	15075	594	397	972	1193	1905
DT	7193	1425	658	2054	1716	1439
XSS	9962	2540	481	3498	2464	2156
ComSpm	2608	377	131	543	380	298
EmExt	23898	364	233	483	655	2125

Table 2: Daily attack activity statistics

4.1.2 - Attacks Trends

Analyzing the number of attacks and the attacks' distribution over the last 6 months shows the following general trends:

- The total number of attacks against the 40 observed applications is in the range 130-385 thousand per month, with peak attacks reaching 38K.
- The total number of attacks, as well as the number of observed attack of each type, shows considerable variation. Long term trends are overshadowed by seasonal trends and especially occasional attack bursts.
- RFI, LFI, SQLi, DT and XSS attacks peaked during August.
- LFI is usually twice as common as RFI. SQLi is more prevalent than RFI. Except for August, each was generally observed between 10 and 40 thousand times a month.[6]

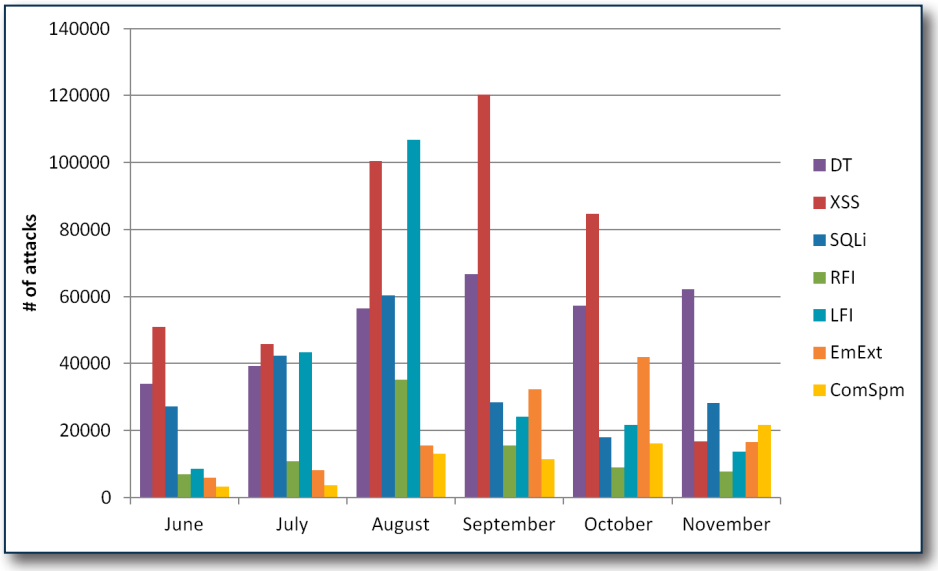

Figure 3: Monthly counts of attacks types

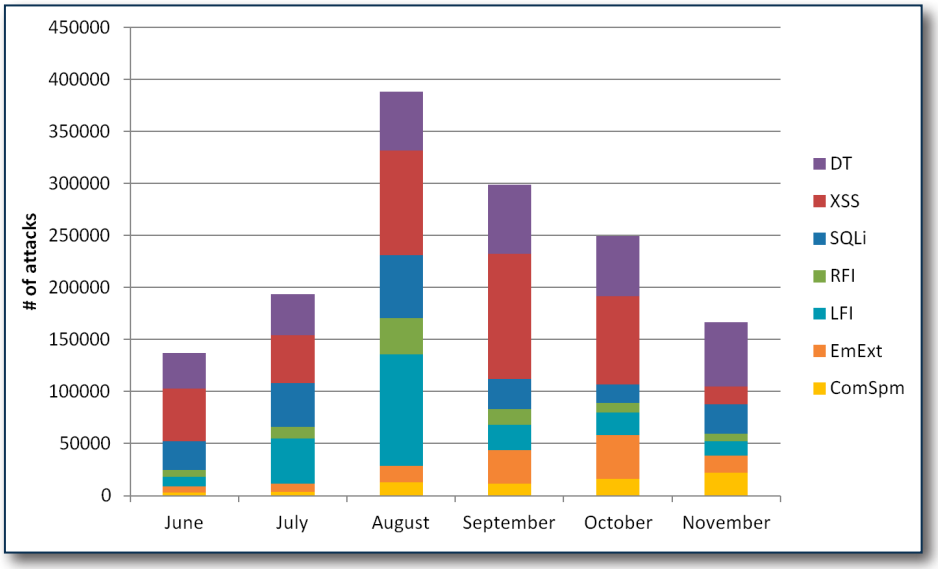

Figure 4: Monthly volume of attacks

[6] Ironically, neither gets the respect they deserve from the security community. For more on why this is the case: http://blog.imperva.com/2011/12/why-rfi-gets-no-respect.html.

4.1.3 - Geographic Dispersion

Even though the identity of the host that initiated an attack is not necessarily indicative of the identity of the attacker that controls it, we have investigated the geographic distribution of the attack-initiating hosts as determined by their IP addresses (Table 3 and Table 4)[7]:

- For all attack types the attackers were spread around the world.
- The leading source countries were rather consistent across all attack types: Most of the attacks (both in absolute numbers and counting the distinct hosts initiating the attacks) were from the United States for RFI, LFI, SQLi and DT. This follows the trend we saw in our previous Web Application Attack Report.
- However, some attack types showed distinct geographic characteristics:
  - We observed a significant portion of SQLi attacks coming from a relatively small number of Chinese hosts. This follows the trend we saw in our previous report.
  - Email extraction was very distinctive since it was dominated by hosts based in African countries.
  - An unusual portion of the Comment-spamming activity was observed from eastern-European countries.

The extent to which each attacking host is used for a specific attack type varies. However, looking at the ratio of attacks to attacking hosts (Table 5), we noticed that hosts that participate in SQLi, RFI or LFI attacks often issue more attack-related traffic than hosts that participate in Directory Traversal, Email Extraction or Comment Spam. We can speculate about the reasons for this difference:

- Automatic tools used in the former attacks are more generic. Since the tool targets a wider variety of applications, scans more options for vulnerabilities and attempts to use a wider set of exploits, the host on which it runs generates more observable attack traffic.
- Reputation-based defenses are more commonly used against BLAs, if for no other reason than that identifying these attacks based on attributes of each HTTP request is harder. Therefore, attackers spread BLAs over a larger number of hosts to reduce the risk and effect of their black-listing.

RFI		SQLi		DT		LFI		Email extraction		ComSpm	
Country	Attacks	Country	Attacks	Country	Attacks	Country	Attacks	Country	Attacks	Country	Attacks
USA	33	USA	143	USA	741	USA	50	European Union	34	Russian Federation	14
France	16	China	24	Canada	153	France	33	Senegal	33	Ukraine	11
Italy	5	United Kingdom	10	France	34	Canada	15	Ivory Coast	13	USA	7
Germany	3	Sweden	9	Germany	22	Germany	15	USA	7	Germany	7
Russian Federation	3	Germany	4	United Kingdom	17	Spain	13	Italy	6	China	5
Republic of Korea	3	Netherlands	4	Russian Federation	14	Norway	11	Thailand	4	European Union	3
Canada	3	Ecuador	4	Spain	14	Italy	10	Malaysia	3	Spain	3
European Union	1	Vietnam	3	Italy	13	Russian Federation	10	Germany	3	Latvia	3
Malaysia	1	Romania	3	Norway	12	Singapore	7	Ghana	2	Australia	2
Netherlands	1	India	2	Singapore	7	Netherlands	6	Nigeria	2	France	2

*Table 3: Countries from which most attacks were initiated (**attacks count in thousands**)*

[7] These geographic figures are not normalized by the overall traffic from these countries or the total number of host IPs in each country.

RFI		SQLi		DT		LFI		Email extraction		ComSpm	
Country	Attackers	Country	Attackers	Country	Attackers	Country	Attackers	Country	Attackers	Country	Attackers
USA	790	USA	10124	USA	10157	USA	1537	Senegal	3299	USA	1078
Germany	138	China	922	United Kingdom	1495	Germany	239	Ivory Coast	2382	Russian Federation	823
France	125	Russian Federation	348	Russian Federation	1280	France	190	European Union	1270	China	632
European Union	80	United Kingdom	289	Ukraine	602	European Union	166	Thailand	665	Ukraine	571
United Kingdom	79	European Union	211	Germany	559	United Kingdom	122	USA	423	Germany	370
Canada	74	Germany	159	Canada	515	Canada	119	Malaysia	282	Sweden	255
Republic of Korea	73	Canada	140	China	466	Netherlands	112	Ghana	176	European Union	225
Russian Federation	63	Ukraine	111	European union	354	Republic of Korea	80	Nigeria	174	France	110
Italy	58	France	73	France	298	Italy	75	China	148	United Kingdom	101
Netherlands	58	Republic of Korea	61	Netherlands	187	Russian Federation	69	India	143	Poland	96

*Table 4: Countries with the most distinct **attacking hosts***

	RFI	SQLi	DT	LFI	Email extraction	ComSpm
Median	10	8	3	28	8	6
Average	21	62	10	65	17	12
Standard Deviation	35	360	33	114	23	25

*Table 5: Ratio of **attacks to attacking hosts** per attack type*

4.2 Attack Automation

Cyber-criminals are increasingly using automation to carry out their attacks on web applications. This phenomenon has several reasons:

- Automatic tools enable an attacker to attack more applications and exploit more vulnerabilities then any manual method possibly could.
- The automatic tools that are available online save the attacker the trouble of studying attack methods and coming up with exploits to applications' vulnerabilities. An attacker can just pick a set of automatic attack tools from the ones that are freely available online, install them, point them at lucrative targets and reap the results.
- The tools use resources, like compromised servers that are employed as attack platforms, more efficiently.

Automatic tools open new avenues for evading security defenses. For example, such a tool can periodically change the HTTP User Agent header that is usually sent in each request to an application and that may be used to identify and block malicious clients. As another example, sophisticated automatic tools can split the attack between several controlled hosts, thus evading being black-listed.

Warning signs of automated attacks detailed in section 4.2.1. Most of them are demonstrated in sections 4.2.2 and 4.2.3, where we analyze specific automatic attack campaigns that were extracted from the observed traffic. We talk about the general-purpose nature of automatic attack platforms in section 4.2.4. Finally, an indicator of automatic attacks at the HTTP protocol level is described in section 4.2.5.

4.2.1 - Indicators of Automatic Attacks

There are several generic indicators that an automated attack is in progress. For example:

- Traffic characteristics like attack rate, attack rate change and attack volume. Obviously, if a single host issues HTTP requests to an application at a high rate (e.g., 20 requests per minute) or interacts with it over a very long period, this traffic is software-generated.
- Specific values in the HTTP User-Agent header. Many tools used to attack applications are actually developed as vulnerability scanners for use by the application's administrator. By default, these tools sign the traffic they generate with a special User-Agent header. Attackers sometimes leave this default behavior as is when they abuse the scanner for their own purposes. A black list of User-Agent values helps identify malicious scanning.
- Usually, HTTP requests generated by automation tools do not contain standard headers[8] that web browsers usually send, like *Accept-Language* or *Accept-Charset*. These headers show the languages and character sets that client software, like a browser, can handle. Automatic tools usually do not send them, since they do not present the application's responses as Web pages. It must be emphasized that benign automatic tools, like search engine crawlers, also do not send these headers while scanning a site. Nevertheless, the absence of headers usually sent by browsers should trigger close monitoring of the traffic.

Additional indicators of automated attack are more specific to the attack tool or the target application. Examples include:

- Attack tools use a predefined set of attack vectors. For example, SQL Injection tools generate attacks containing fragments of SQL code. Sometimes these attacks are unusual enough that they can become a fingerprint of the tools, and identify the attack with high certainty.
- Attack tools intentionally access resources of the application that are not exposed to a human user browsing the site. For example, some RFI attacks directly access scripts in specific Content Management Systems in order to exploit their vulnerabilities. Anomalies of this kind should be closely monitored.

4.2.2 - Example 1: Gaining access to an application's database

The following is an example of a 2-stage attack on a single application from a single host. In the reconnaissance stage, the attacker looked for potential targets – scanning the site for deployed web servers. In the second stage which occurred *4 days later*, a specific attack was directed from the same host to a single web application on this site. The attacking host is apparently a compromised server located in the United States.

4.2.2.1 - Reconnaissance stage

The Dfind scanner is a known scanner that has been in use for several years[9]. It has a fingerprint of accessing the URL "/w00tw00t.at.isc.sans.dfind:)" while looking for IP addresses of computers hosting web applications. Depending on the configuration of the host, the returned error code can show the attacker that a web server is available there.

In the observed attack incident, Dfind was used to scan 87 IP addresses for web servers during a period of 21 minutes.

4.2.2.2 - Brute force login attack

phpMyAdmin is a free software tool written in PHP intended to handle the administration of MySQL database over the World Wide Web. phpMyAdmin supports a wide range of operations with MySQL, from managing databases, tables, fields, relations, indexes, users, and permissions, to the ability to directly execute any SQL statement.[10]

On one of the IP addresses that were scanned by Dfind, We have observed an automated brute-force attack on the authentication of phpMyAdmin users. By using a vocabulary of commonly used weak passwords (e.g., "123456" and "1qa2ws") the tool tries to gain access to the web interface of the database administration application. The entire brute-force login attack was executed over 22 minutes using 6500 HTTP requests.

[8] See: http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html
[9] See: http://isc.sans.edu/diary.html?storyid=900
[10] See: http://www.phpmyadmin.net/home_page/index.php

4.2.2.3 - Early warning signs of automatic attack

This attack incident exhibited several of the warning signs discussed above:

- High rate of traffic – 4 requests per minute in the first stage and over 296 requests per minute over the second stage.
- Tool signature – The Dfind scanner has a unique fingerprint that shows up in the traffic.
- HTTP headers – the Dfind requests were sent without a User Agent header. However, the request during the login attack were sent with an innocent-looking generic "Mozilla/4.0 (compatible; MSIE 5.00; Windows 98)" User Agent. The requests in both stages were sent without browser-like Accept headers.
- IP reputation – the malicious activity identified in the first stage of the attack could have been used to add the attacking host to a blacklist. This would have helped in identifying and blocking the second stage of the attack.

These signs, and especially their combination, could be used to identify and block this kind of automated attack, or at least slow it down and make it considerably less effective.

4.2.3 - Example 2: Automated SQL injection vulnerability scan

The following is an example of a scanner searching for SQL injection vulnerabilities. In this case of automatic reconnaissance activity, the tool enables the attacker to quickly try thousands of combinations of URLs and HTTP parameters of the application in order to acquire potential targets for exploitation.

4.2.3.1 - Time-based blind SQL injection

The tool sends a harmless executable SQL snippet to the application, and parses the response for indications that it was executed. If the application does not return the expected result of the code execution in the response, the tool uses *time-based blind SQL injection*[11]: the tool sends the following SQL snippet to the application[12]:

```
declare@qvarchar(8000)select@q=0x57414954464F522044454C4159202730303A30303A313527
exec(@q) --
```

Which an MS-SQL database interprets as: `WAITFOR DELAY '00:00:15'` (causing the database to simply wait for the specified time). If the HTTP request takes at least 15 seconds to process then the tool notes that SQL commands can be executed through the current URL and HTTP parameter. It also verified that the application uses MS-SQL database.

4.2.3.2 - Early warning signs of automatic attack

The automatic nature of the attack is seen from:

- The rate of malicious traffic: 880 HTTP requests were observed during 10 minutes (88 requests per minute).
- HTTP headers: 65% of the requests were sent without HTTP User Agent header. However, the others were sent with an innocent-looking "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)" User Agent, using the capability of the attack tool to modify the value of this header. Browser-like Accept headers were not sent in the requests at all.
- The unusual pattern of access to application resources: The attack is directed at 37 application URLs (for example: /myaccount/new_customer_register.asp, /products/subdepartment.asp, /home/home.asp, /products/product.asp, /scripts/contact_submit.asp). These URLs are accessed consecutively from the attacking host. For each URL, 5-120 combinations of parameters are tried, and the injected SQL snippet is placed in a different parameter in each combination. This is very different from the access pattern of a human web user.
- Attack tool fingerprint: The sent SQL snippet is quite distinct and can be used to identify similar attacks from this tool.

Again we see that a combination of early warning signs is useful to identify an automated attack and activate appropriate counter-measures.

[11] See: http://www.imperva.com/download.asp?id=352 Hacker Intelligence Summary Report – An Anatomy of a SQL Injection Attack (Sept 2011)

[12] To bypass input sanitation by the application, the automatic vulnerability scanner actually sends to each combination of URL and tested parameter a sequence of 4 slight variations on the same SQL snippet.

4.2.4 - Combined Attacks

Some hosts are used to initiate *several types* of attacks. For example, looking at very active hosts in RFI, LFI and SQLi attacks during August, two hosts showed up as major sources for both LFI and RFI, and one host was a major source for both RFI and SQLi (Table 6). At least some of these hosts are compromised servers. This shows that using automation, hosts can easily be used as a general-purpose attack platform with multiple malicious capabilities. If a host is identified as malicious due to issuing a specific type of attack, it is likely that it can issue other types of attacks in the near future.

Host	Host Information	LFI (August)		RFI (August)		SQLi (August)	
		#attacks	%	#attacks	%	#attacks	%
188.138.90.137	Compromised server, zulu201.startdedicated.com Location: Germany	9330	8.7	1185	3.3	--	--
94.23.12.105	Compromised server, ns204767.ovh.net Location: France	6529	6.1	12129	34.2	--	--
67.207.202.2	Location: United States	--	--	3042	8.6	3686	4.5

Table 6: Hosts participating in several types of attacks during August

4.2.5 - HTTP Protocol Violations

While not an attack per-se, we have often observed violations of the HTTP protocol in the traffic that accompanies attack attempts. These violations included invalid HTTP methods, inclusion of invalid bytes inside parameters, etc. Obviously, this kind of traffic is generated by custom scripts rather than standard web browsers. Thus, protocol violations are an indicator of automatic attacks. HTTP violations may be generated inadvertently by sloppily-written hacking tools. However, we believe most of these the violations are caused intentionally by attackers who attempt to confuse security measures and avoid blocking. For example, in many LFI and Directory Traversal attacks, a Null byte is added to the end of the path of the accessed file. This is intended to bypass applications' defense mechanism that appends "safe" file-type suffixes (like ".png" for image files) to any file path read from input parameters. The Null byte signals the end of a file path, forcing the file system to ignore the appended suffix.

Looking at the number of HTTP protocol violations that appeared in the monthly traffic (Figure 6) shows a moderate incremental trend in the last 6 months. This is different from the findings of our previous report, where the number of protocol violation showed a sharp increase from December to February, but also a steep decline during March-May.

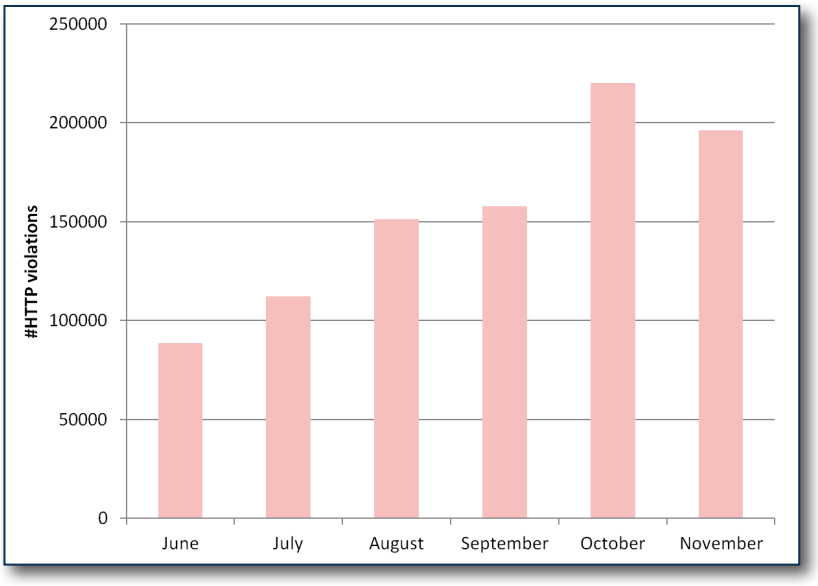

Figure 6: Monthly HTTP protocol violation in the traffic

4.3 Technical Attacks

4.3.1 - SQL Injection

SQL Injection (SQLi) is an attack that exploits a security vulnerability occurring in the database layer of an application (like queries). Using SQL injection the attacker can extract or manipulate the web application's data. The attack is viable when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed.

The monthly number of SQLi attacks is shown in Figure 7. There were 18000-60000 SQLi attacks each month on the observed sites (34100 attacks per month on average). During August there was an unusual SQLi activity: more than 2 times the average rate for the other months (about 28800).

Comparing these results to our previous semi-annual report, the results are similar: the monthly rate of SQLi attacks fluctuates around 30000 attacks per month.

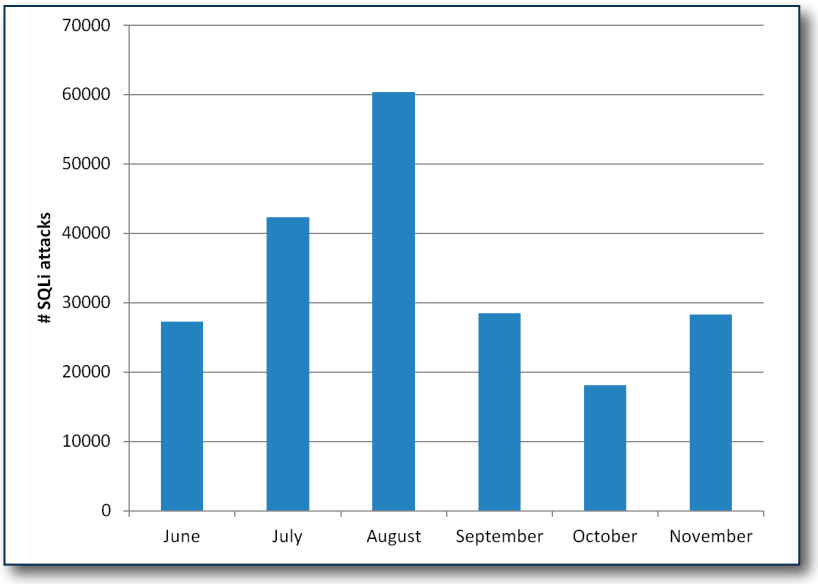

Figure 7: SQLi attacks per month

4.3.2 - Remote File Inclusion

Remote File Inclusion (RFI) is an attack that allows an attacker to include a remote file, usually through a script, on the web server. This attack can lead to data theft or manipulation, malicious code execution on the web server, or malicious code execution on the application's client side (such as Javascript which can lead to other attacks). This vulnerability occurs due to the use of user-supplied input without proper validation.

The monthly occurrence of RFI attacks is summarized in Figure 8. There were 7000-35000 RFI attacks each month on the observed sites (14200 attacks per month on average). In August there were an unusually high number of RFI attacks; excluding it the average monthly rate of RFI attacks is 10000.

The monthly rate of RFI attacks during November-May, as we previously reported, was 5200 on average. This is consistent with the rate we saw on June and November. However, the monthly rate during July-October, and especially during August, was considerably higher. The difference is only partially explained by the addition of applications to our observation set.

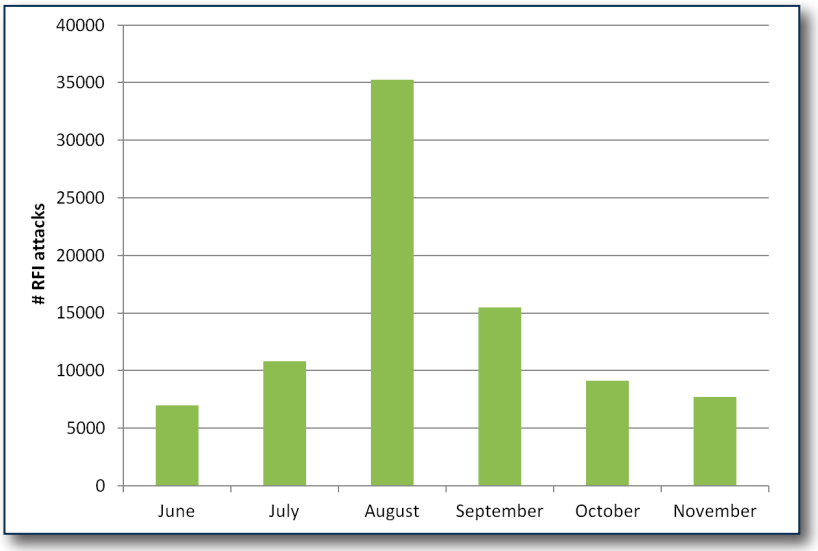

Figure 8: RFI attacks per month

4.3.3 - Local File Inclusion

Local File Inclusion (LFI) is an attack that includes files on a server into the web server. This attack can lead to malicious code execution on the web server. The vulnerability occurs when a page include is not properly sanitized, and allows, for example, directory traversal characters to be injected. LFI attacks often append a Null character to the included file path to bypass value sanitization.

The monthly occurrence of LFI is summarized in Figure 9. There were 8670-106800 monthly LFI attacks, with an average rate of 36400 per month. However, August was clearly an unusual month with respect to LFI attack, which is otherwise observed about 22000 times per month.

LFI attacks were not analyzed independently in our previous Web Attacks Analysis Report, in which they were aggregated into the Directory Traversal category.

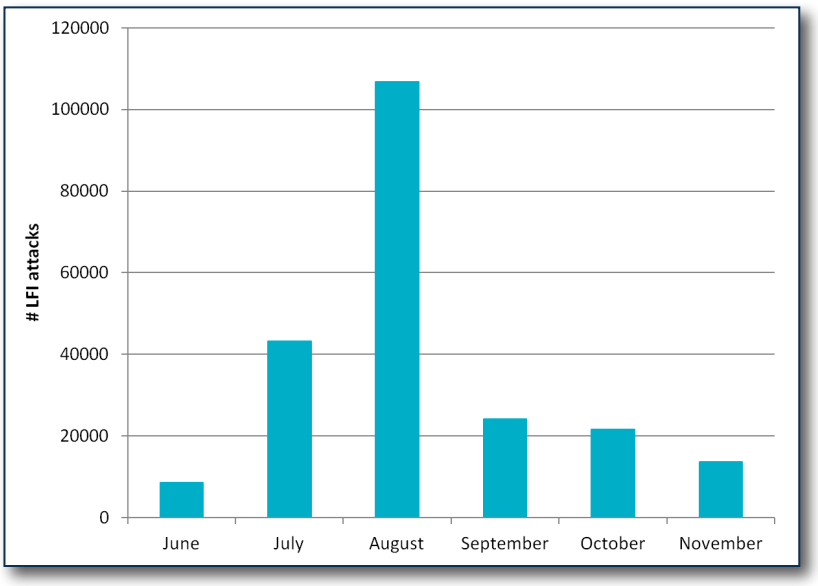

Figure 9: LFI attacks per month

4.3.4 - Directory Traversal

Directory Traversal (DT) is an attack that orders an application to access a file that is not intended to be accessible and expose its content to the attacker. The attack exploits insufficient security validation or insufficient sanitization of user-supplied input file names, so that characters representing "traverse to parent directory" are passed through to the file APIs.

The monthly occurrence of Directory Traversal attacks is summarized in Figure 10. There were 34000-193400 DT attacks each month on the observed sites (141200 attacks per month on average). There were 42000 attacks per month on average during November-May.

According to the data, attackers employ Directory Traversal for several purposes, including attempts to *execute commands*, with patterns like *...\.\...\.\windows\system32\ipconfig.exe* and *../../../winnt/system32/cmd.exe*, and *access sensitive configuration files*, using patterns like *..\\winnt\\win.ini*.

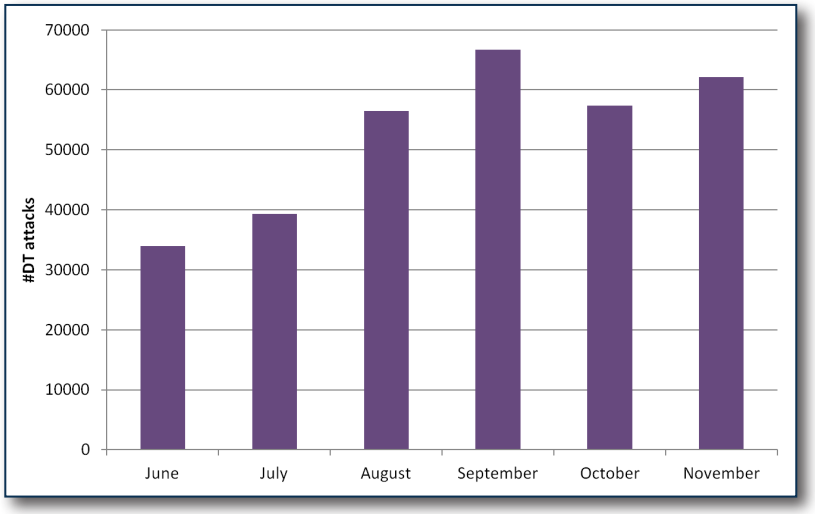

Figure 10: Directory Traversal attacks per month

4.3.5 - Cross Site Scripting

Cross Site Scripting (XSS) is an attack that lets the attacker execute scripts in a victim's browser to hijack user sessions and steal his credentials, deface web sites, insert hostile content, redirect users, hijack the user's browser using malware, etc. XSS flaws occur when an application includes user supplied data in a page sent to the browser without properly validating or escaping that content.

The monthly volume of traffic related to Cross Site Scripting is summarized in Figure 11. Cross Site Scripting is targeted at web users rather than servers. It is used to attack the application's client executing malicious browser code in the context of the trusted application, potentially abusing the trust between the user and the application and vice versa. The traffic we have been able to monitor represents the attacker's set up (see below) and not the traffic of victimized clients. We observed 69800 monthly XSS attacks on average, and an unusually high XSS-related activity during August-October.

On the previous 6 months the average XSS-related attacks were observed 42000 times per month on average. This is consistent with the June-July traffic statistics. The rise in XSS traffic in August-October is mostly related to an unusually high rate of a specific variant of XSS, as detailed below.

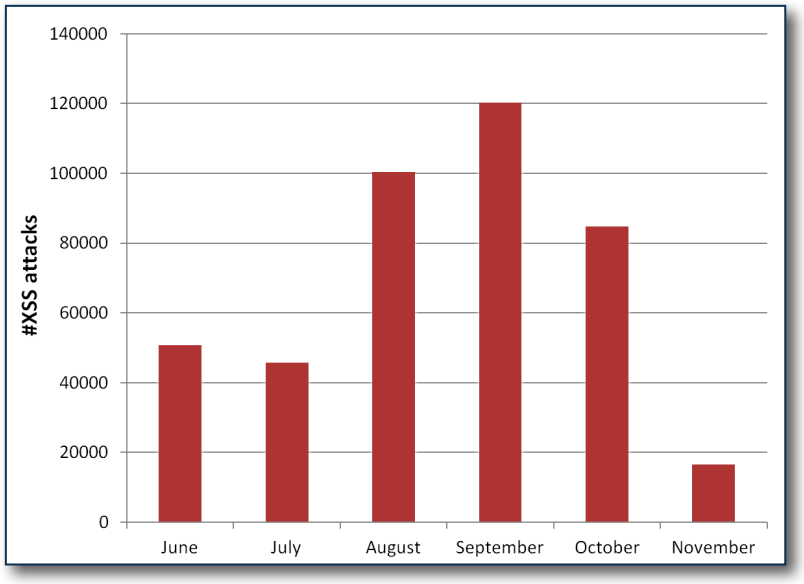

Figure 11: Cross Site Scripting traffic per month

4.4 Business Logic Attacks

A Business Logic Attack (BLA) is an attack which targets the logic of a business application. "Traditional", technical, application attacks contain malformed requests. On the other hand, business logic attacks include legitimate input values. This lack of unusual content attributes makes a business logic attack difficult to detect. BLAs abuse the functionality of the application, attacking the business directly. A BLA is further enhanced when combined with automation, where botnets are used to challenge the business application.[13]

BLAs follow a legitimate flow of interaction of a user with the application. This interaction is guided by an understanding of how specific sequences of operations affect the application's functionality. Therefore, the abuser can lead the application to reveal private information for harvesting, allocate her a disproportionate amount of shared resources, skew information shared with other users, etc. The motivation for BLAs is that the attacker can convert these effects to monetary gains.

[13] See: http://www.imperva.com/resources/glossary/business_logic_attacks.html

4.4.1 - Email Extraction

Email extraction (also called email scraping) is the practice of scanning web applications and extracting the Email addresses and other personal contact information that appear in it. These emails are then used for promotional campaigns and similar marketing purposes. Email extraction is one of several activities that harvest data from web applications against the intent of the data owners and the applications' administrators.

Summary of the monthly Email extraction attacks is shown in Figure 12. On average there were 20000 such attacks each month, but clearly there was a peak of activity during September-October and much lower activity during other months.

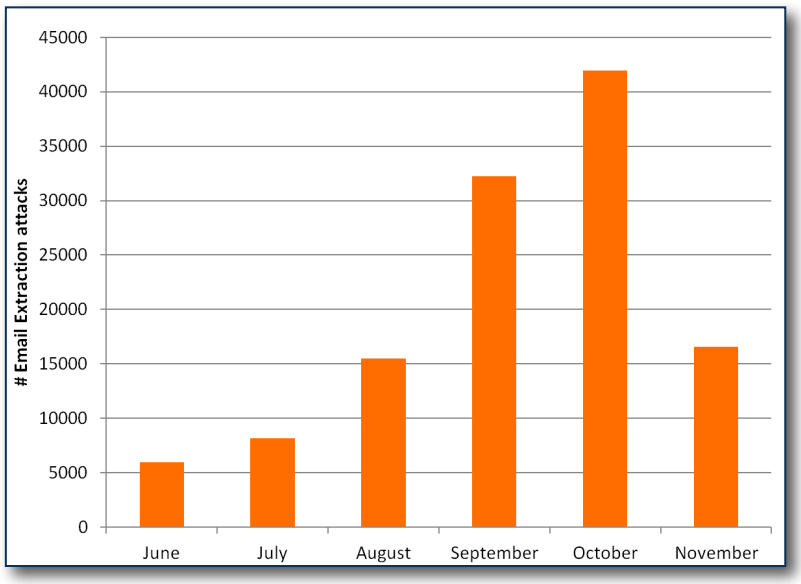

Figure 12: Email extraction attacks per month

Email extraction is a "grey area" practice: attackers earn easy money by selling information extracted illegitimately from web applications. The attack does not exploit vulnerabilities in the application. Rather, the data is extracted by automatically scanning the targeted application, while imitating a user's browsing activity. To speed up the attack and avoid black listing, several scans are run concurrently using web proxies.

Email extraction is offered on the web both as an online service (i.e., "pay on delivery")[14] and as software tool for download. The notorious "Beijing Express Email Address Extractor" (Figure 13), a software tool freely available on the web, was responsible for over 95% of the Email Extraction activity we identified. Usage of the commercial software Advance Email Extractor[15] was also seen in the traffic.

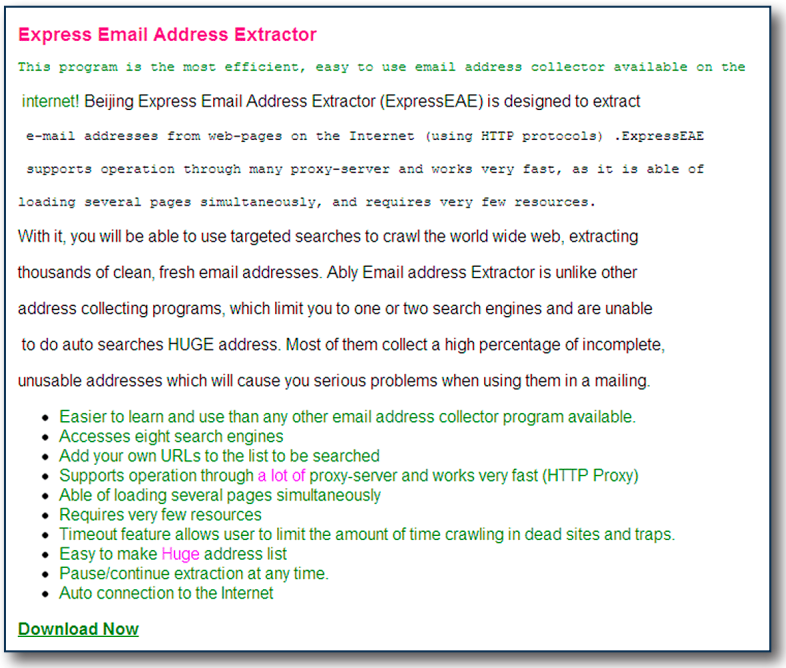

Figure 13: Web description of Beijing Express Email Address Extractor[16]

Hosts that sent Email extraction traffic to the observed application had very unusual geographic locations: Of the 9826 hosts, 3299 (34%) were from Senegal and 2382 (24%) were from Ivory Coast. Other unusual countries (Thailand, Malaysia, Ghana and Nigeria) were also prominent in the list of attacks' geographic sources. Obviously, attackers are hiding their tracks by employing remote and perhaps less monitored hosts for this attack type.

[14] See for example: http://www.iwebscraping.com/webscrapingproject.php?pn=email-scraping-or-extraction-service~~2
[15] Developed by EMMA Labs: http://www.mailutilities.com/
[16] See: http://www.mail-archive.com/lftp-devel@uniyar.ac.ru/msg00903.html

4.4.2 - Comment Spamming

Comment spamming is a way to manipulate the ranking of the spammer's web site within search results returned by popular search engines. A high ranking increases the number of potential visitors and paying customers of this site. The attack targets web applications that let visitors submit content that contains hyperlinks: the attacker automatically posts random comments or promotions of commercial services to publicly accessible online forums, which contain links to the promoted site.

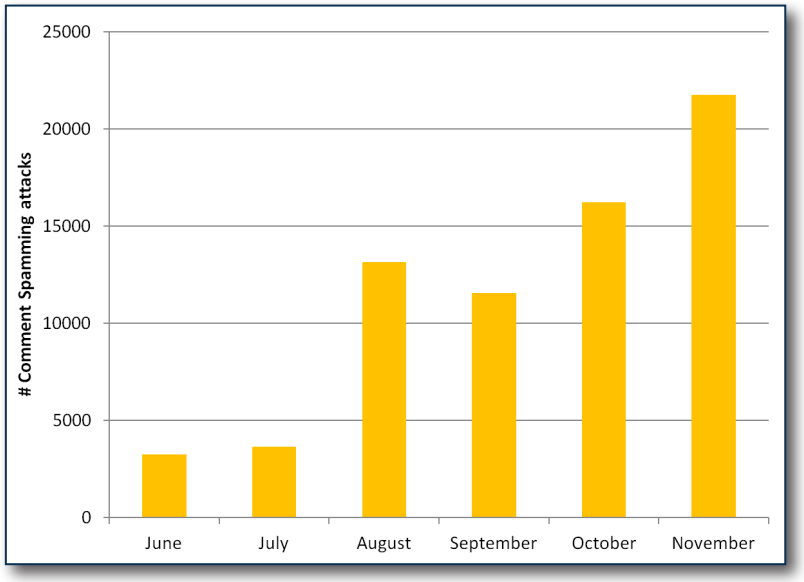

Figure 14: Comment spamming attacks per month

Comment spamming is based on automatic tools that masquerade as a human that surfs the web, but with a "hidden agenda" of leaving traces of good feedback (in various forms) to promoted sites.

The volume of traffic associated with comment spamming is presented in Figure 14. The observations from the last 6 months show a long term trend of growth in traffic related to comment spam. It should be emphasized that not all of this traffic contains the actual spam – the automatic tools must interact with the application like a user (for example, find a forum for posting data, register as a user, login and find a popular thread for posting the spam) before actually injecting the spam link into the site.

We have observed several variants of comment spamming within the monitored traffic. For example:

Figure 15: Facebook page promoted by observed comment spam

- The spammer posted comments to an application's web forum. In some of these posts the *Referer* HTTP header was a URL of a Facebook page promoting specific prescription drugs were given in posts to (see Figure 15). This URL would show up in the spammed site's logs, increasing the ranking of the promoted site in search engine results.
- The spammer promoted the reputation-based ranking of specific answers in a discussion forum. In this application, experts answer questions posted by users. Answers and experts are ranked and displayed based on users' feedback (e.g. based on correctness and usefulness). By artificially increasing the good reputation of specific answers, this promoted content becomes more visible.

An unusual attribute of the observed Comment Spamming attacks is the geographic locations of the involved hosts: Hosts from Russian Federation, Ukraine, Latvia and Poland were very active in this sort of attack. We note that this phenomenon was also detected by other researchers through other means[17].

[17] For example, the top 3 and 7 out of 10 worst spammers ranked by spamhaus are from east-European countries: http://www.spamhaus.org/statistics/spammers.lasso

Comment spamming can be tricky to identify, since a large part of the spammers traffic looks no different than the traffic generated by an innocent user. Good indications of potential malicious activity of this kind are black lists of User Agent values and hosts' IPs, based on activity observed in many applications. Generic indications of automatic attacks, like high rate of requests and missing HTTP headers that are normally sent by browsers, are relevant as well.

One of the mechanisms used by applications to defend against comment spammers is *CAPTCHA* challenges, which require the user to visually identify a specific text within a non-trivial image. We have observed attempts by automatic tools to answer these challenges, probably using a predefined pool of responses to challenges. Even if these attempts are mostly unsuccessful, with enough retries the automatic spamming tool has a chance to eventually get the answer right and complete its spamming task.

⑤ Recommendations

Many of the attacks described in this report are not difficult to mitigate. However, we did find that Web applications face attacks that are becoming more diverse, more technically sophisticated and more difficult to detect and block. Obviously, security counter-measures must keep up with the threats to prevent damages and losses to the business and its customers. What are the right mitigation steps? We've attempted to create a complete list to help security teams improve their odds.

In our last report, we made several recommendations which are still valid:

1. Deploy security solutions that detect automated attacks. This detection must be done as early as possible during the attack. This theme is expanded below.
2. Detect and block attacks against known vulnerabilities. The knowledge base of exploitable weaknesses in the application must be frequently updated.
3. Acquire intelligence on malicious sources and apply it in real time. Black lists of attacking hosts are still an efficient counter measure. However, the lists must be up to date to be effective.
4. Participate in a security community and share data on attacks. The increased automation and scale of attacks leave a large footprint on the web – but it can only be seen by looking at data gathered from a large set of potential victims.

Business Logic Attacks follow a legitimate flow of interaction of a user with the application. Therefore, the usual negative security model of having signatures to detect "known bad traffic" has only limited value against these attacks. Drawing on the insights from this report, we recommend additional steps for dealing with BLAs:

- Acquire intelligence on BLA sources and apply it in real time. For example, comment spammers are active long after they are publicly unmasked. Intelligence must be focused per each attack type, since, as we saw, attackers using Comment Spam and Email Extraction exhibit different attributes.
- Geographic information about traffic can help make security decisions in real time. For example, the analyzed BLA attacks have very distinct geographic characteristics.

The increasing reliance of attackers on automation and the huge volume of malicious traffic the automatic tools generate mean that it is crucial to detect attacks that are generated by such tools quickly, accurately and automatically. As a general guideline, a small set of attributes of the traffic and the web client must be continuously measured and monitored. Deviations from the normal profile of traffic should trigger close scrutiny by dedicated software and personnel.

An anti-automation arsenal is the linchpin of any security mechanism. This is true for traditional technical attacks, and it is also the essence of the described BLAs. For example, if an attacker cannot extract Email addresses quickly using automation then the attack is mitigated, since the attacker will search for a more lucrative target. A checklist for detecting and battling automation includes:

- Reputation-based detection: Acquire and use black lists of hosts employed by attackers.
- High click rates: Traffic shape is the most basic indicator of automated activity. Above an application-related threshold (e.g., 3 clicks per second), the application should delay or block the interchange with the web client.
- Technical attributes of the incoming traffic: Traffic generated by software tools often has technical characteristics (like specific HTTP headers) that are different than traffic generated by common browsers. If this is not an expected usage scenario, block such traffic.
- Repetition of business actions: For example, many login failures may indicate a password brute force attack. Of course, your security device must be able to recognize such "deviations or strange behavior.[18]
- Challenging the application's web client: Test if your application is really interacting with a browser. For example, "fake" browsers do not have capabilities like Javascript execution. The application flow should contain sending Javascript code to the client and verifying that is was actually executed.
- Checking that a human is "in the loop": Test that the end user is human by incorporating capabilities like CAPTCHA.

[18] Web application firewalls can do this.

Hacker Intelligence Initiative Overview

The Imperva Hacker Intelligence Initiative will be going inside the cyber-underground and providing analysis of the trending hacking techniques and interesting attack campaigns from the past month. A part of Imperva's Application Defense Center research arm, the Hacker Intelligence Initiative (HII), is focused on tracking the latest trends in attacks, Web application security and cyber-crime business models with the goal of improving security controls and risk management processes.

Imperva
Headquarters
3400 Bridge Parkway, Suite 200
Redwood Shores, CA 94065
Tel: +1-650-345-9000
Fax: +1-650-345-9004

Toll Free (U.S. only): +1-866-926-4678
www.imperva.com