

Hacker Intelligence Initiative, Monthly Trend Report #3

Hacker Intelligence Summary Report – The Convergence of Google and Bots: Searching for Security Vulnerabilities using Automated Botnets

In this monthly report from Imperva's Hacker Intelligence Initiative (HII), we describe how popular search engines are used as an attack platform to retrieve sensitive data, a.k.a. "Google Hacking". This attack is further enhanced by deploying bots to automate the process and to evade anti-automation detection techniques commonly deployed by the search engine providers. Although Google Hacking has been around – in name – for some time, some new innovations by hackers require another, closer look. Specifically, Google, and other search engines, put in place anti-automation measures to stop hackers from search abuse. However, by using distributed bots, hackers take advantage of bot's dispersed nature, giving search engines the impression that individuals are performing a routine search. The reality? Hackers are conducting cyber reconnaissance on a massive scale. Imperva's Application Defense Center (ADC) has followed up on a particular botnet and has witnessed its usage against a well-known search engine provider. By tracking this botnet, they found how attackers lay out the groundwork to simplify and automate the next stages in an attack campaign against web applications. In this report, we describe the steps that hackers take to leverage on the power of search engines to successfully carry out their attacks to massively collect attack targets. Our findings show that during an attack, hackers can generate more than 80,000 daily queries to probe the Web for vulnerable Web applications. We provide essential advice to organizations on how to prepare against exploits tailored against these vulnerabilities. We also propose potential solutions that leading search engines such as Google, Bing and Yahoo can employ in order to address the growing problem of hackers using their platform as an attacker tool.

Our findings show that during an attack, hackers can generate more than **80,000** daily queries to probe the Web for vulnerable Web applications.

An Overview of Google Hacking

On the Internet, search engines have emerged as powerful tools in an attacker's arsenal, providing a way to gather information about a target and find potential vulnerabilities in an anonymous and risk-free fashion. This activity is typically called "Google Hacking". Although the name emphasizes the search-engine giant, it pertains to all search engine providers. Collecting information about an organization can set the stage for hackers to devise an attack tailored for a known application. The specialized exploitation of known vulnerabilities may lead to contaminated web sites, data theft, data modification, or even a compromise of company servers.

Search engines can be directed to return results that are focused on specific potential targets by using a specific set of query operators. For example, the attacker may focus on all potential victims in a specified geographic location (i.e. per country). In this case, the query includes a "location" search operator. In another scenario, an attacker may want to target all vulnerabilities in a specific web site, and achieves this by issuing different queries containing the "site" search operator. These particular search queries are commonly referred to as "Google Dorks", or simply "Dorks".

Automating the query and result parsing enables the attacker to issue a large number of queries, examine all the returned results and get a filtered list of potentially exploitable sites in a very short time and with minimal effort.

In order to block automated search campaigns, today's search engines deploy detection mechanisms which are based on the IP address of the originating request.

What's new about this attack campaign that we witnessed? Our investigation has shown that attackers are able to overcome these detection techniques by distributing the queries across different machines. This is achieved by employing a network of compromised machines, better known as botnet.

Hackers also gain the secondary benefit of hiding their identity behind these bots, since it is the compromised host which actually performs the search queries. In effect, the attacker adds a layer of indirection between herself and the automated search queries. This makes the task of tracking back the malicious activity to the individual attacker all the more difficult.

The Hacker's 4 Steps for an Industrialized Attack:

1. **Get a botnet.** This is usually done by renting a botnet from a bot farmer who has a global network of compromised computers under his control.
2. **Obtain a tool for coordinated, distributed searching.** This tool is deployed to the botnet agents and it usually contains a database of dorks.
3. **Launch a massive search campaign through the botnet.** Our observations show that there is an automated infrastructure to control the distribution of dorks and the examination of the results between botnet parts.
4. **Craft a massive attack campaign based on search results.** With the list of potentially vulnerable resources, the attacker can create, or use a ready-made, script to craft targeted attack vectors that attempt to exploit vulnerabilities in pages retrieved by the search campaign. Attacks include: infecting web applications, compromising corporate data or stealing sensitive personal information.

Detailed Analysis

Mining Search Engines for Attack Targets

Search engine mining can be used by attackers in multiple ways. Exposing neglected sensitive files and folders, collecting network intelligence from exposed logs and detecting unprotected network attached devices are some of the perks of having access to this huge universal index. Our report focuses on one specific usage: massively collecting attack targets. Specially crafted search queries can be constructed to detect web resources that are potentially vulnerable. There is a wide variety of indicators, starting from distinguishable resource names through banners of specific products and up to specific error messages. The special search terms, commonly referred to as "Dorks"¹, combine search terms and operators that usually correlate the type of resource with its contents. Dorks are commonly exchanged between hackers in forums. Comprehensive lists of dorks are also being made available through various web sites (both public and underground). Examples include the legendary Google Hacking Database at <http://johnny.ihackstuff.com/ghdb/> and the up-to-date sites <http://www.1337day.com/webapps> and <http://www.exploit-db.com/google-dorks/>. As the latter name suggests, the site contains an exploit database demonstrating how dorks and exploits go hand in hand.

¹ <http://www.danscourses.com/Network-Security+/search-engine-hacking-471.html>

GOOGLE
HACKING-DATABASE
Welcome to the google hacking database

We call them 'googledorks': Inept or foolish people as revealed by Google. Whatever you call these fools, you've found the center of the Google Hacking Universe!

Search Google Dorks

Category: Free text search:

Latest Google Hacking Entries

Date	Title	Category
2011-07-26	inurl:server-info intitle:"Server Information...	Files containing juicy info
2011-07-26	inurl: "9000" PacketVideo corporation	Various Online Devices

Figure 1: Banner from the Google Hacking Database

EXPLOIT DATABASE

The Exploit Database

The Exploit Database (EDB) - an ultimate archive of exploits and vulnerable software. A great resource for penetration testers, vulnerability researchers, and security addicts alike. Our aim is to collect exploits from submittals and mailing lists and concentrate them in one, easy to navigate database.

Figure 2: Banners from the Exploit Database

Some resources classify dorks according to platform or usage as can be seen from the screenshot below:



Figure 3: Searching dorks by class

An attacker armed with a browser and a dork can start listing potential attack targets. By using search engine results an attacker not only lists vulnerable servers but also gets a pretty accurate idea as to which resources within that server are potentially vulnerable.

For example, the following query returns results of online shopping sites containing the Oscommerce application.

"Powered By Oscommerce" 'catalog'

21,000,000 results

[osCommerce - LeBaron Bonney Company - Antique Auto Upholstery](#)
Use keywords or item number to find the product you are looking for. Copyright © 2005
LeBaron Bonney Company **Powered by osCommerce**
[redacted] - [Cached](#)

[osCommerce. Open Source Online Shop E-Commerce Solutions](#)
Everything you need to get started in selling physical and digital goods over the internet,
from the **Catalog** frontend that is presented to your customers, to the ...
[oscommerce.com](#) - [Cached](#)

[You can now control your allergy - Allergy Control Products](#)
... not responsible for pricing errors, and may not be able to honor orders for items where
the pricing is incorrect. Copyright © 2003 AllergiesPlus.com **Powered by osCommerce**
[redacted] [Cached](#)

[Possible to Remove "Powered by Oscommerce" Link - osCommerce ...](#)
... Its possible to remove "Powered by Oscommerce ... The separate text "Powered by
osCommerce", or similar text, displayed on both Administration Tool and **Catalog**
modules ...
[forums.oscommerce.com/...remove-powered-by-oscommerce-link](#) - [Cached](#)

[osCommerce - \[redacted\] Home](#)
This product was added to our **catalog** on Wednesday 19 October, 2005. ... Template
Copyright © 2004 osCommerce-templates.co.uk · **Powered by OSCommerce**
[redacted] [Cached](#)

[redacted]
... **Catalog: My Account | Cart Contents | Checkout ... Copyright © 2011 Powered by
osCommerce**
[redacted] - [Cached](#)

[redacted] - [Master Distributor ...](#)
Call (785) 392- [redacted] to speak with a Sales ... **Catalog Download ... Copyright © 2004-2011
G.L. Huyett Powered by osCommerce**
[redacted] - [Cached](#)

[osCommerce](#)
Home » **Catalog: My Account | Cart Contents | Checkout ... Welcome Guest! Would you
like to log yourself in? ... powered by osCommerce**
[redacted] - [Cached](#)

Figure 4: results returned from a dork search

The following screenshot returns results of a dork search for FTP configuration results

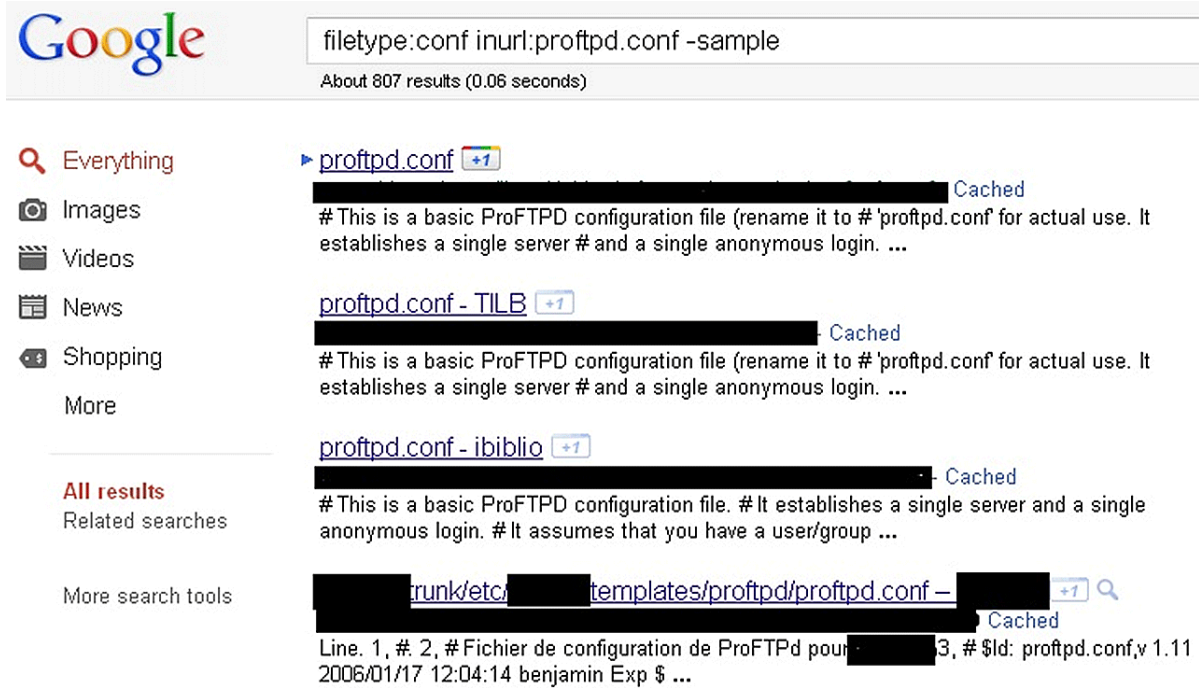


Figure 5: results returned from a dork search

Automating the Usage of Dorks

Tools to automate the use of dorks have been created over the years by attacker groups. Some of them are desktop tools and some are accessible as an online service. Some automate just the collection of targets and others automate the construction of exploit vector and the attack itself.

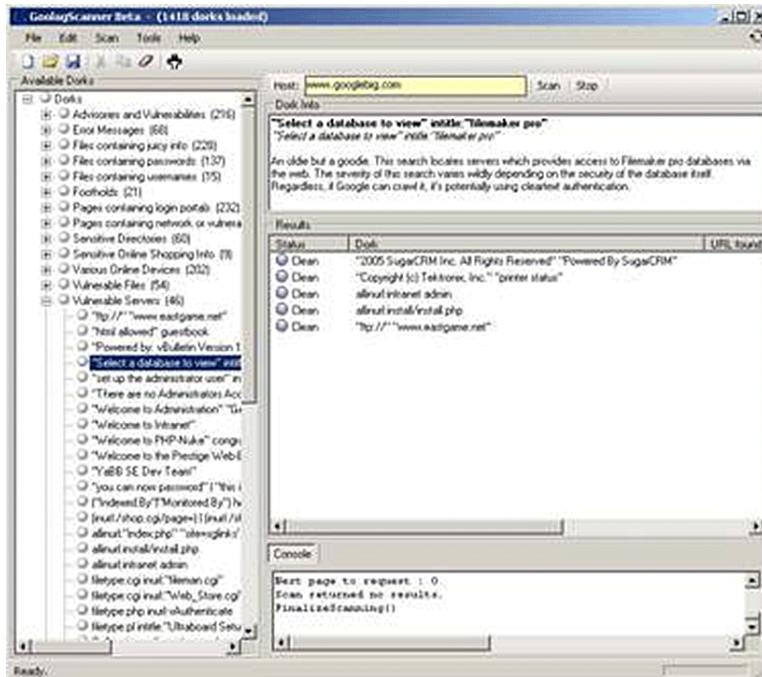


Figure 6: Desktop tool for automated Google Hacking



Figure 7: Online service for automated search and attack campaigns

In view of this threat, most search engines have implemented anti-automation measures that rely (mainly) on the following attributes:

- › Number of search queries from a single source (IP / session)
- › Frequency of queries from a single source
- › Massive retrieval of results for a single query

The anti-automation measures taken by search engine operators forced attackers to look for new alternatives for search engine hacking automation. They found it in the form of botnet based search engine mining. By harnessing the power of botnets, attackers launch distributed coordinated search campaigns that evade the standard anti-automation mechanisms. The inherent distributed nature of the attack helps avoid the single source issue. The use of special search operators that artificially split the search space (e.g. by country or by partial domain), overcomes the limitation enforced by search engines over the number of results that can be retrieved per query. In addition, the attacker creates yet another layer of indirection through the use of “search proxies”. This extra layer makes it even harder to identify the true source of the attack and the whereabouts of the attacker.

In the following section we will show evidence of these techniques as seen in the wild.

A Typical Dork-Search Attack

We have observed a specific botnet attack on a popular search engine during May-June 2011. The attacker used dorks that match vulnerable web applications and search operators that were tailored to the specific search engine. For each unique search query, the botnet examined dozens and even hundreds of returned results using paging parameters in the query.

The volume of attack traffic was huge: nearly 550,000 queries (up to 81,000 daily queries, and 22,000 daily queries on average) were requested during the observation period. It is clear that the attacker took advantage of the bandwidth available to the dozens of controlled hosts in the botnet to seek and examine vulnerable applications.

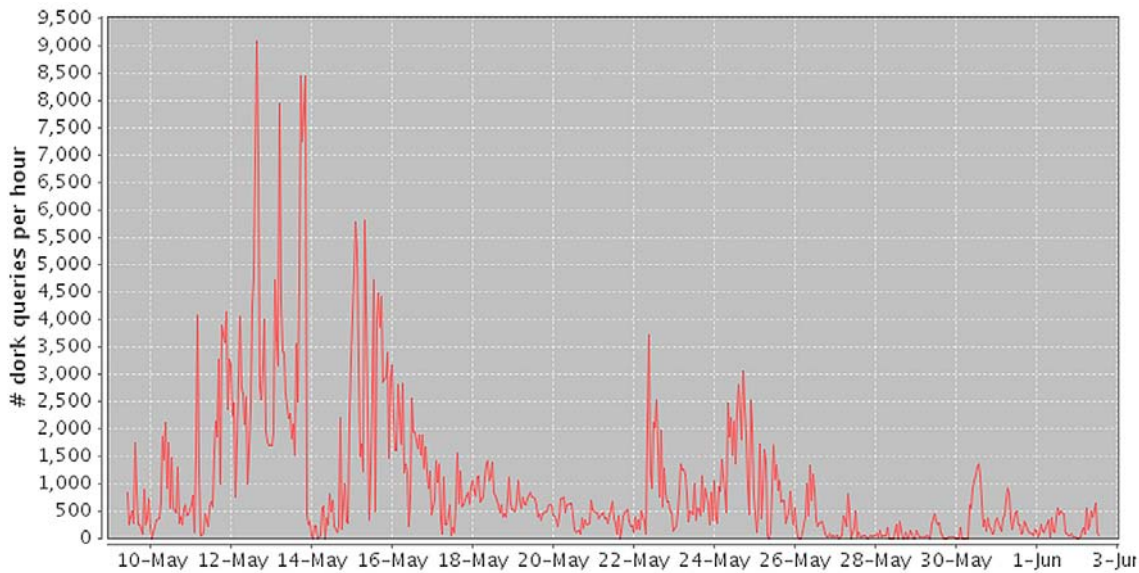


Figure 8: dork queries per hour

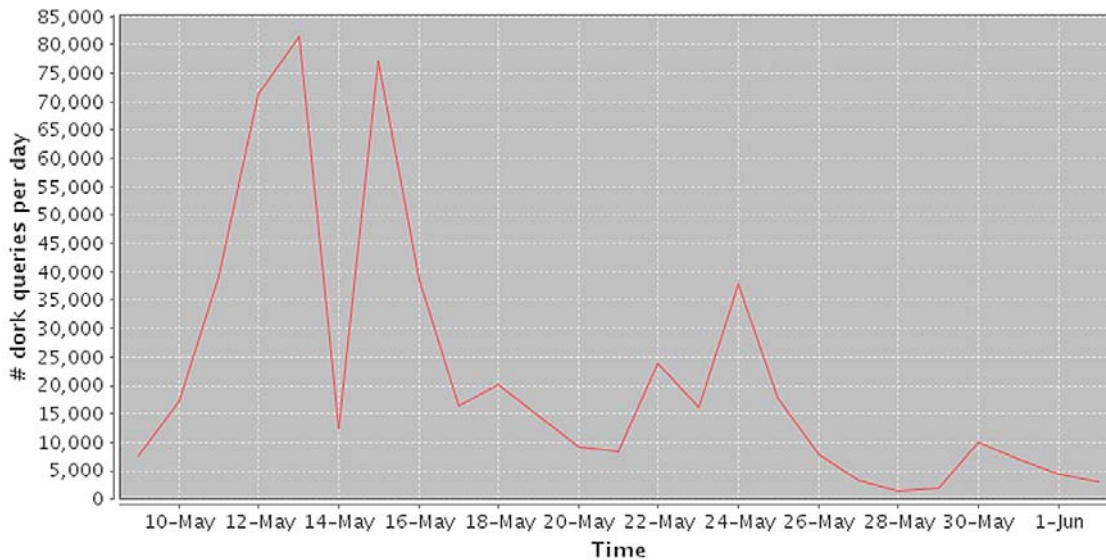


Figure 9: dork queries per day

Search Engine Dorks

Most of the Dorks used in the observed attack were related to Content Management Systems and e-commerce applications. Content Management Systems manage the work flow of users in a collaborative environment and enable a large number of people to contribute to a site and to share stored data (for example, an eCommerce system or a forum for users of a game to share playing tips). These systems are naturally more open and allow external users to contribute content and even upload entire files. Thus, security vulnerabilities they contain can be easily exposed and exploited. E-commerce systems, on the other hand, manage and store financial information about their customers, and a successful attack on such a site can be immediately monetized.

Some examples of the observed dorks used in the attack are shown below. As can be seen, the search terms include various free text words that identify vulnerable applications, as well as search operators that focus the query to specific sites, domains or countries.

Search Query	Target application	Example of vulnerabilities associated with the application ²
"Powered By Oscommerce" 'catalog'	Oscommerce: online shop e-commerce solution	SQL injection vulnerability in shopping_cart.php (CVE-2006-4297)
"powered by oscommerce" shopping	Oscommerce	See above
"powered by e107" site:.ch	e107 CMS; limited to servers in Switzerland	allows remote attackers to execute arbitrary PHP code (CVE-2010-2099)
"*.php?cPath=25" ranking	Oscommerce	See above
"powered by osCommerce"	Oscommerce	See above
"powered by zen cart" payment.php	Zen Cart Ecommerce; e-commerce web site platform	Allows remote attackers to execute arbitrary SQL (CVE-2009-2254)
"powered by e107" global	e107 CMS	See above
"fpw.php" site:.ir	e107 CMS - password reset page; limited to servers in Iran	See above
Herzlich Willkommen Gast! site:.de	Oscommerce German welcome page; limited to servers in Germany	See above
"powered by e107" site:.org	e107 CMS; limited to domains with org suffix)	See above
"by BigCommerce" joomla.ze	BigCommerce e-commerce software integrated with Joomla CMS	See above
"The Appserv Open Project" site:.th	AppServe application development platform; limited to servers in Thailand.	XSS vulnerability allows remote attackers to inject arbitrary web script (CVE-2008-2398)
"Powered by e107 Forum System" site:.com	e107 CMS; limited to domains with com suffix	See above
Joomla! es Software Libre distribuido bajo licencia GNU/GPL.	Joomla CMS - Spanish version	See above
"com_rokdownloads" site:jp	Joomla CMS; limited to servers in Japan	Directory Traversal vulnerability in RokDownloads component of Joomla (CVE-2010-1056)

Table 1: Examples of observed dork queries

The additional operators (domain, language, etc.) as well as specification of the wanted page of results are used for several purposes:

- › Creating more focused result sets that allow construction of more accurate attack vectors
- › Artificially splitting the search space in a way that distributes the workload of exhaustively examining the entire result set between the bots in the net

Overall we have seen 4719 different dork variations being used in the attack (where "powered by e107" site:.ch and "powered by e107" site:.fr are variation on the same basic dork). The 30 most-used dorks were related to osCommerce e-commerce solution, and each of these variation appeared in 1,600-3,900 queries. The e107 application was the next popular attack target based on the number of observed dorks.

² For the applications that the attackers sought, these are examples of publicly disclosed vulnerabilities. However, these are not necessarily the vulnerabilities that the attackers actually tried to exploit.

Botnet Hosts

Search engine providers identify malicious attacks based on a high volume or a high frequency of queries from the same source. Yet we have witnessed how attackers bypass these detection mechanisms by employing a botnet.

During our observation period we have identified 40 different IP addresses of hosts that participate in the attacking botnet. The hosts are not all active at the same time. The attack is distributed and coordinated. Thus, different hosts handle different dorks and each host produces low rate search activity. We found that most hosts issue no more than one request every 2 minutes. However, four hosts together issue 2-4 requests per minute. This rate does not trigger the search engine’s anti-automation policy as it normally cannot be considered abusive. In addition, the requests simulate a true browser activity rather than a script by constantly changing the *user-agent* field. Consequently, the attack campaign can go on for a long time, allowing the attacker to collect a substantial amount of target resources. An example of a coordinated distributed dork search was for the dork “e107” using 99 different argument for the *site* search operator: 5 different hosts issued these queries over the entire observation period.

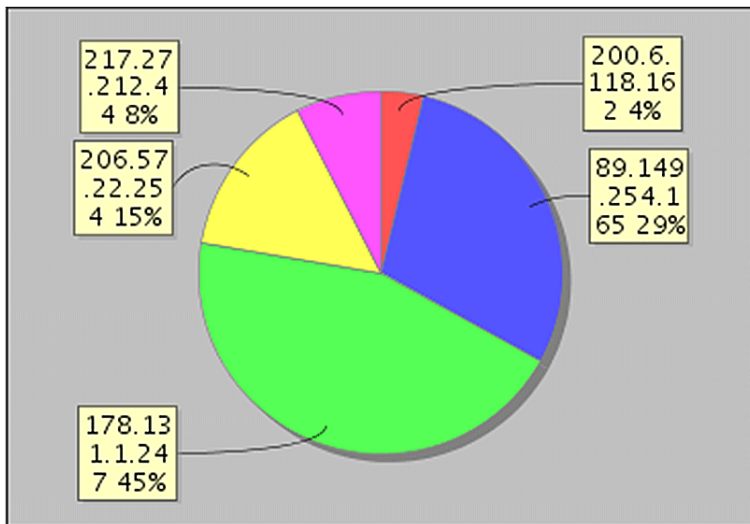


Figure 10: hosts searching for the dork “e107” with a “site” operator

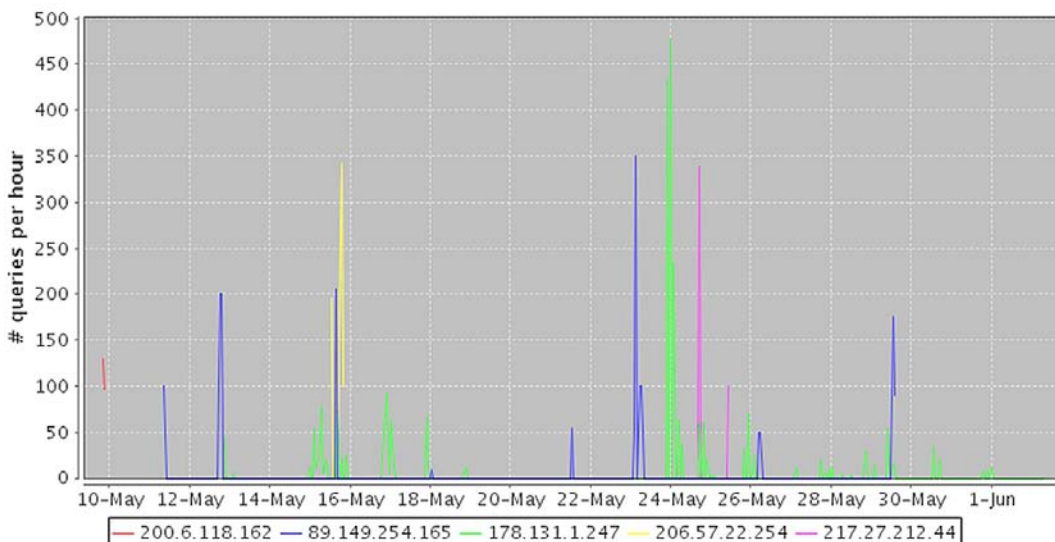


Figure 11: queries for the dork “e107” with a “site” operator

The botnet hosts are distributed all over the world. This is not surprising, since the attacker does not care about the location or ownership of the abused hosts and just needs the ability to take control of these machines and add them to her network of compromised computers. Thus, the identities of the botnet hosts give no direct indication to the identity of the hacker that uses them for malicious attacks. However, it is interesting to note that the observed botnet has a disproportionate number of servers in Iran, Hungary and Germany, and a low number of servers in the United States. Also, some of the dork queries specifically limited results to servers in Iran or Germany. This combination may be a hint to the interests of the attacker.

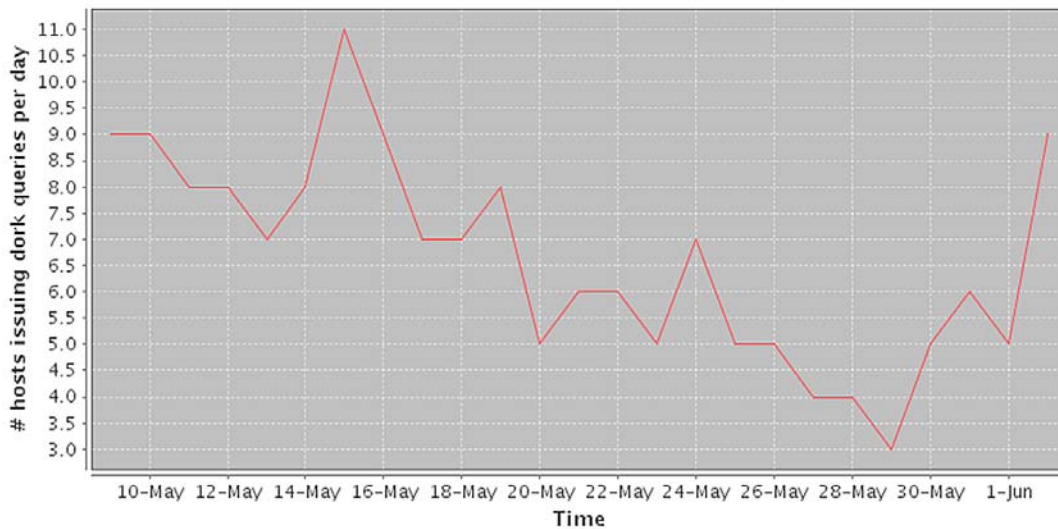


Figure 12: number of hosts issuing dork queries

Country	# dork queries	Percentage of dork queries
Islamic Republic of Iran	227554	41
Hungary	136445	25
Germany	80448	15
United States	19237	3.5
Chile	17365	3
Thailand	16717	3
Republic of Korea	11872	2
France	10906	2
Belgium	10661	2
Brazil	7559	1.5
Other	8892	2

Table 2: Countries of hosts issuing dork queries

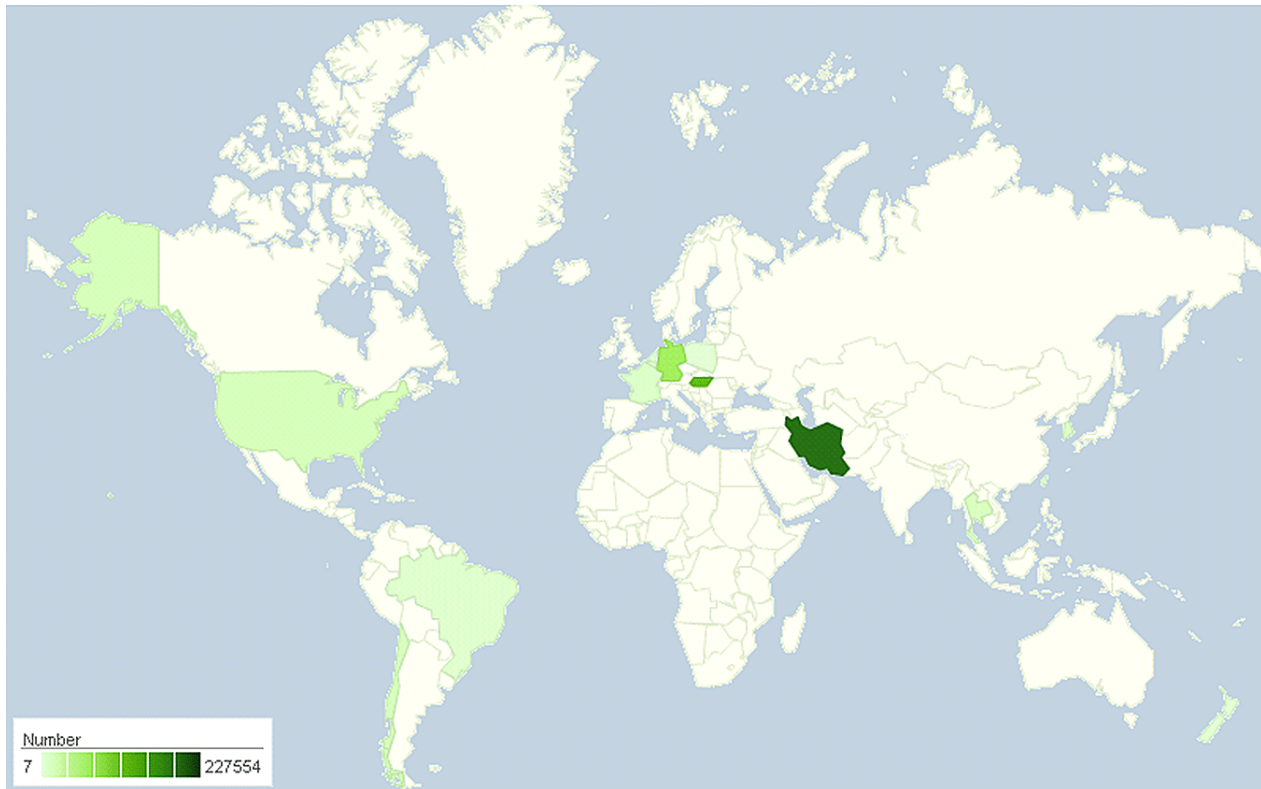


Figure13: Countries of hosts issuing dork queries

Summary and Conclusions

We have observed a high-volume mining campaign of a botnet through a popular search engine. The campaign was focused on finding resources that use specific content management frameworks that can be exploited.

While none of the components of the attack (use of botnets deployed on compromised servers, exploiting search engine using dorks) are unique, it is interesting to observe the potential for automation and flexibility of the attack. Each component may be replaced or reconfigured easily, while the attacker and tools remain hidden from targeted servers and even the abused search engine. The impact of which would be for the attacker to create a map of hackable targets on the Web.

This type of abuse should concern both the search engine providers as well as organizations. Search engines have a responsibility to prevent attackers from taking advantage of their platform to carry out their attacks. At the same time, search engines are in a unique position to identify botnets that abuse their services thus shedding light on the attackers. Organizations should protect their applications from being publicly exposed through the search engines.

Recommendations to the Search Engines

Search engine providers are expected to perform a detailed analysis of network traffic which allows the flagging of suspicious anomalies in the query traffic. Search engines typically look for low-level anomalies like high frequency or high volume of requests from a host. As this report indicates, they should start looking for unusual suspicious queries – such as those that are known to be part of public dorks-databases, or queries that look for known sensitive files (/etc files or database data files).

A list of IPs suspected of being part of a botnet and a pattern of queries from the botnet can be extracted from the suspicious traffic that is flagged by the analysis. Using these black-lists, search engines can then:

- › Apply strict anti-automation policies (e.g. using CAPTCHA) to IP addresses that are blacklisted. Google has been known³ to use CAPTCHA in recent years when a client host exhibits suspicious behavior. However, it appears that this is motivated at least partly by desire to fight Search Engine Optimization and preserve the engine's computational resources, and less by security concerns. Smaller search engines rarely resort to more sophisticated defenses than applying timeouts between queries from the same IP, which are easily circumvented by automated botnets.
- › Identify additional hosts which exhibit the same suspicious behavior pattern to update the IPs blacklist.

Search engines can use the IPs black list to issue warning to the registered owners of the IPs that their machines may have been compromised by attackers. Such proactive approach could help make the Internet safer, instead of just settling for limiting the damage caused by compromised hosts.

Recommendations to the Organization

Organizations should be aware that with the efficiency and thorough indexing of corporate information – including Web applications – the exposure of vulnerable applications is bound to occur. While attackers are mapping out these targets, it is essential that organizations prepare against exploits tailored against these vulnerabilities. This can be done by deploying runtime application layer security controls:

- › A Web Application Firewall should detect and block attempts at exploiting applications vulnerabilities.
- › Reputation-based controls could block attacks originating from known malicious sources. As our 2011 H1 Web Application Attack Report (WAAR) has shown, attacks are automated. Knowing that a request is generated by an automated process, such as coming from a known active botnet source, should be flagged as malicious.

Hacker Intelligence Initiative Overview

The Imperva Hacker Intelligence Initiative goes inside the cyber-underground and provides 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.

³ See: <http://googleonlinesecurity.blogspot.com/2007/07/reason-behind-were-sorry-message.html>