

/ 2015-02-14 / Web-Pentest

## Shell the web-Métodos de um Ninja

Por Zenodermo Javanicus



Em Nome de ALLAH, o Beneficente e o Misericordioso

O upload de shell é um dos ataques mais importantes que podemos encontrar em uma aplicação web. Uma vez que um atacante é capaz de fazer upload de seu shell, ele pode obter acesso completo ao aplicativo e também ao banco de dados. Nesse tutorial, não vou contar a parte básica do upload do shell, mas discutiremos alguns títulos de upload usados e como podemos contorná-los.

Aqui está o conteúdo que irei discutir neste tutorial.

### 1. Ignorar filtros do lado do cliente

- Desative o JavaScript no navegador.
- HTTP Live Headers para reproduzir a solicitação adulterada.
- Adulterar dados usando o complemento Firefox.
- Proxifique o aplicativo e adultere a solicitação.

### 2. Ignorando a verificação de conteúdo/tipo

- Altere o tipo de conteúdo usando modificação de solicitação.
- Verificação do lado do servidor tolo usando GIF89a; cabeçalho
- Injete sua carga útil em metadados/comentários de imagem

### 3. Ignorando a lista negra de extensões

- Experimente outras extensões executáveis.
- Ignorar filtro sensível a maiúsculas e minúsculas.
- Desvio idiota do filtro Regex.
- Adicione shell ao executável usando o arquivo .htaccess.

### 4. Ignorando a lista branca de extensões

- Injeção de Byte Nulo
- Ignorar usando extensão dupla
- Ignorar extensão inválida

### 5. Ignorando as verificações de comprimento de conteúdo e script malicioso

- Ignorar comprimento de conteúdo
- Ignorar verificações de script malicioso

### 6. Carregar Shell usando SQLi

## 7. Desvio de upload de shell usando LFI

Primeiro de tudo, espero que você saiba o básico sobre o que é um shell e como fazer upload de um shell e usá-lo, então deixando tudo isso de lado, vamos nos concentrar nos desvios de upload do shell aqui:

### 1. Ignorar filtros do lado do cliente

Em primeiro lugar, vamos deixar claro o que são filtros do lado do cliente? Os filtros do lado do cliente são os filtros que são baseado em navegador ou podemos usar javascript para validar o tipo de arquivo que estamos enviando. Se o arquivo não parece válido, então se dá um erro. Tudo bem, tudo está bem até aqui, mas o problema com tais títulos baseados em javascript dependem muito do navegador e um invasor também pode adulterar a solicitação antes de chegar ao servidor. Aqui estão alguns dos truques que um invasor pode usar ignorar esses títulos:

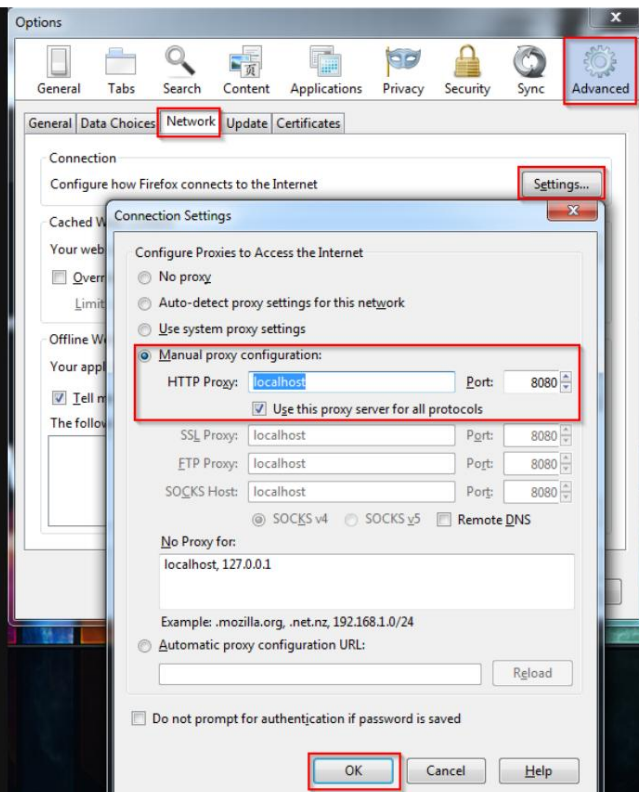
1. Desative o JavaScript no navegador.
2. HTTP Live Headers para reproduzir a solicitação adulterada.
3. Adulterar dados usando o complemento Firefox.
4. Faça proxy do aplicativo e adultere a solicitação.

Como todos os itens acima são o mesmo tipo de bypass e sabendo que pelo menos um deles funcionará para você, então eu usaremos a última abordagem neste tutorial. É realmente simples configurar o proxy BURP com seu navegador e o jogo começa. Vou mostrar os passos básicos para usar o BURP.

Passo 1: Abra seu proxy Burp e certifique-se de que ele esteja conectado à porta 8080:

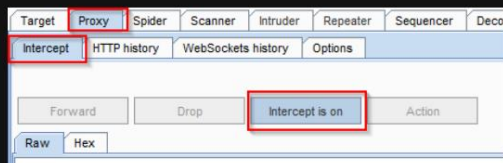


Passo 2: Configure seu Firefox para enviar o tráfego via porta Localhost 8080, vá para Ferramentas-> Opções->Avançado->Rede->Configurações faça as alterações mostradas na imagem.



Você redirecionou o tráfego via BURP com sucesso. Agora vá para Proxy-> Intercept Tab e ligue

a interceptação está desativada, para que você possa alterar o conteúdo da solicitação antes que ela chegue ao servidor:



Agora digamos que há um site onde você está tentando fazer upload do shell e mostra um erro, que você pode

carregar apenas arquivos de imagem, simplesmente renomeie seu shell.php para shell.php.jpg e carregue o arquivo. Quando você

clicar em enviar, uma solicitação sairá do BURP. Mude o nome do arquivo de volta para shell.php e, felizmente, se

não há verificação no lado do servidor, então você fará o upload do seu shell.

## 2. Ignorando a verificação de conteúdo/tipo

1. Altere o tipo de conteúdo usando modificação de solicitação.
2. Verificação do lado do servidor idiota usando GIF89a; cabeçalho
3. Injete sua carga útil em metadados/comentários de imagem

### Altere o tipo de conteúdo usando modificação de solicitação.

Muitas vezes o desenvolvedor confia na solicitação "content-Type", o script de upload verifica o conteúdo-tipo e se for tipo de imagem, apenas o arquivo será carregado. O problema aqui novamente é o tipo de conteúdo variável pode ser alterada antes de chegar ao servidor. Como você pode ver na imagem o tipo de conteúdo é "application/octet-stream", mude para "image/gif" e espere que funcione para você.

```

Connection: keep-alive
Content-Type: multipart/form-data; boundary=-----210193087832301
Content-Length: 65017

-----210193087832301
Content-Disposition: form-data; name="noplugin"

54d77bc27c398
-----210193087832301
Content-Disposition: form-data; name="token"

4d34772ca0aecf777878f134a4c8c0a2
-----210193087832301
Content-Disposition: form-data; name="import_type"

server
-----210193087832301
Content-Disposition: form-data; name="import_file"; filename="disk.php"
Content-Type: application/octet-stream

<?php
//
//

```

Verificação do lado do servidor tolo usando GIF89a; cabeçalho

Às vezes, a verificação de assinatura de conteúdo do lado do servidor pode ser enganada usando "GIF89a;" cabeçalho em seu shell.

Então aqui está um exemplo:

```

GIF89a;
<?
system($_GET['cmd']);//ou você pode inserir seu código shell completo
?>

```

Injete sua carga útil em metadados/comentários de imagem

Bem, há muitos hacks que podemos fazer com nosso arquivo de imagem, alguns deles injetam a carga útil em

o cabeçalho de metadados usando exiftools ou você pode usar uma ferramenta chamada "edjpgcom.exe". Usar comando

linha "edjpgcom.exe yourimagefilename.jpg" para adicionar comentários à sua imagem.

3. Ignorando a lista negra de extensões

Algumas vezes os desenvolvedores usam a abordagem de lista negra contra o upload do shell, o problema com

O aplicativo da lista negra é sempre o mesmo, ou seja, você sempre se esquece de bloquear algo ou um novo

bypass pode foder sua segurança. Aqui também é o mesmo, digamos que um desenvolvedor esteja filtrando arquivos php

do upload pelo servidor. Temos várias maneiras de contornar isso.

1. Experimente outras extensões executáveis.
2. Ignore o filtro sensível a maiúsculas e minúsculas.
3. Desvio idiota do filtro Regex.
4. Adicione shell ao executável usando o arquivo .htaccess.

Experimente outras extensões executáveis.

Primeiro, temos várias extensões php que o desenvolvedor pode ter esquecido para que possamos renomear

nosso arquivo para `shell.php1`

`shell.php2`

`shell.php2`

`shell.php4`

`shell.php5`

`shell.phtml`

Podemos até tentar executar o shell perl com uma extensão `.pl` ou `.cgi`.

Ignorar filtro sensível a maiúsculas e minúsculas.

Se todos estiverem bem na lista negra, ainda podemos tentar alterar maiúsculas de minúsculas para ver se o filtro diferencia maiúsculas de minúsculas ou não, em

palavras simples experimente:

`shell.PhP`

`shell.Php1`

`shell.PhP2`

`shell.pHP2`

shell.php4  
shell.php5  
shell.phpml

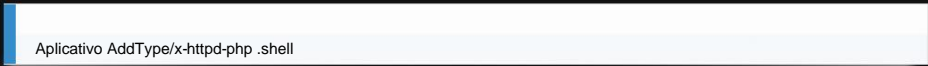
#### Desvio idiota do filtro Regex, \_\_\_\_\_

Muito poucas vezes você pode encontrar uma verificação de extensão de arquivo usando regex, tais casos podem levar a uma falha de expressão regular. Aqui o programador pode ter feito um regex incorreto que está apenas verificando a presença de ".jpg" no nome do arquivo, portanto, tais casos podem ser contornados usando extensão dupla como **shell.jpg.php** e assim por diante.

#### Adicione shell ao executável usando o arquivo .htaccess \_\_\_\_\_

Mas se tivermos azar e todas as extensões acima não funcionarem, ainda teremos uma boa chance de obter um shell no site usando o arquivo .htaccess.

Um arquivo htaccess é o arquivo de configuração no servidor Apache. Usando uma de suas configurações, podemos alterar o comportamento do tipo de arquivo. Agora vamos escolher uma extensão de arquivo que não esteja na lista negra, uma das minhas favoritas nesses casos é a extensão .shell. Então aqui está uma configuração do htaccess que você deve lidar em um Arquivo .htaccess e depois carregue na pasta e depois carregue seu shell php com o nome shell.shell e bum!! ele será executado.



Aplicativo AddType/x-httpd-php .shell

#### 4. Ignorando a lista branca de extensões \_\_\_\_\_

Em alguns casos, os desenvolvedores usaram a lista branca de extensões, ignorar essa segurança geralmente é um servidor web ou desvios baseados em idioma. É um caso em que o desenvolvedor não permite qualquer outra extensão diferente de algumas extensões da lista branca, como digamos que seja uma função de upload de imagem, então apenas jpg, jpeg, gif, png, bmp etc são permitidos. Podemos tentar os seguintes truques:

1. Injeção de Byte Nulo
2. Ignorar usando extensão dupla
3. Desvio de extensão inválido

#### Injeção de Byte Nulo \_\_\_\_\_

Nosso primeiro truque é a injeção de bytes nulos, onde algumas vezes quando inserimos um nome de arquivo como **shell.php%00.jpg** a parte após %00 foi anulada pelo idioma e o arquivo foi carregado com o nome shell.php.

#### Ignorar usando extensão dupla \_\_\_\_\_

Nesses casos, podemos usar **shell.php.jpg**, **shell.php:.jpg**, **shell.php;.jpg** às vezes pode levar a execução de shell, mas geralmente é um servidor web ou bypass baseado em sistema operacional. Então não podemos realmente culpar o programador nesses casos. Mas deixar os nomes dos arquivos inalterados é uma má prática de programação.

#### Ignorar extensão inválida . \_\_\_\_\_

Aqui está outra exploração do lado do servidor, às vezes quando usamos extensões como .test que não é reconhecido pelo sistema operacional, então a primeira extensão é usada, para que possamos tentar fazer o upload shell.php.test.

#### 5. Ignorando as verificações de comprimento de conteúdo e script malicioso \_\_\_\_\_

##### Ignorar comprimento de conteúdo

Às vezes nos deparamos com uma verificação do tamanho do conteúdo que de qualquer forma não é tão comum, mas sabemos que existe não há fim para a estupidez humana. Tendo isso em mente, existem algumas maneiras muito fáceis de contornar tal



## Postagens recentes



13 de novembro de 2018

XXECheatSheet-SecurityIdiotas



13 de novembro de 2018

Execução de diferentes  
contextos para XSS

## Inscriva-se em novas postagens

Assine nossa newsletter e enviaremos a você os e-mails das últimas postagens.

Se inscrever

yyyyyy

2021 © SecurityIdiotas. Criado e projetado por Artem Sheludko.