

Notícias de Insegurança

Como preaver-se contra ataques XSS

Kurt Seifried

XSS e `ap_escape_html()`

O problema de *Cross Site Scripting* (XSS) tem aparecido nos noticiários, então, inicio esta nova coluna com uma abordagem para esse clássico problema de segurança. Como o usuário final nada pode fazer para impedir isso, o XSS é uma forma de ataque adorada pelos mal-intencionados. Um ataque XSS dá ao agressor a possibilidade de inserir código *Javascript* hostil dentro de páginas web de um site em que os usuários confiam. O que torna esse problema particularmente ruim é que ele ocorre no próprio servidor web (servidor `httpd` do *Apache*). Mesmo que você garanta que suas aplicações baseadas em web estejam seguras contra o XSS, um agressor ainda poderia executar o ataque através do software servidor do *Apache*.

O servidor *Apache* implementa a função `ap_escape_html()` para processar as entradas e escapar devidamente os caracteres de controle `<`, `>` e `&`, utilizados com frequência em ataques XSS. Por exemplo, se um agressor puder inserir esses caracteres em uma mensagem de erro enviada para a vítima, ele ganhará acesso ao servidor.

Na maioria dos casos em que a saída é passada para uma vítima em potencial, a `ap_escape_html()` é utilizada da forma adequada para processar a saída e garantir que não passe nada hostil. Infelizmente, a equipe do *Apache* se esqueceu de utilizar essa função de correção em alguns lugares.

A atualização recente do servidor `httpd` do *Apache* inclui não apenas uma correção para uma vulnerabilidade XSS, mas também implementa a `ap_escape_html()` em vários lugares por onde os dados são passados para vítimas potenciais.

O código no **exemplo 1** está no formato universal `diff` (em que `-` significa que uma linha foi removida e `+` denota uma linha que a substituiu). O uso da `ap_escape_html()` é bem simples e não afeta o funcionamento normal do software.

Assim, a alteração mostrada no **exemplo 1** reflete uma situação clássica: um problema ainda fica rondando o código, mesmo com uma solução relativamente simples disponível. A solução simplesmente não é implementada, levando a uma vulnerabilidade de segurança.

e condições de erros inesperados. Esse princípio é conhecido como *projeto livre de falhas* (*fail safe design*). O problema é que a maioria dos softwares, incluindo os de código aberto, não são projetados para serem livres de falhas. Na realidade, os programas frequentemente falham de formas espetaculares, como os estouros de *buffer* que levam à execução de código arbitrário, corrupção de dados e assim por diante.

Se não se pode contar com um sistema livre de falhas, o que fazer?

A resposta é que simplesmente não se pode esperar que um sistema seja livre de falhas. Para os administradores de sistemas e redes, isso significa implementar múltiplas camadas de segurança; em caso de falhas, espera-se que a camada seguinte impeça o desenrolar do problema. Nesse caso, a solução mais óbvia é implementar algum tipo de *proxy* (como o servidor *proxy Squid*) que entenda os dados a serem enviados e possa filtrá-los; por exemplo, bloquear a passagem de caracteres `<` ou `>` junto à URL requisitada.

Além disso, o administrador também pode usar o *proxy* com os dados da saída e verificá-los em busca de possíveis conteúdos hostis; entretanto, o fato de o site provavelmente também incluir códigos *Javascript* legítimos

Livre de falhas

Um dos princípios fundamentais de projeto de sistemas é que eles devem lidar de forma “educada” com as falhas

Item de segurança do mês

Saiu a versão 3.1 da ferramenta de teste de penetração *Metasploit* [2]. A maior mudança é a nova interface gráfica. Preciso dizer que me sinto incomodado com a facilidade de uso das ferramentas para ataques de rede, mas a realidade é que elas já estão tão comuns e fáceis de utilizar que, adicionar uma interface gráfica não muda muito a questão. Afinal, existem tantos ataques e usuários hostis na Internet que, quem não mantiver seus sistemas atualizados e seguros será atingido mais cedo ou mais tarde.

Exemplo 1: Compilação

```
01 ~/src/olamundo $ make
02 Scanning dependencies of target olamundo
03 [100%] Building C object CMakeFiles/ola.dir/main.o
04 Linking C executable olamundo
05 [100%] Built target olamundo
06 ~/src/olamundo $ ./olamundo
07 Ola mundo!
```

cria ainda mais dificuldade para a identificação de scripts hostis.

O usuário não pode esperar que todos os sites tomem medidas para evitar ataques XSS (lembre-se de que muitos sites nem se incomodam em seguir padrões W3C!). Infelizmente, os usuários têm poucas opções para se protegerem contra ataques XSS, e hoje em dia o conselho-padrão de não clicar em links de origem desconhecida não tem mais o menor sentido.

NoScript

Para os usuários do Firefox, a melhor opção é utilizar a extensão NoScript [1]. Embora o bloqueio de Javascript fornecido pelo NoScript não possa fazer muito (em alguns casos, os usuários já colocaram o site na lista de permissão da extensão), o NoScript fornece um bloqueio rudimentar de ataques via scripting XSS, passados pelas requisições de URL, alertando

os usuários e dando-lhes a opção de recarregar a página.

Patch

Para um desenvolvedor, é inaceitável querer jogar fora seu código e recomençar do zero. O melhor caminho é aplicar patches da melhor maneira possível. Porém, uma vez que o problema for solucionado, é interessante testar com o Apache, procurando a mesma condição de erro em outras partes do código.

Conclusão

Neste artigo, mostrei exatamente por que eu não sou um programador. Eu não sou nem inteligente nem meticuloso o suficiente para escrever um código seguro. Um passo de correção perdido, um pequeno erro de lógica ou mesmo um engano na digitação podem levar a resultados potencialmente desastrosos. Mesmo administradores que façam tudo da

forma correta podem enfrentar problemas de segurança e, por motivos da interconectividade natural das redes de computadores (ou mesmo a rede que nos conecta a tudo — incluindo sua torradeira, um dia), algum problema de segurança que porventura apareça também pode se tornar o seu problema de segurança. ■

Mais Informações

[1] NoScript add-on:
<http://noscript.net/>

[2] Metasploit:
<http://metasploit.com/>

Sobre o autor

Kurt Seifried é um Consultor de Segurança da Informação, especializado em Linux e redes desde 1996. Ele é casado e tem quatro gatos mas nenhum peixe (porque os gatos têm mais fome que medo de água). Ele freqüentemente se pergunta como essa tecnologia funciona em larga escala, mas freqüentemente falha em escalas menores.

Linux decola em 2008*

*Fonte: The Economist dez/2007

Treine na IMPACTA e torne-se um profissional Linux

O Linux vem crescendo exponencialmente, tendo lugar no mundo Linux em...
exaustivamente a sua aplicabilidade. Quanto aos desktops, creio que este será o ano do Linux, pois existem facilidades



Preparatório para a Certificação LPI

Linux LPI 101 - Fundamentos | Linux LPI 101 - Implementação e Adm.

Linux LPI 102 - Implementação de Infra-estrutura de Redes

Linux LPI 102 - Gerenciamento e Manutenção

Treinamentos avançados

Linux Shell Script | LDAP | Apache | Samba | Firewall

Tel: (11) 3254-2200

Av. Paulista, 1009 - 9º andar | www.impacta.com.br



O conhecimento sem limites