Laboratório - Calculando Hashes

Objetivos

Parte 1: Hash de um arquivo de texto com OpenSSL

Parte 2: Verificando Hashhes

Histórico/Cenário

Funções de hash são algoritmos matemáticos projetados para tomar dados como entrada e gerar uma cadeia de caracteres única de tamanho fixo, também conhecido como o hash. Projetado para ser rápido, as funções de hash são muito difíceis de reverter; é muito difícil recuperar os dados que criaram qualquer hash determinado, com base apenas no hash. Outra propriedade importante da função hash é que mesmo a menor alteração feita para os dados de entrada produz um hash completamente diferente.

Embora o OpenSSL possa ser usado para gerar e comparar hashes, outras ferramentas estão disponíveis. Algumas dessas ferramentas também estão incluídas neste laboratório.

Recursos necessários

- Máquina virtual CyberOps Workstation

Instruções

Parte 1: Hash de um arquivo de texto com OpenSSL

OpenSSL pode ser usado como uma ferramenta autônoma para hash. Para criar um hash de um arquivo de texto, siga as etapas abaixo:

a. Na máquina virtual CyberOps Workstation, abra uma janela de terminal.

b. Como o arquivo de texto para hash está no diretório /home/analyst/lab.support.files/, mude para esse diretório:

```
[analyst@secOps ~]$ cd /home/analyst/lab.support.files/
```

c. Digite o comando abaixo para listar o conteúdo do arquivo de texto letter_to_grandma.txt na tela:

```
[analyst@secOps lab.support.files]$ cat letter_to_grandma.txt
Oi vovó,
Estou escrevendo esta carta para agradecer pelos biscoitos de chocolate que você me
enviou. Comprei-os esta manhã e já comi metade da caixa! Eles são absolutamente
deliciosos!


Desejo-lhe tudo de bom. Amor,
Seu neto comedor de biscoitos.
```

d. Na janela do terminal, execute o comando abaixo para hash do arquivo de texto. O comando usará SHA-2-256 como o algoritmo de hash para gerar um hash do arquivo de texto. O hash será exibido na tela depois que o OpenSSL o calculou.

```
[analyst @secOps lab.support.files] $ openssl sha256 letter_to_grandma.txt
SHA256 (letter_to_grandma.txt) =
deff9c9bbece44866796ff6cf21f2612fbb77aa1b2515a900bafb29be118080b
```

Observe o formato da saída. O OpenSSL exibe o algoritmo de hash usado, SHA-256, seguido pelo nome do arquivo usado como dados de entrada. O hash SHA-256 em si é exibido após o sinal de igual ('=').

e. Funções de hash são úteis para verificar a integridade dos dados, independentemente de se tratar de uma imagem, uma música ou um arquivo de texto simples. A menor alteração resulta em um hash completamente diferente. Os hashes podem ser calculados antes e depois da transmissão, e depois comparados. Se os hashes não corresponderem, os dados foram modificados durante a transmissão.

Vamos modificar o arquivo de texto letter_to_grandma.txt e recalcular o hash MD5. Execute o comando abaixo para abrir o **nano,** um editor de texto de linha de comando.

```
[analyst@secOps lab.support.files]$ nano letter_to_grandma.txt
```

Usando **nano**, altere a primeira frase de **'Hi Grandma'** para **'Hi Grandpa'**. Observe que estamos mudando apenas um caractere, **'m'** para **'p'**. Após a alteração ter sido feita, pressione as teclas **<CONTROL+X** para salvar o arquivo modificado. Pressione **'Y'** para confirmar o nome e salvar o arquivo. Pressione a <Enter> tecla e você sairá de nano para continuar na próxima etapa.

f. Agora que o arquivo foi modificado e salvo, execute o mesmo comando novamente para gerar um hash SHA-2-256 do arquivo.

```
[analyst@secOps lab.support.files]$ openssl sha256 letter_to_grandma.txt
SHA256 (letter_to_grandma.txt) =
43302c4500b7c4b8e574ba27a59d83267812493c029fd054c9242f3ac73100bc
```

O novo hash é diferente do hash calculado no item (d)? Quão diferente?

g. Um algoritmo de hash com comprimento de bits mais longo, como SHA-2-512, também pode ser usado. Para gerar um hash SHA-2-512 do arquivo letter_to_grandma.txt, use o comando abaixo:

```
[analyst @secOps lab.support.files] $ openssl sha512 letter_to_grandma.txt
SHA512 (letter_to_grandma.txt) =
7c35db79a06aa30ae0f6de33f2322fd419560ee9af9cedeb6e251f2f1c4e99e0bbe5d2fc32ce5
01468891150e3be7e288e3e568450812980c9f8e3a31d3
[analyst@secOps lab.support.files]$
```

h. Use **sha256sum** e **sha512sum** para Generatesha-2-256 e SHA-2-512 hash do arquivo letter_to_grandma.txt:

```
[analyst @secOps lab.support.files] $ sha256sum letter_to_grandma.txt
43302c4500b7c4b8e574ba27a59d83267812493c029fd054c9242f3ac73100bc
letter_to_grandma.txt


[analyst @secOps lab.support.files] $ sha512sum letter_to_grandma.txt
7c35db79a06aa30ae0f6de33f2322fd419560ee9af9cedeb6e251f2f4e99e0bbe5d2fc32ce501
468891150e3be7e288e3e568450812980c9f8288e3a1d3 letter_to_grandma.txt
```

Os hashes gerados com **sha256sum** e **sha512sum** correspondem aos hashes gerados nos itens (f) e (g), respectivamente? Explique.

Nota: SHA-2 é o padrão recomendado para hash. Embora o SHA-2 ainda não tenha sido efetivamente comprometido, os computadores estão se tornando cada vez mais poderosos. Espera-se que esta evolução natural em breve torne possível que os atacantes quebrem SHA-2.

SHA-3 é o mais novo algoritmo de hash e, eventualmente, ser o substituto para a família SHA-2 de hashes.

Observação: a VM CyberOps Workstation inclui suporte somente para SHA-2-224, SHA-2-256 e SHA-2-512 (**sha224sum, sha256sum**e **sha512sum**, respectivamente).

Parte 2: Verificando Hashhes

Como mencionado anteriormente, um uso comum para hashes é verificar a integridade do arquivo. Siga as etapas abaixo para usar hashes SHA-2-256 para verificar a integridade de sample.img, um arquivo baixado da Internet.

a. Junto com sample.img, Sample.img_sha256.sig também foi baixado. Sample.img_sha256.sig é um arquivo contendo o hash SHA-2-256 que foi computado pelo site. Primeiro, use o comando cat para exibir o conteúdo do arquivo Sample.img_sha256.sig:

```
[analyst@secOps lab.support.files]$ cat sample.img_SHA256.sig
c56c4724c26eb0157963c0d62b76422116be31804a39c82fd44ddf0ca5013e6a
```

b. Use SHA256Sum para calcular o hash SHA-2-256 do arquivo.img de exemplo:

```
[analyst @secOps lab.support.files] $ sha256sum sample.img
c56c4724c26eb0157963c0d62b76422116be31804a39c82fd44ddf0ca5013e6a sample.img
```

O sample.img foi baixado sem erros? Explique.

Observação: Embora a comparação de hashes seja um método relativamente robusto para detectar erros de transmissão, há maneiras melhores de garantir que o arquivo não tenha sido adulterado. Ferramentas, como o **gpg**, fornecem um método muito melhor para garantir que o arquivo baixado não foi modificado por terceiros e é de fato o arquivo que o editor pretendia publicar.