Práctica Seguridade Informática
Colisión Hash MD5

ESCENARIO

Máquina virtual ou física:
RAM ≤ 2048MB CPU ≤ 2 PAE/NX habilitado
ISO/CD/DVD/USB: Live amd64 - Calquera distribución baseada en Debian
REDE: DHCP (NAT)
BIOS: Permite arranque dispositivo extraíble: CD/DVD, USB

LIMITACIÓN DE RESPONSABILIDADE O autor do presente documento declina calquera responsabilidade asociada ao uso incorrecto e/ou malicioso que puidese realizarse coa información exposta no mesmo. Por tanto, non se fai responsable en ningún caso, nin pode ser considerado legalmente responsable en ningún caso, das consecuencias que poidan derivarse da información contida nel ou que esté enlazada dende ou hacia el, incluíndo os posibles erros e información incorrecta existentes, información difamatoria, así como das consecuencias que se poidan derivar sobre a súa aplicación en sistemas de información reais e/ou virtuais. Este documento foi xerado para uso didáctico e debe ser empregado en contornas privadas e virtuais controladas co permiso correspondente do administrador desas contornas.

Práctica

Arrancar coa distro Live amd64 baseada en Debian

1. Abrir un terminal e executar:

   $ git clone https://github.com/cr-marcstevens/hashclash #Descargar o repositorio hashclash de git de Marc Stevens
   $ cd hashclash #Acceder ao directorio hashclash que contén o repositorio descargado.
   $ sudo apt update #Actualizar o listado de paquetes dos repositorios (/etc/apt/sources.list, /etc/apt/sources.list.d/). E mediante o comando sudo permitir esta acción a través da configuración deste comando (/etc/sudoers, visudo)
   $ sudo apt -y install autoconf automake libtool zlib1g-dev libbz2-dev #Instalar os paquetes autoconf, automake, libtool, zlib1g-dev e libbz2-dev. Co parámetro -y automaticamente asumimos yes a calquera pregunta que ocorra na instalación do paquete. E mediante o comando sudo permitir a instalación a través da configuración deste comando (/etc/sudoers, visudo)
   $ ./build.sh #Executar o script build.sh
   $ cd bin #Acceder ao directorio bin
   $ ./md5_fastcoll -o out1.bin out2.bin #Executar o script md5_fastcoll xerando dous arquivos binarios out1.bin e out2.bin, os cales son distintos pero posúen o mesmo hash md5, é dicir, executamos o comando que crea a colisión hash nos ficheiros out1.bin e out2.bin
   $ file out1.bin out2.bin #Determinar o tipo de ficheiros que son out1.bin e out2.bin; neste caso ficheiros binarios (data).
   $ md5sum out1.bin out2.bin #Crear hash MD5 dos ficheiros out1.bin e out2.bin
   $ diff out1.bin out2.bin #Comprobar as diferencias entre os 2 ficheiros (out1.bin e out2.bin)
   $ xxd out1.bin | tee 1.bin #Visualizar en hexadecimal o contido do ficheiro out1.bin e ademais mediante o comando tee crear con esa saída o ficheiro 1.bin
   $ xxd out2.bin | tee 2.bin #Visualizar en hexadecimal o contido do ficheiro out2.bin e ademais mediante o comando tee crear con esa saída o ficheiro 2.bin
   $ diff 1.bin 2.bin #Comprobar as diferencias entre os 2 ficheiros (1.bin e 2 .bin)
2. Que acontece no comando executado **md5sum**? Por que?
3. Que acontece nos comandos executados **xxd**? Por que?
4. Que acontece nos comandos executados **diff**? Por que?

Ricardo Feijoo Costa