



ALGUIEN EN LA FISI

CUANDO LA SIMPLE EXISTENCIA TOMA FORMA Y ESPACIO


Página principal	SQL Injection	Canal de Youtube	Scripts	Contacto
----------------------------------	-------------------------------	----------------------------------	-------------------------	--------------------------


Lo más visto de la semana


Hacking Nucom R5000UNv2

Hacking el router ZTE ZXV10 W300 v2.1 (El Portaretratos)


Lista enlazada simple en C/C++

Hacking WiFi - ZTE ZXHN H108N

Hacking WiFi - ZTE ZXHN H108N - Parte II

Ordenamiento por mezcla - MergeSort

Subvector de suma máxima

Ordenamiento por montículos - HeapSort

Archivo del blog

► 2014 (8)

▼ 2013 (20)

▼ diciembre (3)

Buscar subdominios con Google - Script en python

Hacking WiFi - ZTE ZXHN H108N

Compilación cruzada con Buildroot

► octubre (2)

► agosto (2)

► junio (1)

► mayo (2)

► abril (3)

► marzo (4)

► enero (3)

► 2012 (30)

sábado, 7 de diciembre de 2013

Compilación cruzada con Buildroot

En este post quiero mostrar el uso de una herramienta de compilación cruzada llamada **"Buildroot"** para poder escribir programas en C que se ejecuten sistemas Linux embebidos, como los que hay por ejemplo en routers, camaras ip, DVRs, etc.

Muchos de estos dispositivos permiten conectar por SSH o Telnet y obtener una shell, otros tienen vulnerabilidades de inyección de comandos (como este router o este otro) y a otros se puede acceder por puerto serial con algo de hardware hacking. De una u otra manera, lo primero que debes conseguir es una linea de comandos en el dispositivo.

¿Qué es Buildroot?

Traducción monse: *Buildroot es un conjunto de "Makefiles" y parches que facilitan la creación de sistemas Linux embebidos completos. Buildroot puede generar las herramientas de compilación cruzada, un sistema de ficheros raíz, una imagen del kernel y una imagen de bootloader.*

Más info: <http://buildroot.uclibc.org/>

Nosotros lo utilizaremos para crear el **"Toolchain"** de compilación cruzada y con él poder compilar nuestros programas para la arquitectura del cacharro en cuestión.

El primer paso será descargar Buildroot:

```
$ curl http://buildroot.uclibc.org/downloads/buildroot-2013.11.tar.gz -o buildroot-2013.11.tar.gz
```

Luego lo desempaquetamos:

```
$ tar xvzf buildroot-2013.11.tar.gz
```

Y por lo pronto ya está. Antes de continuar asegurate de tener instaladas todas las dependencias (o por lo menos las de la sección 2.1.1)

<http://buildroot.uclibc.org/downloads/manual/manual.html#requirement>

Obteniendo información

Antes de continuar con la configuración de Buildroot debemos obtener algo de información sobre la arquitectura objetivo. Para ello podemos hacer lo siguiente:

En la terminal del dispositivo, ejecutamos:

```
$ cat /proc/version
```

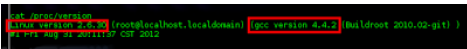



Fig. 1 - Obtener versión de Linux y GCC

Algo que ayuda bastante es analizar algún binario que venga con el sistema embebido. Podemos escoger, por ejemplo, cualquiera de los que esté en **"bin"** y descargarlo a nuestro PC usando netcat o por ftp (o de cualquier otra forma). Una vez tengas el binario, analízalo con el comando **"readelf"**.

```
$ readelf -h [BINARIO]
```


Blogroll

Arthusu BLOG

[Parte 5] CURL en PHP

Hace 6 horas


Comunidad PeruCrack

Entrevista a Marco Morales CCIE R&S, Security,

SP y CCDE


Hace 6 horas

Un informático en el lado del mal

Heartbleed puede desangrarte vivo. Tómalo en serio.

Hace 16 horas


Security By Default

¿Te gusta

SecurityByDefault?


Hace 17 horas

ReYDeS Blog

Crear la Imagen Forense desde una Unidad utilizando dd


Hace 20 horas

Hispasec @unaaldia

Mozilla publica Firefox 29 y corrige 15 nuevas vulnerabilidades


Hace 1 día

hackplayers

Denegar tráfico con rutas nulas


Hace 2 días

EnHacke

FakeInst, el troyano que te roba los datos si quieres ver porno


Hace 6 días

Blog de Omar

ANÁLISIS CAPTCHA: INYECCIÓN DE CÓDIGO SQL-PARTE II

Hace 1 semana

Juan Oliva

Publicado mi Paper :

► 2011 (72)

► 2010 (53)

► 2009 (9)

Visitas

183,230



Fig. 2 - Analizando binario.

Configurando Buildroot

Ahora toca setear los parametros que hemos obtenido en la configuración de Buildroot. Para ello primero ubicate en la carpeta de Buildroot:

```
$ cd buildroot-2013.11
```

Y ejecuta:

```
$ make menuconfig
```

Esto iniciará el asistente de configuración de Buildroot.

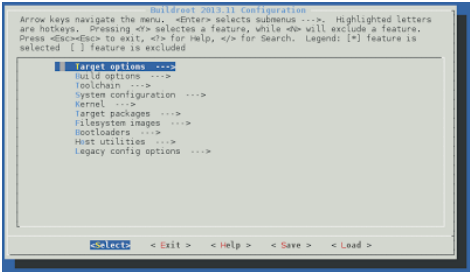


Fig. 3 - Configuración de Buildroot

En "Target options" > "Target Architecture" seleccionamos el tipo de arquitectura. En mi caso "MIPS (big endian)".

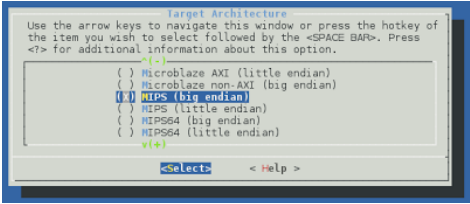


Fig. 4 - Configurar arquitectura

Luego, si es necesario, ajustamos las opciones de la variante de MIPS. En mi caso no es necesario.

Seguridad en Implementaciones de Voz Sobre IP

Hace 2 semanas

El Cerebro de Dedalo

Mis próximas fechas en las canchas

Hace 2 semanas

Blackploit [PenTest]



Vulnerabilidad CRÍTICA en OpenSSL (CVE-2014-0160) #Heartbleed [Exploit + Demo + Mass Scanner]

Hace 3 semanas

(In)seguridad



¿Cómo se llama la película?

Hace 3 semanas

Carluys Blog



Accediendo a la configuración de un Router Wifi desde Android

Hace 5 semanas

INSEGUROS



OSSIM.

Pentesting continuo como si estuvieses en primero

Hace 1 mes

ConexionInversa



WINDBG y KD

(INTRODUCCION FORENSE)

Hace 1 mes

DaboBlog



Comentario en MorterueloCON, info sobre herramientas (y demo con parte de mi intervención). por #BOFHers Herramientas para Google Chrome | SoydelBierzo

Hace 2 meses

NoxSoftNoxSoft



¿Censura a NoxSoft.Net?

Hace 2 meses

Pentester.es



Vulnerabilidad en el iBoot de iPhone 4S

Hace 2 meses

Open-Sec -Seguridad y Contraseguridad



Informática EXTRAYENDO CONTRASEÑAS DE LA RAM CON MIMIKATZ 2.0 (ALFA)

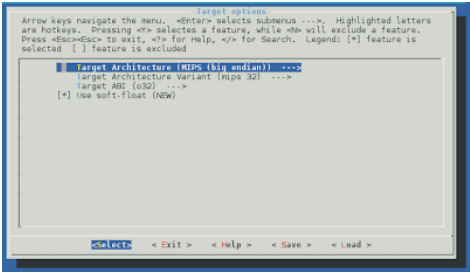


Fig. 5 - Variante (mips 32) ABI (o32)

Ahora en **"Toolchain"** configuramos la versión del kernel Linux. En mi caso **"2.6.30"**. Si no está en la lista, podemos asignar la versión manualmente.

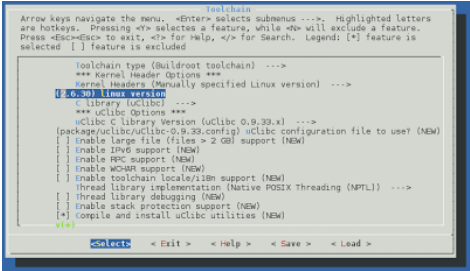


Fig. 6 - Configurar versión de Linux

En **"Toolchain"** también podemos configurar la versión del compilador GCC. En mi caso **"4.4.x"**.

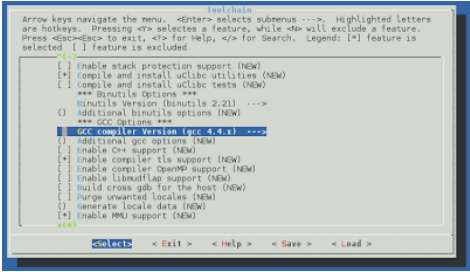


Fig. 7 - Configurar versión de GCC.

Si nuestro código requiere librerías adicionales, en **"Target packages"** > **"Libraries"** podremos buscarlas por categorías y seleccionarl

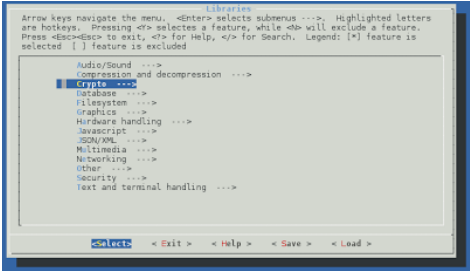


Fig. 8 - Configuración de librerías

Finalmente guardamos la configuración con la opción **"Save"**. Es importante guardar la configuración en el mismo directorio y con el nombre **".config"**. En otro caso no reconocerá la configuración.

ibapp/admin/pans

Hace 3 meses

Walter Cuestas

Agramonte, aka, El Enano

Memorias del 2013 y algo más...

Hace 4 meses

ProxyTemper Secure

Phishing Banco Chileno BCI.cl

Hace 8 meses

Los Caballeros

Conferencia UAT - USB Attack Toolkit - GuadalajaraCON

Hace 1 año

El blog de Thor

Cifrando un ejecutable en C,

problemas (2ª parte)

Hace 2 años

Todo por el Vicio

Hackeando Routers con aplicación de Facebook

Hace 2 años

Un Derrame de Ideas Demenciales

De vuelta al Blog y algo del Owasp Latam Tour 2011 - Perú

Hace 2 años

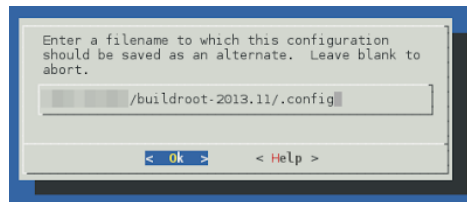


Fig. 9 - Guardar configuración

Y ya podemos cerrar el asistente de configuración.

Construyendo el Toolchain

Esto es muy fácil. Simplemente nos ubicamos en la carpeta de buildroot y ejecutamos **"make"**.

```
$ make
```

Buildroot leerá la configuración e iniciará la descarga de todos los paquetes requeridos para finalmente compilarlos y organizarlos. Esto va a demorar un buen rato, así que ve a comer algo, duerme, comunícate con otros mortales... vamos, deja la maquina trabajando y busca otra cosa que hacer.

Cuando termine, tendremos los binarios para compilación cruzada en el directorio **"output/host/usr/bin/"**. Para acceder fácilmente a ellos configuramos las variables de entorno:

```
export TOOLCHAIN_MIPS=[RUTA AL DIRECTORIO BUILDROOT]
export PATH=$TOOLCHAIN_MIPS/output/host/usr/bin:$PATH
export AR=$TOOLCHAIN_MIPS/output/host/usr/bin/mips-linux-ar
export AS=$TOOLCHAIN_MIPS/output/host/usr/bin/mips-linux-as
export CC=$TOOLCHAIN_MIPS/output/host/usr/bin/mips-linux-gcc
export CPP=$TOOLCHAIN_MIPS/output/host/usr/bin/mips-linux-cpp
export CXX=$TOOLCHAIN_MIPS/output/host/usr/bin/mips-linux-g++
export LD=$TOOLCHAIN_MIPS/output/host/usr/bin/mips-linux-ld
export GCC=$TOOLCHAIN_MIPS/output/host/usr/bin/mips-linux-gcc
export NM=$TOOLCHAIN_MIPS/output/host/usr/bin/mips-linux-nm
```

Puedes añadir las líneas anteriores al **".bashrc"** de tu home para no tener que hacer la configuración cada vez.

Compilando un "Hola Mundo"

Siguiendo con la tradición, haremos nuestro clásico **"Hola Mundo"** y lo compilaremos para MIPS big endian (o la arquitectura que hayas configurado).

holamundo.c

```
1 | #include <stdio.h>
2 |
3 | int main(int argc, char **argv) {
4 |     printf("Hola Mundo!\n");
5 |     return 0;
6 | }
```

Colocamos el código anterior en **"holamundo.c"** y lo compilamos con el siguiente comando:

```
$ mips-linux-gcc holamundo.c -o holamundo
```

Y ya está.

Ahora subimos el binario compilado al cacharro, le damos permisos de ejecución y lo corremos:

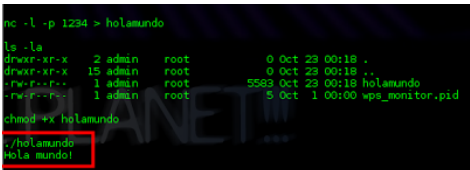


Fig. 10 - Hola Mundo en MIPS big endian

Es todo por ahora. Hasta la próxima.

Un saludo.

Publicado por Alguien el 12/07/2013 02:27:00 p. m.

Etiquetas: big endian, buildroot, compilacion cruzada, linux, mips

No hay comentarios:

Publicar un comentario en la entrada

Introduce tu comentario...

Comentar como:

Seleccionar perfil.

Publicar

Vista previa

Enlaces a esta entrada

Crear un enlace