Periódico de la Comunidad
Perl de Capital Federal

http://cafe.pm.org/boletines/

# *Cafe Perl v0.5*

# Table of Contents
# Cafe Perl v0.5

# CaFe Perl v0.5 - Periódico de la Comunidad Perl de Capital Federal

## Editorial

Un mes complicado, difícil y con sus bemoles.
Como saben este mes no está la sección Mordiditas de aquí y de allá, pero en verdad debo agradecer a Cristhian de codigolibre.org y a Manuel Sanguino por el ofrecimiento de ayuda. Lamentablemente no pudimos plasmarlo en algo concreto, pero seguramente más adelante van a aparecer por CaFe Perl.
Este mes la entrevista de PERLitas es a GomoR al creador de Net::Packet. En esta nota recibí la ayuda de Leonardo Pigñer, que fue quién me sugirió a esta persona para la entrevista (y debo decir que no se equivocó al recomendármelo) y me dio las preguntas para hacerle. Algo para no perderse.
Así que ya sabés, si querés entrevistar a alguien podés mandarme un e-mail y coordinamos como hacerlo y qué preguntas mandarle.
Espero que lo disfruten.
Hasta la próxima taza de CaFe Perl !!! ... eso sí, café del bueno ;-).

Víctor A. Rodríguez (Bit-Man)

## PERlitas

Entrevista realizada y traducida al español por Víctor A. Rodríguez

Net::Packet es un conocido framework para enviar y recibir frames desde los layers 2 a 7 de una forma fácil. Simple de decir y simple de usar, pero qué hay acerca de GomoR, su creador ?? Ahora vas a saber acerca de él en esta entrevista, pero no te engañes : es alguien que conoce sobre seguridad, eligió Perl para desarrollar sus propias herramientas, tiene sus propias ideas y le gustan los Martinis con una aceituna. Alguien con clase.

**Por favor GomoR, una introducción para el grupo CaFe.pm**

Bien, mi trabajo mayormente es hacer asesoría de seguridad a varios niveles, desde red hasta aplicaciones web. Amo desarrollar algunos tests de vulnerabilidad y otros programas de red o seguridad, y ya que no quiero gastar mi tiempo en desarrollo, comencé a aprender Perl, hace dos años.

**Así que decidió usar Perl solo porque estaba a la mano, porque era la duct tape, fue introducido por otra persona u otra razón ??**

Elegí Perl debido a la multitud de módulos disponibles, y porque si hay un lenguaje que ‘satisface todas las necesidades‘, creo que es Perl. Porque desarrollo programas para una variedad de campos (desde paquetes de red hasta frontends web y bases de datos), yo escogí Perl. Lo aprendí yo mismo leyendo los libros de O‘Reilly, y leyendo código de distintos módulos. también amo el código que Perl genera, me gusta llamar a mi código como cierta forma de arte.

**Por qué cree que algunos frames para generar exploits, como CORE Impact o CANVAS, principalmente fueron desarrollados usando Python en lugar de Perl ??**

Bueno, es una buena pregunta. Escuché que Metasploit también migrará a Python. Tengo ciertas ideas sobre eso, y creo que es porque es más fácil inyectar un intérprete Python en la memoria de un proceso remoto y Perl, debido a su estructura de filesystem, debería ser más difícil. Tengo un proyecto para trabajar sobre esto (que es inyectar el intérprete de Perl en un proceso remoto), sólo por el desafío de hacerlo. Pero no estoy seguro que pueda hacerlo. Tengo demasiadas cosas en que trabajar ahora.

**Por qué decidió hacer Net::Packet cuando ya había hechos otros módulos que se encargaban del trabajo ??**

Bueno, esta pregunta es siempre la misma: "Por qué reinventar la rueda ?". Principalmente respondo con dos razones. Primero, creo que la mejor forma de aprender sobre algo es desarrollarlo. Por ejemplo, para aprender cómo se transmite un paquete, hacé vos mismo el paquete. La segunda razón es que siempre estaba escribiendo las mismas funciones (abrir un descriptor usando Net::Pcap, poner siempre los mismos header IPv4 usando NetPacket o Net::RawIP, abrir el socket, enviar el paquete, analizar siempre

los mismos campos para encontrar las respuestas, bla bla bla). No me gusta escribir siempre el mismo código, especialmente cuando es realmente fácil. Así que comencé a desarrollar Net::Packet desde cero (tengo como tarea el remover Net::Pcap como dependencia de Net::Packet), con una interfaz simple, y un poderoso motor para manejo de layers. Básicamente, se crea un frame object, y se ponen dentro todos los layers para llevar a cabo la tarea. Así, con Net::Packet es posible encapsular cualquier-cosa en cualquier-cosa (por ejemplo Net::Packet::VLAN), mandar el frame fácilmente y capturar la respuesta automágicamente. Los descriptores se abren automáticamente, sin necesidad de preocuparse, el sniffing the los frames se hace con un forked process, y el proceso principal puede llamar al método receive en forma asincrónica.
No creo que los módulos actuales puedan hacer eso :-)

**Y los mayores pros y contras entre Net::Packet y los otros módulos ??**

Creo que Net::Packet es muy pesado en cuanto a memoria, y debido a la interfaz del framework, podría tardar más en ejecutar que el resto. La simplicidad tiene un precio. Trabajé para optimizar el uso de memoria y la velocidad, y tuve éxito con la velocidad. Creo que gané un factor de 10 desde Net::Packet 1.28.

**Mas allá de la habilidad de Perl, cuáles son los diferencias entre Net::Packet y libs en otros lenguajes tales como libpcap/libnet ??**

Bueno, no hay equivalente de Net::Packet en C/C++ que yo conozca. Cuando escribí programas de red en C, me encontré lo mismo que en Perl y decidí desarrollar Net::Packet. Puedo comparar libnet con NetPacket o Net::RawIP, o libpcap, bueno, que en realidad es Net::Pcap.
Hace tiempo comencé a escribir una lib en C que haga el trabajo de Net::Packet, pero el tiempo de desarrollo terminó por matarme.

**A propósito, puede mencionar algún trabajo que haya hecho con Net::Packet ??**

Si, el mejor OS fingerprinter que haya existido (chiste :-) ) : Net::SinFP. También desarrollé Net::Packet para que haga fácilmente pruebas de OS fingerprint y sin la limitación en la forma que se hacen los paquetes : http://www.gomor.org/cgi-bin/index.pl?mode=view;page=net_sinfp

Este realmente no tiene POD :-)

**Y que hay acerca de este módulo. Lo escribió como una prueba conceptual o fue concebido como un módulo desde el principio??**

Esto no es una PoC (N.del.T : prueba de concepto). Fue hecho como un módulo, porque integré el OS fingerprinting en otros proyectos en los que trabajo. Así que es un módulo verdadero que se puede usar para hacer OS fingerprinting en sus programas.Y la base de firmas no está vacía :-)
El archivo sinfp.pl que viene con el módulo es el programa que se compara con nmap -O. Pero hay más que OS fingerprinting activo, lo hay también pasivo, y fingerprints sobre IPv4 e IPv6. Este es el primer programa que hace OS fingerprinting para stacks IPv6. Aliento al lector a leer más sobre SinFP en el link anterior.
Para este módulo también desarrollé DBIx::SQLite::Simple, para un fácil acceso a tablas de base de datos y otros elementos. haciendo un mapping entre entradas en las tablas y objetos :
http://www.gomor.org/cgi- bin/index.pl?mode=view;page=dbix_sqlite_simple

**Podría comparar la comunidad Perl en Francia con la de otros países ??**

No puedo. No estoy involucrado en absoluto en ninguna comunidad Perl. Como un principio general no me gustan las comunidades. Podemos tener una entrevista filosófica para describir mis sentimientos sobre eso.

**OK, iba a preguntar sobre "la Comunidad de Seguridad", pero en su lugar voy a preguntar "Podría comparar al movimiento de Seguridad en Francia con los de otros países" ??**

Pienso que el "toque Francés"en el campo de la seguridad es cada vez más grande. Hace no muchos años, no había casi franceses hablando sobre seguridad en las conferencias mundiales. Y ahora, en Francia, tenemos nuestra propia conferencia de seguridad que se pone mejor y mejor cada año. Se llama SSTIC (www.sstic.org).
Principalmente esta conferencia es organizada por una revista francesa de seguridad llamada MISC (www.miscmag.com). También escribí dos artículos sobre OS fingerprinting en esta revista.

**Y cuál cree que fueron los factores que empujaron a Francia hacia este nuevo estado ??**

Siempre hay una brecha entre USA y el viejo continente. Y ahora la llenamos. Los nuevos estudiantes están más interesados en la seguridad, y los gusanos más importantes han puesto una advertencia de seguridad a los altos directivos en las compañías top. Así, hay más y más traajos de seguridad en Francia, y más y más franceses brillantes "en escena". Pero esto es una opinión personal, por supuesto.
Otra cosa, cuando comencé a aprender sobre seguridad, cerca de 7 años atrás, no había tanta documentación en Internet como hay ahora. Hoy es más fácil aprender sobre muchos campos del área de seguridad.

**Podríamos decir que la competencia elevó la calidad, no cree ??**

Bueno, creo que la competencia es algo bueno. Voy a tomar un ejemplo personal. Antes de comenzar a investigar en OS fingerprinting, la mejor herramienta (según la opinión pública general) era nmap O. Así que necesitaba hacer algo mejor que nmap.
Ahora dejo que otra gente juzgue si tuve algún éxito o no.

**Creo que hubo una exposición pública de la seguridad desde 6 o 7 años a esta parte, y que ayudó a diseminarlo, y convertirlo en "algo de lo que no se puede evitar hablar"**

Bueno, no en Francia. Puede ser en otros países, pero no aquí. En Francia, si querés encontrar fácilmente un trabajo en seguridad, tenés que ir a París. En otras ciudades tenés que tener mucha suerte.

**Y hablando de la seguridad, cuál es su opinión acerca de la revelación de vulnerabilidades hecha por Lynn en la convención Black Hat ?? (la que generó el "ciscogate") ??**

Me gusta la pregunta. por lo que leí su "anuncio" no fue nada nuevo. FX, de phenoelit, ya habló acerca de desarrollar IOS shellcodes. El mayor problema en el caso de Lynn fue que trabaja en USA, y especialmente para ISS. Malo para él. Cisco e ISS son dos empresas con base en USA. La libertad en USA se está más y más cerca a la de un país con un régimen dictatorial. Lo que hace, para esa nación, es algo que no es políticamente correcto en absoluto.

**Bueno, se sabe que esta idea genera algunos problemas, principalmente cuando son sobre dinero, ganancias, exposición pública y cosas como esas :-)**

Si. No hablo acerca de la revelación de vulnerabilidades. Sólo digo mi forma personal de actuar es a través de la revelación responsable. Encontré algunas pequeñas vulnerabilidades en el producto BEA WebLogic mientras estaba haciendo un test de penetración (pentest) para mi actual cliente, y trabajé con BEA para resolverlo. OK, le toma toma tiempo al fabricante arreglar el problema, peor trabajé para una compañía de software antes, y se de lo que se está hablando.

**Seguro. Siempre los puntos extremos no están bien posicionados (tienen una "postura radical"). Creo que la verdad está en el medio, entre los puntos extremos, actuando con responsabilidad**

Si, y requiere cierta madurez el poder ver el área gris.

**Siempre, y principalmente el tomar la responsabilidad depende de lo que hagas se puede afectar la vida diaria de millones de personas. Para bien o para mal**

Exacto. Siempre debe poder evaluarse las consecuencias de nuestros actos.

**Y cuáles son sus próximos proyectos ??**

Una verdadera base de datos de vulnerabilidades. el objetivo son las personas que testean vulnerabilidades (pentesters)

**Puede darme más detalles, suena jugoso !!**

Actualmente implementé la base de datos. No hay descripción de la vulnerabilidad. Sólo mantengo información sobre vulnerabilidades que son útiles. Por ejemplo, una vulnerabilidad de consumo de memoria (memory exhaustion) no es muy útil. Pero una vulnerabilidad que es (o fue) explotada masivamente si lo es. El tipo de consecuencia del exploit es la información más grande. Un buffer overflow, en un producto, basado en un stack es información útil. Normalmente, en la bases de datos de vulnerabilidades (VDB), esta información está enmascarada en el campo de descripción.
También agrego los requerimientos para explotarlo, por ejemplo credenciales válidas, o en el caso de la última vulnerabilidad MS05-039 es requerida una sesión nula. Mi VDB podrá (será) usable en herramientas de análisis de vulnerabilidades.

Así que el API de la base de datos está implementada, y comencé a llenar la base con datos. Actualmente tiene 56 vulnerabilidades importantes, con links a los código de explotación y las referencias cruzadas más importantes.

**Volviendo a Net::Packet (donde comenzó esta charla) : que consejo le darías a alguien que recién comienza en la programación de aplicaciones de red y que quiere usar Net::Packet ??**

Que lea el libro de W. Stevens book : TCP/IP Illustrated volume I. Y puede ser también el volume II, pero personalmente nunca lo leí. Y también que lea todos los libros de O'Reilly sobre programación Perl. No estoy aquí para contestar preguntas sobre Perl (cuando respondo a preguntas sobre mis módulos en CPAN)

**Y que consejo de su parte (además de leer libros) ?? ;-)**

Lean, siempre lean. Nunca paren de leer.

**Cómo puede alguien colaborar con Net::Packet o en cualquier proyecto que usted está llevando a cabo ??**

Denme algunas capturas de paquetes de nuevos protocolos de nivel 2, 3 o 4. Esperen a que los implemente y prueben mis implementaciones (funcionen o no).

**Y hablando de CPAN (en la pregunta previa), algún módulo favorito de CPAN ??**

Puede ser LWP::* , o DBI::* y DBD::*

**Qué considera más útil de esos módulos ??**

Que esconden las tareas de bajo nivel que no quiero aprender.

**Y qué usos de su trabajo lo sorprendieron ??**

Un hombre de la NASA ;-) No se en qué proyecto está trabajando, pero me gusta saber que Net::Packet se usa en la NASA :-)

**Parece como si Net::Packet viajó a la ISS (estación espacial internacional) !!**

Hehe ;-) puede ser. Pero no lo creo.

**Tiene alguna experiencia (divertida o no) que tuvo mientras estaba haciendo Net::Packet y que quiere compartir con nosotros ??**

El llamado código portable de un OS a otro. No existe tal cosa como el código portable antes que escribas las funciones portables.

**Bueno, creo que cuando se escribe código de bajo nivel y se lo quiere hacer portable aún un lenguaje portable no es suficiente. Hay cierto trabajo extra por hacer para esos bits que no fueron pensados para caber en el lenguaje**

Si.

**Bueno. Finalmente, algo más que nos quiera decir y no le haya preguntado ??**

Bueno, creo que hablamos sobre las ideas más importantes.

## SudorNews

### *Pugs 6.2.9*

La última versión de Pugs ( http://pugscode.org/dist/Perl6-Pugs-6.2.9.tar.gz ) fue liberada. Pugs es una implementación del esperado Perl 6 escrita en Haskell. Para todos aquellos que quieran jugar con esta nueva versión pero no quieran (o no puedan) instalarla existe un Live CD que puede ser bajado desde http://linide.sf.net/pugs-livecd-6.2.9.iso

### *WWW::Kontent 0.01*

Y ya empiezan a aparecer los primeros códigos escritos en Perl 6. Este es un web content management que puede ser ejecutado con Pugs. En particular este no está completo en todas sus funcionalidades ni está listo para producción, pero si para hacer un download, enterarse de qué se trata y jugar un poco. Para los interesados pueden dirigirse a http://search.cpan.org/~brentdax/WWW-Kontent-0.01/ y leer el archivo INSTALL para ver como dejarlo funcionando. Tengan cuidado, porque a pesar que está en CPAN no puede instalarse como un módulo. Quedan advertidos.

### *Portando Test::Builder a Perl 6*

Parece que con el avance de Pugs y Parrot todo se está moviendo alrededor de Perl 6. Así que algunos está portando su código, y chromatic no podía dejar de poner sus manos en Test::Builder y portarlo a Perl 6. Algo digno de ser leído puede encontrarse en
http://www.perl.com/pub/a/2005/07/28/test_builder_p6.html

### *Parrot 0.2.3 "Serenity" liberado*

La nueva versión de Parrot fue liberada, y puede bajarse desde
http://www.cpan.org/authors/id/L/LT/LTOETSCH/parrot-0.2.3.tar.gz. Parrot es una máquina virtual diseñada para ejecutar Perl6 y otros lenguajes dinámicos.

### *Introducción al scripting Open Source en Mac OS X*

No será una maravilla, pero realmente parece una buena introducción a la diversidad y oferta del Open Source en cuanto a scripting. Véanlo en
http://developer.apple.com/internet/opensource/opensourcescripting.html

### *Introducción al lenguaje Sleep*

Por suerte hay mucho que creen que todavía Perl tiene muchas ideas útiles, y entre ellos está Raphael Mudge que inventó el lenguaje Sleep () basado principalmente en Perl y lo convirtió en un lenguaje de scripting que puede ser embebido en Java. Si quieren un poco más lean el artículo que está en http://today.java.net/pub/a/today/2005/07/14/sleep.html

### *Calificación de sistemas Open Source*

Carnegie Mellon University, CodeZoo (O'Reilly), Intel y SpikeSource están lanzando un sistema de calificación orientado a enterprise adopters y desarrolladores. Comúnmente dentro de la industria hay cierta reticencia a usar open source mayormente por desconocimiento, y el que haya un proceso consensuado y open source hace que la adopción pueda verse facilitada. Más data en
http://www.openbrr.org/

### *El sistema LAMP pierde cuota de mercado*

Según un estudio de Evans Data (http://www.javahispano.org/news.item.action?id=196370513), ha disminuido el número de programadores que usan PHP para realizar páginas web, como también ha disminuido el número de desarrolladores que planean usar PHP para futuros proyectos. Así como ha bajado el índice para PHP, también han disminuido el número de programadores de Perl y Python. Según el estudio, la disminución ha sido porque no se ha conseguido una importante cuota de mercado en el ámbito empresarial.

### *Programming in Perl 6 style using Perl 5*

Un libro que promete llenar la brecha entre ambas versiones de una forma elegante , y ayudando a esclarecer y entrever ese futuro llamado Perl 6 (
http://books.slashdot.org/article.pl?sid=05/08/16/0511217 )

### *Grupo Unix de los laboratorios Bell desmantelado*

Indudablemente Unix es uno de los pilares de la industria informática, y como tal nació en los laboratorios del por entonces monopólico Bell Telephone. En este mes, ese mítico laboratorio fue oficialmente desmantelado. Como todo, lo bueno y lo malo, termina inexorablemente (
http://www.unixreview.com/documents/s=9846/ur0508l/ur0508l.html )

### *Búsqueda de código fuente de libros*

Una nueva facilidad para regocijo de todos los habitantes del ciberespacio : búsqueda de código en una buena cantidad de libros, fácil y rápido en http://perl.codefetch.com/

### *Para el CV : yo participo en un proyecto Open Source*

Una de las opciones mas codiciadas en el último año de una carrera son las pasantías a una empresa, las que pueden resultar en un aporte substancial al CV o bien en una oportunidad laboral. Ahora bien si este codiciado premio no llega, no tenemos más que hacer de nuestras vidas ?? Básicamente no desesperarse es lo primero, y participar en un proyecto Open Source también agrega substancia al CV. Recordemos que lo preciado es el contacto con el mundo real :
http://www.onlamp.com/pub/a/onlamp/2005/08/01/opensourcedevelopers.html

### *Evadiendo el fingerprinting pasivo*

Como leyeron en el artículo de PERLitas de este número, Perl juega un papel importante en el ambiente de seguridad. Uno de nuestros integrantes no es ajeno a esto, y puede leerse un artículo de ca0s en el último número de la revista electrónica SET : http://www.set-ezine.org/archivo/txt/set31.zip

### *Building a 3D Engine in Perl*

Y si señores, no es fanatismo pero Perl da para todo. Que no ?? Por qué no te fijás entonces en cómo hacer un engine 3D en Perl : http://www.perl.com/pub/a/2004/12/01/3d_engine.html,
http://www.perl.com/pub/a/2004/12/29/3d_engine.html,
http://www.perl.com/pub/a/2005/02/17/3d_engine.html y
http://www.perl.com/pub/a/2005/08/04/3d_engine.html

### *Para una buena programación, buenas herramientas*

Y ya que Perl es el mejor lenguaje este trío sería fantástico e imbatible. Bueno, lean
http://www.perl.com/pub/a/2005/08/25/tools.html y si bien la oferta de hoy no es la panacea que deseamos/necesitamos/anhelamos estamos en la buena senda.

### *Y seguimos indexando ...*

El objetivo del POD Indexing Project ( http://pod-indexing.annocpan.org/ ) es hacer la documentación de Perl más accesible e indexable, específicamente agregar las palabras claves correspondientes en el código POD y generar herramientas para aprovecharlas, tanto en el indexado como en la búsqueda.

### *OpenOffice::OODoc 2.003 disponible en CPAN*

Y en el orden de buenas noticias salió la versión en CPAN para poder manejar documentos de OpenOffice.org en sus versiones 1.x y 2.x ( http://search.cpan.org/dist/OpenOffice-OODoc ) ... vamos vamos, a actualizar esos módulos !!

***Querés probar tus conocimientos ...***

simple, entrá a http://brainbench.com/xml/bb/common/testcenter/taketest.xml?testId=2365 y hacé el test !!!

***Testing automatizado de GUI***

Si trabajas en Windows y no querés romperte el coco dando clicks a troche y moche para probar tus programas, ahora podés usar el módulo Win32::GuiTest module. Ahora si no querés perder el tiempo para empezar a usarlo, leete este articulito : http://www.perl.com/pub/a/2005/08/11/win32guitest.html

## POC (peace of code)

Este mes Andrés está ausente con aviso, así que lo voy a reemplazar yo ... bueno no exactamente yo sino el autor del artículo Evadiendo el Firngerprinting Pasivo ( http://www.set-ezine.org/archivo/txt/set31.zip ). Lo lamento, esta vez no hay acertijo :-(

Básicamente este script sirve para tomar un paquete de red, previo desvío hacia este script con iptables, y modificarlos a nuestro antojo (si leen el artículo se van a dar cuenta de la utilidad).

```
use IPTables::IPv4::IPQueue qw(:constants);
use NetPacket::IP qw(:ALL);
use NetPacket::TCP qw(:ALL);

use constant TIMEOUT => 1_000_000 * 2;

my $queue = new IPTables::IPv4::IPQueue(
            copy_mode => IPQ_COPY_PACKET,
            copy_range => 2048)
        or die IPTables::IPv4::IPQueue->errstr;

while (1) {
    my $msg = $queue->get_message(TIMEOUT);

    if (!defined $msg) {
        next if IPTables::IPv4::IPQueue->errstr eq 'Timeout';
        die IPTables::IPv4::IPQueue->errstr;
    }

    my $ip = NetPacket::IP->decode($msg->payload());
    my $tcp = NetPacket::TCP->decode($ip->{data});

    ########
    ## Aqui podemos modificar IP "$ip" o TCP "$tcp",
    ## antes de volver a enviarlo al Kernel
    ########

    $ip->{data} = $tcp->encode($ip);
    my $raw = $ip->encode;

    $queue->set_verdict($msg->packet_id(), NF_ACCEPT,
                length($raw), $raw);

}
```