Construcciones Reproducibles

Por Gunnar Wolf

Gunnar Wolf es administrador de sistemas para el Instituto de Investigaciones Económicas de la UNAM y desarrollador del proyecto Debian GNU/Linux.
http://gwolf.org

El premio Turing [1] de 1983 fue otorgado a Ken Thompson y Dennis Ritchie por «su desarrollo de la teoría genérica de los sistemas operativos, y específicamente, por la implementación del sistema operativo Unix». Su discurso de aceptación del premio, «Reflections on Trusting Trust» [2] (pensamientos acerca de confiar en la confianza) ha sido uno de los pilares de la práctica de la seguridad informática.

En dicho trabajo, Thompson sostiene que mientras haya gente como él, capaces de escribir un compilador a pelo, no podremos confiar auditoría alguna que hagamos a nuestro código —su artículo demuestra cómo se puede troyanizar un compilador para ocultar en él comportamiento maligno que sólo se dispara al compilar un programa determinado— o al compilar a una nueva copia del compilador mismo.

Si bien ya ha llovido en los más de treinta años que han transcurrido desde este artículo [3] el principal argumento de Ken Thompson se sostiene: en el supuesto de que hagamos una auditoría al código de determinado proyecto, ¿qué garantiza que ese sea el código que realmente ejecuta el despliegue productivo?

El planteamiento que hago no tiene nada de hipotético, y seguramente muchos de ustedes podrán encontrar ejemplos de código que no concuerda a detalle con el que pasó un proceso de certificación; parte del papel de quien audite y certifique un sistema debe ser una forma de asegurar que el software estudiado sea efectivamente el empleado.

EL ECOSISTEMA DE LAS DISTRIBUCIONES DE SOFTWARE LIBRE

Conviene mencionar, pues, el punto de partida del proyecto del cual voy a hablar en esta columna. El proyecto de Construcciones Reproducibles [4] nace de la inquietud de un grupo de desarrolladores de software libre, en un principio mayormente desarrolladores de la distribución de Linux Debian.

Debian es uno de los proyectos de software libre más grandes y longevos que hay que sigue llevándose a cabo como hace veinte años: como un esfuerzo comunitario, colaborativo; como la suma de miles de pequeñas contribuciones personales. No existe una compañía detrás de Debian, y todos quienes participamos en su desarrollo lo hacemos como voluntarios y a título personal.

Esto puede verse tanto como una ventaja (el soporte para cada una de las ideas que forman parte del proyecto depende del interés de un individuo, no hay una decisión corporativa que pueda "matar" ideas que no generen ingresos) como una desventaja (al depositar la confianza de nuestra infraestructura en un proyecto tan profundamente descentralizado, estamos confiando en cada uno de los individuos involucrados). Y las Construcciones Reproducibles nacen como respuesta a esta desventaja.

La principal garantía de funcionamiento no-malicioso que ofrecen los proyectos de software libre a sus usuarios es que, dado que el código fuente está disponible, cualquier usuario preocupado de que su sistema pueda estar de alguna manera troyanizado o capturando y reportando datos sin su autorización puede inspeccionar el código fuente de los programas que emplea y compilarlos. Como dicen, ¡pare de sufrir! Sin embargo…esta solución no escala.

Hay distribuciones de Linux, como Gentoo o Arch, que basan su modelo de operación únicamente en la distribución de fuentes, y cada uno de los usuarios compila los paquetes al instalarlos. Estas distribuciones gozan del favor de los usuarios con inclinaciones más técnicas, pero si bien ganan en flexibilidad, resultan imprácticas para su uso en producción para la mayor parte de los operadores. La mayor parte de las distribuciones opta por la distribución de binarios, paquetes precompilados, con información para una resolución automática de dependencias.

Todas las distribuciones de Linux hoy en día proporcionan repositorios de software con verificaciones criptográficas fuertes, asegurando que éste fue construido en el sistema que sus políticas de desarrollo depositen su confianza. Para muchos, y por muchos años, eso ha sido suficiente.

¡NO ES PARA QUE CONFÍEN EN MÍ!

Sin embargo, en junio de 2013, Mike Perry (desarrollador de la red anonimizadora Tor [5] pasó dos meses afinando el navegador derivado de Firefox que emplea Tor para que, si se compila dos veces, el resultado sea idéntico bit por bit. Al explicar su motivación para hacer esto, envió un mensaje [6] del cual traduzco y reproduzco porciones a continuación:

No pasé seis semanas logrando una construcción reproducible para demostrar que soy honesto o confiable. Lo hice porque no creo que los modelos de desarrollo de software basados en la confianza a una única persona puedan estar seguros contra adversarios serios.

Los últimos años hemos visto un sostenido crecimiento en el (..) uso de explotación de software por múltiples gobiernos. (..)

Esto significa que el desarrollo de software debe evolucionar más allá del "confía en el repositorio firmado por GPG de mi máquina confiable". (..) Aquí es donde las construcciones reproducibles entran en juego: cualquier individuo puede usar nuestra red de anonimato, bajar nuestro código fuente, verificar ante una serie de repositorios públicos, firmados, auditados y replicados, y reproducir nuestra construcción exactamente, asegurándose no ser víctima de tales ataques dirigidos.

Hay antecedentes previos, pero este es el primer esfuerzo serio y dedicado para lograrlo; quien esté familiarizado con la red Tor comprenderá la importancia que históricamente han dado a una fuerte protección al anonimato mediante herramientas técnicas apropiadas. Perry puso el tema sobre la mesa, y pocos meses más tarde, Jérémy Bobbio comenzó a explorar lo que haría falta para aplicar las modificaciones y principios que Perry fue descubriendo y documentando al repositorio Debian —formado por cerca de 25,000 paquetes independientes.

El trabajo iniciado por Jérémy demostró resonar en la mente de diversos desarrolladores interesados en la seguridad. El proyecto aceleró velozmente, y dos años más tarde habían ya logrado una amplia concientización respecto al problema y lograr la construcción de la mayoría del archivo; participantes de otros proyectos de software libre comenzaron a verlo con interés, y a fines de 2015 celebraron un primer congreso en Atenas.

Si bien Debian es el proyecto con mayor avance, al día de hoy están trabajando en conjunto con Coreboot, OpenWRT, NetBSD, FreeBSD, Archlinux y Fedora. En consonancia con el tema de este número de Software Gurú, pueden ver en la herramienta de integración continua [7] el impresionante grado de avance que han logrado: Al momento de escribir este artículo, más del 93% de los paquetes de la próxima versión estable de Debian logra una construcción reproducible; apenas hace un año y medio la meta era llegar a un 80%.

¿Y DÓNDE RADICA EL PROBLEMA?

Muchos de ustedes recordarán sus años formativos, y puede que cueste un poco ver el problema. Estamos hablando de la compilación de software. Si yo tengo el mismo código fuente y lo compilo dos veces, ¿no debería obtener exactamente el mismo resultado? ¿Por qué este punto resulta problemático?

Hagan la prueba. Compilen su proyecto favorito. Guarden el resultado en un tar.gz (o en un zip, dependiendo de su sistema favorito). Salgan al patio, tómense un café, y vuelvan a hacerlo. Hagan un checksum sencillo de los archivos. Les garantizo que será diferente. Imaginen ahora hacerlo desde la computadora de un compañero de trabajo, o de arbitrariamente cualquier otra computadora en el mundo (que tenga el mismo compilador instalado).

Algunas de las fuentes de variación que el proyecto de construcciones reproducibles ha tenido que abordar son:

- La hora y fecha de compilación
- Algunas de las variables de entorno (TZ, LANG, USER, etc.)
- El nombre del directorio desde donde se compila
- El ordenamiento de los archivos que presenta el sistema de archivos subyacente
- Nombre de la computadora
- Versión del kernel
- ID del usuario y grupo, permisos de sesión (p.ej. umask)

Claro, parte importante del trabajo ha radicado en identificar y aislar factores que llevan a que una construcción no sea reproducible; si el tema les interesa, los invito a probar la sorprendente herramienta diffoscope [8], que encuentra y presenta las diferencias entre dos archivos de entre un tipo impresionante de formatos.

EN SUMA

El problema planteado por Thompson puede ampliarse, dada la realidad del mundo interconectado en el que vivimos, a que a un agente suficientemente poderoso le basta hacerse del control en una computadora (la de un desarrollador de software que trabaje dentro de una distribución de Linux, en nuestro ejemplo) para, desde ella, hacerse de privilegios elevados en millones de computadoras en todo el mundo. Las construcciones reproducibles harán evidente cuando un ataque de este tipo se dirija al producto binario.

Y no es que examinar a detalle el código fuente buscando código malicioso sea sencillo, pero... por lo menos, es posible hacerlo. Este proyecto plantea un fuerte cambio en el modelo de seguridad que ofrecen los proyectos de software libre, con el que todos los usuarios ganarán mucho mayor confianza en que el software que corren está efectivamente libre de código malicioso.

Referencias y notas

[1] Al que comúnmente se hace referencia como el premio Nóbel de la computación
[2] https://doi.org/10.1145/358198.358210
[3] Para los interesados en el argumento académico, David A. Wheeler hizo su tesis doctoral en 2009 acerca de cómo «contrarrestar la confianza en la confianza mediante la doble compilación diversa», https://www.dwheeler.com/trusting-trust/
[4] https://reproducible-builds.org/
[5] https://torproject.org/
[6] https://mailman.stanford.edu/pipermail/liberationtech/2013-June/009257.html
[7] https://tests.reproducible-builds.org/
[8] https://diffoscope.org/