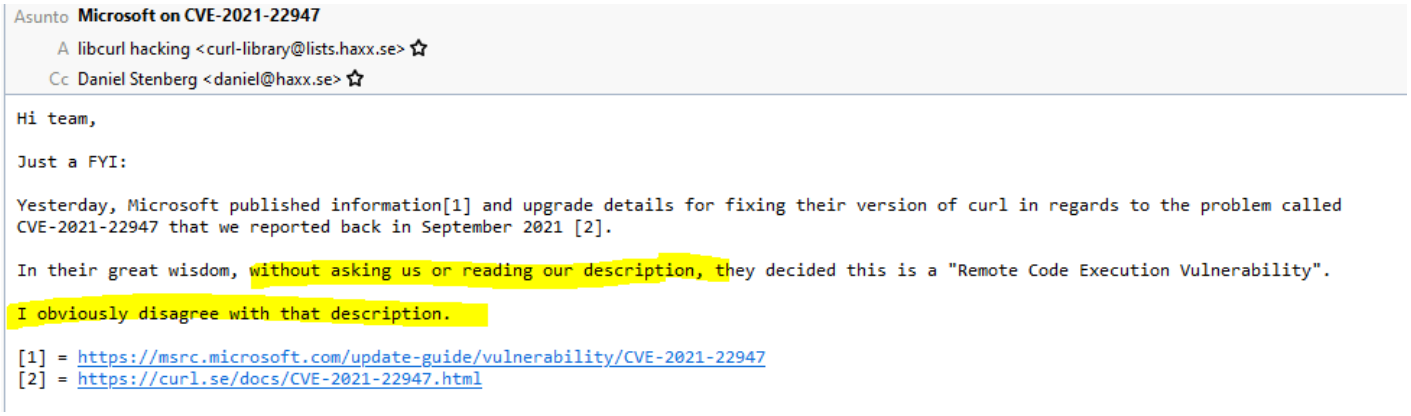
Instalando la última versión de curl

https://twitter.com/ks7000/status/1481278787748872198

En nuestra publicación en Twitter denotamos la noticia que recibimos por el boletín de noticias por correo electrónico del señor Daniel Stenberg, creador y mantenedor, desde 1998, de la poderosa herramienta **curl**.

Allí él explica que la empresa Microsoft®, **de manera increíble e inconsulta**, decidió publicar como una vulnerabilidad *(ejecución remota de* código o RCE) un error que permite enviar información sin cifrar con **curl** en algunos protocolos. A mí en particular me preocupa en protocolo SMB ya que usamos ese software para nuestros clientes.



Asunto **Microsoft on CVE-2021-22947**
A libcurl hacking <curl-library@lists.haxx.se>
Cc Daniel Stenberg <daniel@haxx.se>

Hi team,

Just a FYI:

Yesterday, Microsoft published information[1] and upgrade details for fixing their version of curl in regards to the problem called CVE-2021-22947 that we reported back in September 2021 [2].

In their great wisdom, without asking us or reading our description, they decided this is a "Remote Code Execution Vulnerability".

I obviously disagree with that description.

[1] = https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-22947
[2] = https://curl.se/docs/CVE-2021-22947.html

curl CVE-2021-22947

De manera pública en Twitter hicimos saber que esto está mal porque:

- Enviar información sin cifrar es peligroso, sí, pero *¿Es la única posibilidad de que se envíen credenciales que puedan ser interceptadas y utilizadas para ejecutar código remoto? ¿Qué posibilidad hay de eso?*
- **Además**, esa falla fue corregida en septiembre de 2021, así que si tenemos instalada la última versión de **curl**, pues listo, sin problema alguno.

https://www.youtube.com/watch?v=1y7BR0LZEuM
curl 7.79.0 with Daniel Stenberg

El asunto está que al momento de escribir estas líneas es la versión 7.81.0 y en Ubuntu 20.4 apenas van por la versión 7.74, vulnerabilidad incluida comentada. **¿Por qué ese retraso?** Pues porque se debe compilar y comprobar que las nuevas versiones de **curl** vengan "limpias", para evitar que "desequilibren" a Ubuntu... Pero obvio que no se dieron cuenta de la falla, reconocida públicamente y corregida ya. **O sea, que quedamos en las mismas**.

Ahora que criticar es sumamente fácil, pero poner manos a la obra ya es distinto: debemos ver nuestras propias fallas primero antes de poder abrir la boca. Es así que escribimos esta guía para tener siempre la última versión instalada de **curl**, veamos.

Descargando el código fuente

Investigamos la sección de instalación de **curl** y conseguimos de manera muy clara sobre cómo descargar la última versión, de manera rápida:

- Verificar si tenemos las herramientas de compilación.
- Descargar el código fuente de **curl**.
- Configurar y compilar.

Herramienta gcc

Instalamos mediante **apt** el comando **gcc** (colección de compiladores GNU):

```
$ sudo apt --assume-yes install gcc
```

Tranquilamente también podemos usar **apt-get** ya que se mantienen los alias para mantener la retrocompatibilidad. Como ejemplo colocamos capturas de pantalla en Ubuntu 16.04 (haga clic para ampliar):

sudo apt --assume-yes install gcc

Herramienta make

Instalamos mediante **apt** el comando **make** (gestión de dependencias):

```
$ sudo apt --assume-yes install make
```

Tranquilamente también podemos usar **apt-get** ya que se mantienen los alias para mantener la retrocompatibilidad. Como ejemplo colocamos capturas de pantalla en Ubuntu 16.04 (haga clic para ampliar):

sudo apt --assume-yes install make

Herramienta git

Instalamos mediante **apt** el comando **git** (control de versiones):

```
$ sudo apt --assume-yes install git
```

Tranquilamente también podemos usar **apt-get** ya que se mantienen los alias para mantener la retrocompatibilidad. Como ejemplo colocamos capturas de pantalla en Ubuntu 16.04 (haga clic para ampliar):

sudo apt --assume-yes install git

Descargando el código fuente de curl

Acá la primera opción es directamente desde GitHub:

```
$ git clone https://github.com/curl/curl.git
```

De aquí puede saltar directamente a compilar o, si por alguna extraña razón no está disponible GitHub, pues es posible descargar desde la propia página de descargas con **curl** (sintaxis larga):

```
$ curl --remote-name https://curl.se/download/curl-7.81.0.tar.gz
```

O con su sintaxis corta:

```
$ curl -O https://curl.se/download/curl-7.81.0.tar.gz
```

O con **wget** si la distro utilizada no tiene **curl** en sus repositorios:

```
$ curl -O https://curl.se/download/curl-7.81.0.tar.gz
```

Para descomprimir debe utilizar **tar**:

```
$ tar xzvf curl-7.81.0.tar.gz
```

Compilando e instalando curl

Lo primero que se debe hacer es entrar en el repositorio clonado con el código fuente de **curl**:

```
$ cd curl
```

O, en su defecto, si descomprimimos el código

```
$ cd curl-7.81.0
```

Ahora se aplica el fichero de configuración incluido por defecto, en mi caso opto por usar la librería **OpenSSL** para todo lo que es el cifrado de datos (que es la preocupación principal de este artículos):

```
$ sudo ./configure --with-openssl
```

Si aparece este error (este ejemplo es con Ubuntu 16.04):

--with-openssl was given but OpenSSL could not be detected

Pues habrá que echar mano del cifrado con GNU tools:

```
$ sudo ./configure --with-gnutls
```

Y saldrá una pantalla similar a esta:

--with-gnutls

Echamos mano del comando **make**:

```
$ sudo make
```

Y tardará algo de tiempo, poco menos de un minuto:

sudo make (curl compilation)

Ahora viene la instalación con:

```
$ sudo make install
```

En el caso de Ubuntu 16.04 y debido a la versión instalada previamente del repositorio de Ubuntu,

obtendrá este mensaje de advertencia:

curl and libcurl versions do not match

Investigando un poco sobre el mensaje de error anterior, observo que *aparentemente no reviste mayor importancia* sin embargo noto que si es llamado con **sudo** o sin él devuelve diferentes versiones, he allí el meollo del asunto:

Revisando la documentación me encuentro con que debemos colocar este parámetro:

```
export LD_LIBRARY_PATH=/usr/local/lib
```

Y repetir todo el proceso de compilación. Se puede utilizar **make test** para observar de manera

detallada el proceso de compilación. Esta captura fue hecha en Ubuntu 18.04 y tardó 5 minutos en completar la tarea:

make test (curl compilation)