Administrador de IP's

Jose Alberto Barrantes Abellan
Universidad de Costa Rica
Escuela de Ingeniería Eléctrica

Alejandro Cedeño Uribe
Universidad de Costa Rica
Escuela de Ingenieria Electrica

ÍNDICE

***Resumen*—Se planea desarrollar un programa cuya funcionalidad principal sea la del manejo de sitios seguros (https) e inseguros para que su usuario tenga más seguridad a la hora de surfear por la red.**

I. Objetivo General

- Construir un programa demonio, que utilice herramientas o librerias de sistema, para determinar las conexiones por internet actuales y advertir al usuario de aquellas potencialmente inseguras o sin encriptación.

II. Objetivos específicos

- Construir una lista de puertos seguros (yaml). Construir una lista de ip's inseguros.
- Construir un programa residente (demonio) que constantemente determine la lista de conexiones peligrosas.
- Modificar dicho programa para que despliegue cada una de las conexiones peligrosas del usuario en un formato agradable.

III. Justificación

El Internet es una herramienta que se ha desarrollado de manera exponencial, casi rayando con lo caótico, y esto trae ventajas tales como avances tecnológicos imprescindibles, otro ventaja, aunque por lo general se menosprecia, es el acorte de distancias gigantescas entre civilizaciones a básicamente un "click de distancia", y muchas más, pero lamentablemente, a su vez trae ciertas desventajas, pues es un sistema que te "expone", por así decirlo, al resto del mundo, y esto conlleva a hacer al usuario, en ciertos casos, vulnerable ante ataques de otros usuarios con malas intenciones, por ello, es imprescindible tener formas de protegerse a si mismo contra ataques de otros internautas maliciosos. Para ello se debe definir lo que es un "ámbito seguro" para los usuarios. Por ejemplo un antivirus protege al computador de ataques de "malware" que, por lo general, buscan extraer datos importantes del usuario. Con el antivirus este problema se reduce a una amenaza casi insignificante para el usuario, lamentablemente, esa no es la única forma de extraer datos. Existen diversas formas en las que se pueden robar datos, pero la que nos interesa de momento es el método de extracción de datos mediante una red wi-fi.

En general nos sentimos seguros al conectarnos a cualquier red wi-fi pública, pero esto nos deja vulnerables ante el "snooping" que se define como el acto de revisar los datos no encriptados que un usuario ingresa a un "browser" mientras

el mismo está conectado a una red wi-fi pública que por lo general no encripta estos datos, que pueden variar entre simple historiales de búsqueda a claves y otras cosas. La idea de este proyecto es crear una interfaz que proteja al usuario de sitios inseguros o no encriptados para que sus datos no se vean comprometidos facilmente, dicha interfaz tendrá un tono ameno para que el usuario no se sienta presionado ni molesto cuando se le advierta que su seguridad podría estar en riesgo y que el usuario decida si proseguir en dicho dominio o no.

IV. Metodología

Lo primero que se piensa desarrollar es la lista de ip's seguros, pues es basicamente lo más fundamental del programa, debido a que se necesita saber previamente que ip's son seguros o no para poder construir el programa a partir de las características de dichos programas no seguros, además de los que si son seguros, y que el programa pueda diferenciarlos. Posteriormente se necesita investigar la documentación de los programas pcax, python tcp, netstat, unbound, y ss-t pues todos estos programas van a ser utilizados en el proyecto y, por lo tanto, es necesario manejarlos lo mejor posible. Finalmente, para desarrollar el programa en sí, se planea utilizar Python en su mayoría pues, la lista de programas mencionados con anterioridad están en python, mas no se cierran las puertas al uso de otros lenguajes como C, por el momento es preferible usar python.

V. Marco Teórico

V-A. TCP/IP

El modelo TCP/IP es un sistema de protocolos y guías las cuales permiten que un ordenador se logre conectar a una red. Este sistema permite manejar la conexión de puerto a puerto de manera que cada detalle de la misma, ya sea, las características de los datos, o los datos en sí, sean cambiados al placer del destinatario. Para asegurar que el intercambio de datos es seguro, es necesario tener un sistema complejo de software del manejo de los mismos, que está diseñado a partir de un sistema de capas, que manejan los datos capa por capa y están diseñados para que una capa haga que la información sea más segura al manejar que la anterior.

V-B. Dirección IP

Una dirección IP, es un arreglo de números los cuales se asignan a un aparato en específico, ya sea un computador, una tablet, un celular o cualquier dispositivo con acceso a la internet, de manera sencilla, y obliga al usuario a entrar en el sistema de identificación del TCP/IP.

V-C. Daemon

Un demonio o servicio es un programa que se ejecuta en segundo plano, fuera del control interactivo de los usuarios del sistema ya que carecen de interfaz con estos. Basicamente, es un programa que corre en un bucle sin necesidad que el usuario interactue más de la cuenta con el mismo.

Por lo general, un demonio empieza a correr al inicio del programa, pero también pueden iniciarse después de que el programa encuentra las condiciones predilectas por el programador. En ciertos casos los programas demonio inclusive, se pueden usar para configurar sistemas de hardware, por ejemplo, cuando un computador inicia el modo de "sleepz espera a que el usuario oprima de nueva el botón de encendido.

V-D. Dominio

Un dominio en redes de computadoras, se refiere a la compañia que maneja una cierta cantidad de IP's en un URL específico. También puede verse como una red de identificación que se puede asociar a un grupo específico. El objetivo principal de los nombres de dominio en Internet y de los DNS's, es traducir las direcciones IP de cada nodo activo en la red, a términos simples para que el usuario no tenga problema en recordarlos.

VI. Estado del arte

En la actualidad, los programas para manejar los URL's a los que el usuario decida acceder, son bastos, por así decirlo, no hay razón por la cual esto no sea el caso, después de todo, la seguridad al navegar es primordial, para evitar muchos problemas como internautas. Hoy en día, si una persona común navega buscando un "manager" de IP's o de URL's se encontrará con una vasta cantidad de estos, disponibles a una descarga. Entre ellos tenemos:

VI-A. Thompson IP Manager

Este programa, desarrollado por una compañía a nombre de Thomson Reuters, tiene todo lo necesario para un usuario en busca de seguridad, en un formato bastante ameno a los usuarios "casuales" por así decirlo. Al inspeccionar el programa, se puede notar el tacto de los desarrolladores en este proyecto, que se encuentra en sus etapas finales de construcción, si no ya construido en su gran mayoría, solo en busca de actualizaciones que le den un poco de mejor calidad de vida al programa. El programa permite manejar las conexiones actuales al sistema y brinda seguridad de punta a la hora de navegar.

VI-B. TCP/IP Manager project

Este proyecto promete bastante ya que es un open source que permite al usuario manejar las conexiones del computador en cualquier lugar, ya sea público o privado. También, permite manejar o guardar las "settings"que el usuario desee tener en su computador. Finalmente, como se desea por lo general en programas de este tipo, el programa tiene una interfaz fácil y amigable para el uso común.

VII. Manual de Usuario

En el siguiente manual de usuario se indica todo lo necesario para instalar, ejecutar, y usar el Administrador de IP's en su distribución de GNU/Linux.

VII-A. Instalación

Para poder ejecutar este programa, es necesario que el computador en el que se ejecute disponga de python, por lo general las distribuciones de Linux ya traen Python instalado previamente, pero en el caso en el que no lo tenga, es primordial que sea instalado antes de ejecutar el programa. Después de asegurarse de lo anterior, siga los siguientes pasos:

- Descargue la carpeta comprimida (en formato .tar.gz) con el nombre "Manager"que contiene los archivos correspondientes.
- Descomprimala, escribiendo el comando ***tar -xvf Manager.tar.gz*** en alguna terminal de Linux.
- Asegúrese que los archivos dentro de esta carpeta tengan permisos de ejecución.

VII-B. Ejecución

- Vaya al directorio correspondiente donde está ubicada la carpeta "Manager". De no haberla movido de lugar el directorio debería ser: **/home/user/Download/Manager**
- Ejecute el programa escribiendo: ***./iptools.py*** en la terminal. O bien, ejecutando primeramente ***ipython*** y luego corriendo el archivo (**run iptools.py**)
- Le aparecerá una lista con todas las conexiones actuales y de ingresar a alguna de las conexiones "prohíbidas en la lista le aparecerá un pop-up advirtiendole del caso.

VIII. Conclusiones

En conclusión, los resultados obtenidos fueron, en su mayoría, los que se desearon para este proyecto, sin embargo, aunque los objetivos específicos de dicho proyecto se cumplieron en su totalidad, hubieron otros objetivos aparte que no se lograron obtener, por ejemplo, el programa no funcionó con los puertos incluidos, un problema el cual se busca atacar de nuevo con un nuevo script que los pueda leer, la razón por la cual esto no se intentó en este proyecto fue simplemente por la falta de tiempo, ya con más tiempo disponible, el programa debería poder obtener los puertos de los URL's que se ingresen y cumplir la funcionalidad que se tenía pensada en un principio en su totalidad. Aparte de eso, el programa se puede mejorar en varias áreas, por ejemplo, la lista que se tiene es bastante, limitada, por así decirlo, ya que es una lista hecha a mano, se podría decir que es una lista hecha a, fuerza bruta, pues esto implica que para que el proyecto sea bastante exitoso en su trabajo, se necesita de una lista basta que permita a un usuario promedio ingresar a los sitios que le plazcan y que el programa le advierta de aquellos que no son seguros, pero por supuesto, tener todos los IP's o puertos inseguros de la internet, es una tarea bastante difícil, por no decir imposible, claramente, de encontrar un patrón en estos puertos e IP's inseguros, la tarea se vería completa casi en su totalidad, pero como este no es el caso, solo se puede esperar que la lista sea lo suficientemente completa para que un usuario este seguro en su recorrido por la internet, a la medida de lo posible. Por otro lado, se pueden denotar ciertos aspectos de los programas o métodos usados en el proyecto, por ejemplo el sistema "demonio.º "Daemon.en inglés, fue un poco interesante al usar, pues este permitió hacer un bucle del programa, cosa que, se acostumbra hacer con un "While" pero al usar el Daemon se vuelve un programa un poco más ameno a la hora de intentar hacerlo lo más simple posible. Finalmente, se puede hablar del proyecto a futuro, hacia donde va este programa. Por ahora, lo que se desea es simplemente construir un programa más amigable, con más funcionalidades, y aún más seguro. Para la primera parte, el formato de interacción entre programa y usuario se encuentra en una estado, poco amigable, por así decirlo, esto lo podemos ver en el pop-up, que simplemente anuncia el ingreso a un URL inseguro, cosa que no es bastante útil, a la hora de evitar que el usuario obtenga malware en IP's no seguros, se desea hacer que el pop-up de la opción al usuario de bloquear dicho URL, y que cierre la pestaña con dicho URL inmediatamente. Para la segunda parte, se había pensado tener una opción de agregar sus propios URL's "prohibidos" al usuario, esto por si el usuario desea tener un tipo de "control parental" o algo por el estilo con las navegaciones hechas en su computador. Para el último aspecto se desea simplemente que la lista de IP's sea mucho más amplia, además de que el programa pueda manejar también los puertos, aspecto del que ya se habló. En general estas son las áreas en las que el programa puede mejorarse.

IX. Anexos

IX-A. Cronograma

	Fecha	**Actividad**
1	06 al 12 de Junio	Creación de la lista de ip's no seguros y de la lista de puertos seguros
2	06 al 12 de Junio	Investigar la documentación de los programas a utilizar en el proyecto.
3	06 al 12 de Junio	Posible inicio de la base del programa.
4	13 al 19 de Junio	Reunión con el profesor.
5	13 al 19 de Junio	Finalizar lista de ip's y puertos, investigar la construcción de pop-ups en python, además de la construcción de programas demonio.
6	20 al 26 de Junio	El programa ya debería diferenciar ip's seguros de inseguros además de correr constantemente y mostrar los pop-ups.
7	20 al 26 de Junio	Finalizar el script.
8	27 de Junio al 3 de Julio	Toques finales al proyecto, presentación del proyecto con demostración de funcionalidad

X. Referencias

Referencias

[1] Adrian-Costin Tundrea. TCP/IP Manager project. Recuperado el Sábado 2 de Julio del 2016 en el sitio web: http://tcpipmanager.sourceforge.net/

[2] Jonathan Muller. How to run a shell script at startup. Extraído el Lunes 13 de Junio del 2016 del sitio web: http://stackoverflow.com/questions/12973777/how-to-run-a-shell-script-at-startup

[3] Jesús Torres. Servicios y demonios en Linux. Recuperado el Jueves 16 de Junio del 2016 del sitio web: https://jmtorres.webs.ull.es/me/2013/05/servicios-y-demonios-en-linux/

[4] NLnet Labs. Unbound. Recuperado el Viernes 24 de Junio del 2016 del sitio web: http://unbound.net/documentation/index.html

[5] Margaret Rouse. TCP/IP (Transmission Control Protocol/Internet Protocol). Recuperado el Sábado 2 de Julio del 2016 en el sitio web: http://searchnetworking.techtarget.com/definition/TCP-IP