Sendmail Restricted Shell

Amaia Echeandia Paredes
Diseño y Programación Seguras
Universidad de León
León, España
aechep00@estudiantes.unileon.es

***Abstract*—Este documento expone el caso de estudio del diseño del programa *smrsh*, utilizado para el servidor de correo Sendmail. En el programa fueron descubiertas dos vulnerabilidades introducidas durante mantenimiento de código que, más tarde, fueron solucionadas para versiones posteriores.**

Descriptores—vulnerabilidades, smrsh, diseño de software, /bin/sh,

I. Resumen

Sendmail es un servidor de correo que utiliza el programa *smrsh* para mejorar laseguridad del servidor. El programa principalmente se encarga de restringir los programas que un usuario puede ejecutar. Este programa se hizo con la idea de reemplazar el conocido /bin/sh e incluir, gracias al directorio */etc/smrsh*, una lista de programas que se pueden ejecutar en Sendmail. Así, en el caso de que un intruso accediera a Sendmail, este no podría ejecutar ningún programa que no estuviera definido en la lista.

A su vez, *smrsh* lo que hace es prohibir la escritura de algunos caracteres en los comandos para evitar que se produjeran ataques en el servidor. Por ejemplo, la manipulación del Shell utilizado en Sendmail.

Esta prohibición incluía caracteres tales como '< > ^ & \n', pero después de un asesoramiento llevado a cabo por "SecuriTeam", se descubrieron dos métodos de ataque en la versión Sendmail 8.12.6-15 y anteriores. Estos ataques se producían introduciendo una secuencia de caracteres [2]. El primer método de ataque aprovechaba que la lista de caracteres ilegales no incluía otros tantos caracteres con los que el usuario podría aprovechar para evitar las restricciones de ejecución. Esta práctica utilizaba caracteres como '||' o '/' en los comandos. A continuación podemos verun ejemplo:

```
$ echo "echo unauthorized execute" > /tmp/unauth
$ smrsh -c ". || . /tmp/unauth || ."
  /bin/sh: /etc/smrsh/.: is a directory
  unauthorized execute
```

A pesar de que "*/tmp/unauth*" no está listado en el directorio */etc/smrsh* que permite la ejecución de los programas, éste se ejecuta igualmente. Esto ocurre porque el programa lo primero que comprueba es que '.' sí que existe y no hace una comprobación de ficheros posteriores a '||' [3].

El segundo método de ataque se basa en una rutina que existe en el fichero *smrsh.c.* Un exploit elimina las restricciones de *smrsh* y, además, permite la ejecución de programas aún no teniendo acceso al shell del sistema. Esto le permite al ciberdelincuente la escritura de comandos arbitrarios en el sistema de manera que pueda escalar privilegios.

Estas dos vulnerabilidades fueron corregidas por sendmail.org a pocos días de detectarlas.

II. Conclusiones

Gracias a este caso de estudio se ha podido mostrar la importancia de un buen diseño de programación, que incluso el más experimentado profesional es susceptible a este tipo de errores.

Algo a tener en cuenta a la hora de diseñar un software seguro es la paciencia y el cuidado del planteamiento preliminar del programa. Esto incluye también la escritura de codigo lo más simple posible para poder detectar errores más fácilmente.

Por último, es importante anticiparse a los posibles problemas que se pueden producir y que, a su vez, el código pueda ser adaptable para que, en el futuro, sea posible mejorar el diseño.

III. Referencias

[1] Mark G. Graff, Kenneth R. van Wyk, "Secure Coding: Principles & Practices", Chapter 3: Design. O'Reilly Media, Inc. ISBN:0-596-00242-4, June 2003

[2] Incibe-CERT, CVE-2002-1165, https://www.incibe-cert.es/alerta-temprana/vulnerabilidades/cve- 2002-1165

[3] Linux man pages, Section 8: smrsh, https://linux.die.net/man/8/smrsh