Linux Kernel−Level Trojan − Kernel Intrusion System (KIS)

Source: http://www.derkeiler.com/Mailing−Lists/securityfocus/focus−linux/2001−07/0008.html

---

From: Timothy Lawless (*lawless@netdoor.com*)
Date: 07/22/01

```
Date: Sun, 22 Jul 2001 15:53:32 −0400 (EDT)
From: Timothy Lawless <lawless@netdoor.com>
To: <focus−linux@securityfocus.com>
Subject: Linux Kernel−Level Trojan − Kernel Intrusion System (KIS)
Message−ID: <Pine.LNX.4.33.0107221551540.28815−100000@pantheon.wwjh.net>
```

This document describes the Kernel Intrusion System (KIS) trojan that affects Linux 2.2 and 2.4 systems. The specific version of the KIS trojan analyzed is labeled 0.9.

1. Introduction

At the Defcon Conference in Las Vegas, NV at 10:00am PST on July 14th 2001, the KIS trojan was published by an individual who is identified as Optyx. The trojan is designed to automate the loading of a kernel module. Once loaded the kernel module will attempt to conceal its presence, and listen to the network for instructions.

2. Description

The KIS trojan is a hybrid between zombie daemons which came to light as a result of DDOS attacks on major sites at the beginning of 2000 and kernel level rootkits that are used by hostile entities to conceal their presence on a system after a successful compromise.

In its remote control client, the KIS trojan delivers a similar look and feel as is associated with Back Orifice or SubSeven.

By issuing commands from a remote KIS client, an individual is capable of executing processes on a victim host while hiding arbitrary files, child processes and network connections.

The KIS trojan is introduced into a system in the form of a regular executable binary that contains the KIS kernel module and the trojan.

3. Operation

The KIS trojan is inserted on a victim host by executing a binary that

installs the trojan, and loads the KIS trojan kernel module.

The trojan is installed into the system by replacing the /sbin/init binary with the trojan. Upon bootup, the trojaned /sbin/init will load the KIS kernel module and subsequently call the original "init" binary that has been moved to a hidden directory. This ensures that the KIS trojan is the first kernel module loaded on the system.

In the testing of the KIS system, it appears it was designed only to load from init. Multiple runs of the trojan binary, such as what would occur if it were to replace /bin/sh or another binary that runs often, can cause the system to hang, generate "Out of Memory" messages or become unstable.

During loading, the KIS kernel module performs several tasks:

−− Conceals the Modules Presence by Removing the Module from the modules_list structure.

−− Replaces key system calls.

−− Replaces portions of the vfs structures for the net/tcp, net/udp, and net/raw files in the procfs.

−− Spawns a kernel_thread to process incoming commands from the network.

−− Replaces the ip_packet_type structure with a new structure to allow KIS to monitor all ip based network traffic and add observed commands to queue.

Commands are sent to the KIS trojaned system from a KIS client console. The commands are sent via directed IP packets with a specific length to match a modulus and remainder defined in the KIS module upon compile.

If the packet matches the length requirements and decrypts into a valid command packet, then the command is added to a queue for processing.

The queue manager takes a queued command off of the queue and performs the directed command.

Valid commands include:

−− Execution of A Process
−− Hiding a running process
−− Revealing a hidden process
−− Hiding a file
−− Revealing a file
−− Hiding a connection
−− Revealing a connection
−− Ping
−− Shutdown and Removal of the Trojan

The queue manager is always running, monitoring the incoming queue of commands. As a result, the load on a victim system will never fall below a load of 0.80.

Additionally, as a result of the replaced systemcalls and the requirements to manage hidden files and processes, filesystem operations such as listing or even compiling a kernel consume up to 30% more system time then the victim system would consume in a non−trojaned state.

4. Risk

The KIS system permits a remote execution of processes on a victim system. Combined with its ability to conceal such executions, files, and network activity from normal processes, the KIS system provides a prime platform from which attacks against the integrity and availability of other compromised systems may be launched.

Despite the need to compile a KIS trojan for each kernel, a pre−compiled KIS trojan could be packaged and distributed to victim hosts that are running stock kernels.

If such a pre−compiled binaries were to be included into a RPM or DEB package, a KIS trojan could be introduced to victim systems by administrators installing a new or updated package.

5. Detection

The KIS system can be detected on initial load by the StMichael 0.05 Linux Kernel Module. The StMichael Linux kernel module is a integrity monitor for the 2.2 and 2.4 Linux kernels.

On systems not running StMichael, or that are suspected of being infected by the KIS trojan, the following observations can be made to determine if the system is actually infected:

During normal operations (ie, runlevel 3 or 5)

−− "ps −auwx | grep init" will reveal two "init" processes.

−− The system load will be high, not falling below 0.8 even when all observable processes are idle or sleeping.

−− Repeated runs of "init", such as switching between runlevels or rebooting will cause the system to hang, generate error messages or crash.

−− Portscans on the system reveal open and listening ports that are not reflected in a netstat.

Alternatively, if the system can be safely brought down and booted off of a clean or rescue media, search the filesystem for a file called "ht"

with mode 000. This is the hidden file table used by KIS to list hidden files.

```
find / −name ht −mode 000 −print
```

Also, the trojaned init process is copied to a file in the same directory as the original init binary, but named "init."

```
find / −name init\. −print
```

Additionally, to determine if files are hidden by KIS or any other trojan the following method may be applied:

1. Boot the system normally. Generate a file list via:

   ```
   find / −type f −print > listing.1
   ```

2. Boot the system into single user mode using 'clean' media such as a rescue disk. Mount all the filesystems (ie, on /mnt)

   ```
   find /mnt −type f −print | sed −e "/\/mnt//g" > listing.2
   ```

3. Diff the two listings to determine the hidden files

   ```
   diff −u listing.1 listing.2
   ```

6. Cleaning

To remove KIS, the trojaned init binary needs to be replaced with a clean init binary or the original init binary.

The original init binary is stored in the same directory that the ht file is stored in. There are two files that are named simularly: "init." and "init" .

"init" is the original init binary.
"init." is a copy of the trojan.

Copy the "init" file over the trojaned init file:

```
cp init /sbin/init
```

The "ht" file will contain files and directories that are hidden. These files may contain information about the attack and what the trojaned system was used for.

If the source for KIS is recovered, the server.h file will contain the keys, modulus, and remainder needed to communicate with the KIS server. Using this information it may be possible to sniff the network, identifying packets that are sent in attempts to contact the KIS server. In some circumstances, such as with the PING command, the origin IP

address will likely not be spoofed.

Tim Lawless
lawless@netdoor.com

- ***Previous message:*** Brian Kejser: "IPTables Upgrade"
- ***Messages sorted by:*** [date] [thread] [subject] [author] [attachment]