Global Information Assurance Certification Paper

Bob root mys
RAMEN – A Linux Worm
Jack R. Collins (collins002)
Level One, Security Essentials Track, December 2000

Summary: The goal of this assignment was to research an exploit or vulnerability using the resources available on the Internet. Choosing the recent "ramen worm" as an example, we found security advisories warning of its existence, clean-up scripts for removal of the worm from an infected host, thorough analyses of its mechanism of infiltration and replication, documentation of its "mutation" over time, its impact on an infected host and the internet as a whole, its "signature" for detecting it on a host and a network, and recommendations for easily patching the vulnerabilities so as to prevent an infection in the first place. What is truly interesting about this particular worm is the fact that all of the exploits used to gain entry into a host had been noted and patches for the vulnerabilities were available several months prior to the appearance and rapid spread of the ramen worm. Further, the fact that all of the information listed above was gathered and made available in approximately one month (mid-January to mid-February in 2001) is truly impressive. The ramen worm underscores the growing concern of security professionals about the relative lack of security consciousness in the computer community as a whole. Here, we emphasize the information freely available for detecting, understanding, and eliminating malware such as ramen, and re-emphasize the message of security organizations, such as SANS and CERT, regarding security awareness. Finally, several programs that are freely available and, if installed on all of the linux boxes infected by ramen, would have essentially stopped this worm before it started.

What is the ramen worm?

The ramen worm is a self-replicating program created from commonly available hacking tools.(1,2) Using three well-known vulnerabilities in default installations of Red Hat linux systems, the worm gains access to a computer and sets-up programs to scan for more vulnerable systems on the network. The worm appears to be a simple collection of hacking tools (rootkits), readily available on the Internet, bound together in shell scripts creating a rapidly self-replicating package.(3) Ramen's name comes from a unique feature of the worm once inside the infected host. It defaces web pages by replacing index.html files with its own version that includes the words "RameN Crew", an image of a package of Ramen Noodles, and the phrase "Hackers looooooooove noodles!". The worm appears to do no further damage other than use a large amount of network bandwidth scanning for vulnerable targets. No other worm has left such a public display of its presence, especially without doing any further damage to the host system itself. Quite interestingly, in an apparent effort to prevent the worm from attacking hosts it has already infected, ramen actually closes the vulnerabilities of the systems that it infects. The contents of the ramen worm, in its original form, listed here are reproduced from Daniel Martin's document on the Ramen Worm.(4,5)
(http://members.home.net/dtmartin24/ramen_worm.txt)

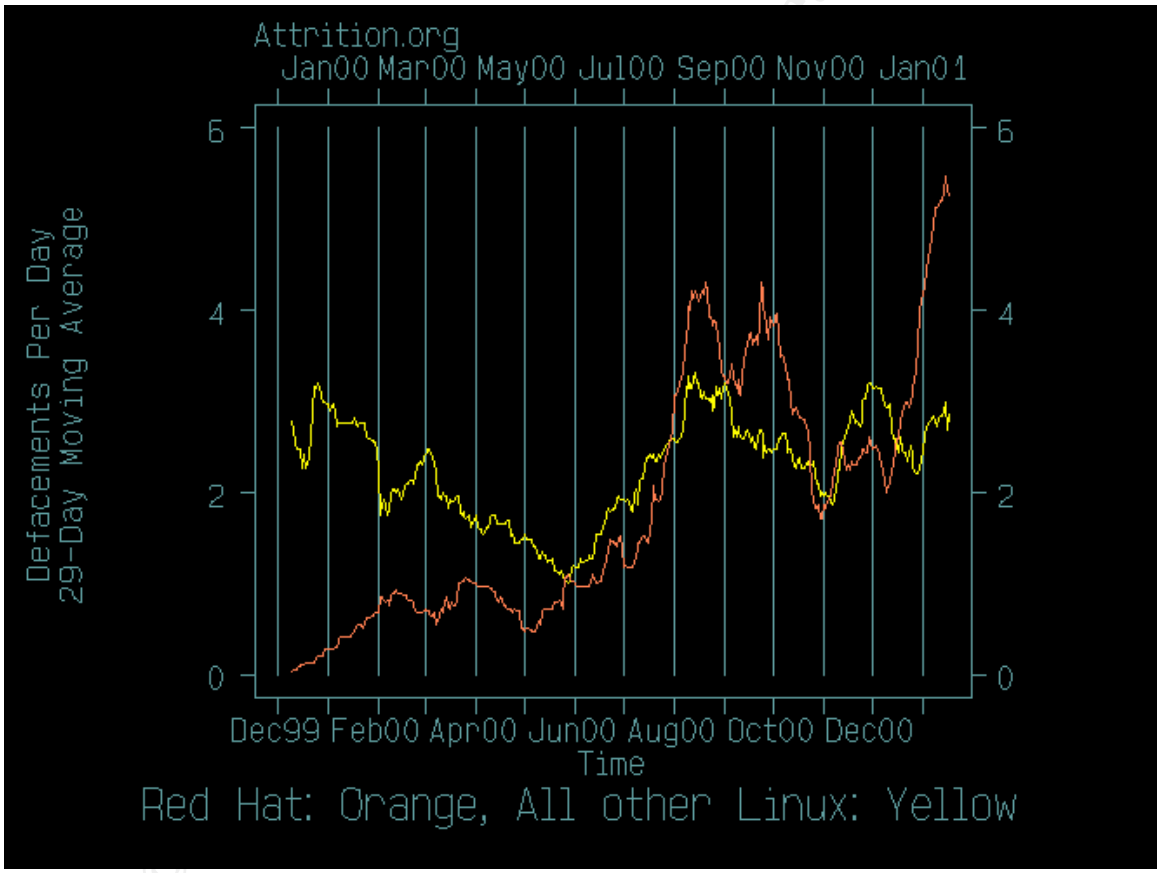
Contents of ramen.tgz

asp:	An xinetd config. file that will start up the fake webserver Used on RedHat 7.0 victim machines.
asp62:	HTTP/0.9-compatible server that always serves out the file /tmp/ramen.tgz to any request - NOT stripped
asp7:	RedHat 7-compiled version - NOT stripped
bd62.sh:	Does the setup (installing wormserver, removing vulnerable programs, adding ftp users) for RedHat 6.2
bd7.sh:	Same for RedHat 7.0
getip.sh	Utility script to get the main external IP address
hackl.sh:	Driver to read the .l file and pass addresses to lh.sh
hackw.sh:	Driver to read the .w file and pass addresses to wh.sh
index.html:	HTML document text
l62:	LPRng format string exploit program - NOT stripped
l7:	Same but compiled for RedHat 7 - stripped
lh.sh:	Driver script to execute the LPRng exploit with several different options
randb62:	Picks a random class-B subnet to scan on - NOT stripped
randb7:	Same but compiled for RedHat 7 - NOT stripped
s62:	statdx exploit - NOT stripped
s7:	Same but compiled for RedHat 7 - stripped
scan.sh:	get a class B network from randb and run synscan
start.sh:	Replace any index.html with the one from the worm; run getip; determine if we're RedHat 6.2 or 7.0 and run the appropriate bd*.sh and start*.sh
start62.sh:	start (backgrounded) scan.sh, hackl.sh, and hackw.sh
start7.sh:	Same as start62.sh
synscan62:	Modified synscan tool - records to .w and .l files - stripped
synscan7:	Same but compiled for RedHat 7 - stripped
w62:	venglin wu-ftpd exploit - stripped
w7:	Same but compiled for RedHat 7 - stripped
wh.sh:	Driver script to call the "s" and "w" binaries against a given target
wu62:	Apparently only included by mistake. "strings" shows it to be very similar to w62; nowhere is this binary ever invoked.

A bit of epidemiology

Once introduced into the wild, the ramen worm spread quite rapidly through vulnerable computers running the Red Hat linux (versions 6.2 and 7.0) operating systems in their default configurations. Red Hat has reissued version 7 as a “respin” that has changed the default installation to close the known vulnerabilities. According to Mihai Moldovau, a Romanian network administrator, the worm scanned two class-B networks, approximately 130,000 addresses, in about 15 minutes. This ability to rapidly scan for vulnerable systems and quickly spread accounts for the sharp upturn in compromised

systems among the Red Hat population of linux hosts when compared to other variations of linux. The effect of the ramen worm on web sites is graphically portrayed in the figure below, taken from Attrition.org. There appears to be a clear correlation between compromised linux servers and the CERT incident note in mid-January. Even though the original ramen worm appeared to be relatively harmless, the worm has been modified by mid-February to include a rootkit called "knark", a bind 8.2 scanner and exploit, a Trojan version of sshd, an RPC scanner called pscan and an exploit, and possibly an extra ftp server. This entire process has taken less than two months to play-out on the Internet. Clearly the simple framework of easily available tools implemented using shell scripts and the easy access to other "cracker" tools via the Internet has produced a fertile ground for the "evolution" of ramen into much more malicious code. Even though the level of sophistication needed to create the ramen worm is low, the impact can be quite far-reaching, as evidenced by the thousands of infected systems.



Patches were available prior to the appearance of ramen.

What may be the most disturbing feature of this worm is the fact that all of vulnerabilities that were exploited had been known for approximately six months and that fixes were readily available on the Internet. CERT had posted warnings regarding each of these vulnerabilities long before the ramen worm was found in the wild. (See section on Relevant Security Advisories.) According to Lance Spitzner, coordinator of the Honeynet Project, "It's lack of awareness. Not enough people are taking measures to secure the default installations."(3) He further noted, "most default installations are insecure".(3) This incident points out the need for software vendors to ship their products with tighter security as the default, and allow users to relax security policies as needed, in addition to an increased effort to instill security awareness into the end users.

Detection of the ramen worm:

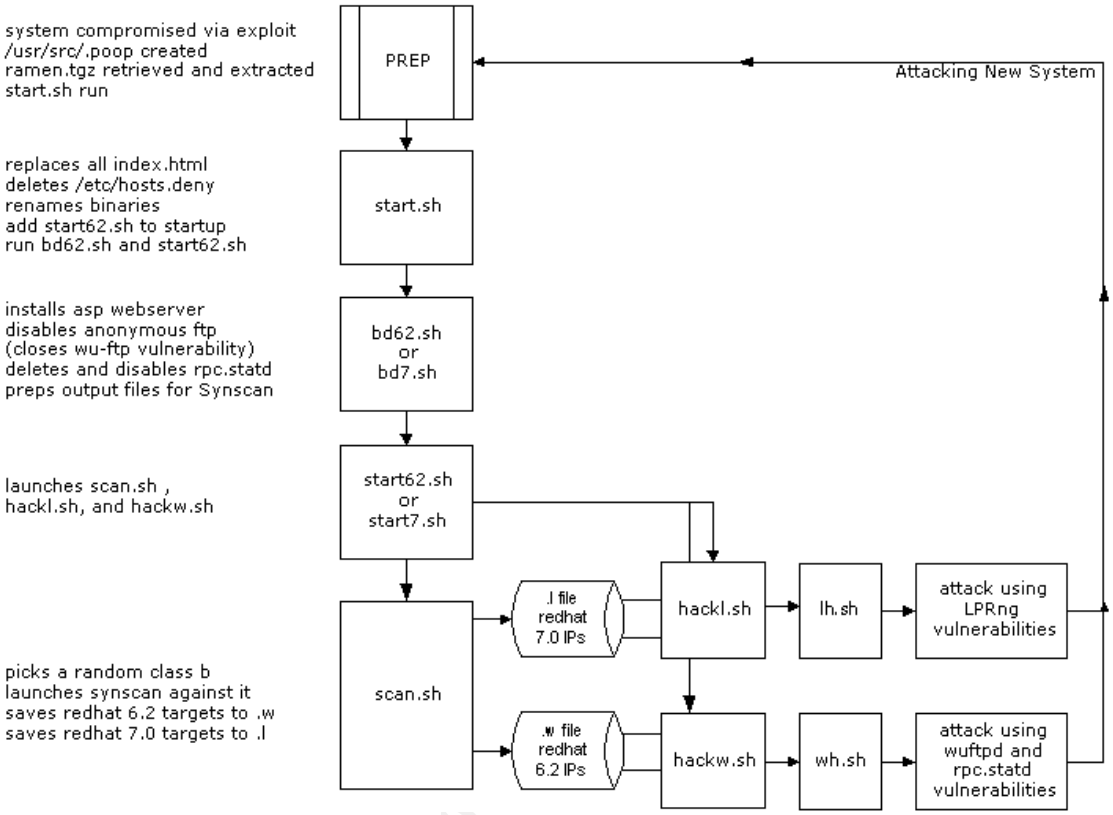
Detection of the ramen worm is quite straightforward since it leaves a quite distinct and visible trail. William Stearns has written a script to detect the ramen worm, and it has been posted at the SANS site (http://www.sans.org/y2k/ramen.htm) for public download. Even without the script, ramen can easily be detected. First, index.html files will be changed and these web pages will display the "ramen logo" in a very public manner. Second, ramen creates a directory called /usr/src/.poop (note the dot in front of poop) in which the script installs itself. Third, ramen adds a file called /sbin/asp. Fourth, ramen modifies /etc/rc.d/rc.sysinit by adding lines referring to the directory /usr/src/.poop. In later versions of ramen, including the knark rootkit and a trojanized version of sshd, one can try to ssh to port 5555 and see if it displays a prompt.

Analysis of ramen's exploits and replication

The ramen worm was thoroughly analyzed quite quickly once the ramen.tgz file had been obtained by programmers and analysts around the world. The following diagram and text summarize the exploits of the initial ramen worm based on an analysis by Max Vision at whitehats.com.(6) (http://www.whitehats.com/library/worms/ramen/index.html).

The worm starts its life cycle when the ramen.tgz package is extracted into a directory called /usr/src/.poop., and the start.sh script is run. The script defaces all index.html files on the server, copies binaries to appropriate locations, and adds the worm start script to /etc/rc.d/rc.sysinit so the worm will start again upon reboot. Perhaps one of the most curious actions is next. After installing a webserver, ramen disables anonymous ftp, closing the wu-ftp vulnerability, and disable rpc.statd (if Red Hat 6.2). The worm then launches a synscan attack on a class B network and stores potential targets running Red Hat 6.2 and 7.0. The worm then attacks RH6.2 systems through the wuftpd and rpc.statd vulnerabilities, and RH7.0 systems through the LPRng vulnerability. Once the worm compromises another machine, it starts the cycle over again.

Other variations of the worm, containing the knark rootkit and trojanized version of sshd, appear to proceed similarly but infect deeper into the system and leave backdoors for re-entry.(6)



Instructions for removing the ramen worm

Instructions for removing the ramen worm are freely available from sources such as SANS (http://www.sans.org/y2k/ramen.htm), securityfocus.com, and linuxlock.org (http://www.linuxlock.org/features/ramenfix.html). Once all of the worm's files have been removed and the relevant files in /etc repaired, the machine should be rebooted to kill the processes started by ramen. Finally, security patches that address the vulnerabilities exploited by ramen should be applied before opening the system to the Internet. A short summary from William Stearn's document is given here:

1. Delete: /usr/src/.poop and /sbin/asp.
2. If it exists, remove: /etc/xinetd.d/asp
3. Remove all lines in /etc/rc.d/rc.sysinit which refer to any file in /etc/src/.poop.
4. Remove any lines in /etc/inetd.conf referring to /sbin/asp
5. Reboot the system or manually kill any processes such as synscan, start.sh, scan.sh, hackl.sh, or hackw.sh.
6. ISS recommends that ftp, rpc.statd, or lpr are not enabled until updates have been installed.

Procedures for protecting against ramen and similar malware

The first and most critical step in protecting against malware such as ramen is a concern for and awareness of security issues. In this case simply following the recommendations of the vendor (Red Hat) for applying patches to the linux operating system would have virtually eliminated the risk posed by ramen. (Recall that risk is vulnerability times threat. [R=V x T]) A second step is employing a tool such as Bastille for hardening a linux system. Bastille is essentially a step-by-step walkthrough of steps to address known security vulnerabilities in linux systems. A third step in protecting a linux system against attacks such as the ramen worm is to install Tripwire and perform regular file system scans.

Relevant Security Advisories:

- CERT Incident Note IN-2001-01, "Widespread Compromises via 'ramen' Toolkit", January, 18, 2001 (http://www.cert.org/incident_notes/IN-2001-01.html)
- SANS Global Incident Analysis Center, "Ramen Worm", Updated February 15, 2001 (http://www.sans.org/y2k/ramen.htm)
- CERT Advisory CA-2000-17, "Input Validation Problem in rpc.statd", August 18, 2000; last revision September 6, 2000 (http://www.cert.org/advisories/CA-2000-17.html)
- CERT Advisory CA-2000-13, "Two Input Validation Problems in FTPD", July 7, 2000; last revision November 21, 2000 (http://www.cert.org/advisories/CA-2000-13.html)
- CERT Vulnerability Note VU#382365, "LPRng can pass user-supplied input as a format string parameter to syslog() calls", First published December 4, 2000 (http://www.kb.cert.org/vuls/id/382365)
- CERT Vulnerability Note VU#34034, "rpc.statd vulnerable to remote root compromise via format string stack overwrite", First published October 30, 2000 (http://www.kb.cert.org/vuls/id/34043)
- CERT Vulnerability Note VU#29823, "Format string input validation error in wu-ftpd site_exec() function", First published October 2, 2000 (http://www.kb.cert.org/vuls/id/29823)
- SANS Institute resources (IDFAQ), "Knark: Linux Kernel Subversion", Jonathan Clemens, (http://www.sans.org/newlook/resources/IDFAQ/knark.htm)
- CERT Advisory CA-2000-22, "Input Validation Problems in LPRng" (http://www.cert.org/advisories/CA-2000-22.html)

References:

1. Dev Zaborav, "Stopping the Ramen worm:Linux and Unix administrators need to be more vigilant in their security measures", http://www.unixinsider.com/unixinsideronline/swol-02-2001/swol-0202-unixsecurity-dv.html
2. Robert Lemos, "Internet worm squirms into Linux servers", January 17, 2001 http://news.cnet.com/news/0-1003-200-4508359.html
3. William Stearns, url: http://www.securityfocus.com/archive/75/163010
4. Daniel Martin, "Ramen Worm", http://members.home.net/dtmartin24/ramen_worm.txt
5. Max Vision, "Ramen Internet Worm Analysis" http://www.whitehats.com/library/worms/ramen/index.html
6. Ryan Hilton, "Modified Ramen found in the wild", http://www.securityfocus.com/archive/75/163619

Glossary:

Malware

Malware is a term given to software introduced into a system for malicious intent.

Synscan

An extremely fast portscanner checking for socks, proxys, cgi-bins, rfc vulnerabilities and others. It was reengineered to include rcp scanning and implemented named query and subnet options.(http://www.psychoid.lam3rz.de/synscan.html)

Knark

Knark is one of the second generation of a relatively new form of rootkit—a loadable kernel module (LKM) designed to mask the presence of system activity. The author places an explicit disclaimer in the code and readme file, indicating that it is not to be used for illegal activity. However, it is easily used for this purpose, and covert usage has indeed been reported to the author.
(http://www.sans.org/newlook/resources/IDFAQ/knark.htm)

CERT (Computer Emergency Response Team)

An organization set up after the Morris worm in 1988 to coordinate efforts to promote computer security by providing critical information about vulnerabilities and incidents to the computer community at large, as well as other advisories and security information. (www.cert.org)

Bastille

The Bastille Linux Project was started by Jon Lasser, of UMBC, Ben Woodard, at VA Linux systems, and an informal group that met at a SANS 98 Conference. They were later joined by a large group of developers and beta testers, including the maintainer and first author of the Hardening Program, Jay Beale. Bastille is available from Bastille-linux at (http://bastille-linux.sourceforge.net/).

Tripwire

A file system integrity package for unix and Windows NT systems. It monitors a file system for modified, added, or deleted files and reports changes to the owner. Tripwire is also freely available for linux at (http://www.tripwire.com/downloads/).

Related Articles from SANS:

- "The Fundamentals Of Computer HACKING", Ida Mae Boyd, December 3, 2000 http://www.sans.org/infosecFAQ/hackers/fundamentals.htm
- "Knark: Linux Kernel Subversion", Jonathan Clemens, March 26, 2000 http://www.sans.org/newlook/resources/IDFAQ/knark.htm
- "Understanding the Attackers Toolkit", Sunnie Hawkins, January 13, 2001 http://www.sans.org/infosecFAQ/linux/toolkit.htm
- "The Process of Hardening Linux", Chris Koutras, January 11, 2001 http://www.sans.org/infosecFAQ/linux/hardening.htm
- "Tripwire – An Integrity Assessment Tool", Paula McKeehan, January 24, 2001 http://www.sans.org/infosecFAQ/audit/tripwire.htm
- "A Real Vulnerability: Rogue System Libraries and Binaries", Manny D. Peterson, December 14, 2000 http://www.sans.org/infosecFAQ/threats/rogue.htm
- "How Do You Know? Using an integrity checker to verify system binaries.", John F. Rovert, December 9, 2000 http://www.sans.org/infosecFAQ/start/verify.htm

Last Updated: July 24th, 2017

Upcoming Training

SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
Community SANS Omaha SEC401*	Omaha, NE	Aug 14, 2017 - Aug 19, 2017	Community SANS
Community SANS Trenton SEC401	Trenton, NJ	Aug 21, 2017 - Aug 26, 2017	Community SANS
Virginia Beach 2017 - SEC401: Security Essentials Bootcamp Style	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Community SANS Pasadena SEC401 @ NASA	Pasadena, CA	Aug 23, 2017 - Aug 30, 2017	Community SANS
Mentor Session - SEC401	Minneapolis, MN	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
Mentor Session - SEC401	Edmonton, AB	Sep 06, 2017 - Oct 18, 2017	Mentor
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Mentor Session - SEC401	Ventura, CA	Sep 11, 2017 - Oct 12, 2017	Mentor
Community SANS Albany SEC401	Albany, NY	Sep 11, 2017 - Sep 16, 2017	Community SANS
Community SANS Columbia SEC401	Columbia, MD	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Dallas SEC401	Dallas, TX	Sep 18, 2017 - Sep 23, 2017	Community SANS
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Copenhagen 2017	Copenhagen, Denmark	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Boise SEC401	Boise, ID	Sep 25, 2017 - Sep 30, 2017	Community SANS
Baltimore Fall 2017 - SEC401: Security Essentials Bootcamp Style	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS New York SEC401	New York, NY	Sep 25, 2017 - Sep 30, 2017	Community SANS
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Charleston SEC401	Charleston, SC	Oct 02, 2017 - Oct 07, 2017	Community SANS
Mentor Session - SEC401	Arlington, VA	Oct 04, 2017 - Nov 15, 2017	Mentor