PRODAFT

WEAPONIZING RSYNC ODAY VULNERABILITY

EGE B. - TAHA H.

INFO@PRODAFT.COM
WWW.PRODAFT.COM

$WHOAMI

PRODAFT

EGE BALCI – Threat Intelligence Division Manager

@egeblc

/in/egebalci

ege@prodaft.com

/egebalci

PRODAFT

PTI REPORTS

[PYSA] Ransomware Group In-Depth Analysis

The group behind PYSA ransomware has earned notoriety for targeting government agencies, educational institutions, and the healthcare sector. The group is known to carefully research high-value targets before launching its attacks, compromising enterprise systems and forcing organizations to pay large ransoms to restore their data. They are listed...

APRIL 13, 2022 17:55

[CONTI] Ransomware Group In-Depth Analysis

PRODAFT Threat Intelligence (PTI) Team has obtained valuable insights on the inner workings of the Conti ransomware group. The PTI team accessed Conti's infrastructure and identified the real IP addresses of the servers in question. This report provides unprecedented detail into the way the Conti ransomware gang works, how they select their targets...

NOVEMBER 18, 2021 09:57

[SOLARMARKER] In-Depth Analysis Report

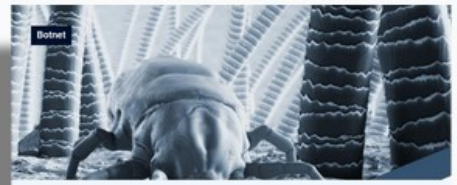
The PRODAFT Threat Intelligence (PTI) team has assembled this report to provide in-depth knowledge about Solarmarker malware and the threat actors behind it. Solarmarker is a multipurpose backdoor first discovered sometime around September 2020. This report brings new, exclusive information about Solarmarker C&C infrastructure to the...

OCTOBER 28, 2021 10:43

[FLUBOT] New Masssive Mobile Malware Ring Tar...

PRODAFT Threat Intelligence (also known as "PTI") Team has just uncovered a massive banking malware operation which primarily targets banking users in Spain. According to our findings, this new operation (referred as "FluBot") sets a new precedent of spreading methods and DGA implementations. Currently, the malware has collected more than ~11 Mil...

MARCH 2, 2021 19:36



[TODDLER] Mobile Banking Botnet Analysis Report

Starting from the second half of 2020, PRODAFT Threat Intelligence ("PTI") team witnessed a rising trend of mobile banking malware attacks against the European countries; primarily targeting customers of banking institutions based in Spain, Germany, Switzerland, and Netherlands. Toddler is considered to be an important example of this trend in term...

JULY 16, 2021 09:01

[LOCKBIT] Behind The Lines Of LockBit R.a.a.S.

PRODAFT Threat Intelligence (also known as "PTI") Team has analyzed critical LockBit ransomware infrastructure and gained in-depth knowledge about the threat actors who operate LockBit ransomware. The PTI team was able to decrypt most of the LockBit victims and uncovered the inner workings of a semi-automated R.a.a.S. platform. Our report includes...

JUNE 16, 2021 10:42

[SILVERFISH] Global Cyber Espionage Campaign ...

The PRODAFT Threat Intelligence (PTI) Team has uncovered a global cyber-espionage campaign, which has strong ties with the SolarWinds attack and the EvilCorp. Victims include dozens of Fortune500 companies, a three letter US organization, and various ministries/departments (from the U.S. and the E.U.) which have previously admitted being breache...

JUNE 17, 2021 22:31

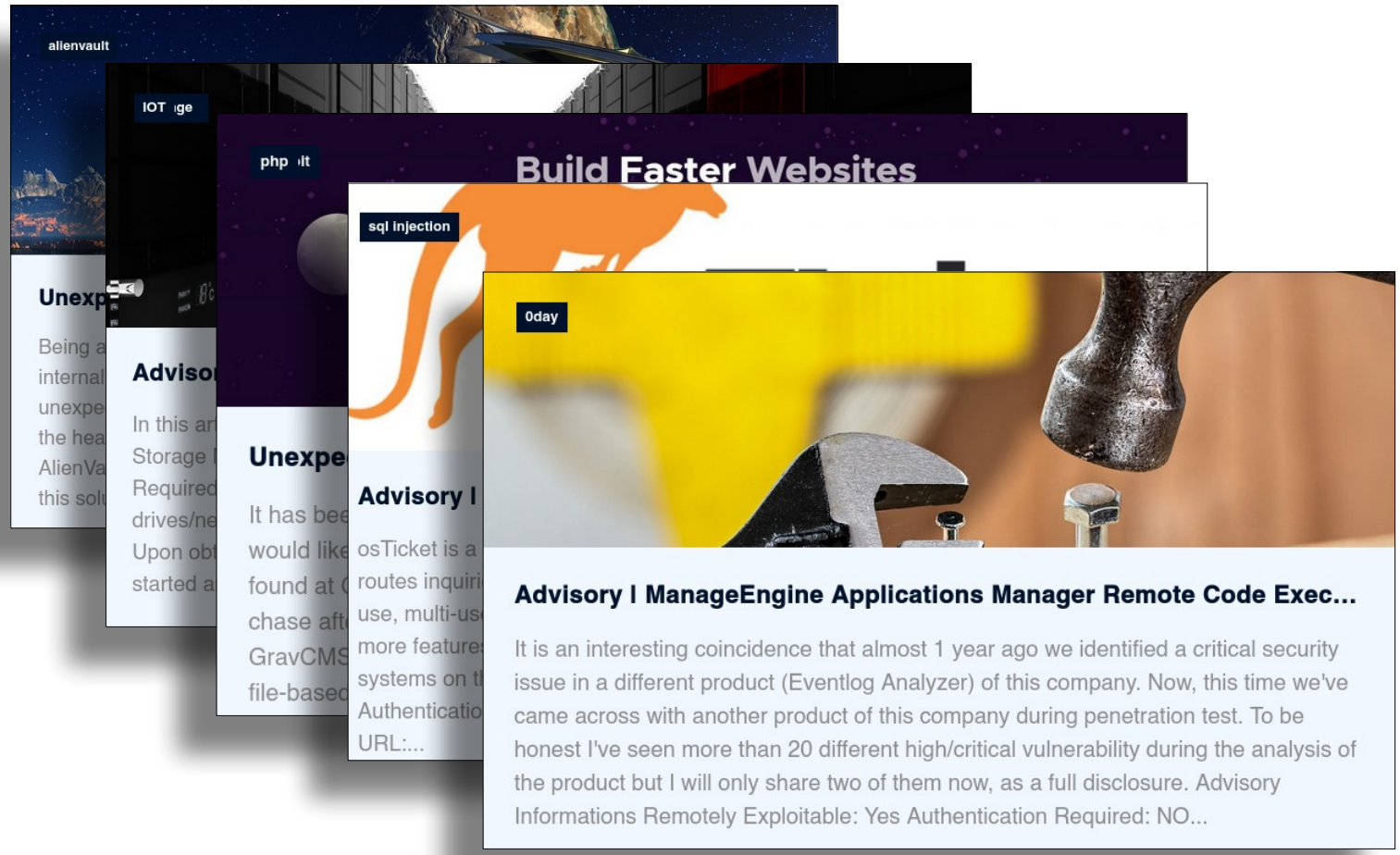
[BRUNHILDA] DaaS Malware Analysis Report

This report is based on an analysis of the Brunhilda dropper service which is detected by The PRODAFT Threat Intelligence (PTI) Team. Brunhilda is a dropper service that utilizes the Google Play Store to distribute various malware. While cybercrimegroups tend to start operating as MaaS businesses, currently there is an upward trend of DaaS (Drop...

NOVEMBER 14, 2020 23:58

PRODAFT

VULNERABILITY RESEARCH



PRODAFT

CURRENT RESEARCH LIST

- Wireguard
- OpenVPN
- ElectrumWallet
- Rsync
- Torify
- Croc
- WasabiWallet
- ToxChat
- Metasploit
- Pidgin
- FileZilla
- RocketChat
- Mattermost
- Onionshare
- MyMonero
- Adium

PRODAFT

RSYNC APPLICATIONS

Program	Operating system			Free software	Description
	Linux	macOS	Windows		
Back In Time	Yes	No	No	Yes	
BackupAssist	No	No	Yes	No	Direct mirror or with history, VSS.
cwRsync	No	No	Yes	No	Based on Cygwin.
Grsync	Yes	Yes	Yes[42]	Yes	Graphical Interface for rsync.
GS RichCopy 360	No	No	Yes[43]	No	Designed only for MS Windows workstations and servers with VSS support.
LuckyBackup	Yes	Yes	Yes	Yes	
rclone	Yes	Yes	Yes	Yes	Inspired by rsync and supports more than 50 cloud storage providers and other high latency storage services. Does not actually use rsync or support rolling checksums and partial file synchronization.
rsnapshot	Yes	Yes	No	Yes	A filesystem snapshot utility based on rsync.
Syncrify	Yes	Yes	Yes	No	Uses rsync over HTTP(S).
tym	Yes	Yes	Yes	Yes	Time rsYnc Machine – backup à la Time Machine – Bash script

PRODAFT

RSYNC APPLICATIONS

This list of topics was included inside the leaked training materials :

- How to build a Cobalt Strike executable
- An introduction to avoiding security products using compiler-based obfuscation techniques.
- A tutorial on using rclone to pull victim data to secure cloud storage accounts on MEGA.
- How to establish a remote connection to the victim's network and gain persistence using AnyDesk and Atera.
- How to connect to hacked networks with RDP using Ngrok secure tunnel.
- A guide to performing SMB brute-force attacks.
- A tutorial on operating system and anonymizing internet traffic via Tor network.
- A how-to on privilege escalation and gaining administrative rights inside a target network.

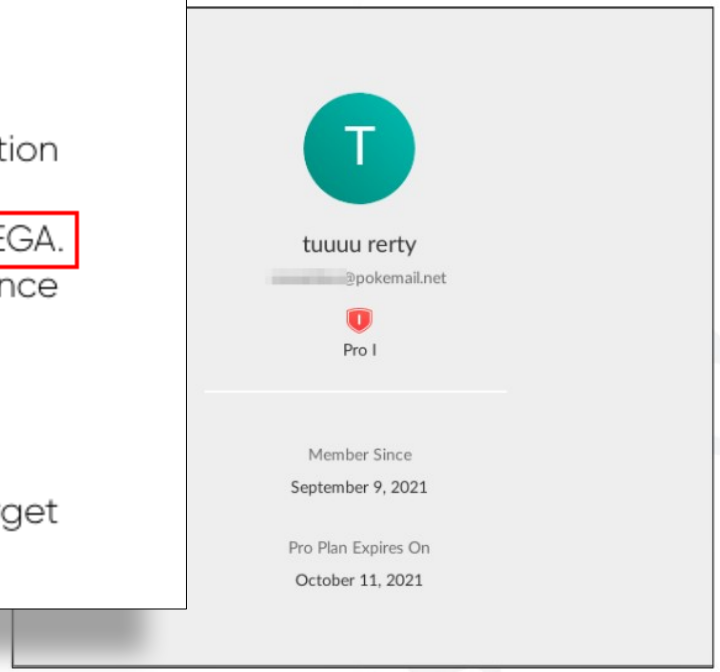


Figure 21. Mega account

Analyzing the MEGA account in above image 21 revealed that the threat actor created the account using the TOR browser and made the purchase using Bitcoin. This transaction occurred while the ransomware attack against the victim was happening. Afterwards, this account was used with a variety of different operating systems before Conti attackers uploaded company data to this account using the rclone 3.2.2 tool.

Table 2 contains the login application, IP address, and access date for the MEGA account discovered by the PTI team.

3.3 Extortion Servers

The victim's stolen data is typically transferred to an extortion server (using rclone or similar data transfer tool) via a proxy network established using Wireguard VPN before deploying the ransomware. We detected several storage servers containing the victim's data during the PTI team's investigation. Interestingly, we identified several folders belonging to the victims who agreed to pay the ransom. This finding is an excellent example that we should not trust ransomware operators. Table 6 shows the excerpt list of the storage servers.

PRODAFT

WHAT IS RSYNC?

WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Current events
Random article
About Wikipedia
Contact us
Donate

Contribute
Help
Learn to edit
Community portal
Recent changes
Upload file

Article Talk

rsync

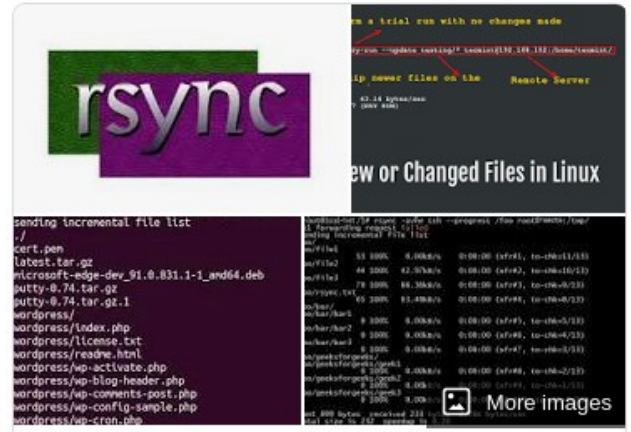
From Wikipedia, the free encyclopedia

rsync is a utility for efficiently transferring and synchronizing files between a computer and a storage drive and across networked computers by comparing the modification times and sizes of files.[9] It is commonly found on Unix-like operating systems and is under the GPL-3.0-or-later license.[5][6][10][11][12][13]

Rsync is written in C as a single threaded application.[14] The rsync algorithm is a type of delta encoding, and is used for minimizing network usage. Zlib may be used for additional data compression,[9] and SSH or stunnel can be used for security. Rsync is the facility typically used for synchronizing software repositories on mirror sites used by package management systems.[15][16]

Rsync is typically used for synchronizing files and directories between two different systems. For example, if the command `rsync local-file user@remote-host:remote-file` is run, rsync will use SSH to connect as `user` to `remote-host`.[17] Once connected, it will invoke the remote host's rsync and then the two programs will determine what parts of the local file need to be transferred so that the remote file matches the local one.

Rsync can also operate in a daemon mode (rsyncd), serving and receiving files in the native rsync protocol (using the "rsync://" syntax).



rsync

Computer application

rsync is a utility for efficiently transferring and synchronizing files between a computer and a storage drive and across networked computers by comparing the modification times and sizes of files. It is commonly found on Unix-like operating systems and is under the GPL-3.0-or-later license. Wikipedia

Repository: github.com/WayneD/rsync/

License: 2007: GPL-3.0-or-later; 2007: GPL-3.0-only; 2007: GPL-2.0-only; 1996: GPL-2.0-or-later

Developer(s): Wayne Davison

Initial release: June 19, 1996; 25 years ago

Original author(s): Andrew Tridgell, Paul Mackerras

Stable release: 3.2.4 / 15 April 2022; 2 months ago

Written in: C

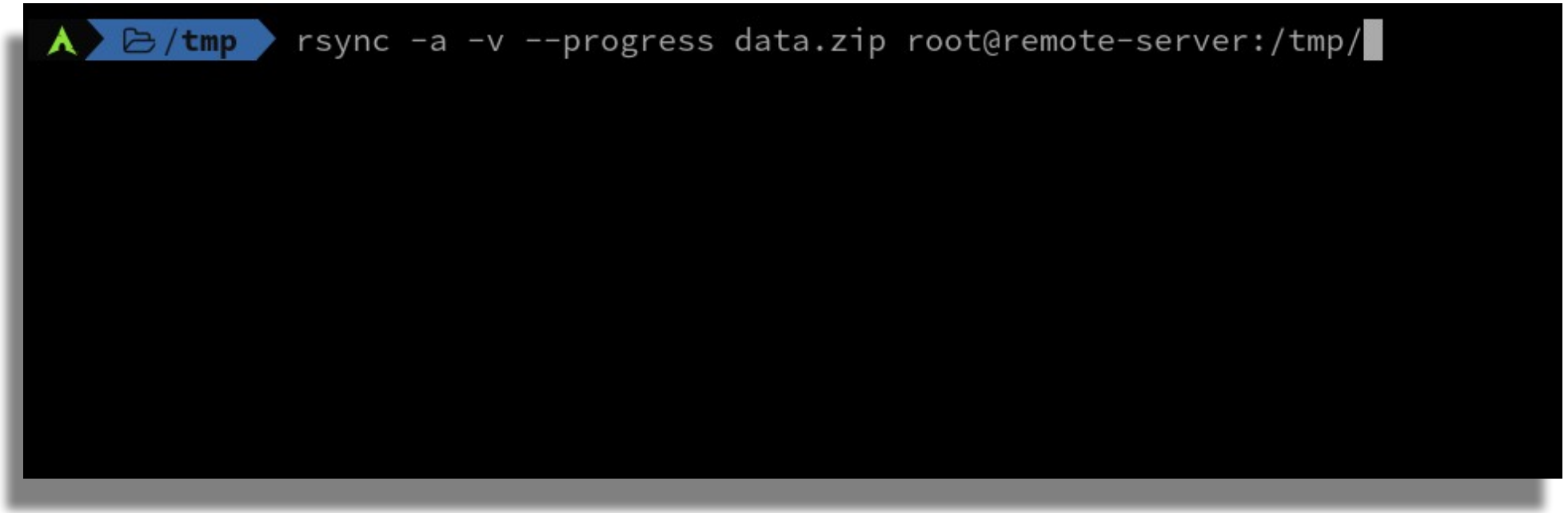
PRODAFT

#112 UBUNTU POPULARITY CONTEST

100	hdparm	2791558	3114	2787310	34	1100	(Stephen Gran)
101	libklibc	2791342	2751	2787432	72	1087	(Maximilian Attems)
102	klibc-utils	2791300	3296	2787001	76	927	(Maximilian Attems)
103	ftp	2791230	95	2789834	12	1289	(Alberto Gonzalez Iniesta)
104	lshw	2791112	284	2789569	21	1238	(Ghe Rivero)
105	iptables	2791026	2672	2787141	43	1170	(Laurence J. Lane)
106	libsqlite3-0	2790813	4208	2783030	60	3515	(Laszlo Boszormenyi)
107	telnet	2790710	115	2789342	14	1239	(Alberto Gonzalez Iniesta)
108	python	2790706	4486	2785703	138	379	(Matthias Klose)
109	pciutils	2790484	1392	2787761	198	1133	(Debian Pciutils Maintainers)
110	libmagic1	2790367	2601	2786894	52	820	(Daniel Baumann)
111	eject	2790162	461	2788496	38	1167	(Frank Lichtenheld)
112	rsync	2790140	854	2788056	54	1176	(Paul Slootman)
113	sysv-rc	2790140	4875	2784603	94	568	(Debian Sysvinit Maintainers)
114	ifupdown	2790123	4730	2784733	75	585	(Anthony Towns)
115	libdrm2	2790037	3264	2769719	71	16983	(Debian X Strike Force)
116	file	2789949	2188	2786982	39	740	(Daniel Baumann)
117	iputils-tracepath	2789919	115	2788648	17	1139	(Noah Meyerhans)
118	libdbus-glib-1-2	2789701	4452	2784527	83	639	(Utopia Maintenance Team)
119	busybox-initramfs	2789326	3295	2785048	78	905	(Debian Install System Team)
120	tcpdump	2789240	81	2787931	33	1195	(Romain Francoise)

PRODAFT

WHAT IS RSYNC?



PRODAFT

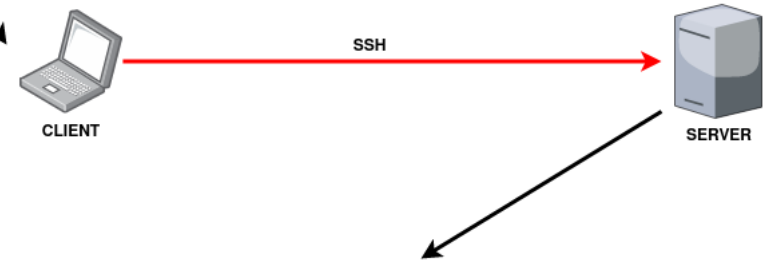
HOW DOES IT WORK?

```
/tmp rsync -a -v root@remote-server:/ ./
```

PRODAFT

HOW DOES IT WORK?

```
/tmp rsync -a -v root@remote-server:/ ./
```

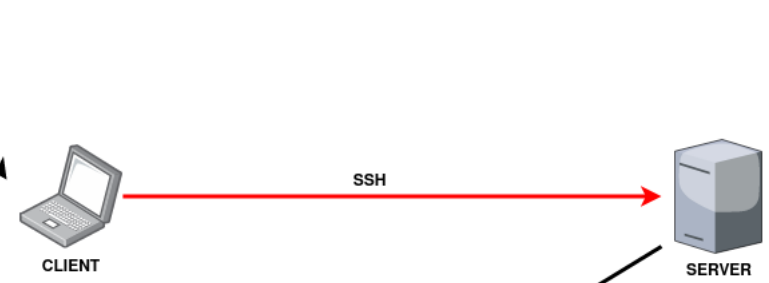



```
root     1924307  5.0  0.0 231404  3408 ?        Ss   10:46   0:00 rsync --server --sender -vlogDtpre.iLsfxCIvu . /
```

PRODAFT

HOW DOES IT WORK?

```
/tmp rsync -a -v root@remote-server:/ ./
```




```
root     1924307  5.0  0.0 231404  3408 ?        Ss   10:46   0:00 rsync --server --sender -vlogDtpre.iLsfxCIvu . /
```

INTERNAL OPTIONS

The options **--server** and **--sender** are used internally by rsync, and should never be typed by a user under normal circumstances. Some awareness of these options may be needed in certain scenarios, such as when setting up a login that can only run an rsync command. For instance, the support directory of the rsync distribution has an example script named rrsync (for restricted rsync) that can be used with a restricted ssh login.

PRODAFT

HOW DOES IT WORK?

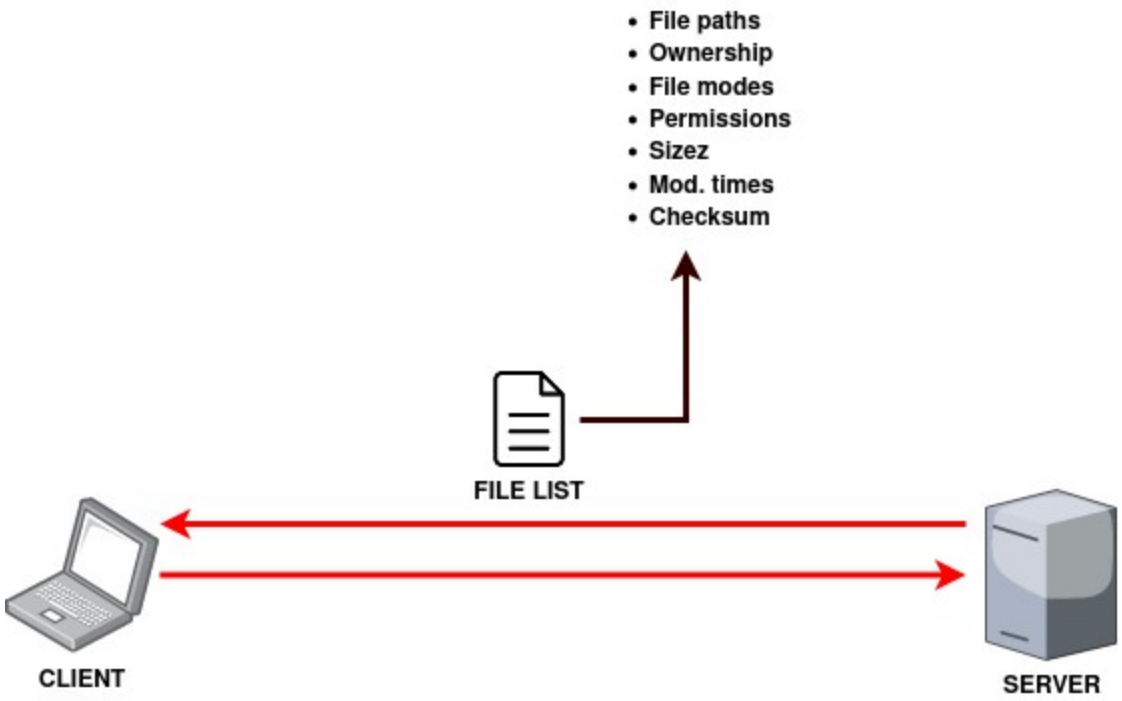
```
/tmp rsync -a -v root@remote-server:/ ./
```


```
root     1924307  5.0  0.0 231404  3408 ?        Ss   10:46   0:00 rsync --server --sender -vlogDtpre.iLsfxCIvu . /
```

```
INTERNAL OPTIONS
       The options --server and --sender are used internally by rsync, and should never be typed by a u
ser under normal circumstances.  Some awareness of these options may be needed in certain scenarios,
       such  as  when  setting up a login that can only run an rsync command.  For instance, the suppor
t directory of the rsync distribution has an example script named rrsync (for restricted rsync) that
       can be used with a restricted ssh login.
```

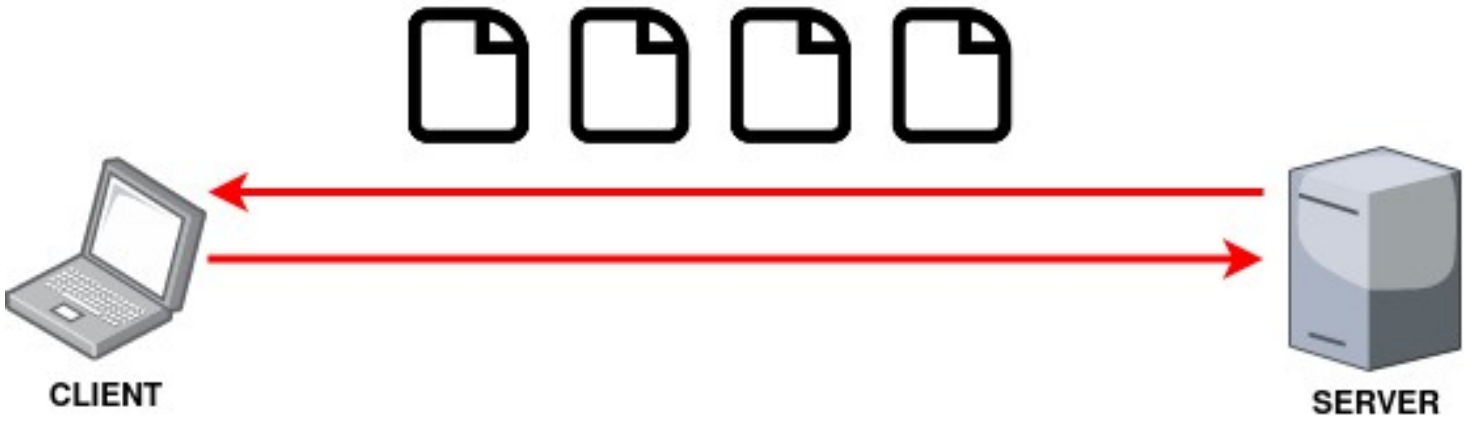
PRODAFT

HOW DOES IT WORK?



PRODAFT

HOW DOES IT WORK?



PRODAFT

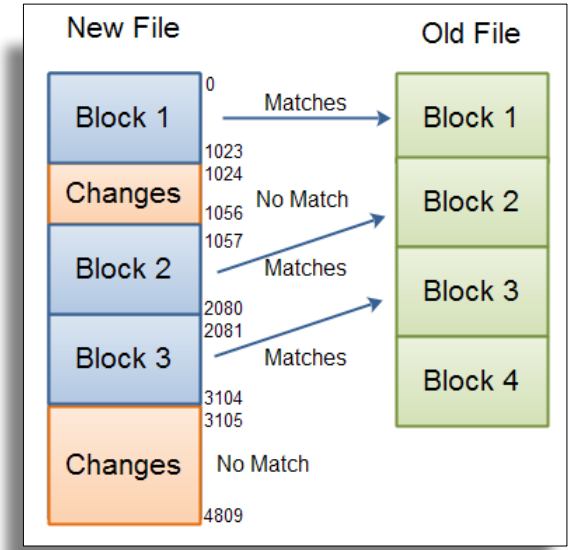
DELTA TRANSFER ALGORITHM

Suppose we have two general purpose computers α and β. Computer α has access to a file A and β has access to file B, where A and B are ``similar''. There is a slow communications link between α and β.

The rsync algorithm consists of the following steps:

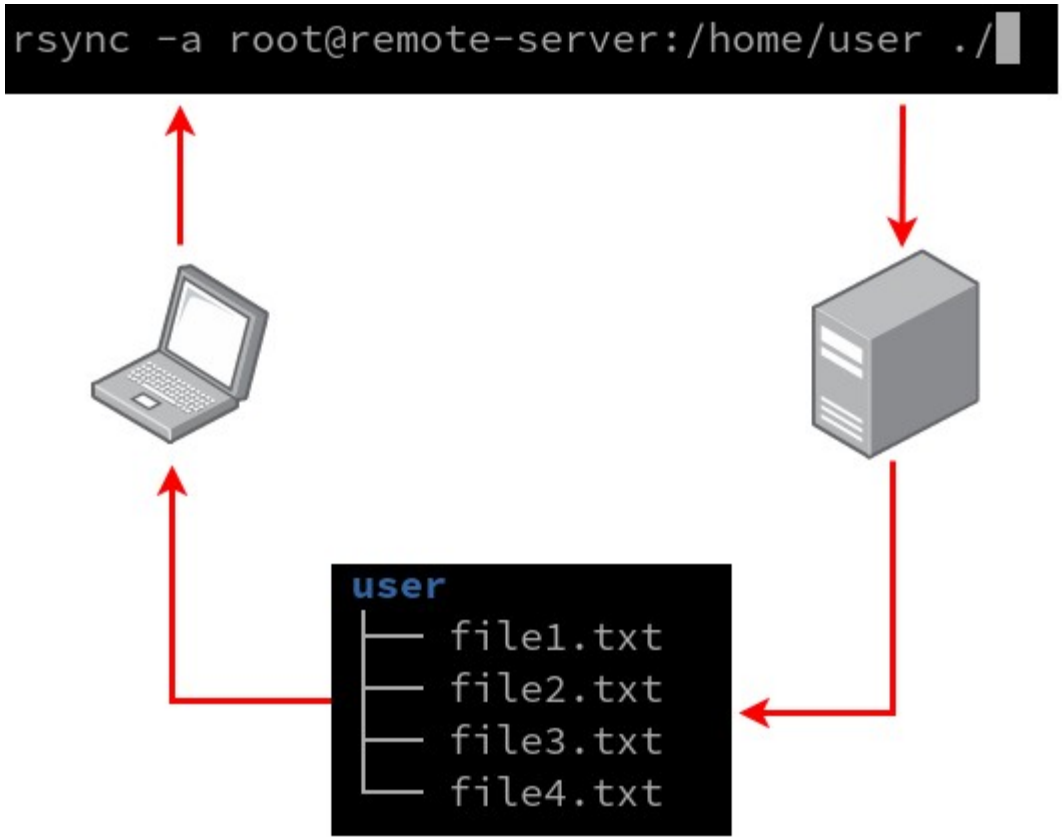
1. β splits the file B into a series of non-overlapping fixed-sized blocks of size S bytes[1]. The last block may be shorter than S bytes.
2. For each of these blocks β calculates two checksums: a weak ``rolling'' 32-bit checksum (described below) and a strong 128-bit MD4 checksum.
3. β sends these checksums to α.
4. α searches through A to find all blocks of length S bytes (at any offset, not just multiples of S) that have the same weak and strong checksum as one of the blocks of B. This can be done in a single pass very quickly using a special property of the rolling checksum described below.
5. α sends β a sequence of instructions for constructing a copy of A. Each instruction is either a reference to a block of B, or literal data. Literal data is sent only for those sections of A which did not match any of the blocks of B.

The end result is that β gets a copy of A, but only the pieces of A that are not found in B (plus a small amount of data for checksums and block indexes) are sent over the link. The algorithm also only requires one round trip, which minimises the impact of the link latency.



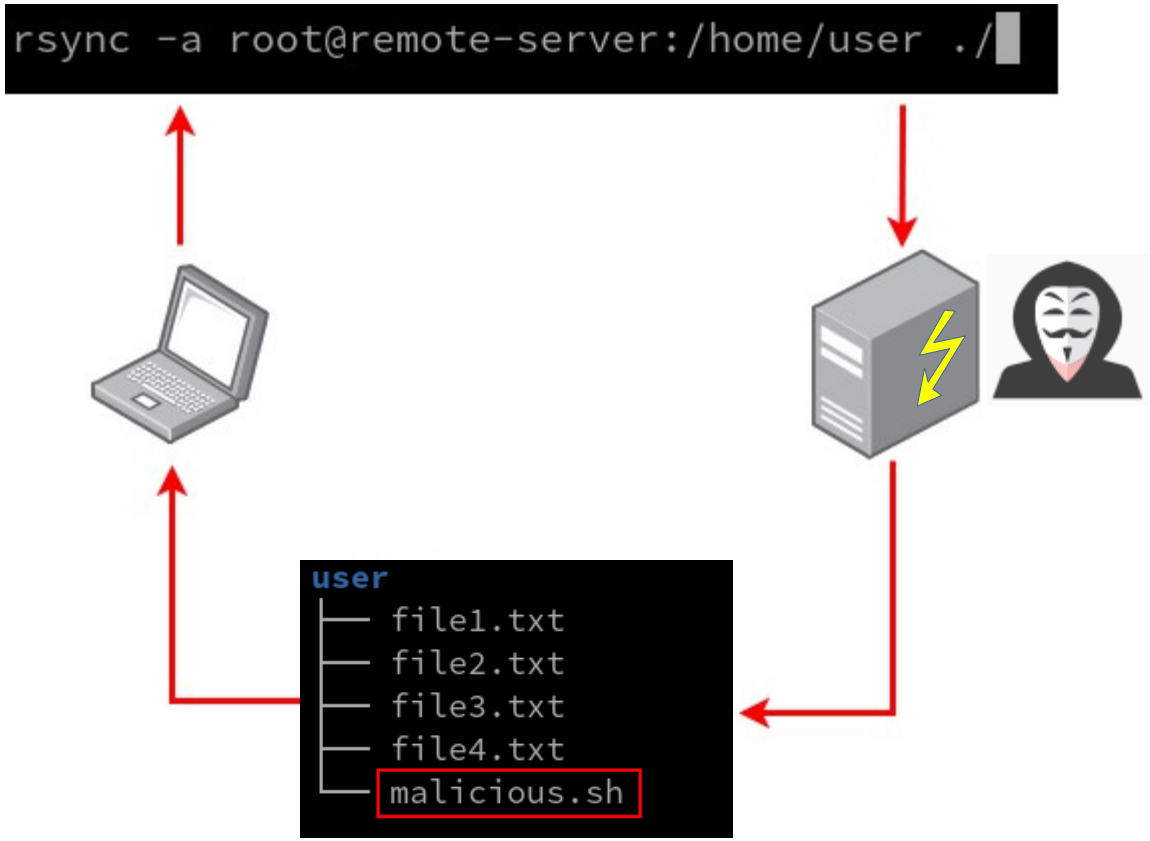
PRODAFT

CLIENT KNOWS NOTHING!



PRODAFT

CLIENT KNOWS NOTHING!



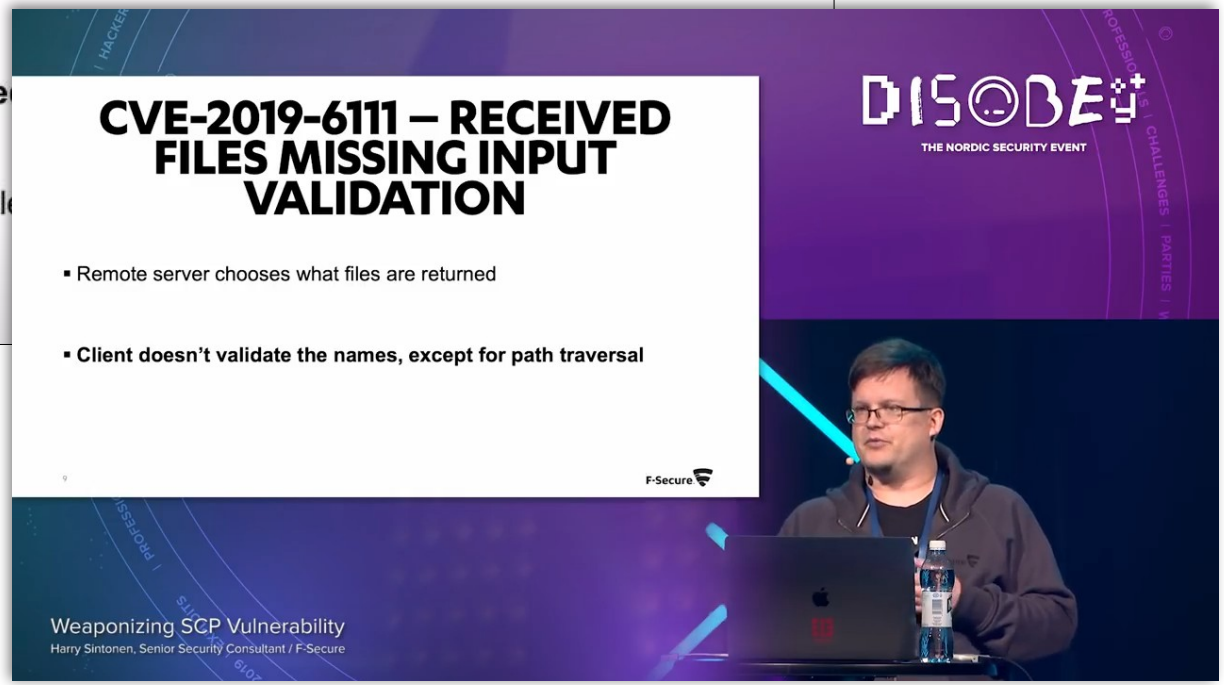
PRODAFT

PRIOR RESEARCH

CVE-2019-6110 – CLIENT OUTPUT SPOOFING

- Remote server can modify the client output at will
- **stderr is forwarde**
- Send "cursor up, cl
  being sent

10



PRODAFT

FILES FOR CODE EXECUTION

- /etc/passwd
- /etc/shadow
- /etc/profile
- /etc/*bash.bashrc*
- /etc/cron.*
- /etc/zsh/*
- /proc/*
- /etc/X11/xinit/xinitrc
- /etc/ld.so.preload
- ~/.bashrc
- ~/.zshrc
- ~/.vimrc
- ~/.profile
- ~/.alias
- ~/.oh-my-zsh
- ~/.ssh/*
- ~/.gitconfig

PRODAFT

FILES FOR CODE EXECUTION

- /etc/passwd
- /etc/shadow
- /etc/profile
- /etc/*bash.bashrc*
- /etc/cron.*
- /etc/zsh/*
- /proc/*
- /etc/X11/xinit/xinitrc
- /etc/ld.so.preload

Requires Permission

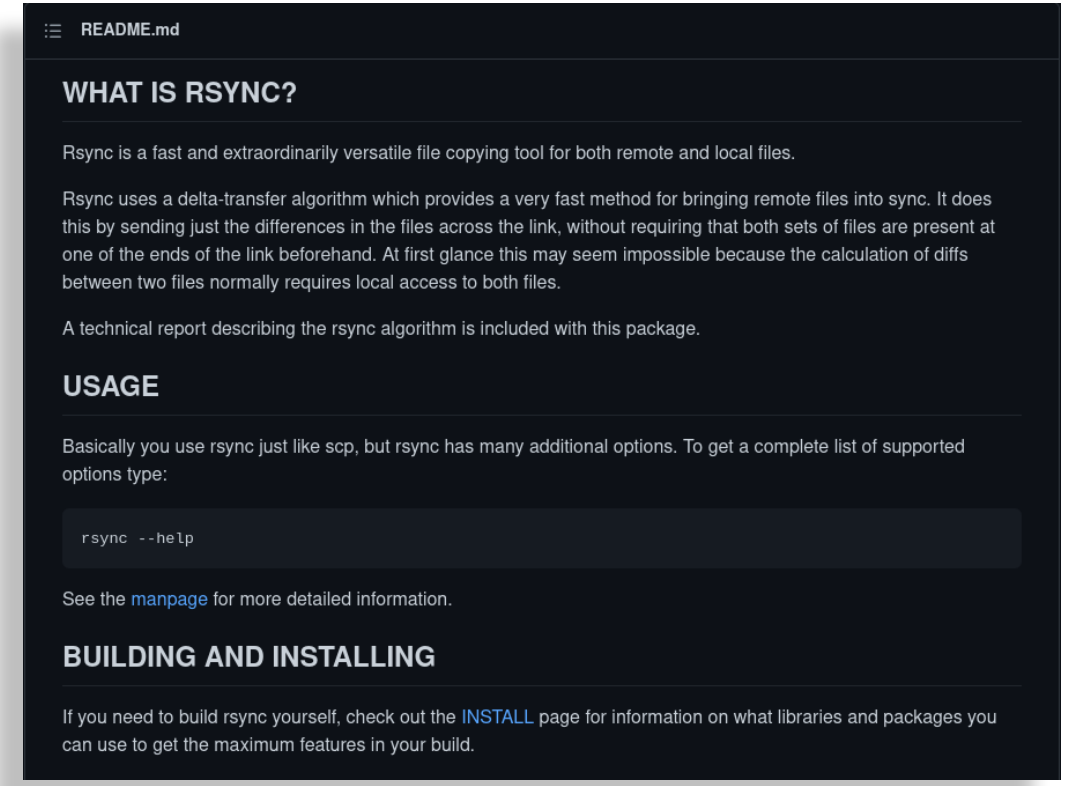
- ~/.bashrc
- ~/.zshrc
- ~/.vimrc
- ~/.profile
- ~/.alias
- ~/.oh-my-zsh
- ~/.ssh/*
- ~/.gitconfig

Accessable to user

PRODAFT

SOURCE CODE ANALYSIS

$ apt source rsync



README.md

WHAT IS RSYNC?

Rsync is a fast and extraordinarily versatile file copying tool for both remote and local files.

Rsync uses a delta-transfer algorithm which provides a very fast method for bringing remote files into sync. It does this by sending just the differences in the files across the link, without requiring that both sets of files are present at one of the ends of the link beforehand. At first glance this may seem impossible because the calculation of diffs between two files normally requires local access to both files.

A technical report describing the rsync algorithm is included with this package.

USAGE

Basically you use rsync just like scp, but rsync has many additional options. To get a complete list of supported options type:

```
rsync --help
```

See the manpage for more detailed information.

BUILDING AND INSTALLING

If you need to build rsync yourself, check out the INSTALL page for information on what libraries and packages you can use to get the maximum features in your build.

PRODAFT

SOURCE CODE ANALYSIS

```
if (argc == 0 && (recurse || xfer_dirs || list_only)) {
        argc = 1;
        argv--;
        argv[0] = ".";
}

flist = send_file_list(f_out,argc,argv);
if (!flist || flist->used == 0) {
        /* Make sure input buffering is off so we can't hang in noop_io_until_death().
        io_end_buffering_in(0);
        /* TODO:  we should really exit in a more controlled manner. */
        exit_cleanup(0);
}
```

PRODAFT

SOURCE CODE ANALYSIS

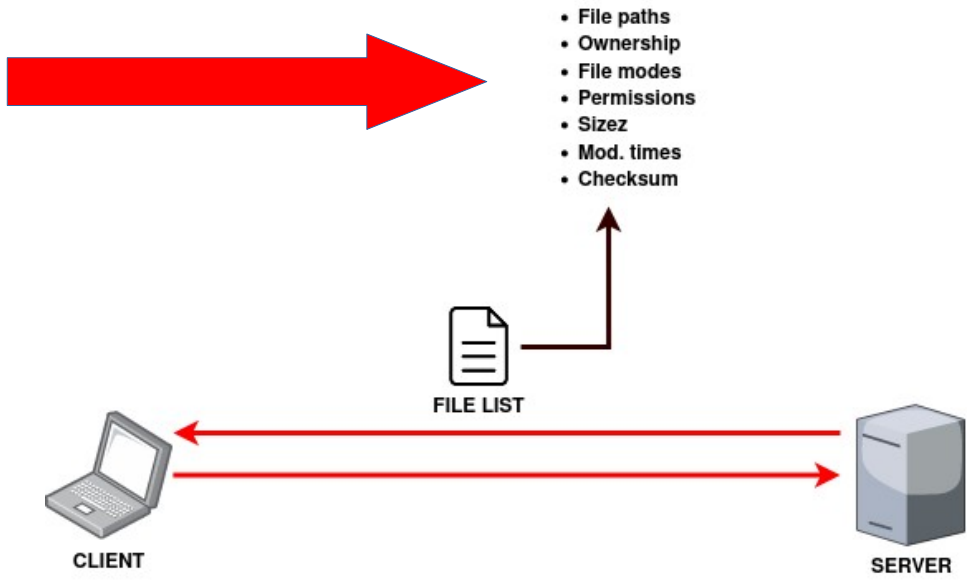
```
struct file_list *send_file_list(int f, int argc, char *argv[])
{
	static const char *lastdir;
	static int lastdir_len = -1;
	int len, dirlen;
	STRUCT_STAT st;
	char *p, *dir;
	struct file_list *flist;
	struct timeval start_tv, end_tv;
	int64 start_write;
	int use_ff_fd = 0;
	int disable_buffering, reenable_multiplex = -1;
	int flags = recurse ? FLAG_CONTENT_DIR : 0;
	int reading_remotely = filesfrom_host != NULL;
	int rl_flags = (reading_remotely ? 0 : RL_DUMP_COMMENTS)
#ifdef ICONV_OPTION
		      | (filesfrom_convert ? RL_CONVERT : 0)
#endif
		      | (eol_nulls || reading_remotely ? RL_EOL_NULLS : 0);
	int implied_dot_dir = 0;

	rprintf(FLOG, "building file list\n");
	if (show_filelist_progress)
		start_filelist_progress("building file list");
	else if (inc_recurse && INFO_GTE(FLIST, 1) && !am_server)
		rprintf(FCLIENT, "sending incremental file list\n");

	start_write = stats.total_written;
	gettimeofday(&start_tv, NULL);

	if (relative_paths && protocol_version >= 30)
		implied_dirs = 1; /* We send flagged implied dirs */

#ifdef SUPPORT_HARD_LINKS
	if (preserve_hard_links && protocol_version >= 30 && !cur_flist)
		init_hard_links();
#endif
```



PRODAFT

SOURCE CODE ANALYSIS

```
                rprintf(FERROR, "Invalid flist flag: %x\n", flags);
                exit_cleanup(RERR_PROTOCOL);
            }
            err = read_varint(f);
            if (!ignore_errors)
                io_error |= err;
            break;
        }
    }

    flist_expand(flist, 1);
    file = recv_file_entry(f, flist, flags);

    if (inc_recurse) {
        static const char empty_dir[] = "\0";
        const char *cur_dir = file->dirname ? file->dirname : empty_dir;
        if (relative_paths && *cur_dir == '/')
            cur_dir++;
        if (cur_dir != good_dirname) {
            const char *d = dir_ndx >= 0 ? f_name(dir_flist->files[dir_ndx], NULL) : empty_dir;
            if (strcmp(cur_dir, d) != 0) {
                rprintf(FERROR
```

PRODAFT

WEAPONIZED SOURCE CODE

```
if (argc == 0 && (recurse || xfer_dirs || list_only)) {
    argc = 1;
    argv--;
    argv[0] = ".";
}

news = inject_files(argv, &argc, &malicious_files);
flist = send_file_list(f_out, argc, news);

if (!flist || flist->used == 0) {
    /* Make sure input buffering is off so we can't hang in noop_io_until_death(). */
    io_end_buffering_in(0);
    /* TODO:  we should really exit in a more controlled manner. */
    exit_cleanup(0);
}
```

PRODAFT

ATTACK SCENARIO

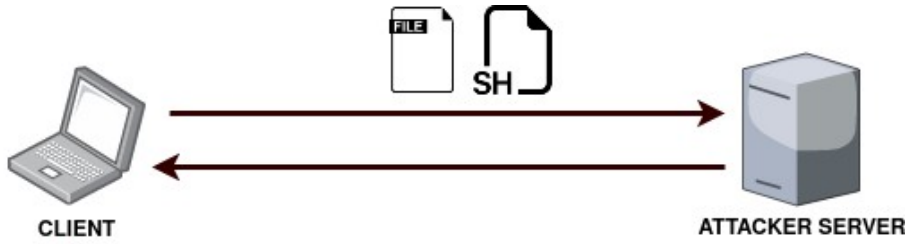
```
/* Make path appear as if a chroot had occurred.  This handles a leading
 * "/" (either removing it or expanding it) and any leading or embedded
 * ".." components that attempt to escape past the module's top dir.
 *
 * If dest is NULL, a buffer is allocated to hold the result.  It is legal
 * to call with the dest and the path (p) pointing to the same buffer, but
 * rootdir will be ignored to avoid expansion of the string.
 *
 * The rootdir string contains a value to use in place of a leading slash.
 * Specify NULL to get the default of "module_dir".
 *
 * The depth var is a count of how many '..'s to allow at the start of the
 * path.
 *
 * We also clean the path in a manner similar to clean_fname() but with a
 * few differences:
 *
 * Turns multiple adjacent slashes into a single slash, gets rid of "." dir
 * elements (INCLUDING a trailing dot dir), PRESERVES a trailing slash, and
 * ALWAYS collapses ".." elements (except for those at the start of the
 * string up to "depth" deep).  If the resulting name would be empty,
 * change it into a ".". */
char *sanitize_path(char *dest, const char *p, const char *rootdir, int depth, int flags)
{
        char *start, *sanp;
        int rlen = 0, drop_dot_dirs = !relative_paths || !(flags & SP_KEEP_DOT_DIRS);

        if (dest != p) {
                int plen = strlen(p); /* the path len INCLUDING any separating slash */
                if (*p == '/') {
                        if (!rootdir)
                                rootdir = module_dir;
                        rlen = strlen(rootdir);
                        depth = 0;
                        p++;
```

PRODAFT

ATTACK SCENARIO

You can’t escape **above** the destination directory.

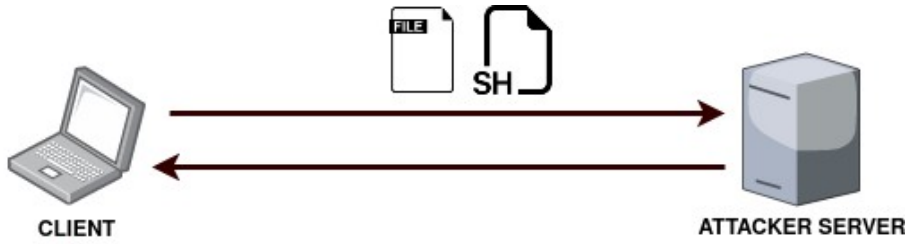



```
home
└── user
    ├── .bash_history
    ├── .bashrc
    ├── Downloads
    │   └── backups
    │       └── backup.tar.gz
    ├── .gitconfig
    ├── .profile
    ├── .vim
    ├── .viminfo
    └── .vimrc
```

PRODAFT

ATTACK SCENARIO

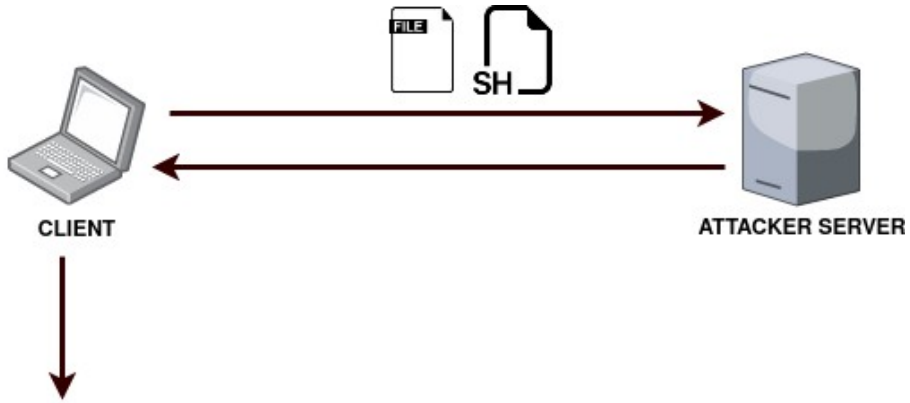
You can’t escape **above** the destination directory.




```
home
└── user
    ├── .bash_history
    ├── .bashrc
    ├── Downloads
    │   └── backups
    │       └── backup.tar.gz
    ├── .gitconfig
    ├── .profile
    ├── .vim
    ├── .viminfo
    └── .vimrc
```

PRODAFT

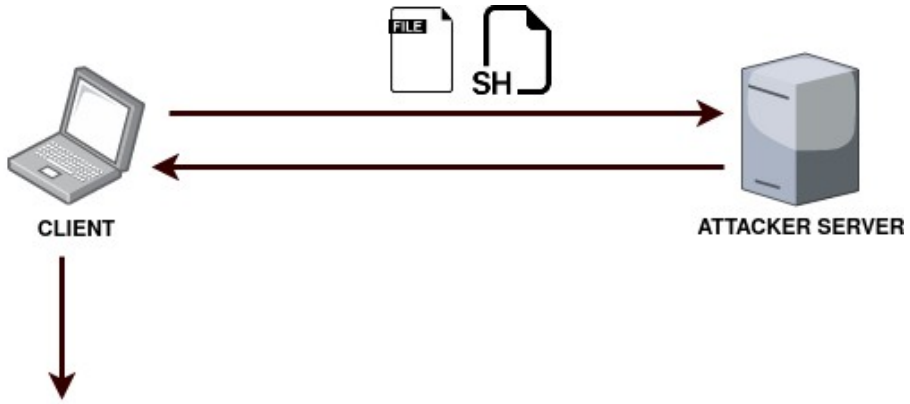
ATTACK SCENARIO

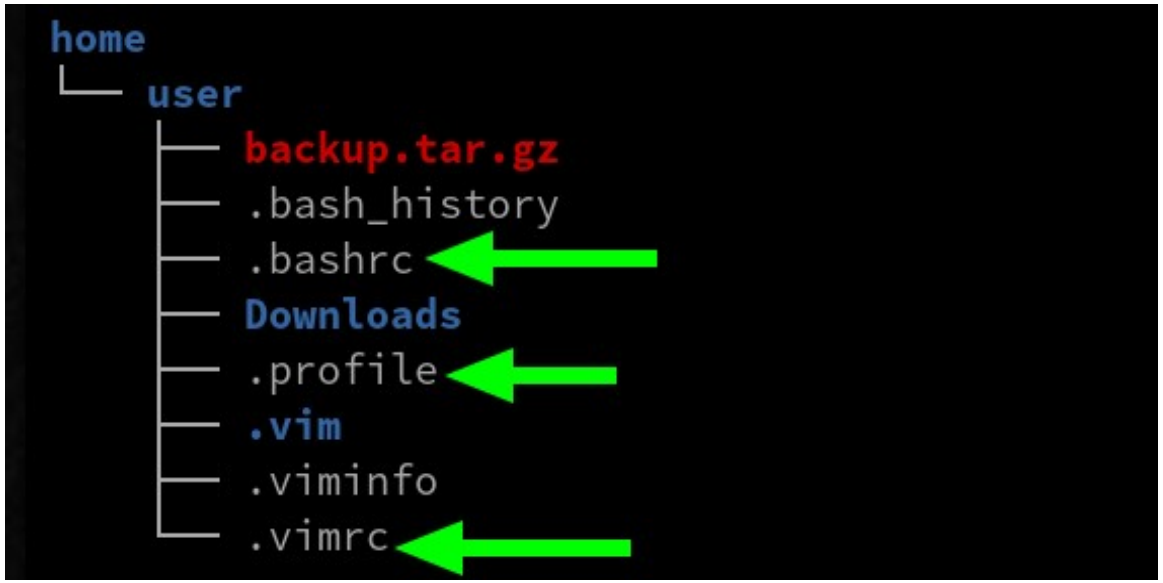



```
home
└── user
    ├── .bash_history
    ├── .bashrc
    ├── Downloads
    │   └── backups
    │       └── backup.tar.gz
    ├── .gitconfig
    ├── .profile
    ├── .vim
    ├── .viminfo
    └── .vimrc
```

You can't escape **above** the destination directory.

ATTACK SCENARIO

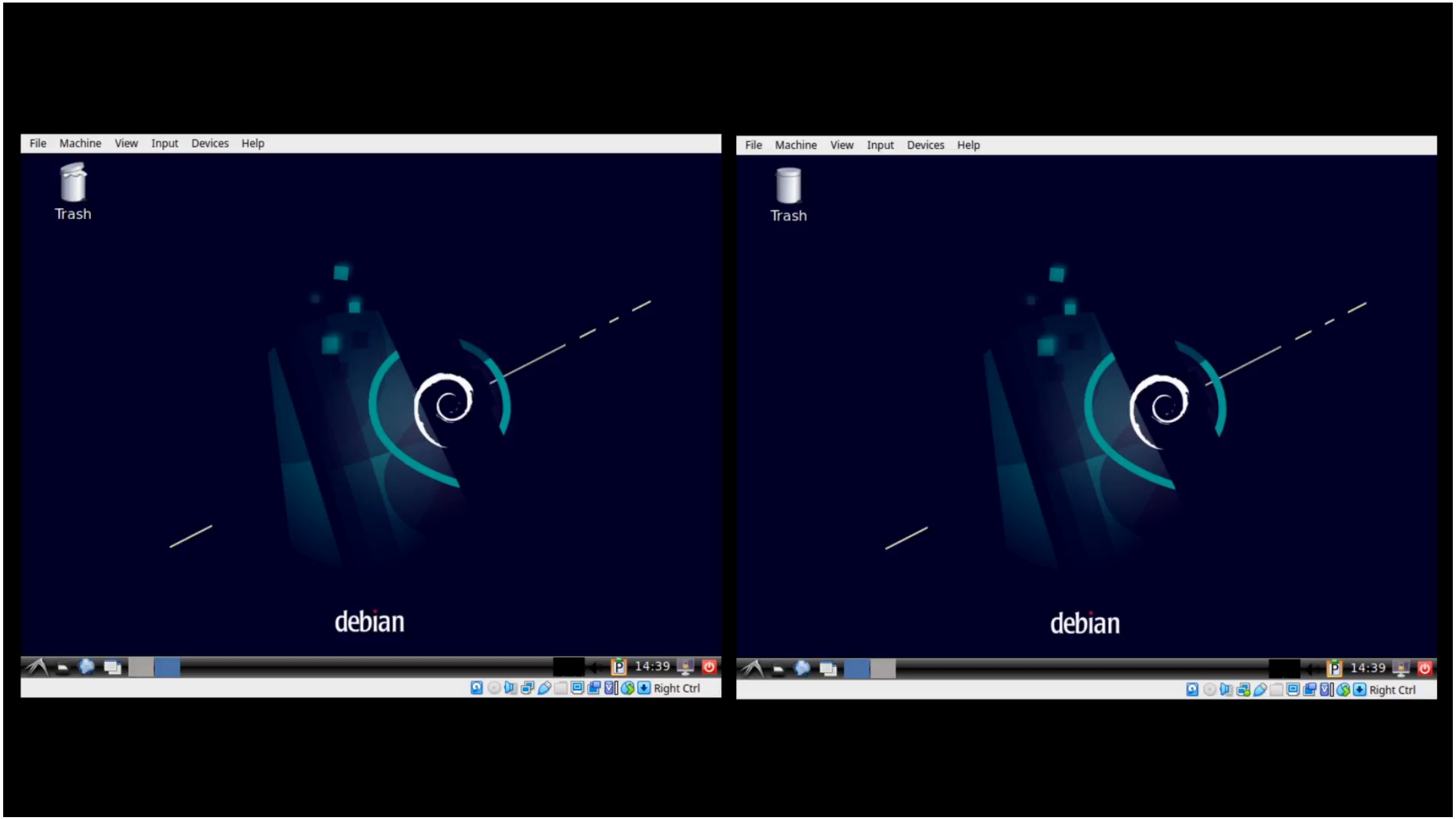




You can overwrite files **in & under** the destination directory.

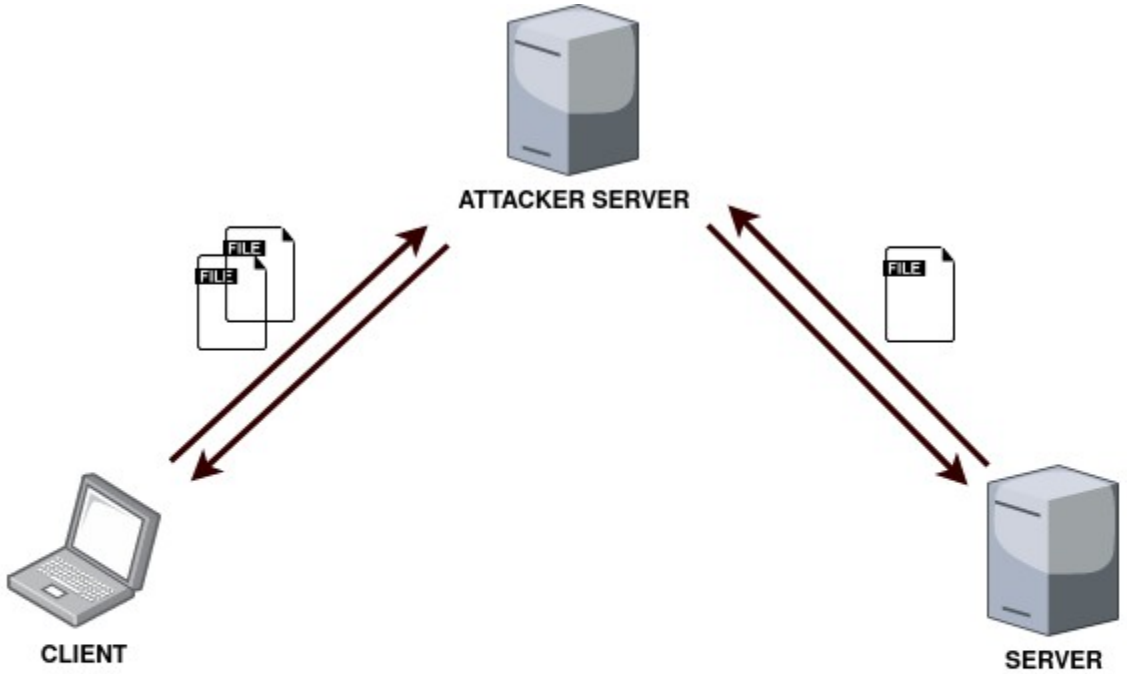
CLIENT FILE REPLACE DEMO

PRODAFT


File Machine View Input Devices Help
Trash
debian
14:39
Right Ctrl
File Machine View Input Devices Help
Trash
debian
14:39
Right Ctrl

PRODAFT

MITM ATTACKS

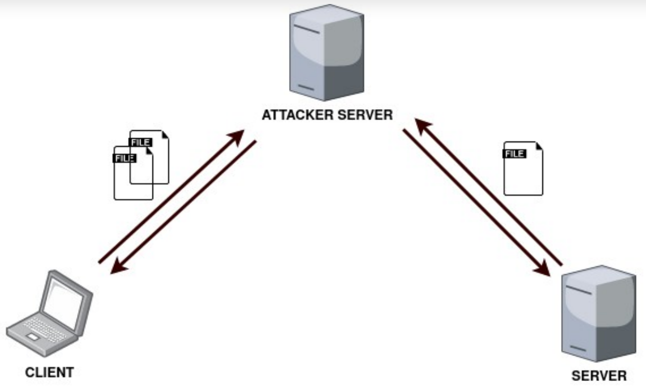



```
/tmp ssh root@remote-server
The authenticity of host 'remote-server (46.101.178.128)' can't be established.
ED25519 key fingerprint is SHA256:LRSBD1RRAjtnyw4XtejZ/2FospS0x1LlIMsmH0ThF4I.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

PRODAFT

MITM ATTACKS

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the ECDSA key sent by the remote host is
SHA256:K/jEKNQCYYOilJxOZc7qAWlu4xu0nW+MD09DfJL7+gc.
Please contact your system administrator.
Add correct host key in /home/sk/.ssh/known_hosts to get rid of this message.
Offending ECDSA key in /home/sk/.ssh/known_hosts:11
  remove with:
  ssh-keygen -f "/home/sk/.ssh/known_hosts" -R "192.168.225.52"
ECDSA host key for 192.168.225.52 has changed and you have requested strict checking.
Host key verification failed.
```



PRODAFT

POSSIBLE GIT TRICKS

What is a git pre-commit hook?

Git hooks in general, are scripts that get run when a specific event occurs in git. They can be written in any language and do anything you like. The only requirement for a git hook script is that it is an executable file.

What events in git can trigger a git hook? Git hook events can trigger on the server-side (where the code is stored remotely -- GitHub, GitLab, etc.) or locally (on your computer). Local git events that can trigger a hook script to run include:

- `pre-commit` (occurs before a git commit is accepted)
- `post-commit` (occurs immediately after a git commit is accepted)
- `post-checkout` (occurs before a git checkout)
- `pre-rebase` (occurs before a git rebase)

```
.git
├── branches
├── config
├── description
├── FETCH_HEAD
├── HEAD
└── hooks
    ├── applypatch-msg.sample
    ├── commit-msg.sample
    ├── fsmonitor-watchman.sample
    ├── post-update.sample
    ├── pre-applypatch.sample
    ├── pre-commit.sample
    ├── pre-merge-commit.sample
    ├── prepare-commit-msg.sample
    ├── pre-push.sample
    ├── pre-rebase.sample
    ├── pre-receive.sample
    ├── push-to-checkout.sample
    └── update.sample
```

PRODAFT

POSSIBLE GIT TRICKS

```
/tmp/test git master git remote add origin git@caz24fh2vtc0000mqeg0gfxtn8hyyyyyb.interact.sh:xxx/yyy.git
/tmp/test git master
```

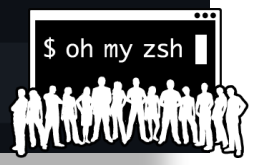
```
/tmp/test git master git remote show origin
The authenticity of host 'caz24fh2vtc0000mqeg0gfxtn8hyyyyyb.interact.sh (46.101.25.250)' can't be established.
ED25519 key fingerprint is SHA256:soCPJOlK+nn+fmnlCDtU3MA8YW1lhZ3Qz96MW4bC2Ms.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

Git auto-fetch

Automatically fetches all changes from all remotes while you are working in a git-initialized directory.

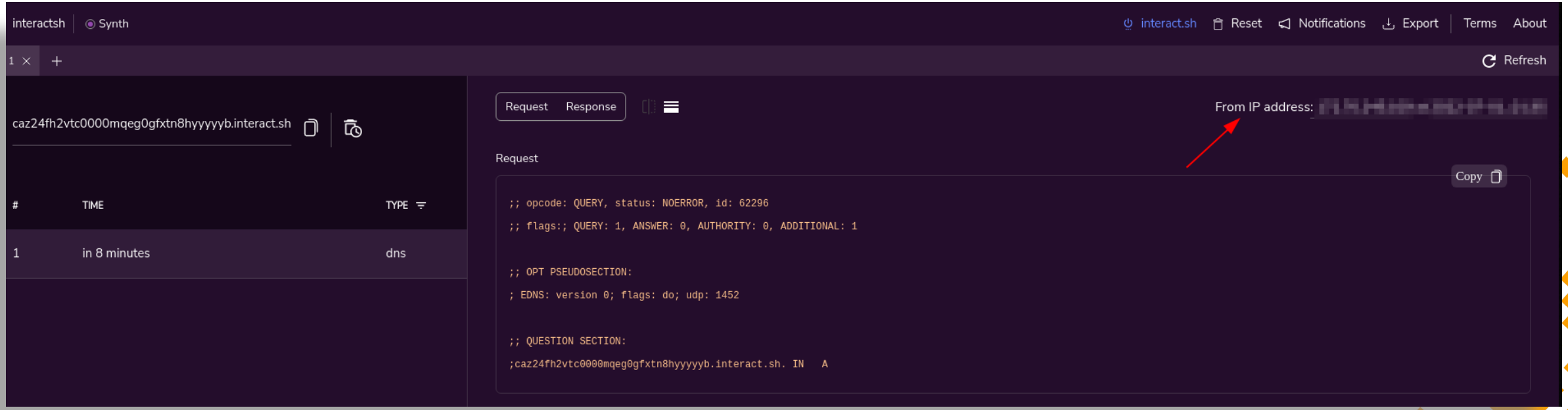
To use it, add `git-auto-fetch` to the plugins array in your zshrc file:

```
plugins=(... git-auto-fetch)
```



PRODAFT

POSSIBLE GIT TRICKS

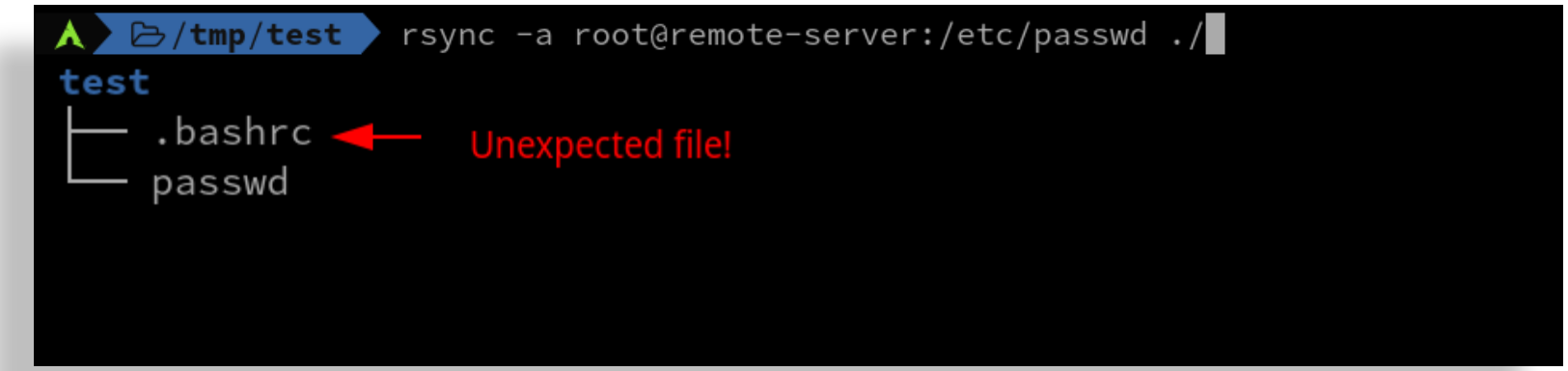


POSSIBLE MITIGATIONS?

Client knows the name and exact number of expected entries inside the local destination directory.

POSSIBLE MITIGATIONS?

Client knows the name and exact number of expected entries inside the local destination directory.



PRODAFT

POSSIBLE MITIGATIONS?

Client knows the name and exact number of expected entries inside the local destination directory.

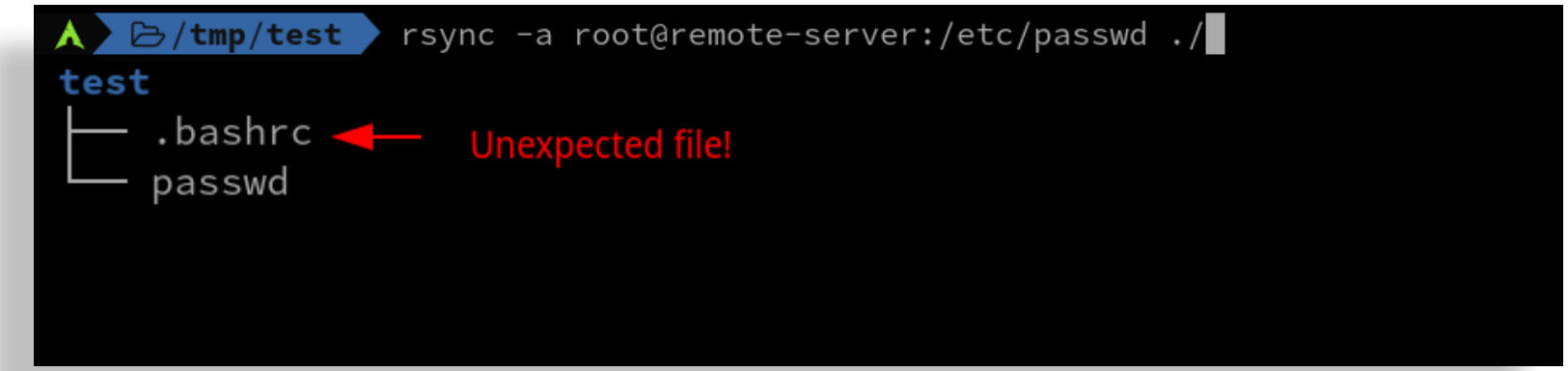
```
/tmp/test rsync -a root@remote-server:/etc/passwd ./
test
├── .bashrc  <-- Unexpected file!
└── passwd
```

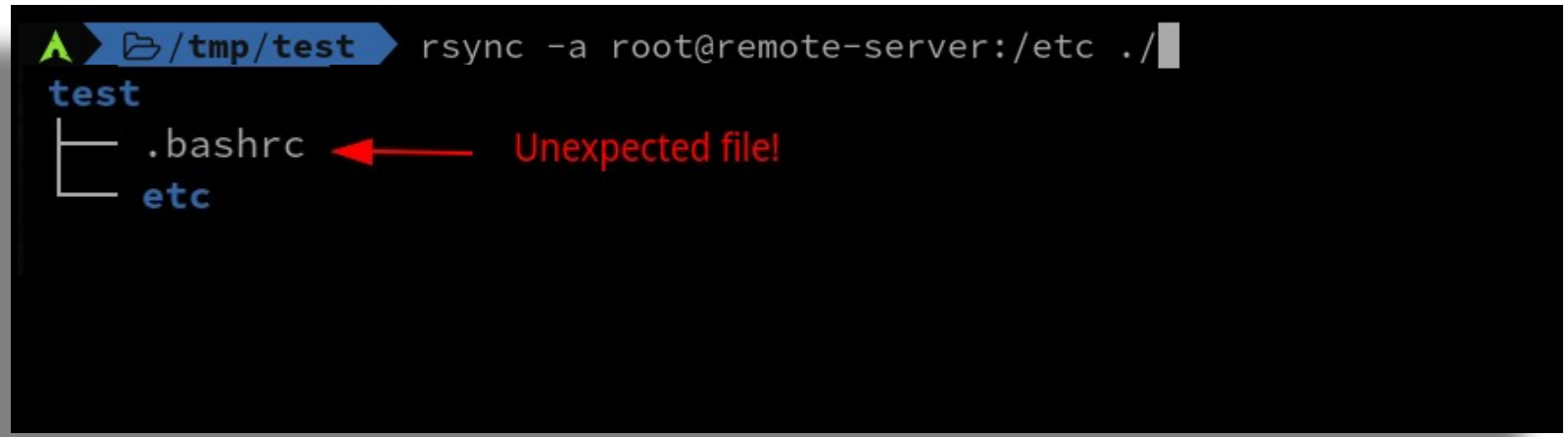
```
/tmp/test rsync -a root@remote-server:/etc ./
```

PRODAFT

POSSIBLE MITIGATIONS?

Client knows the name and exact number of expected entries inside the local destination directory.

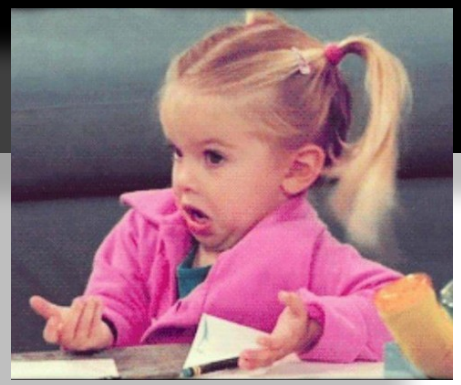




POSSIBLE MITIGATIONS?

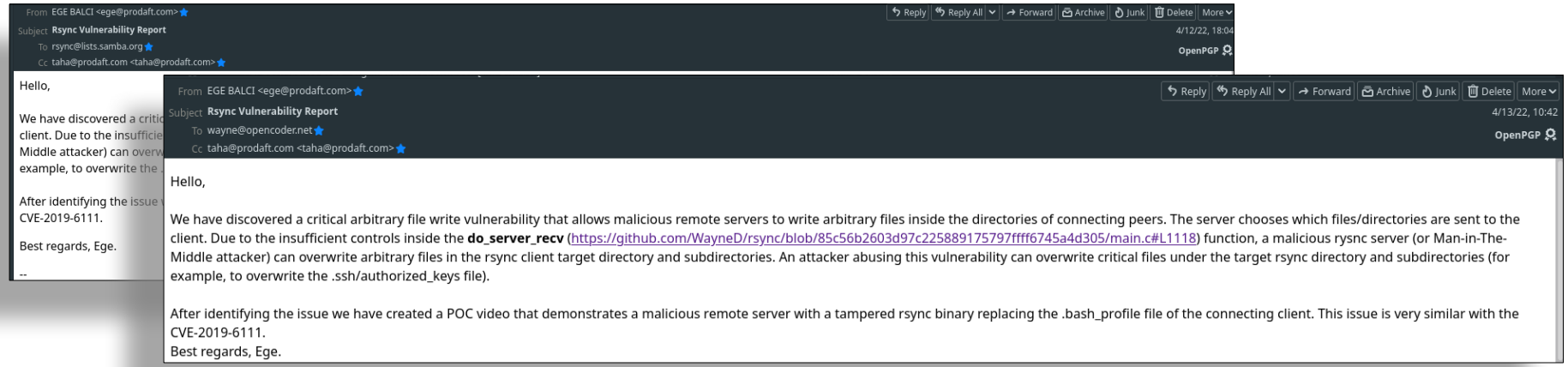
~~Client knows the name and exact number of expected entries inside the local destination directory.~~

```
/tmp/test rsync -a root@remote-server:/etc/ ./
```



PRODAFT

CVE-2022-29154 STILL EXPLOITABLE!



From EGE BALCI <ege@prodaft.com>
Subject **Rsync Vulnerability Report**
To rsync@lists.samba.org
Cc taha@prodaft.com <taha@prodaft.com>
Reply | Reply All | Forward | Archive | Junk | Delete | More
4/12/22, 18:04
OpenPGP

Hello,

We have discovered a critic
client. Due to the insufficie
Middle attacker) can overw
example, to overwrite the

After identifying the issue
CVE-2019-6111.

Best regards, Ege.

--

From EGE BALCI <ege@prodaft.com>
Subject **Rsync Vulnerability Report**
To wayne@opencoder.net
Cc taha@prodaft.com <taha@prodaft.com>
Reply | Reply All | Forward | Archive | Junk | Delete | More
4/13/22, 10:42
OpenPGP

Hello,

We have discovered a critical arbitrary file write vulnerability that allows malicious remote servers to write arbitrary files inside the directories of connecting peers. The server chooses which files/directories are sent to the client. Due to the insufficient controls inside the **do_server_recv** (https://github.com/WayneD/rsync/blob/85c56b2603d97c225889175797ffff6745a4d305/main.c#L1118) function, a malicious rsync server (or Man-in-The-Middle attacker) can overwrite arbitrary files in the rsync client target directory and subdirectories. An attacker abusing this vulnerability can overwrite critical files under the target rsync directory and subdirectories (for example, to overwrite the .ssh/authorized_keys file).

After identifying the issue we have created a POC video that demonstrates a malicious remote server with a tampered rsync binary replacing the .bash_profile file of the connecting client. This issue is very similar with the CVE-2019-6111.
Best regards, Ege.

BUG REPORTS

The bug-tracking web page has full details on bug reporting.

That page contains links to the current bug list, and information on how to do a good job when reporting a bug. You might also like to try searching the Internet for the error message you've received, or looking in the mailing list archives.

To send a bug report, follow the instructions on the bug-tracking page of the web site.

Alternately, email your bug report to rsync@lists.samba.org.


THANK YOU
TWITTER:
@EGEBLC
EGE@PRODAFT.COM
WWW.PRODAFT.COM
PRODAFT