Ministry of Defence of the Netherlands uncovers COATHANGER, a stealthy Chinese FortiGate RAT

This advisory is a joint publication of the MIVD and AIVD, the intelligence and security services of the Netherlands. An accompanying press release was published on the website of the Ministry of Defence of the Netherlands.

1. Overview

- The Ministry of Defence (MOD) of the Netherlands was impacted in 2023 by an intrusion into one of its networks. The effects were limited because of prior network segmentation.
- Incident response uncovered previously unpublished malware, a remote access trojan (RAT) designed specifically for Fortigate appliances. It is used as second-stage malware, and does *not* exploit a new vulnerability. Intelligence services MIVD & AIVD refer to the malware as COATHANGER based on a string present in the code.
- The COATHANGER malware is stealthy and persistent. It hides itself by hooking system calls that could reveal its presence. It survives reboots and firmware upgrades.
- MIVD & AIVD assess with high confidence that the malicious activity was conducted by a state-sponsored actor from the People's Republic of China. This is part of a wider trend of Chinese political espionage against the Netherlands and its allies.
- MIVD & AIVD assess that use of COATHANGER may be relatively targeted. The Chinese threat actor(s) scan for vulnerable edge devices at scale and gain access opportunistically, and likely introduce COATHANGER as a communication channel for select victims.
- Organizations that use FortiGate devices can check if they are affected using the detection methods described in section 4 of this report. Refer to section 5 for advice for incident response.
- Action that organizations can take to prevent future malicious activity: for all internet-facing (edge) devices, install security patches from the vendor as soon as they become available. More preventive steps are described in section 5 of this report.

2. Incident at the Ministry of Defence of the Kingdom of the Netherlands

The Ministry of Defence (MOD) of the Kingdom of the Netherlands was impacted in 2023 by an intrusion into one of its networks. The effects of the intrusion were limited because the victim network was segmented from the wider MOD networks.

The victim network had fewer than 50 users. Its purpose was research and development (R&D) of unclassified projects and collaboration with two third-party research institutes. These organizations have been notified of the incident.

2.1 Attribution

MIVD & AIVD assess with high confidence that the intrusion at the MOD, as well as the development of the malware described in this report, was conducted by a state-sponsored actor from the People's Republic of China.

MIVD & AIVD emphasize that this incident does not stand on its own, but is part of a wider trend of Chinese political espionage against the Netherlands and its allies.

2.2 Actor activity

Chinese threat actors are known to perform wide and opportunistic scanning campaigns for both published (n-day) as well as unpublished (0-day) software vulnerabilities on internet-facing (edge) devices. They do so with a high operational tempo, sometimes abusing vulnerabilities on the day they are published.

For this incident, initial access occurred through exploitation of the CVE-2022-42475 vulnerability for FortiGate devices. The threat actor fired an exploit for this CVE using an obfuscated connection. The second-stage COATHANGER malware described below was then downloaded from another host, possibly a staging server.

Although this incident started with abuse of CVE-2022-42475, the COATHANGER malware could conceivably be used in combination with any present or future software vulnerability in FortiGate devices.

Post compromise, the actor conducted reconnaissance of the R&D network and exfiltrated a list of user accounts from the Active Directory server.

The impact of the intrusion was limited because the victim network was segmented from the wider MOD networks.

3. The COATHANGER malware for FortiGate devices

During an incident response case, the Netherlands' MIVD found a Remote Access Trojan (RAT) present on the FortiGate device that had been used for initial access.

MIVD & AIVD refer to this RAT as COATHANGER. The name is derived from the peculiar phrase that the malware uses to encrypt the configuration on disk: `'She took his coat and hung it up'.`

MIVD notified Fortinet PSIRT of the existence of the malware and cooperated on publication of its blogpost discussing COATHANGER and three other implants.

Please note that second-stage malware like COATHANGER are used in tandem with a vulnerability: the malware is used for persistence to a victim network after the actor gained access. Any published or unpublished vulnerability in a device can be used for initial access to the network, after which COATHANGER is used as a backdoor into the network.

3.1 Characteristics

The COATHANGER malware provides access to compromised FortiGate devices after installation. The implant connects back periodically to a Command & Control server over SSL, providing a BusyBox reverse shell.

Notably, the COATHANGER implant is persistent, recovering after every reboot by injecting a backup of itself in the process responsible for rebooting the system. Moreover, the infection survives firmware upgrades. Even fully patched FortiGate devices may therefore be infected, if they were compromised before the latest patch was applied.

Furthermore, COATHANGER is stealthy: it is hard to detect using default FortiGate CLI commands, because it hides itself by hooking most system calls that could reveal its presence, such as `stat` and `opendir`. It does so by replacing them for any process that is forced to load `preload.so`.

Note that COATHANGER is distinct from BOLDMOVE, another RAT targeting FortiGate devices.

MIVD & AIVD assess that use of COATHANGER may be relatively targeted. The Chinese threat actor(s) scan for vulnerable edge devices at scale and gain access opportunistically, but likely introduce COATHANGER as a communication channel for select victims.

Earlier, the Dutch services found the COATHANGER implant present on a network of a Western international mission, as well as a handful of other victims.

3.2 Behavior

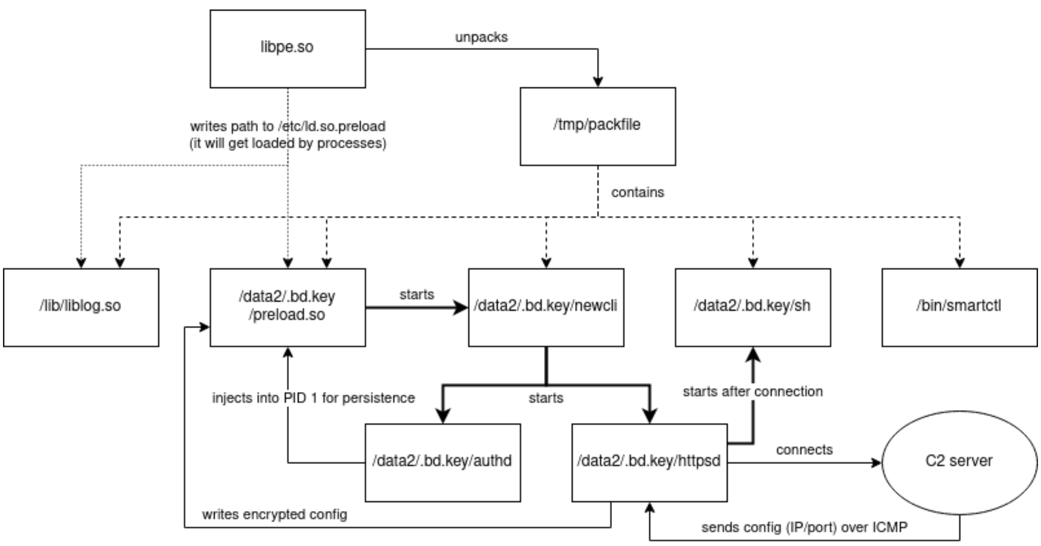
This section describes components of the COATHANGER malware and how they interact.

3.2.1 libpe.so

`libpe.so` is a linux shared object that extracts COATHANGER from a packed file located at `/tmp/packfile`. As far as we are aware, this is the first stage of COATHANGER.

First `libpe.so` will check if the contents of `/proc/[pid]/cmdline` starts with "ls" for its own process. If this is the case it will continue with writing its PID to `/tmp/pepid` and `/sbin/init` and create the `/dev/null` character file. Then it will create the install directory `/data2/.bd.key/`.

From the file `/tmp/packfile` then extracts the files `newcli`, `httpsd`, `preload.so`, `authd`, `sh` and writes them to `/data2/.bd.key/`. It extracts `liblog.so` and



writes it to the /lib/ directory. It extracts smartctl and writes it to the /bin/ directory. **Note that the name of the hidden folder (.bd.key) varies over different samples!**

libpe.so writes "/lib/liblog.so" to /etc/ld.so.preload. It then kills any /bin/miglogd processes and waits for four seconds. Presumably miglogd will restart and be forced to use liblog.so.

It then writes "/data2/.bd.key/preload.so" to /etc/ld.so.preload. This will result in /data2/.bd.key/preload.so being used by any new process.

3.2.2 /bin/smartctl

smartctl is a wrapper of /bin/sh.

3.2.3 /data2/.bd.key/authd

authd is a binary that can inject a library into a running process and then hook an existing function within that process with a new function from that library. It is used by newcli to hook the reboot function in the process with PID 1 with the newreboot function of preload.so.

Without arguments, it returns the following with its process name replacing %s:

```
usage: %s library-to-inject oldfunname newfunname
pid\n
```

3.2.4 /data2/.bd.key/httpsd

httpsd is the main executable of the malware. It can receive and store a config, contact the C2 server and then provide a shell to the actor. It is started by newcli.

3.2.5 Parameter r

If httpsd is started with the parameter 'r', 56 bytes are read from the end of /data2/.bd.key/preload.so where the malware config is stored after initialization.

If a config is present, the ip, port and st value of the config are printed:

```
ip\t %s\nport\t %d\nst\t %d\n
```

3.2.6 Parameter w

If httpsd is started with the parameters 'w <ip> <port> <st>', it will create a new config. If there already is a config present in the last 56 bytes of /data2/.bd.key/preload.so it will be replaced, otherwise it will be appended to the end of this file.

3.2.7 No parameters

If httpsd is started without parameters, it will check if its filename is httpsd, otherwise it will copy itself to /data2/httpsd and execute that file as /bin/httpsd. At the same time /data2/httpsd is set for removal with unlink(). If the filename is indeed httpsd, it will daemonize, set its GID to 90 (in order to be hidden by preload.so) and fork itself.

The child process of the fork will, after three minutes, try and decrypt a config from /data2/.bd.key/preload.so. If successful, it will connect to the C2. Afterwards it will sleep for 4-7 times the st value in seconds before decrypting and connecting again. A default st value of 0 is interpreted as 86400 seconds (24 hours).

3.2.8 C2 communication

Communication to the C2 server is done over a TLS tunnel. COATHANGER first sends the following request to the HTTP GET request to the C2 server:

```
GET / HTTP/2\nHost: www.google.com\n\n
```

Afterwards a pseudo-terminal is opened with openpty. The director side reads and writes to the C2 server, while the performer side starts /data2/.bd.key/sh as /bin/sh. This file is in fact busybox, a tool combining a lot of unix functionality in a single file.

This provides an interface to busybox running on the victims server to the C2 server.

3.2.9 /data2/.bd.key/newcli

newcli will use authd to ensure reboot persistence and starts httpsd. It is started by preload.so.

First it will copy /data2/.bd.key/preload.so to /lib/preload.so. It then forks. The child process will execute /data2/.bd.key/authd as /bin/authd with the arguments "/lib/preload.so reboot newreboot 1". This will inject /lib/preload.so into the process with PID 1 and replace its reboot function with the newreboot function of /lib/preload.so. It then exits.

The parent process will wait for the child to exit. It then removes /lib/preload.so, daemonizes itself and copies /data2/.bd.key/httpsd to /data2/httpsd. It will then start /data2/httpsd as /bin/httpsd and set /data2/httpsd up for removal.

3.2.10 /data2/.bd.key/preload.so

`preload.so` is a linux shared object that handles much of the persistence of COATHANGER, as well as hiding it for any user of the system. It is loaded by `libpe.so`.

First it will check if its process starts with "`/bin/miglogd 1`" or "`ls /tmp/login1`". If this is the case it will write its PID to `/tmp/pid` and start `/data2/.bd.key/newcli`, effectively starting the malware.

Then it will check if the string "`/init`" is part of the process name. This is the case if preload.so is injected into the process with PID 1 (see `newcli`), which starts with "`/sbin/init`". In that case `preload.so` will run a function called backup. This will read the contents of the files `newcli`, `httpsd`, `preload.so`, `authd` and `sh` from the `/data2/.bd.key/` folder and store them in memory.

The function `newreboot` (replacing reboot for the process with PID 1) calls a function called recover. This function writes the string "`/data2/.bd.key/preload.so`" into `/etc/ld.so.preload` and restores all the files read into memory with the backup function back into the `/data2/.bd.key` directory.

This ensures persistence whenever the process with PID 1 will call the function reboot (which happens on a reboot).

`preload.so` will also provide alternatives for a lot of system calls in order to hide the COATHANGER files and running process. It will return with 0 or -1 if the GID of the target is 90 (see `httpsd`) and also when the target contains the string "`ld.so.preload`" or "`.bd.key`". Otherwise the function is called as normal. The following table gives an overview of the replaced functions.

System call	Hides gid 90	Hides "ld.so.preload" and ".bd.key"
__fxstat	X	
__fxstat64	X	
__lxstat	X	X
__lxstat64	X	X
__xstat	X	X
__xstat64	X	X
fdopendir	X	
fstat	X	
fstat64	X	
lstat	X	X
lstat64	X	X
open	X	X
opendir	X	X
readdir	X	X
readdir64	X	X
rmdir	X	X
stat	X	X
stat64	X	X
unlink	X	X
unlinkat	X	X

System functions that are unaffected and used by the malware itself are `access`, `fopen`, `fseek`, `fread` and `fclose`.

3.2.11 /data2/.bd.key/sh

`sh` is a busybox binary. It is used by `httpsd` to provide its functionality to the C2 server, see Section 3.3.2.

3.2.12 /lib/liblog.so

`liblog.so` is a linux shared object that replaces the `read(2)` function. It will call the regular `read(2)` function, but if the target of the read action is `/dev/fgtlog` it will not return any data, effectively disabling reading from `/dev/fgtlog` through `read(2)` by processes that have this library loaded. It is used by `libpe.so`.

3.2.13 /tmp/packfile

`packfile` is a simple container file with the format `[size:4][data:size]` for multiple files. `libpe.so` unpacks all COATHANGER files from this file. It contains the files `newcli`, `httpsd`, `preload.so`, `authd`, `sh`, `liblog.so` and `smartctl`.

3.3 Artifacts

The COATHANGER malware drops the following files.

```
/bin/smartctl or /data/bin/smartctl
/data2/.bd.key/authd
/data2/.bd.key/httpsd
/data2/.bd.key/newcli
/data2/.bd.key/preload.so
/data2/.bd.key/sh
/lib/liblog.so
```

It will in its operation copy some of those files to other locations, but those are removed after use.
Note that the name of the hidden folder (`.bd.key`) varies over different samples!

4. Detection methods

Several methods have been identified to detect COATHANGER implants. These include a YARA-rule, a JA3-hash, different CLI commands, file checksums and a network traffic heuristic.

Note: The described detection methods should be seen as independent methods which differ in approach and reliability. Some methods focus on general indicators of compromise, whereas other methods are tailormade for detecting COATHANGER.

IOCs and additional resources have been made available on Github. Readers can use these tools to check for the presence of COATHANGER, in addition to the below methods.

4.1 YARA

Appendix 1 provides two YARA rules for detection on the COATHANGER samples.

4.2 JA3

The COATHANGER implant communicates to the C2 server using TLS. This TLS connection is fingerprintable using the following JA3-hash:

```
339f6adf54e6076d069dcaac54fddc25
```

This JA3-hash is a fingerprint for connections originating from FortiGate devices that support all encryption and hashing algorithms for doing TLS.

Whereas the far majority of TLS-connections use different parameters, the built-in logging functionality of FortiGate devices seems to make use of identical TLS-parameters, leading to **potential false positive** results from this JA3 hash.

Therefore, traffic should be judged as legitimate (i.e., as false positive indicator of COATHANGER) if it originates from a FortiGate device and has:
- port 541 or 514 as destination port and
- an IP address belonging to Fortinet Inc. or a Fortinet device, such as a FortiManager as destination.

4.3 CLI

With access to the CLI of a FortiGate device, the presence of COATHANGER can be detected in three ways.

Tip: It is recommended to access the CLI using the web interface, instead of using the serial console port.

4.3.1 Deviating modification time

Check if the files `/bin/smartctl` or `/data/bin/smartctl` exist using the following command:

```
fnsysctl ls -la /bin
fnsysctl ls -la /data/bin
```

Inspect the timestamps of `smartctl` and other files in the same directory. If `smartctl` was modified later than the majority of other files or is not a symlink, it is likely that the `smartctl` binary was tampered with. This serves as a first indication that the device may be infected by COATHANGER.

4.3.2 TCP Sockets

The following command shows a list of active TCP sockets (similar to `netstat`):

```
diagnose sys tcpsock
```

Whenever the FortiGate device has internet access and the malware is active (this may take some time after system startup), the outgoing connection will appear in the results. This also displays the process number and name of one of the processes related to the infection, as well as the IP address and listening port of the C2 server.

Note: Whenever the malware has not been able to connect with the C2 server, the TCP socket is not listed in the results. Therefore, the absence of a suspicious entry in the TCP socket list does not indicate that the device has not been infected!

The specific version of COATHANGER that this report describes uses the process name `httpsd` to obfuscate itself.

Therefore, any suspicious outgoing connections to external IP addresses from a process called `httpsd` is a **strong indicator** of the presence of COATHANGER:

```
<device_IP>:<device_port>-><c2_IP>:<c2_port>->state=established  err=0  socktype=1  rma=0  wma=0  fma=0 tma=0 inode=<inode> process=<PID>/httpsd
```

Note that the process name (`httpsd`) may vary over different samples of the malware!

4.3.3 Unusual location in process maps of httpsd

The specific version of COATHANGER that this report describes uses the process name `httpsd` to obfuscate itself.

All active processes can be listed using the following command:

```
fnsysctl ps
```

Running the following command returns all PIDs that are named `httpsd`:

```
diagnose sys process pidof httpsd
```

Running the following command using the retrieved process IDs, yields process information for the processes named `httpsd`.

Tip: The PIDs of process that have suspicious outgoing connections (see Section 4.3.2) can be used as a starting point for conducting this check.

```
diagnose sys process dump <PID>
```

When the process has a GID set to 90, **the device is infected with COATHANGER**.

```
Gid:    90      90      90      90
```

Note that the name of the hidden folder (`.bd.key`) varies over different samples. The process name (`httpsd`) may also vary!

When the process map includes deleted entries linked to `/data2/httpsd` or any entries to `/data2/.bd.key/preload.so`, **the device is infected with COATHANGER**.

```
Maps:
0040.....-0040.....   ....   0000....   b3:..   .....
/data2/httpsd (deleted)
[...]
7f90.....-7f90.....   ....   0000....   b3:..   .....
/data2/.bd.key/preload.so
[...]
```

When entries like the ones displayed above are found in the process map, the full path of the hidden directory where COATHANGER is located is discovered. In this case, COATHANGER is located in `/data2/.bd.key`.

Note that this location is hidden from tools like `fnsysctl`. Therefore, running `fnsysctl ls -la /data2/.bd.key` will result in an error message stating 'No such file or directory'.

FortiGate devices support computing MD5 hashes of files using the following command:

```
diagnose sys csum /data2/.bd.key/httpsd
```

Such checksums can be used to uniquely identify files and verify whether they match the checksums of known malicious COATHANGER binaries (see Section 4.4).

Note however that multiple unique versions of COATHANGER exist. Therefore, the absence of a checksum in lists of known malicious files **does not indicate** that the file is legitimate. Whenever patterns as described above are encountered, one should assume that the device has been compromised.

4.3.4 Deleted /lib/preload.so in process maps of PID1

As described in Section 3.2.3, `preload.so` is injected in PID 1 (`initXXXXXXXXXXX`) to gain persistence. This can be detected by dumping the memory map of process 1 using the following command:

```
diagnose sys process dump 1
```

If the memory map contains deleted entries linked to `/lib/preload.so`, this is a **strong indicator** of the presence of COATHANGER.

```
Maps:
[...]
7f7ff.....-7f7ff.....   ....   0000....   00:..   .....
/lib/preload.so (deleted)
[...]
```

4.4 Checksums

Appendix 2 lists checksums of files related to COATHANGER. Please note that other versions of COATHANGER with different checksums might exist. Therefore fuzzy hashes are provided for the file `httpsd`. These allow for analysis of code similarity with potential (future) variants of COATHANGER.

4.5 Outgoing Connections

It is not expected behavior of an isolated FortiGate device to connect to other domains than those related to Fortinet or domains/IPs listed in the configuration of the device.

Therefore, spontaneous TCP SYN packets initiated by the FortiGate device are a **strong indicator that a FortiGate device has been compromised**. Note that, as described in 3.2.7, it might take several minutes, hours or even days for the beaconing to start.

You can investigate this by isolating a FortiGate device. Attach a second device to the WAN port of the FortiGate device and capture its traffic. Make sure that the second device is assigned the IP address of the configured (WAN) gateway, which causes the FortiGate device to 'believe' it has network access.

5. Advice for mitigation and protection

5.1 Incident response recommendations

If you believe you have been affected by the COATHANGER malware, assume the following:

- The actor may have compromised other hosts reachable via the FortiGate appliance, as well as any devices beyond these.
- There is an increased likelihood of more targeted, hands-on-keyboard activity, i.e., the incident goes beyond opportunistic targeting, especially for longer dwell times.

We recommend you take the following steps:

1. Isolate the affected FortiGate devices immediately.
2. Collect and review relevant logs, data and artifacts from the compromised devices. Extract a forensic image from the device for further detailed analysis of the attack.
3. Consider contacting a third-party specialized in incident response. Assistance in following up on the incident helps ensure that the malicious actor is eradicated from the network. This could avoid a new compromise of the network from the same actor.
4. Report the incident to the NCSC of the Netherlands.

5.2 Removing infections

5.2.1 Wiping

The only currently identified way of removing COATHANGER from an infected FortiGate device involves formatting the device and reinstalling and reconfiguring the device. This method should wipe all traces of COATHANGER. Confirm afterwards that this was successful using the detection methods.

5.2.2 Upgrades don't resolve existing infections

The malware survives a firmware upgrade, meaning that **upgrading firmware is not a solution for removing COATHANGER** from a FortiGate device.

5.3 Preventive measures

To limit risks from adversaries that make use of known vulnerabilities to gain initial access to a victim, it is important to have a robust level of basic information security within your organization. Measures include the hardening, monitoring and response on internet-facing devices.

Specific steps your organization can take to defend against threats similar to COATHANGER:

- Install the most recent security patches from the vendor on internet-facing (edge) devices as soon as they become available. Security patches from the vendor contain fixes for known vulnerabilities. In some cases, an update for a vulnerability is not yet available. Take mitigating measures to lower the risk of such vulnerabilities. Replace software and hardware that are no longer supported by the vendor.
- Implement security best practices from the manufacturer of the device.
- Before adding or enabling features on internet-facing devices, execute a risk analysis for the mandatory and/or needed features before enabling these features on the device. Unnecessary features should be disabled.
- Restrict access to the internet from the internet-facing devices by disabling unnecessary services and ports and disable access to the management interface from the internet.
- Monitor event logs for abnormal activity, such as logons outside of working hours, unusual or unexpected external connections and unauthorized configuration changes on the device. Forward log files and store them in a separate secure network segment to prevent tampering with the log files by a malicious actor. Investigate unusual IP addresses, ports and movement of data.

For more information, read the publication of the NCSC of the Netherlands about the eight steps that every organization should take to prevent incidents.

6. Appendix 1: YARA rules

```
rule COATHANGER_beacon
{
    meta:
        description = "Detects COATHANGER beaconing code (GET / HTTP/2\nHost: www.google.com\n\n)"
        malware = "COATHANGER"
        author = "NLD MIVD - JSCU"
        date = "20240206"
    strings:
        $chunk_1 = {
            48 B8 47 45 54 20 2F 20 48 54
            48 89 45 B0
            48 B8 54 50 2F 32 0A 48 6F 73
            48 89 45 B8
            48 B8 74 3A 20 77 77 77 2E 67
            48 89 45 C0
            48 B8 6F 6F 67 6C 65 2E 63 6F
        }

    condition:
        uint32(0) == 0x464c457f and filesize < 5MB and
        any of them
}


rule COATHANGER_files
{
    meta:
        description = "Detects COATHANGER files by used filenames"
        malware = "COATHANGER"
        author = "NLD MIVD - JSCU"
        date = "20240206"
    strings:
        $1 = "/data2/"
        $2 = "/httpsd"
        $3 = "/preload.so"
        $4 = "/authd"
        $5 = "/tmp/packfile"
        $6 = "/smartctl"
        $7 = "/etc/ld.so.preload"
        $8 = "/newcli"
        $9 = "/bin/busybox"

    condition:
        (uint32(0) == 0x464c457f or uint32(4) == 0x464c457f)
        and filesize < 5MB and 4 of them
}
```

7. Appendix 2: Checksums

Note: fuzzy hashes provided only for file `httpsd`, as it is deemed most likely to be unique.

Filename	Hash	
libpe.so	MD5	6c0adca790235445d07be98cd0f820b5
	SHA1	cd6944926169f56ba78cdf15df6eea44b267bb51
	SHA256	50451bb5b6d68115695a6cb277839a6dd2bad8f70bdb8b79670b18dcde188965
smartctl	MD5	205a8c6049061930490b2482855babcd
	SHA1	77698f3f915e61852b6a79bbd85744d845b112c4
	SHA256	4519baebba73827e2b33f36f835d6cb704755abf1312d8d197be635f4d9ffade
authd	MD5	9124ce7531951456l156d2013fc9d3be
	SHA1	b59d6ec835329ea8982fbbe87bb6b6132514c491
	SHA256	f40c04fb9e2d4157a0bc753925dbc5f757feb77cdd22f90fedf3cc5e095143bc
httpsd	MD5	218a3525ab8e46f7afe252d050a86907
	SHA1	44ed7bf2187c5f7442d8167ef009598dbbed60cb
	SHA256	3ed99aad5922744b6a75ea90ea6ece81ba0d8eb9935aec38b897e44ac3b36c35
	TLSH	T1C7829327B751CA79C099F7B05CAF8AB07836B0F4E722621F2241A6797C647844F0F766
	SSDEEP	192:GTHZecX8f8fUlxblVKmu6Wt9ygq1OtHCj31DM8MC3RUJET+mFG7vSif:kZe18fUl17WlqUtiL1h36iTnYj
newcli	MD5	ab89139e3d47fbaba2da33040da95200
	SHA1	302743eaaa12018647b67b390a270ed98d3219d6
	SHA256	2acc6a2a931db63fe3a875780f00192a60955c9794df68fe0ace0012d309b04f
preload.so	MD5	a62377c01935f366761846b5ceed5a49
	SHA1	c259f0efa8ff0ea798a6a3dda22b8df62627405b
	SHA256	1c437dc9e929669e5a65a1c70afb3107fba471afb9ad35e3848334c9332f2b59
sh	MD5	991461b86aebecfd096dc11ff2a04b4b
	SHA1	dc5074340d4631bbf89adc122e8f1a3ca8d87564
	SHA256	dcd9a5af1c6297ed1a66c851efa305000335d8ade068ba515125a6612f1d5300
liblog.so / ld.so.preload	MD5	e24d14d3e6c6de0ed3db050dd5c935f0
	SHA1	4226726bbdc05cc72e4fbb9bcef4a3b625e8a53b
	SHA256	a79f80158ebbf9e34f6a7ec86b564de2fbee783fe6c1e20eefe2832226e2f827
packfile	MD5	201ee76e996846d5ea3fc03bac3273dd
	SHA1	95c69c5a0751c0c2fc30e9ab5de0af5623b28da3
	SHA256	4591b4fb1c93c27203b36c773597fd3f885338ad7641dcebf8ed2395acdf4a5f
ld.so.preload	MD5	9e333e7b57e5773a68df065477af33ae
	SHA1	97621c409295341808d3697f54dc7b59ee5c42af
	SHA256	80baadc163ab14128a8d3f65de093a400f5ae8e27ec979918cf065cea38af7f8
unpack.so	MD5	fb8bc202b1a6e1661fc6fb72a5b186d6
	SHA1	dbd4726251d710cbaabbd3c345c72df431d7454b
	SHA256	cb284b2e846181a6148059d592c9e6687567433810d1376a8e6f83cb5347c93b
preload.so	MD5	1bc945d6aa5d1b2ad7ccfd3fac82422d
	SHA1	4e365312a06c3da747c7425612ef3fe360b52c05
	SHA256	45fc722b9959384fa46be673c246c9fc94491898a9b1aef6b4a408d81e6fed0f
newcli	MD5	4d0d41064cb24d690226577fe2e52248
	SHA1	ad60b32dce36f45e742db69d3f432c2456abf942
	SHA256	47501cebf0b4ffbf5171d811c1517ea4fce178d925fcc4a3b3057be211add88b
httpsd	MD5	6a1e036b7b7fea19a68b9d05b54dbda6
	SHA1	bd5911d32ddb7893b076ba2dc80b48b3875584df
	SHA256	2a5ea4b166163bf028c4f3c8d4dc1cd6788e991b7300b5ea948e38ec4f6ac8fd
packfile	MD5	dba4fd981120faa360ac5df67d3566aa
	SHA1	cfcae79765e169267445257417cbe06df7d336f4
	SHA256	d2ba18a8b851b87163e42807a3541d17b272b679045d2de00364a718973cb5d4
ld.so.preload	MD5	98a4f08e6617e30ca8f8bd8e5b9177a5
	SHA1	70581b5075d88223cfb830b28e823d2eecd92134
	SHA256	bdf838ca2268c6a33718c3682a0311821365290356 8d66fba362d3ce18b4b4cf

Filename	Hash	
sh	MD5	8d0fffd6b8b127e0972e281c85fbf11c
	SHA1	ddd53cb70798ed530e6e5880bf0182607ca1eecd
	SHA256	218a64bc50f4f82d07c459868b321ec0ef5cf315b012255a129e0bde5cc80320
preload.so	MD5	b37756edc2b756223acacf491be06d48
	SHA1	5edb7c6aa9cc3e441523a32e99e01f83e174949c
	SHA256	7b0709ec1f6e0eda3205a4ebdafbd2484f0590bbfe6ddd7c82d979f0f471e664
newcli	MD5	c6eca7f3a99bff43be8ed5e2a2cd689f
	SHA1	dda35b053f2072fdb30fe21ad3c136b5054817be
	SHA256	7fba5ab17972daa6250f3097c5254c4cf0e5e19889e10c02307f73c7481b4d5e
httpsd	MD5	b9f2ae5082184fdc88b914ab136e54bd
	SHA1	0b209c3b1fd247c56dd003888214bf4ee9a872fc
	SHA256	ce2d55a794bd7f41218796ef4a2cfab9707e8a5e8e971aa02ac8ff908b5f02f6
authd	MD5	92cf72c7e85cc8657644b1e6ca9f8b1e
	SHA1	9211b7cd8d4e8c2416e11a7794a37c31683c2be0
	SHA256	01942a2b1b64446f8bf332004f8f875e66924a8405ac049fd0bb8d03c992fba6