

Viewing TTY Logs with Lighttpd

Enable tty logging in each DShield sensors:

```
guy@picollector: sudo vi /srv/cowrie/cowrie.cfg
```

Search for `ttylog = false` change to: `ttylog = true`

Default `ttylog` → `/srv/cowrie/var/lib/cowrie/tty`

```
$ sudo systemctl restart isc-agent
```

```
$ sudo /srv/dshield/status.sh
```

Install `txt2html` to convert the `ttylogs` to an `html` format to have the ability to access them from ELK

```
$ sudo apt-get install txt2html
```

Install `lighttpd` in the ELK server

```
$ sudo apt-get install lighttpd
```

```
$ sudo systemctl enable lighttpd
```

```
$ sudo systemctl start lighttpd
```

Log location: `/var/www/html`

```
$ ls -la /var/www/html
```

Confirm the webserver is accessible and running then move the default index out of the default directory

Using browser access: `http://elk`

```
$ sudo mv /var/www/html/index.lighttpd.html .
```

Configure TTYLog in Kibana

Management -> Kibana -> Data Views -> `cowrie*`

Edit `TTYLog` and change the IP `192.168.25.231` to your ELK server IP or Name

URL template

```
http://192.168.25.231/{{rawValue}}.html
```

In ELK server, create SSH Shared Keys and don't put a password

```
$ ssh-keygen
```

Copy `id_rsa.pub` over to each sensor(s). Likely the easiest way to copy the public key over might be to scp from DShield sensor

```
$ scp guy@192.168.25.231:/home/guy/.ssh/id_rsa.pub .
```

```
$ cat id_rsa.pub >> .ssh/authorized_keys
```

```
$ rm id_rsa.pub
```

Test SSH between the ELK server to DShield sensor

```
$ ssh -l guy 192.168.25.165 -p 12222
```

Setup DShield Sensor for TTYLogs to HTML Format

There are 2 scripts provided for the sensor

- `replayttylog.sh` → Used when setting up TTYLogs parsing for the first time
- `ttylog.sh` → Runs on a hourly cronjob

If the sensor has been running for a while, un `replayttylog.sh` until it is completed. This could take a while depending on the amount of logs is currently stored. The converted hashes will be stored in the home directory in `~/ttylog`.

```
$ ~/scripts/replayttylog.sh
```

When the script finish to run, it is time to manually transfer the logs to the Elastic server.

Setup Elastic Server for TTYLogs Transfer

The script has completed the log conversion to HTML, the Elastic server should be ready to receive the logs (we assume the webserver is setup as per above) and manually transfer them via scp to ELK:

```
$ sudo scp -P 12222 guy@192.168.25.105:~/ttylog/* /var/www/httpd
```

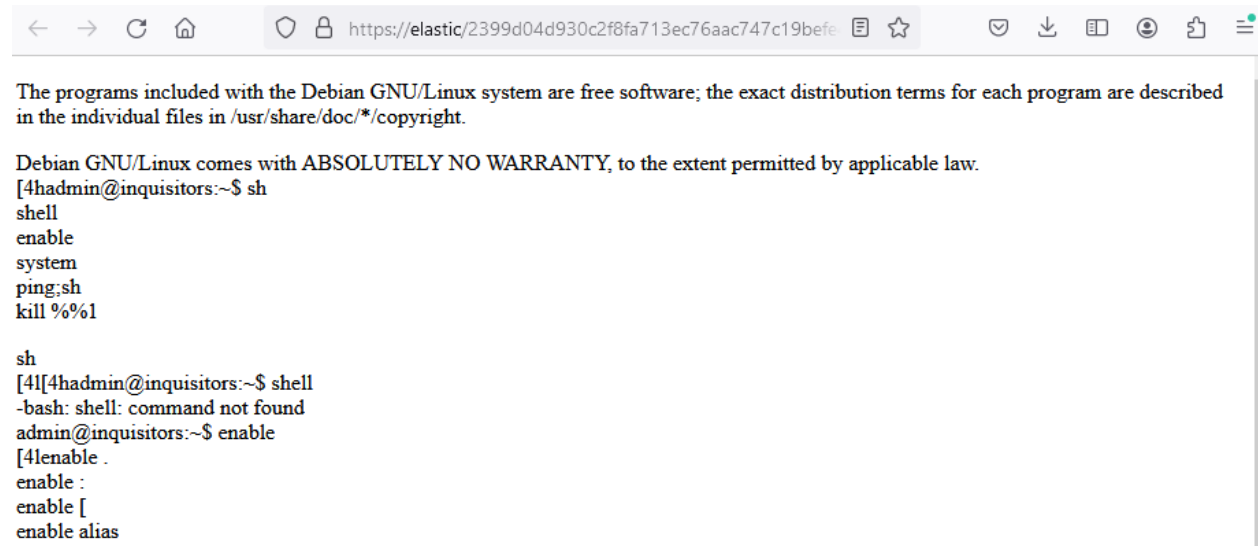
The last step is to enable the hourly cronjob on the server to convert and transfer them over to the web server.

```
$ crontab -e
```

```
1 * * * * /root/scripts/parsing_tty.sh > /dev/null 2>1&
```

This complete converting to HTML the ttylogs and having them available from the ELK server.

This would be what the output looks like:

A screenshot of a web browser window displaying a terminal session log. The browser's address bar shows a URL from an Elastic search endpoint. The log content includes a Debian GNU/Linux copyright notice, a warranty disclaimer, and a series of terminal commands and their outputs, such as 'sh', 'enable', 'system', 'ping:sh', 'kill %1', and 'sh'. The terminal session appears to be a multi-line log of a user's interactions with a system.

```
The programs included with the Debian GNU/Linux system are free software; the exact distribution terms for each program are described
in the individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.
[4]hadmin@inquisitors:~$ sh
shell
enable
system
ping:sh
kill %1

sh
[4]hadmin@inquisitors:~$ sh
-bash: shell: command not found
admin@inquisitors:~$ enable
[4]enable .
enable :
enable [
enable alias
```