

Biohazard Outbreak private server

Outbreakserver community edition on a Raspberry Pi

Foreword and prerequisites

We need some hardware for our system and since I thought this should be as portable as possible I chose a small computer like the Raspberry. Of course you can choose whatever hardware you prefer, but this tutorial is based on Debian Linux. Other flavours like Ubuntu will work with minor tweaks.

I tested this setup on Raspberry 1, 2, 3 and Zero W with differently sized SDHC cards but never less than 4GB. Less might work but hasn't been tested by me.

Also, you'll need to compile custom versions of OpenSSL and Apache2. As long as you stick on the versions in this tutorial, you won't face a lot of problems. But things evolve and at some point this tutorial won't cover all problems you might face. So, you should be able to compile programs under Linux and interpret appearing error messages.

The custom versions are needed because the Playstation 2 is such an old system. Available cipher-suites are all out of date and OpenSSL is usually not compiled with weak ciphers enabled. Apache2 uses mod_ssl and therefore needs to be recompiled.

I once tried to use NginX as webserver but miserably failed. I wasn't able to downgrade HTTP/1.1 protocol to HTTP/1.0 which is a MUST for DNAS emulation. Lighttpd might work but hasn't been tested by me.

Here you see my setup, I attached it to a docking station from my old Motorola phone to have keyboard and screen for the installation. Also, this is a Raspberry PI2 without WLAN and I used a cheap USB-WLAN stick with antenna to it.



Setting up the basic system

First of all get the basic image for the Raspberry, I chose the lite image because of its size and lean environment. You can also use the bigger images if you want to use your PI for more than a private Outbreak server. I for one prefer to buy cards for each project but that's up to you. Also, the bigger images might come with Apache preinstalled which means you need to deactivate that instance before running our own compiled version.

<https://www.raspberrypi.org/downloads/raspbian/>



For bringing the image on your SD-Card follow the installation instructions on <https://www.raspberrypi.org/documentation/installation/installing-images/README.md>

The login for the freshly installed system is **pi** with password **raspberrypi**. Have in mind that y and z might be on other keys if you're for example using a german keyboard.

After logging into your PI we'll do some basic setup with **sudo raspi-config**.

- Change the password to something non-standard or leave it like it is and setup your personal user later
- Setup your network if you're in a WLAN
 - Some fancy hostname like **obcomsrv**
 - Setup WLAN if you want/need to
 - Change locales if needed
 - Enable SSH under **Interfacing Options**
 - Resize your SDHC card to use all available space

Then reboot. At this point you should be able to connect to the internet. Also, you don't need to have a screen and keyboard attached to the PI. All it needs is a tool like PuTTY (<https://www.putty.org/>) or ssh if you're using Linux.

Setting up weak ciphers

Compiling custom OpenSSL

At least for the DNAS server we are forced to use weak ciphers. OpenSSL nowadays isn't compiled with weak cipher activated. Here are the ciphers Playstation 2 is able to use:

```

TLS_DHE_DSS_WITH_RC4_128_SHA (0x000066)
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (0x000016)
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA (0x000013)
TLS_RSA_WITH_3DES_EDE_CBC_SHA (0x00000a)
TLS_RSA_WITH_RC4_128_SHA (0x000005)
TLS_RSA_WITH_RC4_128_MD5 (0x000004)
TLS_DHE_RSA_WITH_DES_CBC_SHA (0x000015)
TLS_DHE_DSS_WITH_DES_CBC_SHA (0x000012)
TLS_RSA_WITH_DES_CBC_SHA (0x000009)
SSL2_DES_192_EDE3_CBC_WITH_MD5 (0x0700c0)
SSL2_RC2_128_CBC_WITH_MD5 (0x030080)
SSL2_RC4_128_WITH_MD5 (0x010080)
SSL2_DES_64_CBC_WITH_MD5 (0x060040)
TLS_DHE_DSS_EXPORT1024_WITH_DES_CBC_SHA (0x000063)
TLS_DHE_DSS_EXPORT1024_WITH_RC4_56_SHA (0x000065)
TLS_RSA_EXPORT1024_WITH_DES_CBC_SHA (0x000062)
TLS_RSA_EXPORT1024_WITH_RC4_56_SHA (0x000064)
TLS_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA (0x000014)
TLS_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA (0x000011)
TLS_RSA_EXPORT_WITH_DES40_CBC_SHA (0x000008)
TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5 (0x000006)
TLS_RSA_EXPORT_WITH_RC4_40_MD5 (0x000003)
SSL2_RC2_128_CBC_EXPORT40_WITH_MD5 (0x040080)
SSL2_RC4_128_EXPORT40_WITH_MD5 (0x020080)
SSL2_RC4_64_WITH_MD5 (0x080080)

```

You can check your OpenSSL installation for the ciphers with **openssl ciphers -V 'ALL'** and see if one of the ciphers is still available. The names are, however, a little bit different to IANA. For a comparison see <https://testssl.sh/openssl-iana.mapping.html>

We need to compile a version of OpenSSL with weak-ciphers enabled. First, install the needed programs, then get the source code (<https://www.openssl.org/source/>) and compile it.

We'll install this version in **/opt** because we don't want to interfere with the already installed openssl.

- wget <https://www.openssl.org/source/openssl-1.0.2g.tar.gz>
- tar xzvf openssl-1.0.2g.tar.gz
- cd openssl-1.0.2g
- ./config --prefix=/opt/openssl-1.0.2 \
- --openssldir=/etc/ssl \
- shared enable-weak-ssl-ciphers \
- enable-ssl3 enable-ssl3-method \
- enable-ssl2 \
- -Wl,-rpath=/opt/openssl-1.0.2/lib
- make
- sudo make install

Let's check the new version for the needed cipher (for example TLS-DHE-DSS-WITH-3DES-EDE-SHA):

```

pi@obcomsrv:~/openssl-1.0.2g $ /opt/openssl-1.0.2/bin/openssl ciphers -V 'ALL' | grep 0x13
0xC0,0x13 - ECDHE-RSA-AES128-SHA    SSLv3 Kx=ECDH     Au=RSA Enc=AES(128) Mac=SHA1
0x00,0x13 - EDH-DSS-DES-CBC3-SHA     SSLv3 Kx=DH      Au=DSS  Enc=3DES(168) Mac=SHA1
pi@obcomsrv:~/openssl-1.0.2g $

```

The last entry here's exactly what we need.

Compiling Apache2

Now that we have custom OpenSSL libraries, we also need to build a webserver against them.

Before we do that we need to edit `/etc/ld.so.conf` (or the appropriate arch file) and add the new SSL lib to it. Otherwise Apache2 won't link against the weak OpenSSL.

- `sudo nano /etc/ld.so.conf.d/arm-linux-gnueabi.hf.conf`

```
GNU nano 2.7.4
# custom OpenSSL
/opt/openssl-1.0.2g/lib

# Multiarch support
/lib/arm-linux-gnueabi.hf
/usr/lib/arm-linux-gnueabi.hf
```

- `sudo ldconfig`

There are some build dependencies we need to sort out before continuing:

- `sudo apt-get update`
- `sudo apt-get install libpcre3 libpcre3-dev`
- `sudo apt-get install libexpat1 libexpat1-dev libxml2 libxml2-dev libxslt1-dev libxslt1.1`

Get the Apache2 from <https://httpd.apache.org/download.cgi#apache24> and compile it. Detailed instructions can be found here: <https://httpd.apache.org/docs/2.4/install.html>

- `wget http://mirror.netcologne.de/apache.org/httpd/httpd-2.4.38.tar.gz`
- `wget http://mirror.funkfreundelandshut.de/apache/apr/apr-1.6.5.tar.gz`
- `wget http://mirror.funkfreundelandshut.de/apache/apr/apr-util-1.6.1.tar.gz`
- `tar xzvf httpd-2.4.38.tar.gz`
- `cd httpd-2.4.38/src/lib/`
- `tar xzvf ~/apr-1.6.5.tar.gz`
- `tar xzvf ~/apr-util-1.6.1.tar.gz`
- `ln -s apr-1.6.5 apr`
- `ln -s apr-util-1.6.1 apr-util`
- `cd ~/httpd-2.4.38/`
- `./configure --prefix=/opt/apache \`
`--with-included-apr \`
`--with-ssl=/opt/openssl-1.0.2 \`
`--enable-ssl`
- `make`
- `sudo make install`
- `nano /opt/apache/bin/envvars`

```
if test "x$LD_LIBRARY_PATH" != "x" ; then
    LD_LIBRARY_PATH="/opt/apache/lib:$LD_LIBRARY_PATH"
else
    LD_LIBRARY_PATH="/opt/apache/lib"
fi
LD_LIBRARY_PATH="/opt/openssl-1.0.2/lib:$LD_LIBRARY_PATH"
export LD_LIBRARY_PATH
#
```

If you want to have Apache added to systemd you need to create a service file and set it up. Finally, we can start the service and see if it works in a browser.

- `sudo nano /etc/systemd/system/apache.service`

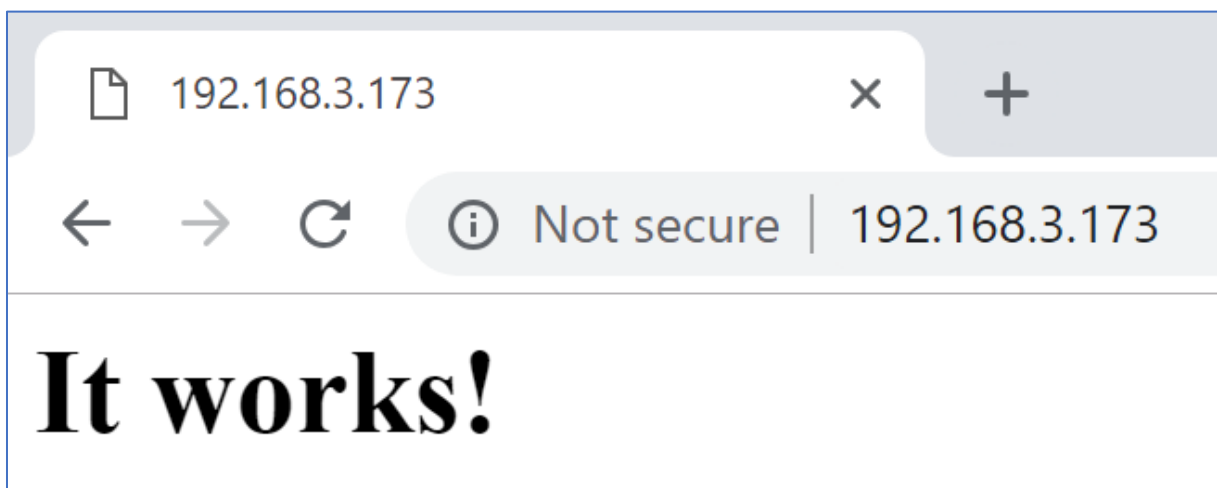
```
GNU nano 2.7.4      Datei: /etc
[Unit]
Description=Apache Server for Outbreak

[Service]
Type=forking
EnvironmentFile=/opt/apache/bin/envvars
PIDFile=/opt/apache/logs/httpd.pid
ExecStart=/opt/apache/bin/apachectl -k start
ExecReload=/opt/apache/bin/apachectl graceful
ExecStop=/opt/apache/bin/apachectl -k stop
KillSignal=SIGCONT
PrivateTmp=true

[Install]
WantedBy=multi-user.target
```

- `sudo systemctl enable apache.service`
- `sudo systemctl start apache`

Quick check in a browser of your choice (no SSL enabled yet):



Setting up DNS

Compared to the other steps, this is quite easy. We'll use dnsmasq for the task of redirecting Playstation 2 to the PI.

- sudo apt-get install dnsmasq dnsutils
- sudo nano /etc/dnsmasq.d/obcomsrv

```
GNU nano 2.7.4                               Datei: /etc/dnsmasq.d/obcomsrv
address=/gatel.jp.dnas.playstation.org/192.168.3.173
address=/www01.kddi-mmbb.jp/192.168.3.173
```

- nano /etc/dnsmasq.conf

```
# Or which to listen on by address (remember to include 127.0.0.1 if
# you use this.)
listen-address=192.168.3.173,127.0.0.1
```

- sudo systemctl restart dnsmasq

Quick check:

```
pi@obcomsrv:~ $ nslookup gatel.jp.dnas.playstation.org 192.168.3.173
Server:                192.168.3.173
Address:               192.168.3.173#53

Name:   gatel.jp.dnas.playstation.org
Address: 192.168.3.173
```

Most probably you need to adjust the IP to the one of your PI. **MAYBE I SHOULD CREATE A WEB USER-INTERFACE FOR THESE THINGS?**

On PS2 side you need to use your PI's IP address as primary DNS. This way queries about the addresses of Sony's DNAS server or KDDI's matchmaking service are answered with the IP address of the PI which will run the needed services in just a few more steps ...

Installing and setting up DNAS

Basically, the DNAS service is build on top of of a webserver running a few PHP scripts. The most problematic issue is to create the needed certificates for SSL. Everything else had been covered by compiling OpenSSI and Apache2 with weak ciphers. Get the zip-file and follow the instructions:

- wget <https://gitlab.com/gh0stl1ne/DNASrep/-/archive/master/DNASrep-master.zip> -O DNASrep.zip
- unzip DNASrep.zip
- sudo mv DNASrep-master/etc/dnas /etc/dnas
- sudo chown -R 0:0 /etc/dnas
- sudo mkdir /var/www
- sudo mv DNASrep-master/www/dnas /var/www/dnas
- sudo chown -R www-data:www-data /var/www/dnas

DNAS emulation is currently done by a few PHP scripts. The lite version of Raspbian stretch doesn't have it preinstalled. Let's adjust the installation:

➤ `sudo apt-get install php7.0-fpm`

Last but definitely not least we'll setup Apache to run a SSL enabled PHP7 webserver with DNAS certificates.

Edit Apache's http.conf with `sudo nano /opt/apache/conf/httpd.conf` You need to enable some modules in order to have SSL and PHP abilities. Search for the following lines and remove the comment sign from the modules **mod_rewrite**, **mod_proxy**, **mod_proxy_fcgi** and **mod_ssl**:

```
LoadModule proxy_module modules/mod_proxy.so
#LoadModule proxy_connect_module modules/mod_proxy_connect.so
#LoadModule proxy_ftp_module modules/mod_proxy_ftp.so
#LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule proxy_fcgi_module modules/mod_proxy_fcgi.so
#LoadModule proxy_scgi_module modules/mod_proxy_scgi.so
#LoadModule proxy_uwsgi_module modules/mod_proxy_uwsgi.so
#LoadModule proxy_fdpass_module modules/mod_proxy_fdpass.so
#LoadModule proxy_wstunnel_module modules/mod_proxy_wstunnel.so
#LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
#LoadModule proxy_balancer_module modules/mod_proxy_balancer.so
#LoadModule proxy_express_module modules/mod_proxy_express.so
#LoadModule proxy_hcheck_module modules/mod_proxy_hcheck.so
#LoadModule session_module modules/mod_session.so
#LoadModule session_cookie_module modules/mod_session_cookie.so
#LoadModule session_dbd_module modules/mod_session_dbd.so
#LoadModule slotmem_shm_module modules/mod_slotmem_shm.so
LoadModule ssl_module modules/mod_ssl.so
```

Also change the user/group from **daemon** to **www-data**:

```
User www-data
Group www-data
```

Then add the server configuration at the end of the file:

```
<IfModule ssl_module>
    Listen 192.168.3.173:443
    SSLEngine on
    # nail it to the securest cipher PS2 understands DHE-RSA-DES-CBC3-SHA
    # check this with openssl
    SSLCipherSuite DHE:!DSS:!AES:!SEED:!CAMELLIA!TLSv1.2

    SSLCertificateFile /etc/dnas/cert-jp.pem
    SSLCertificateKeyFile /etc/dnas/cert-jp-key.pem
    SSLCertificateChainFile /etc/dnas/ca-cert.pem

    ServerName gate1.jp.dnas.playstation.org
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www/dnas
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>

    <Directory "/var/www/dnas">
        Options -Indexes
        Require all granted
    </Directory>

    # rewrite some URLs
    RewriteEngine on
    RewriteRule ^(/.*)/v2\.5_i-connect$ $1/connect.php [PT]
```



```

RewriteRule ^(/.*)/i-connect$ $1/connect.php [PT]
RewriteRule ^(/.*)/v2\.5_d-connect$ $1/connect.php [PT]
RewriteRule ^(/.*)/v2\.5_others$ $1/others.php [PT]
RewriteRule ^(/.*)/others$ $1/others.php [PT]

# send this to php-fpm socket (needs write access!)
<FilesMatch "\.php$">
    SetHandler "proxy:unix:/var/run/php/php7.0-fpm.sock|fcgi://127.0.0.1"
</FilesMatch>

ErrorLog /opt/apache/logs/dnas_error.log
# Possible values include: debug, info, notice, warn, error, crit, alert, emerg.
LogLevel warn
CustomLog /opt/apache/logs/dnas_access.log combined

<FilesMatch "\.(cgi|shtml|phtml|php)$">
    SSLOptions +StdEnvVars
</FilesMatch>

<Directory /usr/lib/cgi-bin>
    SSLOptions +StdEnvVars
</Directory>

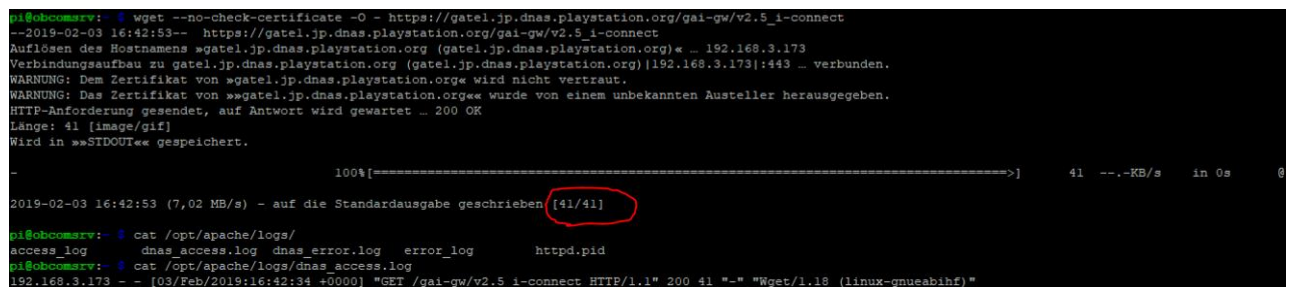
# we need to downgrade protocol for the DNAS browser
BrowserMatch "open sesame asdfjkl" \
    nokeepalive ssl-unclean-shutdown \
    downgrade-1.0 force-response-1.0
</IfModule>

```

Finally you need to restart Apache and voila! Your personal instance of a DNAS server is up and running.

Quick check:

➤ `wget --no-check-certificate -O - https://gate1.jp.dnas.playstation.org/gai-gw/v2.5_i-connect`



```

pi@bobcomrv: ~$ wget --no-check-certificate -O - https://gate1.jp.dnas.playstation.org/gai-gw/v2.5_i-connect
--2019-02-03 16:42:53-- https://gate1.jp.dnas.playstation.org/gai-gw/v2.5_i-connect
Auflösen des Hostnamens »gate1.jp.dnas.playstation.org (gate1.jp.dnas.playstation.org)« _ 192.168.3.173
Verbindungsaufbau zu gate1.jp.dnas.playstation.org (gate1.jp.dnas.playstation.org)|192.168.3.173|:443 _ verbunden.
WARNUNG: Dem Zertifikat von »gate1.jp.dnas.playstation.org« wird nicht vertraut.
WARNUNG: Das Zertifikat von »gate1.jp.dnas.playstation.org« wurde von einem unbekannten Aussteller herausgegeben.
HTTP-Anforderung gesendet, auf Antwort wird gewartet _ 200 OK
Länge: 41 [image/gif]
Wird in »STDOUT« gespeichert.

100%[=====] 41 --KB/s in 0s 0
2019-02-03 16:42:53 (7,02 MB/s) - auf die Standardausgabe geschrieben [41/41]

pi@bobcomrv: ~$ cat /opt/apache/logs/
access_log      dnas_access.log dnas_error.log error_log      httpd.pid
pi@bobcomrv: ~$ cat /opt/apache/logs/dnas_access.log
192.168.3.173 - - [03/Feb/2019:16:42:34 +0000] "GET /gai-gw/v2.5_i-connect HTTP/1.1" 200 41 "-" "Wget/1.18 (linux-gnueabi)"

```

The number in red needs to be 41! Otherwise something's wrong and your DNAS service is not working like expected.

Installing Outbreak server

The Outbreak servers are Java programs using mysql as database backend for managing connections, games, statistics and more. That means we need to install java and mysql.

- sudo apt-get install mariadb-server
- sudo apt-get install php7.0-mysql
- sudo apt-get install openjdk-8-jre-headless
- sudo apt-get install openjdk-8-jre

Copy the server packages to your PI (Filezilla is your friend).

Unpack them with 7z, this tool needs to be installed first

- sudo apt-get install p7zip
- p7zip -d BiohazardOutbreak1Server_CE.7z
- cd BiohazardOutbreak1Server_CE
- sudo mysql -u root < database/bioserver.sql
- sudo mkdir /var/www/bhof1
- sudo cp www/* /var/www/bhof1
- sudo chown -R www-data:www-data /var/www/bhof1
- sudo ln -s /var/www/bhof1 /var/www/dnas/00000002

Same for File#2: **TO BE CHECKED!**

- p7zip -d BiohazardOutbreak2Server_CE.7z
- cd BiohazardOutbreak2Server_CE
- sudo mysql -u root < database/bioserver.sql
- sudo mkdir /var/www/bhof2
- sudo cp www/* /var/www/bhof2
- sudo chown -R www-data:www-data /var/www/bhof1
- sudo ln -s /var/www/bhof2 /var/www/dnas/00000010

Create the bioserver database user:

- sudo mysql -u root bioserver
 - CREATE USER 'bioserver'@'%' IDENTIFIED BY 'xxxSECUREPASSWORDxxx';
 - GRANT ALL PRIVILEGES ON `bioserver`.* TO 'bioserver'@'%';
 - FLUSH PRIVILEGES;
 - exit

Starting the servers is like this: **NEED AUTOSTART SCRIPTS!**

- cd cd BiohazardOutbreak1Server_CE/dist
- java -jar BiohazardOutbreak1Server.jar

OUCH! Error needs fix undefined function mysql_connect ...

Final words