RustDesk Server on Devuan

The RustDesk Server must be accessible from any connecting clients on ports 21114-21119, TCP and UDP. If any workstation will be accessing from outside your network, you will need to forward those ports on your router to your server. Not all are required for all installations. See section *Open Ports* for more information.

We use Devuan as an alternative to Debian as Devuan allows us to choose the init system instead of forcing the use of SystemD. With its many faults, we still choose SysVInit for our init system, and Devuan allows that.

An excellent Rust Server install script is available at GitHub built by techahold. This script has a lot of extra features, but unfortunately, it assumes SystemD, and will fail halfway through as it is attempting to set up the SystemD service. The same is true of the .deb package which is available from RustDesk.

This article describes how to set up the Rust Server on Devuan, and may be helpful for other systems also.

Open Ports

On your firewall/NAT/whatever, you need ports

```
21114,21115,21116,21117,21118,21119
```

open and forwarded to the Rust Server. Additionally, to use the webui, you will need to open a port for that.

Following table shows what the ports are used for.

Port	Purpose	Protocol	Notes
21114	HBBS (RustDesk server)	TCP	Default port for API (not required)
21115	HBBS (RustDesk server)	TCP/UDP	Default port for the RustDesk server.
21116	HBBS (RustDesk server)	TCP/UDP	Used for client connections to the server.
21117	HBBS (RustDesk server)	TCP/UDP	Additional port for server communication.
21118	HBBS (RustDesk server)	TCP/UDP	Another port for server communication.
21119	HBBS (RustDesk server)	TCP/UDP	Used for additional server functionalities.
80	HTTP (optional fallback)	TCP	Commonly allowed port for web traffic; can be used for fallback.
443	HTTPS (optional fallback)	TCP	Secure web traffic; can be used for fallback

Add necessary packages

```
apt install -y curl unzip
```

Set up user

Rust Server does not require any special privileges, so creating a separate user account instead of running as root greatly enhances security. The following few lines assume we will install rust in /opt/rustdesk, and the log files will be stored in /var/log/rustdesk/*, and a system user named rust

setupRustDeskServer

```
#! /usr/bin/env sh

useradd --shell /usr/sbin/nologin --system --user-group --home-dir
/opt/rustdesk rust
mkdir /opt/rustdesk
mkdir /var/log/rustdesk
chown rust:rust /opt/rustdesk
chown rust:rust /var/log/rustdesk
```

Download Server

This will download the latest version of RustDesk Server from their github site. Second two lines stolen directly from techahold's script. He's a much better sh programmer than me.

```
cd /tmp
LATEST=$(curl https://api.github.com/repos/rustdesk/rustdesk-
server/releases/latest -s | grep "tag_name" | awk -F'"' '{print $4}')
wget "https://github.com/rustdesk/rustdesk-
server/releases/download/${LATEST}/rustdesk-server-linux-amd64.zip"
unzip rustdesk-server-linux-amd64.zip
mkdir -p /opt/rustdesk
mv /tmp/amd64/* /opt/rustdesk
chown rust:rust /opt/rustdesk/*
chmod 755 /opt/rustdesk/*
```

This will get three binary files in /opt/rustdesk

- hbbr - The relay server. If a direct P2P connection can not be made, this will be used to relay traffic between clients.
- hbbs - this is the “signal” server. Your clients will ping this to notify they exist, and will use it to set up a connection between clients.
- rustdesk-utils - a utility program that can generate/verify key pairs, and also run some basic tests on your service

Run for first time

The first time you run the signal server (hbbs), it will note that the key pair used for authentication does not exist and generate them. These keys are stored in the files:

- id_ed25519 - the private key
- id_ed25519.pub - the public key required by any client wanting to connect to this server

Note: The check is made in the current working directory, so you **must** run hbbs from within it's home directory (/opt/rustdesk)

```
cd /opt/rustdesk
echo Starting signaling server for testing. Press ^c to exit when you are
happy
sudo -u rust ./hbbs

echo Your key for the clients is
cat id_ed25519.pub
echo To find this again at a later date, just run the command cat
id_ed25519.pub
```

Set automatic run

Everything up to this point will work on all Unix systems, and we have done nothing that techahold's install script will do faster, and more reliably. However, for the Unix systems which do not use SysV, we need a SysV init script. Actually two; one for hbbr and one for hbbs.

Copy the following two files to /etc/init.d (Devuan), or wherever your init scripts are stored. By the way, I built these starting with the template at fhd's Github.

Create the file /etc/init.d/hbbs with the following content to control the signaling server via SysV Init

hbbs

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:          hbbs
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Rust Signaling Server
# Description:       This provides the definition of the signaling
server for rust
### END INIT INFO

dir="/opt/rustdesk"
cmd="/opt/rustdesk/hbbs"
```

```
user="rust"

name=`basename $0`
pid_file="/var/run/$name.pid"
stdout_log="/var/log/rustdesk/$name.log"
stderr_log="/var/log/rustdesk/$name.err"

get_pid() {
    cat "$pid_file"
}

is_running() {
    [ -f "$pid_file" ] && ps -p `get_pid` > /dev/null 2>&1
}

case "$1" in
    start)
    if is_running; then
        echo "Already started"
    else
        echo "Starting $name"
        cd "$dir"
        if [ -z "$user" ]; then
            sudo $cmd >> "$stdout_log" 2>> "$stderr_log" &
        else
            sudo -u "$user" $cmd >> "$stdout_log" 2>> "$stderr_log" &
        fi
        echo $! > "$pid_file"
        if ! is_running; then
            echo "Unable to start, see $stdout_log and $stderr_log"
            exit 1
        fi
    fi
    ;;
    stop)
    if is_running; then
        echo -n "Stopping $name.."
        kill `get_pid`
        for i in 1 2 3 4 5 6 7 8 9 10
        # for i in `seq 10`
        do
            if ! is_running; then
                break
            fi

            echo -n "."
            sleep 1
        done
        echo

        if is_running; then
```

```
            echo "Not stopped; may still be shutting down or shutdown
may have failed"
            exit 1
        else
            echo "Stopped"
            if [ -f "$pid_file" ]; then
                rm "$pid_file"
            fi
        fi
    else
        echo "Not running"
    fi
    ;;
    restart)
    $0 stop
    if is_running; then
        echo "Unable to stop, will not attempt to start"
        exit 1
    fi
    $0 start
    ;;
    status)
    if is_running; then
        echo "Running"
    else
        echo "Stopped"
        exit 1
    fi
    ;;
    *)
    echo "Usage: $0 {start|stop|restart|status}"
    exit 1
    ;;
esac

exit 0
```

Create the file /etc/init.d/hbbr with the following content to control the relay server via SysV Init

hbbr

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:          hbbr
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Rust Relay Server
# Description:       This provides the definition of the relay server
```

```
for rust
### END INIT INFO

dir="/opt/rustdesk"
cmd="/opt/rustdesk/hbbr"
user="rust"

name=`basename $0`
pid_file="/var/run/$name.pid"
stdout_log="/var/log/rustdesk/$name.log"
stderr_log="/var/log/rustdesk/$name.err"

get_pid() {
    cat "$pid_file"
}

is_running() {
    [ -f "$pid_file" ] && ps -p `get_pid` > /dev/null 2>&1
}

case "$1" in
    start)
    if is_running; then
        echo "Already started"
    else
        echo "Starting $name"
        cd "$dir"
        if [ -z "$user" ]; then
            sudo $cmd >> "$stdout_log" 2>> "$stderr_log" &
        else
            sudo -u "$user" $cmd >> "$stdout_log" 2>> "$stderr_log" &
        fi
        echo $! > "$pid_file"
        if ! is_running; then
            echo "Unable to start, see $stdout_log and $stderr_log"
            exit 1
        fi
    fi
    ;;
    stop)
    if is_running; then
        echo -n "Stopping $name.."
        kill `get_pid`
        for i in 1 2 3 4 5 6 7 8 9 10
        # for i in `seq 10`
        do
            if ! is_running; then
                break
            fi

            echo -n "."
```

```
            sleep 1
        done
        echo

        if is_running; then
            echo "Not stopped; may still be shutting down or shutdown
may have failed"
            exit 1
        else
            echo "Stopped"
            if [ -f "$pid_file" ]; then
                rm "$pid_file"
            fi
        fi
    else
        echo "Not running"
    fi
    ;;
    restart)
    $0 stop
    if is_running; then
        echo "Unable to stop, will not attempt to start"
        exit 1
    fi
    $0 start
    ;;
    status)
    if is_running; then
        echo "Running"
    else
        echo "Stopped"
        exit 1
    fi
    ;;
    *)
    echo "Usage: $0 {start|stop|restart|status}"
    exit 1
    ;;
esac

exit 0
```

Now (we're almost done), run the following commands to start the both servers up.

```
chmod 755 /etc/init.d/hbbr
chmod 755 /etc/init.d/hbbs
# test hbbs
/etc/init.d/hbbs start
# test hbbr
/etc/init.d/hbbr start
```

```
# if both worked correctly, run the following command to automatically start at boot
update-rc.d hbbs defaults
update-rc.d hbbr defaults
```

If you made it through the last step with no errors, you should now be able to access the server from one of the clients.

Set automatic log rotate

Your logs can get quite large, so it is best to rotate them occassional. Devuan uses the logrotate script to do this for, and it is fairly simple to add a new definition in /etc/logrotate.d for the next pass.

Following command will create the definition. It will rotate the hbbr/hbbs logs daily, keeping two weeks of logs. All logs except for the current one and the previous one will be compressed.

This is just one command. Just copy and paste it anywhere into the server.

rotatelogs

```
cat << EOF > /etc/logrotate.d/rustdesk
/var/log/rustdesk/*.log /var/log/rustdesk/*.err {
   daily
   rotate 14
   compress
   delaycompress
   create 640 rust rust
   postrotate
      service hbbr restart > /dev/null
      service hbbs restart > /dev/null
   endscript
   sharedscripts
   missingok
   notifempty
}
EOF
```

Links

- Template for creating SysV init scripts
- Excellent install script for open source RustDesk Server on Linux which uses SystemD
- Documentation from RustDesk for installing the Open Source server
- Installer Scripts for RustDesk Client
- Several customized installer scripts for RustDesk Client