RustDesk Server Setup (free)

These are just notes. They are not usable. Ignore this page

The RustDesk Server must be accessible from any connecting clients on ports 21115-21119, 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

Configuration Files

Do not use, these are just notes and they are wrong

```
# hbbs Configuration
[hbbs]
addr = "0.0.0.0:21116"
password = "your_secure_password"
max_connections = 100
timeout = 30

[log]
level = "info"
file = "hbbs.log"

[tls]
enabled = true
cert_file = "/path/to/cert.pem"
key_file = "/path/to/key.pem"

# hbbr Configuration
[hbbr]
addr = "0.0.0.0:21117"
password = "your_secure_password"
max_connections = 100
timeout = 30
enable_file_transfer = true

[log]
level = "info"
file = "hbbr.log"

[tls]
enabled = true
cert_file = "/path/to/cert.pem"
```

```
key_file = "/path/to/key.pem"
```

```
# hbbs Configuration
# rust ID Server
# save as hbbs.toml
# hbbs --config /path/to/hbbs.toml

# The address and port for the ID server
addr = "0.0.0.0:21116"  # Listen on all interfaces

# Optional: Set a password for the ID server
# password = "your_secure_password"

# Optional: limit maximum number of connections
# max_connections = "10"

# Optional: Sets a timeout for client connections
# timeout =

# Optional: Enable logging
[log]
level = "info"  # Log level can be "debug", "info", "warn", "error"
file = "hbbs.log"  # Log file path
```

```
# hbbr Configuration
# save as hbbr.toml
# hbbr --config /path/to/hbbr.toml

# The address and port for the relay server
addr = "0.0.0.0:21117"  # Listen on all interfaces

# Optional: Set a password for the relay server
# password = "your_secure_password"

# Optional: Enable logging
[log]
level = "info"  # Log level can be "debug", "info", "warn", "error"
file = "hbbr.log"  # Log file path

# Optional: Enable TLS for secure connections
[tls]
enabled = true
cert_file = "/path/to/cert.pem"  # Path to the certificate file
key_file = "/path/to/key.pem"      # Path to the private key file
```

SysVInit

Startup script

This is not very good, but it works, so I'm including it. Running this requires you create the /var/log/rustdesk directory first

startRust

```
#! /usr/bin/env bash

# go into correct directory so we can find the key file
cd /opt/rustdesk

# start signal server
/opt/rustdesk/hbbs >>/var/log/rustdesk/signalserver.log
2>>/var/log/rustdesk/signalserver.error &

# start relay server
/opt/rustdesk/hbbr >>/var/log/rustdesk/relayserver.log
2>>/var/log/rustdesk/relayserver.error &
```

Do Not Use, work in progress

hbbs

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:          hbbs
# Required-Start:    $network
# Required-Stop:     $network
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Rust HBBS
### END INIT INFO

# Path to the executable
DAEMON=/opt/rustdesk/hbbs >>/var/log/rustdesk/signalserver.log
2>>/var/log/rustdesk/signalserver.error
DAEMON_NAME=hbbs
PIDFILE=/var/run/$DAEMON_NAME.pid

start() {
    echo "Starting $DAEMON_NAME..."
    if [ -f $PIDFILE ]; then
        echo "$DAEMON_NAME is already running."
        return 1
    fi
    $DAEMON &
    echo $! > $PIDFILE
    echo "$DAEMON_NAME started."
```

```
}

stop() {
    echo "Stopping $DAEMON_NAME..."
    if [ ! -f $PIDFILE ]; then
        echo "$DAEMON_NAME is not running."
        return 1
    fi
    kill $(cat $PIDFILE)
    rm -f $PIDFILE
    echo "$DAEMON_NAME stopped."
}

status() {
    if [ -f $PIDFILE ]; then
        echo "$DAEMON_NAME is running."
    else
        echo "$DAEMON_NAME is not running."
    fi
}

case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    status)
        status
        ;;
    restart)
        stop
        start
        ;;
    *)
        echo "Usage: $0 {start|stop|status|restart}"
        exit 1
        ;;
esac

exit 0
```