MuntsOS Embedded Linux

Application Note #16: Sending Email using an SSH Tunnel

Revision 1
19 March 2025

by Philip Munts
***dba* Munts Technologies**
http://tech.munts.com

Introduction

This application note describes how send email messages from software running on a ***MuntsOS Embedded Linux*** (hereafter just **MuntsOS**) target board.

Prerequisites

The **MuntsOS** software development environment must be installed on a 64-bit x86-64 Linux system (AppNote #1 or AppNote #2).

MuntsOS must be installed on the target computer (AppNote #3 or AppNote #15).

Test Platform Hardware

The test platform for the purposes of this application note consists of any **MuntsOS** target board with Internet connectivity.

Mail Transfer Agent (MTA)

A ***Mail Transfer Agent*** is a system program that accepts an email message text conforming to RFC5322 and then delivers it to one or more recipients.

Examples of MTA's for Unix and Linux include Sendmail, Postfix, and DragonFly Mail Agent. Full service MTA's are often large and difficult to configure and may not suitable for small embedded systems.

By tradition every MTA for Unix and Linux, including **MuntsOS**, provides a command line program named `/usr/sbin/sendmail` that reads an email message text conforming to RFC5322 from standard input and then delivers the message to its recipient(s).

Given a file `message.txt` containing an email message text conforming to RFC5322, the following command sends an email:

```
/usr/sbin/sendmail -t <message.txt
```

The `/usr/sbin/sendmail` included in **MuntsOS** is provided by BusyBox. It just forwards email messages to an SMTP (***Simple Mail Transfer Protocol***) server listening on TCP `port localhost:25`.

MuntsOS does not include a SMTP server by default. See below for a description about how to use an SMTP server on another machine via an SSH tunnel.

Mail User Agent (MUA)

A ***Mail User Agent*** is a user program that generates an email message text conforming to RFC5322 and then passes it to an MTA for delivery. On Unix and Linux, the MUA often runs **`/usr/sbin/sendmail`** using the **`popen()`** Standard I/O Library function and writes the generated email message text to the file object returned by **`popen()`**.

MuntsOS includes the MUA **`/usr/bin/mail`** from the GNU Mailutils package. The mail program reads a message payload from standard input, generates an email message text conforming to RFC5322, and then passes the result to **`/usr/sbin/sendmail`** for delivery.

The following command sends an email with the subject **`Test1`** to recipient **`you@me.com`**:

```
echo "This is a test" | mail -s "Test1" you@me.com
```

A user application program can run **`/usr/bin/mail`** with **`popen()`** to send an email message. A user application program can also act as its *own* MUA. The Ada Web Server library provides email client services for the Ada programming language. Similarly, the .Net Core **`System.Net.Mail`** namespace provides email client services for C# programs.

Configuring an SSH Tunnel to a Remote Computer

If you have administrative access to a Unix (FreeBSD, OpenBSD, Linux, etc.) remote computer that runs a local SMTP server on port 25, that is permitted to send email, and that you can log into with **`ssh`**, you can create a user **`mailtunnel`** that connects to the local SMTP server upon login.

Setup MuntsOS Target Computer

1. Create or modify **`/etc/inetd.conf`** and **`/etc/ssh/ssh_known_hosts`** with the following commands, replacing **`foo.bar`** with the domain name of your remote computer:

```
cat <<EOD >>/etc/inetd.conf
# Mail relay over SSH tunnel
127.0.0.1:25 stream tcp nowait root /usr/bin/ssh -q -T mailtunnel@foo.bar
EOD
ssh-keyscan foo.bar >>/etc/ssh/ssh_known_hosts
```

2. Create **`.ssh/id_rsa`** and **`.ssh/id_rsa.pub`** using **`sysconfig`** option **`Regenerate superuser id_rsa`**.

3. Enable **`inetd`** by setting bit 10 in the **`OPTIONS`** word in **`/boot/cmdline.txt`** (Raspberry Pi) or **`/boot.config.txt`** (Orange Pi Zero 2W or BeaglePlay) using **`sysconfig`** option **`Edit cmdline.txt`** or **`Edit config.txt`**.

4. Copy **`.ssh/id_rsa.pub`** to the remote computer superuser and then reboot.

Setup Remote Computer

1. Become super user with `sudo su -`

2. Modify `/etc/ssh/sshd_config`:

```
cat <<EOD >>/etc/ssh/sshd_config

Match User mailtunnel
        AllowTcpForwarding no
        ForceCommand ncat -4 127.0.0.1 25
EOD
killall -HUP /usr/sbin/sshd
```

2. Create user `mailtunnel` with commands similar to the following:

```
groupadd -g 997 mailtunnel
useradd -c "Mail Tunnel" -m -g 997 -u 997 -s /bin/sh mailtunnel
rm -rf /home/mailtunnel/.*
```

3. Create `/home/mailtunnel/.ssh/authorized_keys` on the remote computer, using commands similar to the following:

```
sudo su -
mkdir -p /home/mailtunnel/.ssh
cat id_rsa.pub >>/home/mailtunnel/.ssh/authorized_keys
chown -R mailtunnel.mailtunnel /home/mailtunnel
chmod 444 /home/mailtunnel/.ssh/authorized_keys
chmod 500 /home/mailtunnel/.ssh
chmod 500 /home/mailtunnel
```

Testing

1. Try to log in from the MuntsOS target computer to the remote computer:

```
ssh mailtunnel@foo.bar
quit
```

You should get responses similar to the following:

```
220 bethel.munts.net ESMTP OpenSMTPD
221 2.0.0 Bye
```

2. Try to send yourself an email with a command similar to the following:

```
echo "This is a test" | mail -s Test1 me@you.com
```