Technical Notebook Pages

One of my online technical notebooks. This one made using BookStack.

- Asterisk
- Axis camera packet capture
- Batteries
- NodeJS
- OpenVPN
- Polycom
- Project: Teensy Climate
- radsecproxy
- Software
- Ubiquiti UniFi
- WiFi Regulatory Notes
- WireGuard
- ProFTPd
    - ProFTPd and SFTP
- Bind9
    - Recursion without recursion to enable forward zones
- GitLab
- PoE Detection and Negotiation
- Virtual disk format conversions
- WiFi Standards
- USB Pendrive

- Trend Micro Vision One Search Auto-Clicker Auto-Refresher

Asterisk

> Content migrated from here.

Asterisk Download Links

http://downloads.asterisk.org/pub/telephony/

http://downloads.asterisk.org/pub/telephony/dahdi-linux/

http://downloads.asterisk.org/pub/telephony/dahdi-tools/

http://downloads.asterisk.org/pub/telephony/libpri/

http://downloads.asterisk.org/pub/telephony/asterisk/

Asterisk SVN Links

http://svnview.digium.com/svn

http://svnview.digium.com/svn/asterisk/

Asterisk Related Links

- Asterisk snmp
- Asterisk 1.4 app_page.c added device state
- Asterisk & rtcp
- Asterisk :: The Open Source Telephony Platform
- Building asterisk

Telephony Related Links

- Analog Voice Fundamentals
- Polycom
- T1 Channel Numbering

Hacks and Patches

- Logrotate configuration
- Music on hold
- Voicemail VM_ALTEXT Patch

Converting audio files for use in Asterisk

```
Resample a stereo wav into a mono 8000Hz 16 bit wav:
sox infile.wav -c mono -r 8000 -b 16 outfile.wav

Convert a properly formatted wav (8000Hz, 16bit mono) file to gsm:
sox infile.wav outfile.gsm
```

Converting mp3 to slin

```
mpg123 -w file.wav file.mp3
sox file.wav -t raw -r 8000 -s -2 -c 1 file.sln
```

Converting wav to slin

```
sox file.wav -t raw -r 8000 -s -2 -c 1 file.sln
```

Upgrading asterisk 1.4 to 1.6

```
NoOp updates

/NoOp

:%s/\(NoOp,\)\(.*\)$/NoOp(\2)/
sed -e 's/\(NoOp,\)\(.*\)$/NoOp(\2)/'

pipe character to comma

:%s/|/,/g

ExecIf updates

/ExecIf

:%s/\(ExecIf(\)\(\$\[.*\]\),\(.*\),\(.*\)$/\1\2?\3(\4)/
sed -e 's/\(ExecIf(\)\(\$\[.*\]\),\(.*\),\(.*\)$/\1\2?\3(\4)/'

Deprecated stuff - dialplan
grep -ir \
-e SetCallerPres \
-e WaitMusicOnHold \
-e SetMusicOnHold \
-e QUEUE_MEMBER_COUNT \
-e Local \
*
```

Faxing

Changes that can be made on the Brother MFC-8860DN to better optimize for using with VOIP.

Resolution press MENU/SET, 2, 2, 2, choose standard. Press MENU/SET, STOP,EXIT.

Baud (baud rate)/ECM press MENU/SET, 2, 0, 1, which is compatiability. Press up or down arrow to choose basic. Press MENU/SET, STOP/EXIT. This sets baud to 9600 and disables ECM.

Building

```
cd /usr/src/asterisk/mpg123-0.59r-gpl
make linux-x86_64 ; make install ; cd ..

cd /usr/src/asterisk/1.6.2.4

cd spandsp-0.0.6
./configure ; make install ; ldconfig ; cd ..

cd corosync-1.2.0
./configure ; make install ; cd ..

cd openais-1.1.2
./configure ; make install ; cd ..

cd dahdi-linux-2.2.1
make install ; cd ..

cd dahdi-tools-2.2.1
./configure ; make menuconfig ; make install ; make config ; make samples ; cd ..

cd libpri-1.4.10.2
make install ; cd ..

cd asterisk-1.6.2.4
./configure ; make menuconfig ; make install ; make samples ; make config ; cd ..

cd asterisk-addons-1.6.2.0
./configure ; make menuconfig ; make install ; make samples ; cd ..

cd ..
```

Simplify Asterisk 1.6 Dialplan Extensions

```
:%s/exten => .*,n/same => n/g
```

Sipura Distinctive Ring

```
exten => s,n,SIPAddHeader(Alert-Info: info=<Bellcore-r2>)
```

Available Distinctive Ring Patterns:

```
Pattern Name   Distinctive Ring Patterns
============   ========================
Bellcore-r1    60(2/4)
Bellcore-r2    60(.8/.4,.8/4)
Bellcore-r3    60(.4/.2,.4/.2,.8/4)
Bellcore-r4    60(.3/.2,1/.2,.3/4)
Bellcore-r5    1(.5/.5)
Bellcore-r6    60(.2/.4,.2/.4,.2/4)
Bellcore-r7    60(.4/.2,.4/.2,.4/4)
Bellcore-r8    60(0.25/9.75)
```

Available Distinctive Call Waiting Tone Patterns:

```
Pattern Name   Tone Pattern
============   ============
Bellcore-r1    30(.3/9.7)
Bellcore-r2    30(.1/.1, .1/9.7)
Bellcore-r3    30(.1/.1, .1/.1, .1/9.7)
Bellcore-r4    30(.1/.1,.3/.1,.1/9.3)
Bellcore-r5    1(.5/.5)
Bellcore-r6    30(.1/.1,.3/.2,.3/9.1)
Bellcore-r7    30(.3/.1,.3/.1,.1/9.1)
Bellcore-r8    2.3(.3/2)
```

Using AWK to find individual calls

The below scripting will find all calls to the extension 11175 and print all the details for each call.

```
grep Dial /var/log/asterisk/full | grep macro-stdexten-v3 | grep "11175," | awk -F\[ '{print $3}' | awk -F\] '{print $1}' | while read x; do echo ""; echo ""; echo ""; echo "identifier: $x"; echo ""; echo ""; echo ""; grep "\[$x\]" /var/log/asterisk/full; done



grep "Called .*11175" /var/log/asterisk/full | awk -F\[ '{print $3}' | awk -F\] '{print $1}' | while read x; do echo ""; echo ""; echo ""; echo "identifier: $x"; echo ""; echo ""; echo ""; grep "\[$x\]" /var/log/asterisk/full; done



grep "Called .*11175" /var/log/asterisk/full | awk -F\[ '{print $3}' | awk -F\] '{print $1}' | while read x; do echo ""; echo ""; echo ""; echo "identifier: $x"; echo ""; echo ""; echo ""; grep "\[$x\]" /var/log/asterisk/full; done | grep -A 5 -e "Called .*11175"
```

Finding hung channels

```
Asterisk 1.4,1.6
asterisk -rx "core show channels concise" | awk -F ! '{print $1,$11}'

asterisk -rx "core show channels concise" | awk -F ! '{print $1,$11}' | while read x y; do if [ $y -gt 60 ] || [ $y -lt 0 ]; then echo "$x - `expr $y / 60` minutes `expr $y % 60` seconds"; fi ; done

Asterisk 1.2
asterisk -rx "show channels concise" | awk -F : '{print $1,$11}'
```

Asterisk and SystemD

See here...

Put the following in /etc/systemd/system/asterisk.service and then run "systemctl daemon-reload && systemctl enable asterisk"

```
[Unit]
Description=Asterisk PBX And Telephony Daemon
Wants=network.target
After=network.target

[Service]
Type=simple
User=root
Group=root
#Environment=HOME=/var/lib/asterisk
#WorkingDirectory=/var/lib/asterisk
ExecStart=/usr/sbin/asterisk -f -C /etc/asterisk/asterisk.conf
ExecStop=/usr/sbin/asterisk -rx 'core stop now'
ExecReload=/usr/sbin/asterisk -rx 'core reload'

LimitNOFILE=65535

# safe_asterisk emulation
Restart=always
RestartSec=10

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

Paste below into a terminal to setup the files:

```
cat << EOF > /etc/systemd/system/asterisk.service
[Unit]
Description=Asterisk PBX And Telephony Daemon
Wants=network.target
After=network.target

[Service]
Type=simple
User=root
```

```
Group=root
#Environment=HOME=/var/lib/asterisk
#WorkingDirectory=/var/lib/asterisk
ExecStart=/usr/sbin/asterisk -f -C /etc/asterisk/asterisk.conf
ExecStop=/usr/sbin/asterisk -rx 'core stop now'
ExecReload=/usr/sbin/asterisk -rx 'core reload'

LimitNOFILE=65535

# safe_asterisk emulation
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target

EOF
```

Loading Dahdi

After some digging, for my purposes I found the best way to load the transcoding module is to use the systemd module loader to load the wctc4xxp transcoding module and UDEV to run dahdi_cfg once the module is loaded.

Prerequisite: Configure the /etc/dahdi/modules and /etc/dahdi/system.conf files as you normally would.

```
cat << EOF > /etc/dahdi/modules
# /etc/modules-load.d/dahdi is symlinked here so systemd will load it on startup
# /etc/udev/rules.d/dahdi-wctc4xxp.rules instructs udev to run dahdi_cfg after wctc4xxp module
is loaded

# Digium TC400B: G729 / G723 Transcoding Engine
wctc4xxp

EOF

ln -s /etc/dahdi/modules /etc/modules-load.d/dahdi.conf

# update udev to run dahdi_cfg after loading the transcoding module
cat << EOF > /etc/udev/rules.d/dahdi-wctc4xxp.rules
```

```
KERNEL=="wctc4xxp" RUN+="/usr/sbin/dahdi_cfg"
EOF
```

The next time you reboot, systemd will load the module, udev will run dahdi_cfg, and then systemd will load asterisk. Granted this is only really needed if you haven't migrated away from MeetMe yet...

GROUP and GROUP_COUNT

func_GROUP

func_GROUP_COUNT

```
Set(GROUP(category)=groupname)
Set(CHECKING=${GROUP_COUNT(groupname@category)})
```

Example:

```
same => n,Set(GROUP(rcf)=${CHANNEL(accountcode)})
same => n,Set(GROUP(rcf${RCF_DID})=${CHANNEL(accountcode)})

same => n,Set(rcfCountAccountcode=${GROUP_COUNT(${CHANNEL(accountcode)}@rcf)})
same => n,Set(rcfCountDID=${GROUP_COUNT(${CHANNEL(accountcode)}@rcf${RCF_DID})})
```

Asterisk comparisons

```
same => n,ExecIf($["${testVariable}" = "SomeValue"]?Set(testvar=TRUE):Set(testvar=FALSE))

; Numberical comparisons require not using quotes
same => n,Set(count=5)
same => n,GofoIf($[${count} < 10]?trueTarget:falseTarget)



; Check if a variable is NOT NULL - This one will set testVariable2=TRUE
same => n,Set(testVariable1=something)
same =>
```

```
n,ExecIf($[!${LEN(${testVariable1})}]?Set(testVariable2=TRUE):Set(testVariable2=FALSE))

; Check if a variable is NOT NULL - This one will set testVariable2=FALSE
same => n,Set(testVariable1=)
same =>
n,ExecIf($[!${LEN(${testVariable1})}]?Set(testVariable2=TRUE):Set(testVariable2=FALSE))



; Multiple comparisons using AND (&)
same => n,ExecIf($["${testVariable1}" = "VALID" & "${testVariable2}" =
"VALID"]?set(status=TRUE))

; Multiple comparisons using OR (|)
same => n,ExecIf($["${testVariable1}" = "VALID" | "${testVariable2}" =
"VALID"]?set(status=TRUE))



; Providing only a false conditional path
same => n,Set(testVariable1=1)
same => n,GotoIf(${}?:falseCondition)
same => n(trueCondition),NoOp(do something on true condition)
same => n,Hangup()
same => n(falseCondition),NoOp(do something on false condition)
same => n,Hangup()
```

[end]

Axis camera packet capture

Download network trace

Source

According to documentation this can be done via the URL below, as well as via the web interface in newer AXIS OS versions.

pcapdump URL

Source

1. Open any browser and paste the following URL: `http://CA.ME.RA.IP/axis-cgi/debug/debug.tgz?cmd=pcapdump&duration=60` (Substitute the "CA.ME.RA.IP" with the IP address of the device.)
2. The browser will ask for credentials. The default ones are: `root/pass`.
3. Once you enter the credentials, a download will start. It will last for `60` seconds. Note that the last part of the URL (`"duration="`) will determine the amount of time the capture will run — you can change this number as needed.
4. Once finished, the file will be usually saved to the computer's `Downloads` folder.

via the web interface

1. Open the camera's web interface.
2. Go to Settings -> System -> Maintenance

Batteries

Lithium Ion Voltage vs. Charge Status

Original Source

```
4.2V – 100%
4.1V – 87%
4.0V – 75%
3.9V – 55%
3.8V – 30%
3.5V – 0%
```

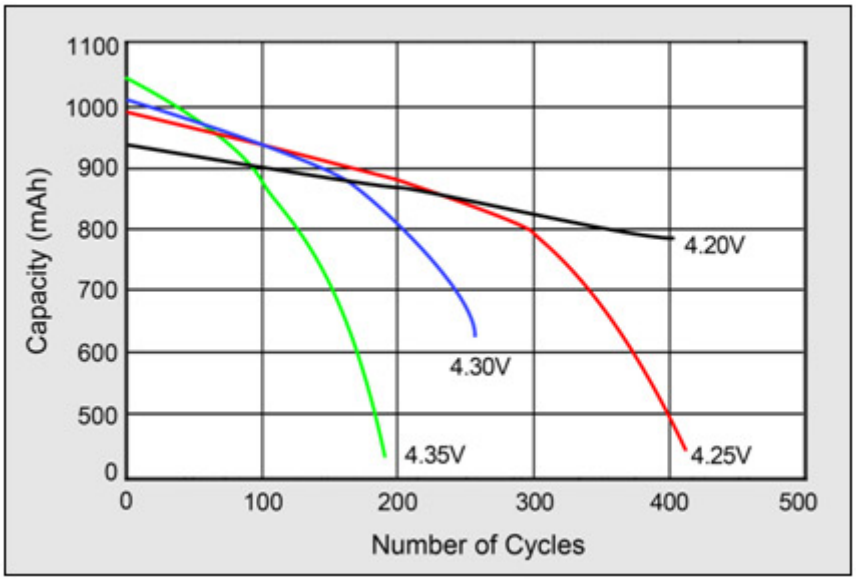
Somewhere in the 3.9V or slightly below area would be ideal for storage. Just don't overdo it. I believe AW batteries generally ship about 40% charge, or you can just discharge them in a light or other device, since that is easy and safe.

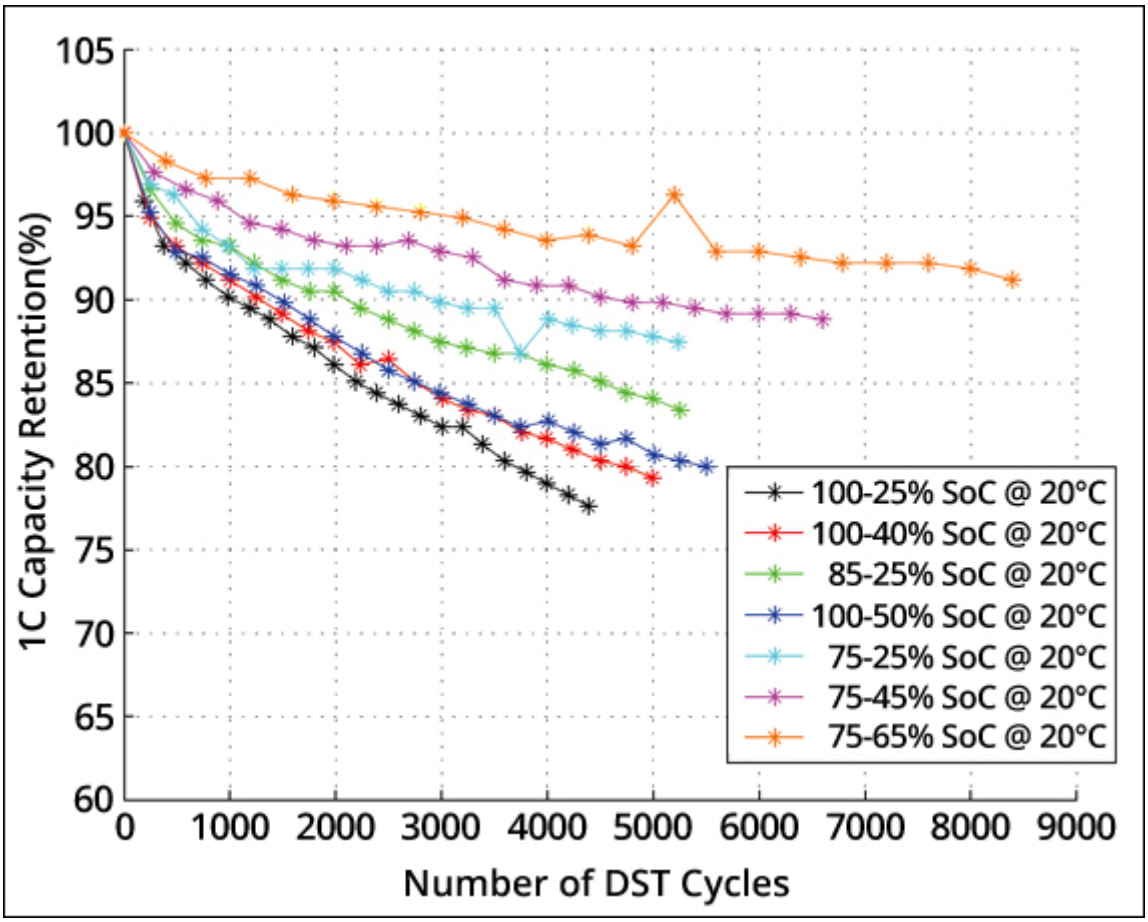
Battery University

Below are links to and summaries of some very informative Battery University articles.

BU-808: How to Prolong Lithium-based Batteries

Original Source





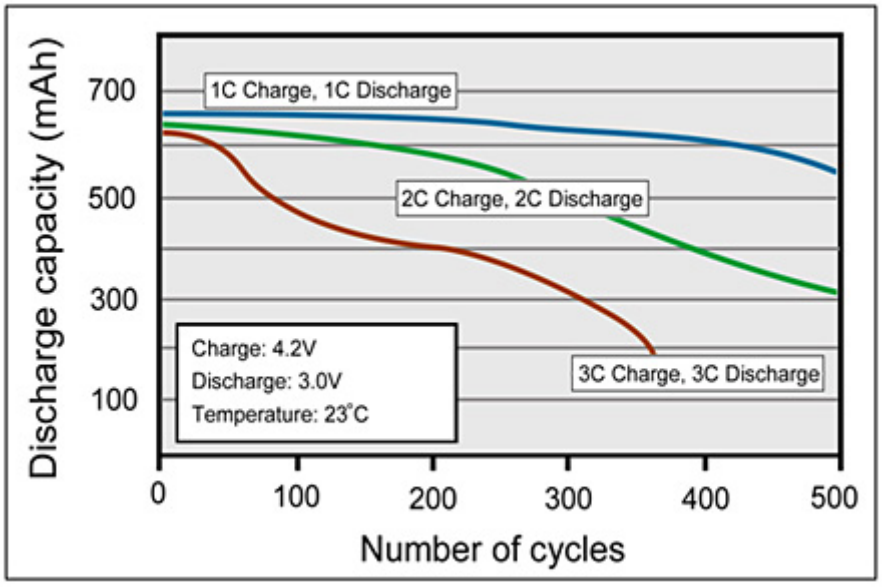
Summary:

- Environmental conditions, not cycling alone, govern the longevity of lithium-ion batteries. The worst situation is keeping a fully charged battery at elevated temperatures. Battery packs do not die suddenly, but the runtime gradually shortens as the capacity fades.
- Lower charge voltages prolong battery life and electric vehicles and satellites take advantage of this. Similar provisions could also be made for consumer devices, but these are seldom offered; planned obsolescence takes care of this.

- A laptop battery could be prolonged by lowering the charge voltage when connected to the AC grid. To make this feature user-friendly, a device should feature a “Long Life” mode that keeps the battery at 4.05V/cell and offers a SoC of about 80 percent. One hour before traveling, the user requests the “Full Capacity” mode to bring the charge to 4.20V/cell.

BU-401a: Fast and Ultra-fast Chargers

Original Source



Summary

- Charging and discharging Li-ion above 1C reduces service life. Use a slower charge and discharge if possible. This rule applies to most batteries.
- All batteries perform best at room temperature and with a moderate charge and discharge. Such a sheltered lifestyle does not always reflect real-world situations where a compact pack must be charged quickly and deliver high currents. Such typical applications are drones and remote control devices for hobbyists. Expect a short cycle life when a small pack must give all it has.

-end

NodeJS

Process Management

- pm2 - Advanced, production process manager for Node.js

ORM for Node.js

- Objection.js - ORM for Node.js
- Sequelize - ORM for Node.js

Template languages for JavaScript

- nunjucks - A powerful templating engine with inheritance, asynchronous control, and more (jinja2 inspired)

Asterisk related moduels for Node.js

- asterisk-manager - NodeJS Asterisk Manager API
- ari-client - Node.js client for ARI

Other useful modules for Node.js

- chalk - Terminal string styling done right
- ipaddr.js - IP address manipulation library in JavaScript (CoffeeScript, actually)

- jsonwebtoken -JsonWebToken implementation for node.js
- node-ssh - SSH2 with Promises
- node-uuid - Generate RFC-compliant UUIDs in JavaScript

OpenVPN

Integrating OpenVPN with RADIUS

Below is the culmination of at least a month working with OpenVPN.

I recently discovered a bug in the Mikrotik OpenVPN server implementation that prevents users from being informed that they entered invalid credentials. (This bug was resolved in RouterOS 7.4.) The project I'm working on requires Azure MFA be part of the equation for VPN authentication. Azure ADDS will lock a user account after three failed authentication attempts. Because of this Mikrotik bug, a user who accidentally mistypes a password has about nine seconds before Azure ADDS locks there account.

I submitted a bug report to Mikrotik, but as much as I am a Mikrotik fanboy, I know they're not going to fix it anytime soon (which they surprisingly did in RouterOS 7.4, but didn't back-port the fix into the ROS v6 release tree.) I can't be the first person to inform them of this bug, and their OpenVPN Server has always left much to be desired and not gotten much attention from them, so I don't expect it to be resolved anytime soon. I also wasn't about to deploy a VPN solution to a user base that was going to be constantly resulting in calls to the helpdesk because of account lockouts.

I wanted to use the Mikrotik router because its very easy to implement firewall policies that are based on the user's group returned from RADIUS.

The goal of this exercise was learning how to integrate RADIUS authentication with MFA PUSH requirements and be able to service similar group based firewall policies.

I have not yet gotten around to the firewall policies implementation, but I do have a good idea of how it can be done.

The configuration below uses the following authentication strategies:

- Private CA created with EasyRSAv3, complete with CRL distribution endpoints. This is not detailed here.
- The Private CA was created an an Intermediate CA signed by another well used CA that I did not want to use for client certificates.
- Server certificates generated by the above Private CA
- User certificates generated by the above Private CA
- The following factors are required for a successful login:
  - A valid user certificate. The CN must match the username of the person logging into the OpenVPN server.

    - Username and password supplied by the user is authentication against a RADIUS server.
        - Using RADIUS allows any number of 2FA/MFA solutions to be leveraged for a third factor such as an MFA PUSH notification via Azure MFA or DUO.
        - The OpenVPN radius plugin does not support radsec, so radsecproxy was also used to provide secure RADIUS queries across the Internet to redundant cloud hosted servers.

If you know what you're doing, you can use the configs and scripts listed below to implement the same solution. □□

Installing OpenVPN and the OpenVPN plugin in Ubuntu

```
apt -y install openvpn openvpn-auth-radius


mkdir -p /etc/openvpn/client /etc/openvpn/server /etc/openvpn/server-ccd
```

/etc/openvpn/server/server.conf

```
server 192.168.168.0 255.255.255.0
proto udp
port 1194
dev tun1
topology subnet

;user nobody
;group nogroup

cipher AES-256-GCM
auth SHA256

management 127.0.0.1 61194 /etc/openvpn/server/pw-file

duplicate-cn

;client-cert-not-required
verify-client-cert require
;username-as-common-name

;auth-user-pass-verify
```

```
ca         /etc/openvpn/server/ca-chained.crt
crl-verify /etc/openvpn/server/crl.pem
cert       /etc/openvpn/server/server.crt
key        /etc/openvpn/server/server.key
dh none
ecdh-curve secp521r1

tls-auth /etc/openvpn/server/tls-auth.key 0
tls-version-min 1.2

verb 1
log               /var/log/openvpn/openvpn.log
status            /var/log/openvpn/openvpn-status.log
client-config-dir /etc/openvpn/server-ccd

script-security 2
tls-verify            /etc/openvpn/server/script-tls-verify
auth-user-pass-verify /etc/openvpn/server/script-auth-user-pass-verify via-env
plugin                /usr/lib/openvpn/radiusplugin.so /etc/openvpn/server/radiusplugin.cnf
client-connect        /etc/openvpn/server/script-client-connect

ifconfig-pool-persist /var/log/openvpn/ipp.txt

push "dhcp-option DNS 8.8.8.8"
push "dhcp-option DNS 8.8.4.4"
;push "dhcp-option DNS 192.168.168.1"
push "dhcp-option DOMAIN ovpn.loc"

; Do you want the default gateway redirected? There's a few ways to do it:
push "redirect-gateway"
;push "redirect-gateway autolocal"
;push "redirect-gateway def1 bypass-dhcp"

; Pick other routes you want to push to the clients:
;push "route 10.0.0.0 255.0.0.0"
;push "route 172.16.0.0 255.240.0.0"
;push "route 192.168.0.0 255.255.0.0"
;push "route 100.64.0.0 255.192.0.0"

keepalive 10 120
```

```
persist-key
persist-tun

;mute 20
;explicit-exit-notify 1
```

/etc/openvpn/server/radiusplugin.cnf

```
# The NAS identifier which is sent to the RADIUS server
# The service type which is sent to the RADIUS server
# The framed protocol which is sent to the RADIUS server
# The NAS port type which is sent to the RADIUS server
# The NAS IP address which is sent to the RADIUS server
NAS-Identifier=something-that-is-meaningful
Service-Type=2
Framed-Protocol=1
NAS-Port-Type=5
NAS-IP-Address=192.168.1.1

# Path to the OpenVPN configfile. The plugin searches there for
# client-config-dir PATH   (searches for the path)
# status FILE                      (searches for the file, version must be 1)
# client-cert-not-required (if the option is used or not)
# username-as-common-name  (if the option is used or not)
OpenVPNConfig=/etc/openvpn/server/server.conf



# Support for topology option in OpenVPN 2.1
# If you don't specify anything, option "net30" (default in OpenVPN) is used.
# You can only use one of the options at the same time.
# If you use topology option "subnet", fill in the right netmask, e.g. from OpenVPN option "--
server NETWORK NETMASK"
subnet=255.255.255.0
# If you use topology option "p2p", fill in the right network, e.g. from OpenVPN option "--
server NETWORK NETMASK"
# p2p=10.8.0.1



# Allows the plugin to overwrite the client config in client config file directory,
```

```
# default is true
overwriteccfiles=true

# Allows the plugin to use auth control files if OpenVPN (>= 2.1 rc8) provides them.
# default is false
useauthcontrolfile=true



# Path to a script for vendor specific attributes.
# Leave it out if you don't use an own script.
#vsascript=/etc/openvpn/server/vsascript.pl

# Path to the pipe for communication with the vsascript.
# Leave it out if you don't use an own script.
#vsanamedpipe=/tmp/vsapipe

# A radius server definition, there could be more than one.
# The priority of the server depends on the order in this file. The first one has the highest
priority.
server
{
        acctport=1813
        authport=1812
        retry=0
        wait=30
        name=127.0.0.1
        sharedsecret=ChangeMe
}
```

/etc/openvpn/server/script-auth-user-pass-verify

The script below is used currently to make sure that the username and the user certificate name match, otherwise, deny the login request.

Unfortunately, OpenVPN runs the RADIUS plugin in a different thread, so if you're using RADIUS to do PUSH notifications to Azure MFA, DUO, Okta, etc., the user will most likely receive a PUSH notification if they have entered correct credentials, even though there may be a certificate mismatch since these two processes are running in parallel.

```
#!/bin/bash
DUMPFILE=/var/log/openvpn/dump.all
```

```
echo "script-auth-user-pass-verify" >> $DUMPFILE
date >> $DUMPFILE
echo "==========" >> $DUMPFILE
for arg in "$@"; do
        echo "ARG: $arg" >> $DUMPFILE
done
echo "==========" >> $DUMPFILE
env >> $DUMPFILE
echo "==========" >> $DUMPFILE
echo "Pulled from environment:" >> $DUMPFILE
echo "common_name: $common_name" >> $DUMPFILE
echo "username:    $username" >> $DUMPFILE
if [ "$common_name" = "$username" ]; then
        echo "Names match!!! Allowing connection!"
        echo "==========" >> $DUMPFILE
        echo "" >> $DUMPFILE
        echo "" >> $DUMPFILE
        echo "" >> $DUMPFILE
else
        echo "$untrusted_ip:$untrusted_port script-auth-user-pass-verify: common_name and
username do not match... denying connection!!!"
        echo "common_name and username do not match... denying connection!!!" >> $DUMPFILE
        echo "==========" >> $DUMPFILE
        echo "" >> $DUMPFILE
        echo "" >> $DUMPFILE
        echo "" >> $DUMPFILE
        exit 1
fi
```

/etc/openvpn/server/script-client-connect

This script doesn't do anything currently but log available environment variables to the dump.all file. It can be used to push additional configuration details to the OpenVPN server for the user that is logging in.

```
#!/bin/bash
DUMPFILE=/var/log/openvpn/dump.all

echo "script-client-connect" >> $DUMPFILE
```

```
date >> $DUMPFILE
echo "==========" >> $DUMPFILE
for arg in "$@"; do
        echo "ARG: $arg" >> $DUMPFILE
done
echo "==========" >> $DUMPFILE
env >> $DUMPFILE
echo "==========" >> $DUMPFILE
echo "" >> $DUMPFILE
echo "" >> $DUMPFILE
echo "" >> $DUMPFILE
```

OpenVPN Client Configuration

```
setenv FRIENDLY_NAME "Meaningful name"
setenv USERNAME "actual.username@some.domain.com"
setenv ALLOW_PASSWORD_SAVE 0

remote vpn.your.domain.here.com 1194

client
proto udp
dev tun1

connect-retry 300
connect-retry-max 0
ping 20

remote-cert-tls server
auth-user-pass
auth-retry interact
cipher AES-256-GCM
auth SHA256

nobind
persist-key
persist-tun
reneg-sec 0

<ca>
```

```
-----BEGIN CERTIFICATE-----
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
-----END CERTIFICATE-----
</ca>

key-direction 1

<tls-auth>
-----BEGIN OpenVPN Static key V1-----
-----END OpenVPN Static key V1-----
</tls-auth>

<key>
-----BEGIN PRIVATE KEY-----
-----END PRIVATE KEY-----
</key>

<cert>
-----BEGIN CERTIFICATE-----
-----END CERTIFICATE-----
</cert>
```

OpenVPN Connect Client - Profile Update

The latest versions of OpenVPN Connect deprecated a few configuration options. The following configuration option that was previously acceptable has been deprecated and needs to be changed to the two separate options. If using udp instead of tcp, just change the options appropriately.

```
proto tcp-client

# needs to be changed to

client
```

```
proto tcp
```

#end

Polycom

Polycom SIP TLS with custom CA

The config files below show how to add a custom CA certificate (customCaCert1) to a Polycom phone for use with SIP TLS registration.

file: ca.cfg

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- PlcmConversionCreatedFile version=1.2 converted=Mon Feb  3 15:31:47 2014 -->
<!-- $Polycom Revision: 1.67 $  $Date: 2005/03/11 17:05:46 $ -->
<!-- certificate below is the Skyhawk CA 2020 public root certificate -->
<polycomConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="polycomConfig.xsd">
    <device.sec.TLS device.set="1"
        device.sec.TLS.customCaCert1="-----BEGIN CERTIFICATE-----
MIIDQzCCAiugAwIBAgIJALTIzl8d4AGkMA0GCSqGSIb3DQEBCwUAMBoxGDAWBgNV
BAMMD1NreWhhd2sgQ0EgMjAyMDAgFw0yMDAzMjAwMTI2MzNaGA8yMDcwMDMwODAx
MjYzM1owGjEYMBYGA1UEAwwPU2t5aGF3ayBDQSAyMDIwMIIBIjANBgkqhkiG9w0B
AQEFAAOCAQ8AMIIBCgKCAQEAplKHp7M3VO4ZGChecOlvaq8I/JOyK7L0/a/naoKp
Q3aP32V8vyK3vZFFeHbyniWcAjwSBXPsOdy2qZcEP5ukXYW8qDl3VB4nb1K/uJk/
U8pEsfKJJ1s/YEvNIoRndnnr2EWa4olH0Ec9G+mxFAsmw3i1T2wrqnaIiBv8VJpA
muJtHdzFySzqxd8Wox7WkVxC2l/PUooB1XiUIK9WOoBRE00z03vILxf7ZGnCjbGd
C1PRVWmTr6PqPHd/Ern47NLmMchJClTEweYweYtw843qfR1WN/wn8ftNrruOPoAA
bOSOF0M4Zm/yTT4qb3n8CX7T9EZg7mA6tjYqvPKsp7rGrQIDAQABo4GJMIGGMB0G
A1UdDgQWBBTWRFxu293dJmwS0RXun/U7teQDrDBKBgNVHSMEQzBBgBTWRFxu293d
JmwS0RXun/U7teQDrKEepBwwGjEYMBYGA1UEAwwPU2t5aGF3ayBDQSAyMDIwggkA
tMjOXx3gAaQwDAYDVR0TBAUwAwEB/zALBgNVHQ8EBAMCAQYwDQYJKoZIhvcNAQEL
BQADggEBAGSwOk7FKJyEcrlh4SXOnS6oLYKYM1u+uymX65Ur3WwJC6+TlBNy6joy
0786NwCZ8Cb7fdOeYC5n89Zb7rNuCGj89N5fHhgjCxJo5mbgcTYLWLE3zp8+rfhC
Q0iKK29dy9qTNTwBiBhmLZWhIJc2W+sPtANqvSGLoIbsTEp6Ktwp77B5mkaLkh09
eAf/yY7CDgDLihYQ8J2FsCeyw2jVAQZXZLHBNNkCHytAKtZqQC/tbe7aPIpAImZn
G5CJ0uKvhOjWckrFWMxMBtgB5pUoAvG/6UTyqU+N4elULVmg4wvi+bf4+gR2aN7p
UGD4YP6SHFz4LsoXH10YZd1znXa1anI=
-----END CERTIFICATE-----">
```

```
      <device.sec.TLS.customCaCert1
           device.sec.TLS.customCaCert1.set="1">
      </device.sec.TLS.customCaCert1>
</polycomConfig>
```

file: phone-prefixXXYYY.cfg

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- PlcmConversionCreatedFile version=1.2 converted=Fri Apr 25 12:00:24 2014 -->
<polycomConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="polycomConfig.xsd">
  <msg>
    <msg.mwi msg.mwi.1.callBack="9YYY" msg.mwi.1.callBackMode="contact" />
  </msg>
  <reg reg.1.address="prefixXXYYY" reg.1.displayName="YYY" reg.1.label="YYY"
reg.1.ringType="ringer3" reg.2.ringType="ringer3" reg.3.ringType="ringer3"
reg.4.ringType="ringer3" reg.5.ringType="ringer3" reg.6.ringType="ringer3">
    <reg.1.auth reg.1.auth.password="SuperSecretPassword" reg.1.auth.userId="prefixXXYYY" />
    <reg.1.server reg.1.server.1.address="pbx.url.com" reg.1.server.1.port="5061"
reg.1.server.1.transport="TLS"/>
  </reg>
</polycomConfig>
```

file: [MAC].cfg

```
<?xml version="1.0" standalone="yes"?>
<APPLICATION APP_FILE_PATH="sip.ld" CONFIG_FILES="phone-prefixXXYYY.cfg, sip.local.cfg,
ca.cfg" MISC_FILES="http://pbx.url.com/polycom-checkcfg.cfg" LOG_FILE_DIRECTORY="pcip/"
OVERRIDES_DIRECTORY="pcip/" CONTACTS_DIRECTORY="pcip/" LICENSE_DIRECTORY="license/"
USER_PROFILES_DIRECTORY="pcip/" CALL_LISTS_DIRECTORY="pcip/" COREFILE_DIRECTORY="pcip/">
   <APPLICATION_SPIP300 APP_FILE_PATH_SPIP300="sip_213.ld" CONFIG_FILES_SPIP300="phone-
prefixXXYYY.legacy.cfg, phone1_213.cfg, sip.local-legacy.cfg, sip_213.cfg"/>
   <APPLICATION_SPIP500 APP_FILE_PATH_SPIP500="sip_213.ld" CONFIG_FILES_SPIP500="phone-
prefixXXYYY.legacy.cfg, phone1_213.cfg, sip.local-legacy.cfg, sip_213.cfg"/>
   <APPLICATION_SPIP301 APP_FILE_PATH_SPIP301="sip_318.ld" CONFIG_FILES_SPIP301="phone-
prefixXXYYY.legacy.cfg, phone1_318.cfg, sip.local-legacy.cfg, sip_318.cfg"/>
   <APPLICATION_SPIP320 APP_FILE_PATH_SPIP320="sip_335.ld" CONFIG_FILES_SPIP320=""/>
   <APPLICATION_SPIP330 APP_FILE_PATH_SPIP330="sip_335.ld" CONFIG_FILES_SPIP330=""/>
   <APPLICATION_SPIP430 APP_FILE_PATH_SPIP430="sip_327.ld" CONFIG_FILES_SPIP430="phone-
```

```
prefixXXYYY.legacy.cfg, phone1_327.cfg, sip.local-legacy.cfg, sip_327.cfg"/>
   <APPLICATION_SPIP501 APP_FILE_PATH_SPIP501="sip_318.ld" CONFIG_FILES_SPIP501="phone-
prefixXXYYY.legacy.cfg, phone1_318.cfg, sip.local-legacy.cfg, sip_318.cfg"/>
   <APPLICATION_SPIP600 APP_FILE_PATH_SPIP600="sip_318.ld" CONFIG_FILES_SPIP600="phone-
prefixXXYYY.legacy.cfg, phone1_318.cfg, sip.local-legacy.cfg, sip_318.cfg"/>
   <APPLICATION_SPIP601 APP_FILE_PATH_SPIP601="sip_318.ld" CONFIG_FILES_SPIP601="phone-
prefixXXYYY.legacy.cfg, phone1_318.cfg, sip.local-legacy.cfg, sip_318.cfg"/>
   <APPLICATION_SPIP670 APP_FILE_PATH_SPIP670="sip_408.ld" CONFIG_FILES_SPIP670=""/>
   <APPLICATION_SSIP4000 APP_FILE_PATH_SSIP4000="sip_318.ld" CONFIG_FILES_SSIP4000="phone-
prefixXXYYY.legacy.cfg, phone1_318.cfg, sip.local-legacy.cfg, sip_318.cfg"/>
   <APPLICATION_SSIP6000 APP_FILE_PATH_SSIP6000="sip_408.ld" CONFIG_FILES_SSIP6000=""/>
   <APPLICATION_SSIP7000 APP_FILE_PATH_SSIP7000="sip_408.ld" CONFIG_FILES_SSIP7000=""/>
</APPLICATION>
```

References:

- How can I setup a TLS connection for SIP signaling and / or troubleshoot this?
- Install custom certificates with config files (Poly Community Forums)

Project: Teensy Climate

The primary purpose of this project was to give me a reason to learn how to develop solutions using the Atmel AVR 8-bit microcontrollers.

The secondary purpose of this project is to develop a climate control system for the home.

Current Project Status

4 Dec 2014:

Really haven't done anything with this project since the last update primarily because I wasn't finding a decent relay module for the fan attic fan controls easily enough. Earlier this week someone showed me the SainSmart relay modules. They just came in today. Hopefully I'll get to play with soon and get them integrated into the project. The ultimate goal of this project was to have a device automatically control the attic fan during the sprint, summer and autumn months. With these boards I now see a light at the end of the tunnel.

30 July 2011:

The code has been cleaned up a little bit (you should have seen versions 0.1-0.4!!!).

While I am using the uthash library, I know I'm not using it properly, but it works. While I wanted to implement a hash table, it's pretty much just creating a linked list for me and I'm iterating the list in various places. I've tried implementing a standard linked list using pointers (in version 0.6) but it's not working properly. The sensor objects are being created with the proper hardware addresses in each location, but the temperature readings are not being stored right, which is just strange. So for now I'm continuing to eat up some extra memory while I use the uthash library.

I'm trying to determine the best relay setup to use to control the attic fan. I guess I just need to get over it and by a $15-20 120-240V relay and be done with it.

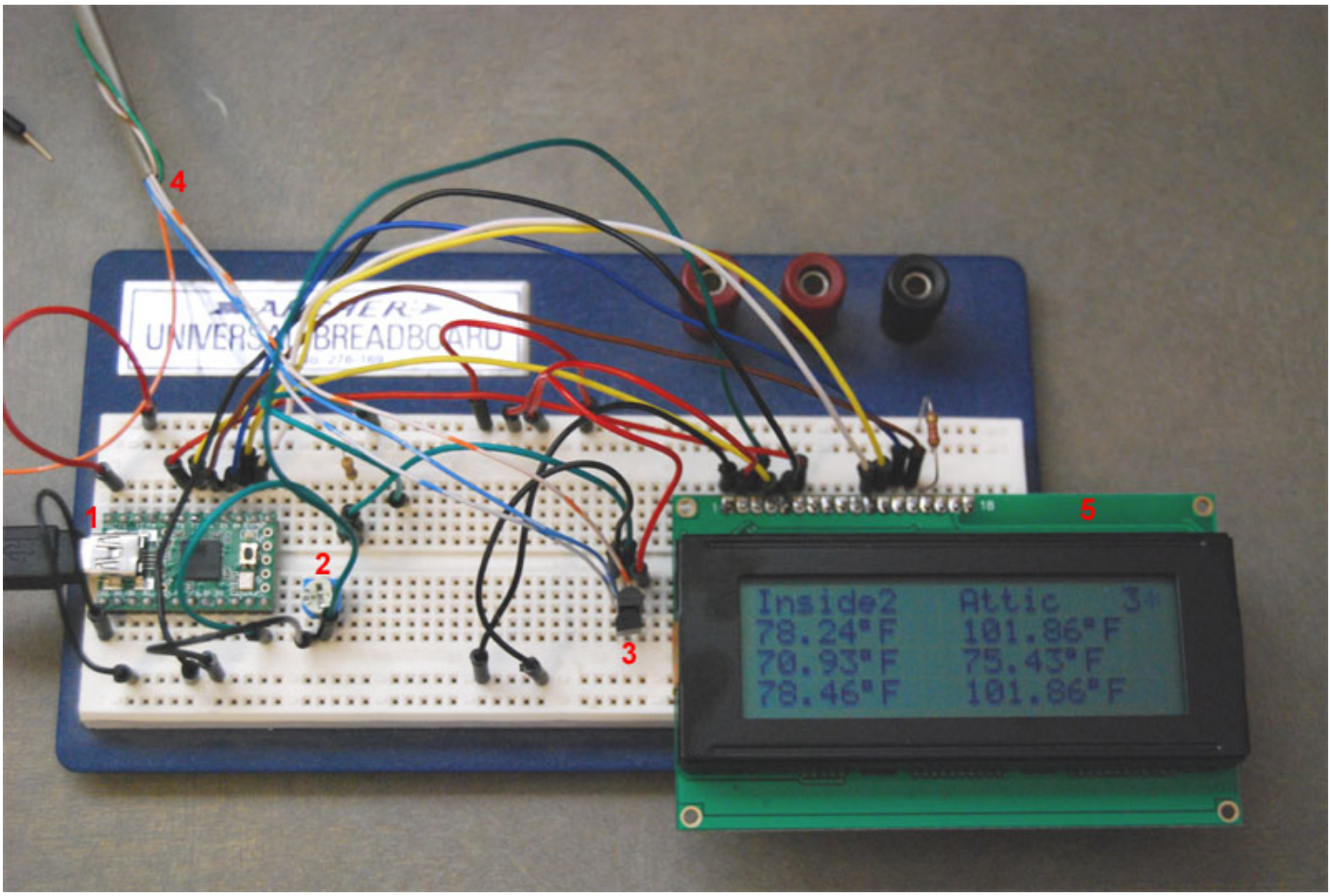
To Date min's, max's and observations

- 7/31/2011 14:11:16 - Max 131.67 @ Attic
- 8/3/2011 14:58:15 - Max: 135.16 @ Attic
- 1/16/2024 - Even though the outdoor temp was in the teens, the attic temp never dipped below freezing
- 1/21/2024 - Pending

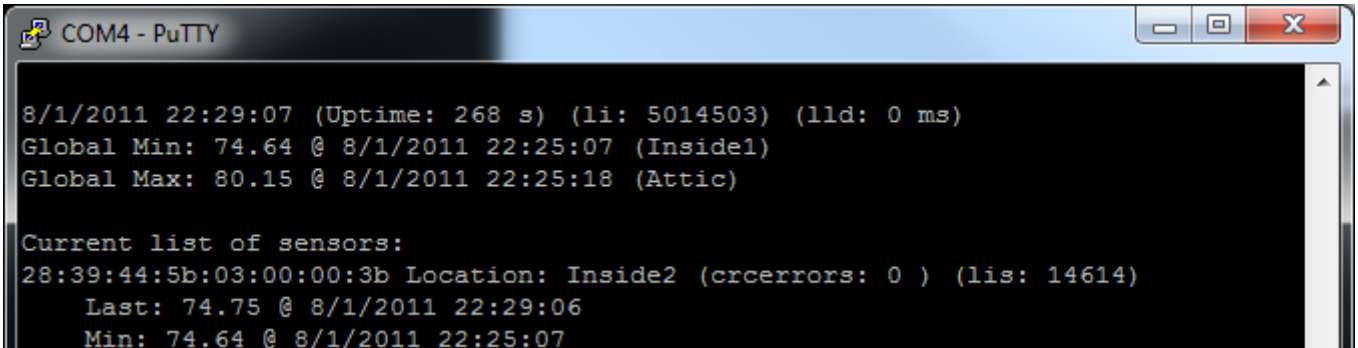
Development Photos

Breadboarded project

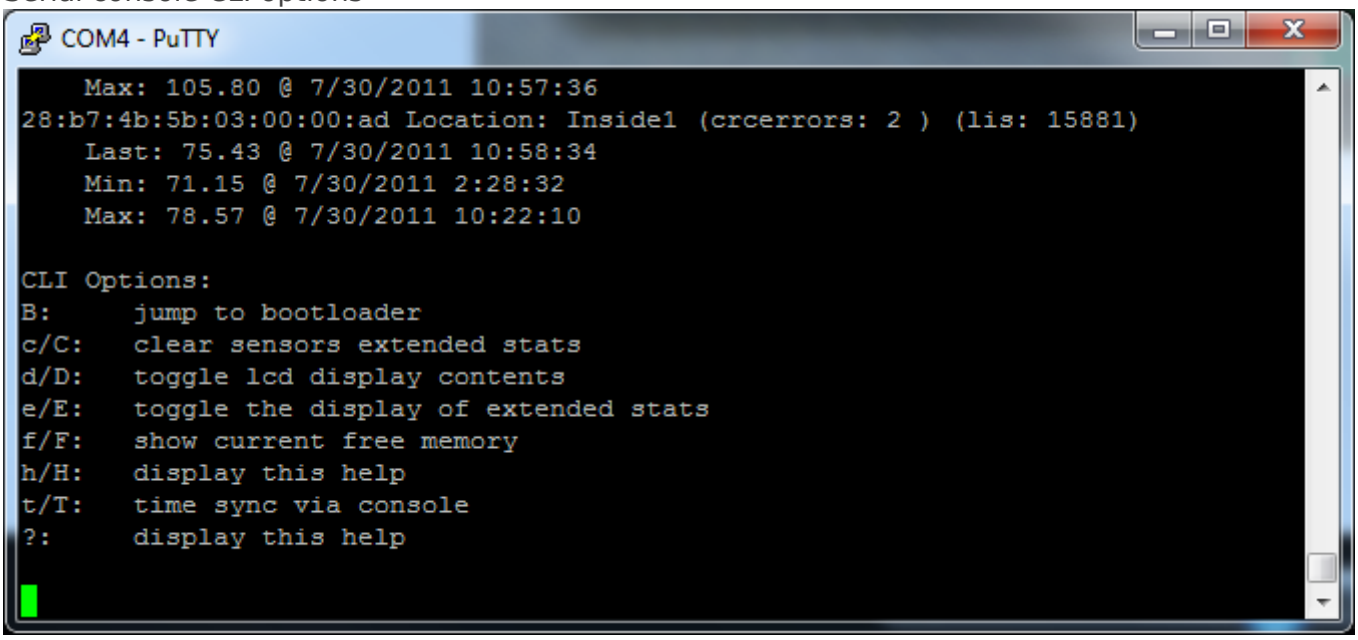
1. Teensy 2.0 Development Board
2. LCD Contrast Potentiometer (10k)
3. DS18B20 Programmable Resolution 1-Wire Digital Thermometer: two on board and one in attic
4. CAT3 extending 1-Wire bus to sensor in attic
5. HD44780 compatible LCD display

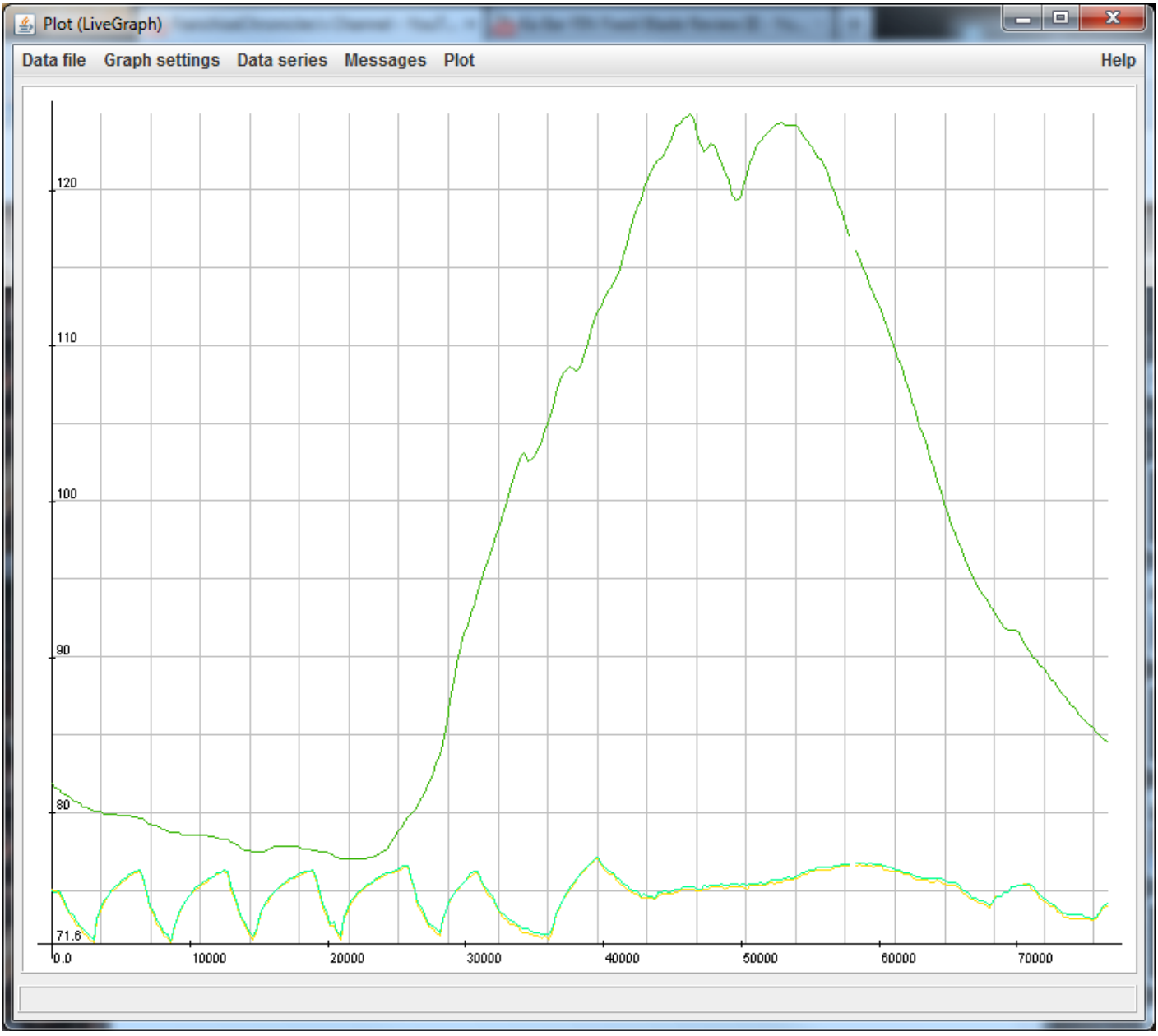


Serial console extended stats



Serial console CLI options





Graphed data from 8 Aug 2011 (dark green is attic temperature, others are inside temperature)

The CLI was updated to export data using a CSV format (millis,now(),each sensors last conversion...). A Python script was written to read the text from the serial interface and write it to a file. LiveGraph was used to read the file in real time and display the graphed data. The graph above begins at about 11:30 PM 7 Aug 2011 and ends about 11:30 PM 8 Aug 2011. One sample is output to the CLI every second. After roughly a 24 hour period the CSV file was 2.69MB in size.

Version

The current version is 0.5c.

Features

As of 0.5c (1 Aug 2011):

- Global min/max with sensor name and timestamps

As of 0.5b (30 July 2011):

- Dynamic sensor discovery on device power on
- Per sensor last/min/max temperatures with timestamps
- Non-blocking temperature conversions
- Sensor data display on LCD (currently limited to the first two sensors)
- Sensor data display on serial console (basic and extended)
- Serial console control
- Time update via serial console

To Do

- Air Conditioner Filter Change Reminder
- Attic fan relay control
- EEPROM storage of configuration information
- Ethernet TCP/IP Interface using WizNet WIZ812MJ module
- Historical statistics other than just current/min/max
  - avg/min/max per day
  - long term five minute interval graphing (via external means, SNMP possible)
- Keypad for complete device control

- PCB design
- Final migration from breadboard to PCB

Source Code

```
/*
Starting point:
 http://tushev.org/articles/electronics/42-how-it-works-ds18b20-and-arduino
 20150822 Forked from temp_05b_hash_realtime_metrof

 -= TODO =-

 A/C Filter Change Reminder

 -= EEPROM STORAGE MAP =-
 1k bytes EEPROM available
 Page size is 8 bytes
 128 Pages

 Reserve the first page for empty (due to slot 0 possiblity of being corrupted)

 Reserve pages 2-9 for configuration data

 TIMEZONE_OFFSET_HOURS - char
 DST - daylight savings time - byte
 NUM_STORED_SENSORS - byte
 AIRFILTER REPLACEMENT DATE
 AIRFILTER LIFESPAN

 Pages 10 and up are reserved for stored sensors

 STORED SENSORS:
 ADDR - byte[8] (1 page)
 NAME - char[10] (1 page plus 2 bytes)
 CALIBRATION_OFFSET - float (2 bytes)

 */
```

```
/*
 * Teensy 2.0 pinout details for this project
 * ==========================================
 * 0  -
 * 1  -
 * 2  -
 * 3  -
 * 4  -
 * 5  - * i2c SCL
 * 6  - * i2c SDA
 * 7  -
 * 8  -
 * 9  -
 * 10 - * OneWire bus
 * 11 - * LED_PIN Used for onboard signalling LED
 * 12 - * LCD Display
 * 13 - * LCD Display
 * 14 - * LCD Display
 * 15 - * LCD Display
 * 16 - * LCD Display
 * 17 - * LCD Display
 * 18 -
 * 19 -
 * 20 -
 * 21 - A0 - Connected to Adafruit GA1A12S202 Log-scale Analog Light Sensor (response from 3
to 55,000 lux)
 *
 * i2c Device Addresses
 * ====================
 * 0x29 - Adafruit TSL2591 High Dynamic Range Digital Light Sensor
 *
 */

// Uncomment the line below to enable DEBUG information. This will cause the compiled code to
increase.
//#define DEBUG 0

#define SKETCHVERSION 7

#define ACTIVITY_CHAR_INTERVAL 250
```

```
#define BLINK_INTERVAL 250
#define CONSOLE_UPDATE_INTERVAL 1000
#define DEFAULT_MIN_TEMP 999
#define DEFAULT_MAX_TEMP -99
#define DEFAULT_TIME_ZONE_OFFSET -5  // CST during DST
#define DS18B20_ID 0x28
#define DS18B20_CONVERSION_WAIT_TIME 750
#define HELP_PAUSE_METRO_INTERVAL 10000
#define LCD_CLEAR_INTERVAL 10000
#define LCD_UPDATE_INTERVAL 1000
#define LCD_STRING_BUFFER_LENGTH 21
#define LED_PIN 11 // physical pin 11

#define ANALOG_LUX_SENSOR_PIN A0 // physical pin 21
#define LIGHTANALOGSENSOR 250
#define LIGHTDIGITALSENSOR 1000

#include <avr/pgmspace.h>
#include <MemoryFree.h>
#include <LiquidCrystal.h>
#include <Metro.h>
#include <OneWire.h>
#include <String.h>
#include "C:\ArduinoProjects\lib\Streaming.h"
#include <Time.h>
#include "C:\ArduinoProjects\lib\uthash-master\uthash.h"

// 20150822 MSHARP - Adding TSL2591 light sensor support
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include "Adafruit_TSL2591.h"

enum console_mode_t {
  standard,
  extended,
  streaming
};

console_mode_t consoleMode = standard;
```

```
// ===== ===== ===== ===== ===== ===== ===== ===== ===== ===== ===== =====
// Using PROGMEM ROM to store strings
// Use printPROGMEMString() to print these strings to the serial port
// ===== ===== ===== ===== ===== ===== ===== ===== ===== ===== ===== =====

const char version_line_1[] PROGMEM = "Sketch version: ";

const char message_settingup [] PROGMEM = "Setting up...";

const char message_timesync_waiting [] PROGMEM = "Waiting for time data in @time_t format...";
const char message_timesync_updated [] PROGMEM = "Sync message received and time updated.";
const char message_timesync_invalid [] PROGMEM = "Invalid sync message received.";

const char message_bootloader_jump_1 [] PROGMEM = "Jumping to bootloader in ";
const char message_bootloader_jump_2 [] PROGMEM = "Make sure you close your serial
console!!!";
const char message_bootloader_jump_jumping [] PROGMEM = "Jumping!";

const char message_pressanykeytoabort [] PROGMEM = "PRESS ANY KEY TO ABORT!!!";
const char message_aborted [] PROGMEM = "Aborted!";

// ===== ===== ===== ===== ===== ===== ===== ===== ===== ===== ===== =====

#define CLI_STRING_BUFFER_LENGTH 50
#define NUM_CLI_LINES 13

const char cli_line_1[] PROGMEM  = "CLI Options:";
const char cli_line_2[] PROGMEM  = "B:     jump to bootloader";
const char cli_line_3[] PROGMEM  = "c:     cycle console mode";
const char cli_line_4[] PROGMEM  = "C:     clear sensors stats";
const char cli_line_5[] PROGMEM  = "d/D:   toggle lcd display contents";
const char cli_line_6[] PROGMEM  = "f/F:   show current free memory";
const char cli_line_7[] PROGMEM  = "G:     reset global sensor stats";
const char cli_line_8[] PROGMEM  = "h:     display this help";
const char cli_line_9[] PROGMEM  = "l:     query analog light sensor and display result";
const char cli_line_10[] PROGMEM = "L:     query digital light sensor and display result";
const char cli_line_11[] PROGMEM = "s/S:   set console mode to streaming";
const char cli_line_12[] PROGMEM = "T:     time sync via console";
const char cli_line_13[] PROGMEM = "v/V:   show version";
```

```
const char* const cli_string_table[] PROGMEM =
{
  cli_line_1,
  cli_line_2,
  cli_line_3,
  cli_line_4,
  cli_line_5,
  cli_line_6,
  cli_line_7,
  cli_line_8,
  cli_line_9,
  cli_line_10,
  cli_line_11,
  cli_line_12,
  cli_line_13
};

// ===== ===== ===== ===== ===== ===== ===== ===== ===== ===== ===== =====

// These are sensors that I know I have. Eventually we'll store this in EEPROM programatically
byte tempSensorAttic[8] = {
  0x28, 0xdb, 0x32, 0x5b, 0x03, 0x00, 0x00, 0x8d
};
byte tempSensorInside1[8] = {
  0x28, 0xb7, 0x4b, 0x5b, 0x03, 0x00, 0x00, 0xad
};
byte tempSensorInside2[8] = {
  0x28, 0x39, 0x44, 0x5b, 0x03, 0x00, 0x00, 0x3b
};

boolean consolePaused = false;
boolean lcdDisplayAddresses = false;
boolean showExtendedStats = false;

unsigned int activityCharIndex = 0;
unsigned int ledStatus = 0;

unsigned long loopIteration = 0;
unsigned long loopStartMillis;
unsigned long lastLoopDuration = 0;
```

```
// LCD Display using 6 pins (12-17)
LiquidCrystal lcd(16, 17, 12, 13, 14, 15);

// Adafruit TSL2591 High Dynamic Range Digital Light Sensor
boolean tslFound = false;
Adafruit_TSL2591 tsl = Adafruit_TSL2591(2591);

// Global variables for templerature sensors
const int luxAnalogNumReadings = 10;
float luxAnalogReadings[luxAnalogNumReadings];
float luxAnalogRaw = 0.0;
int luxAnalogRawIndex = 0;
float luxAnalogRawTotal = 0.0;
float luxAnalogRawAverage = 0.0;
float luxAnalogLog = 0.0;
float luxAnalogLogSmoothed = 0.0;

uint32_t luxDigitalFullLuminosity;
uint16_t luxDigitalIR, luxDigitalFull, luxDigitalCalculated;

// Manually update this number when you add a Metro below
#define NUM_METROS 7

Metro activityCharMetro(ACTIVITY_CHAR_INTERVAL, false);
Metro consoleUpdateMetro(CONSOLE_UPDATE_INTERVAL);
Metro helpPauseMetro(HELP_PAUSE_METRO_INTERVAL, false);
Metro ledMetro(BLINK_INTERVAL, false);
Metro lcdUpdateMetro(LCD_UPDATE_INTERVAL);
Metro lcdClearMetro(LCD_CLEAR_INTERVAL);  // counter to clear the lcd every 10 seconds for
housekeeping
Metro owBusSearchMetro(600000, false);  //  counter to search the bus every 10 minutes for new
devices
Metro lightAnalogSensor(LIGHTANALOGSENSOR, false);
Metro lightDigitalSensor(LIGHTDIGITALSENSOR, false);

// OneWire bus on pin 10
OneWire ds(10);

float globalMin = DEFAULT_MIN_TEMP;
```

```
char * globalMinName = NULL;
time_t globalMinTimeStamp = (time_t) 0;
float globalMax = DEFAULT_MAX_TEMP;
char * globalMaxName = NULL;
time_t globalMaxTimeStamp = (time_t) 0;

struct DS18B20 {
  byte addr[8];              /* key */
  char name[10];
  boolean active;
  boolean converting;  /* true if a conversion has been requested */
  unsigned int crcerrors;
  unsigned long startConversionLI;  // loop iteration of the start conversion
  unsigned long liLastConversion;    // number of loop iterations for the last conversion
  Metro conversionTimer;
  float lastTemp;
  time_t lastTimeStamp;
  float minTemp;
  time_t minTimeStamp;
  float maxTemp;
  time_t maxTimeStamp;
  UT_hash_handle hh;         /* makes this structure hashable */
};

struct DS18B20 *tempSensors = NULL;

boolean compareByteArray(byte a1[], byte a2[]) {
  int arraySize = sizeof(a1) / sizeof(byte);
  if (sizeof(a1) != sizeof(a2)) {
    return false;
  }
  for (int i = 0; i < arraySize; i++) {
    if (a1[i] != a2[i]) {
      return false;
    }
  }
  return true;
}

void add_sensor(byte addr[]) {
```

```
  struct DS18B20 *s;

  s = (DS18B20 *) malloc(sizeof(struct DS18B20));
  for (int i = 0; i < 8; i++) {
    s->addr[i] = addr[i];
  }
  s->active = false;
  s->converting = false;
  s->crcerrors = 0;
  s->conversionTimer = Metro(DS18B20_CONVERSION_WAIT_TIME, false);
  s->liLastConversion = 0;
  s->lastTemp = 0.0;
  s->lastTimeStamp = now();
  s->minTemp = DEFAULT_MIN_TEMP;
  s->minTimeStamp = s->lastTimeStamp;
  s->maxTemp = DEFAULT_MAX_TEMP;
  s->maxTimeStamp = s->lastTimeStamp;
  if (compareByteArray(s->addr, tempSensorAttic)) {
    strcpy(s->name, "Attic");
  }
  else if (compareByteArray(s->addr, tempSensorInside1)) {
    strcpy(s->name, "Inside1");
  }
  else if (compareByteArray(s->addr, tempSensorInside2)) {
    strcpy(s->name, "Inside2");
  }
  else {
    strcpy(s->name, "Unknown");
  }
  HASH_ADD(hh, tempSensors, addr, sizeof(byte) * 8, s);
}

struct DS18B20 *find_sensor(byte addr[]) {
  struct DS18B20 *s;

#ifdef DEBUG
  Serial.print("find_sensor(");
  Serial.print(OneWireaddrtostring(addr, false));
  Serial.println(")");
#endif
```

```
  for (s = tempSensors; s != NULL; s = (DS18B20 *) s->hh.next) {
#ifdef DEBUG
    Serial.println("Comparing:");
    Serial.print(OneWireaddrtostring(s->addr, false));
    Serial.print(" to ");
    Serial.println(OneWireaddrtostring(addr, false));
#endif

    if (compareByteArray(addr, s->addr)) {
#ifdef DEBUG
      Serial.println("  match found... sensor already detected");
#endif
      return s;
    }
  }

#ifdef DEBUG
  Serial.println("no match found... returning NULL");
#endif
  return NULL;
}

void update_console() {
  struct DS18B20 *s;

  // Do a digital lux reading before we attempt displaying anything... this will make the
display update more pleasing
  doLuxReadingDigital();

  if ((consoleMode == standard) || (consoleMode == extended)) {
    // Now lets start putting stuff on the console
    serialPrintDateTime(now());
    Serial << " (Uptime: " << millis() / 1000 << " s) (li: " << loopIteration << ") (lld: " <<
lastLoopDuration << " ms)" << endl;
    Serial << "Global Min: " << globalMin << " @ ";
    serialPrintDateTime(globalMinTimeStamp);
    Serial << " (" << globalMinName << ")" << endl;
    Serial << "Global Max: " << globalMax << " @ ";
    serialPrintDateTime(globalMaxTimeStamp);
```

```
    Serial << " (" << globalMaxName << ")" << endl << endl;
    Serial.println("Current list of sensors:");

    for (s = tempSensors; s != NULL; s = (DS18B20 *) s->hh.next) {
      Serial.print(OneWireaddrtostring(s->addr, false));
      if (s->active) {
        if (consoleMode == standard) {
          // show standard information
          Serial.print(" ");
          Serial.print(s->lastTemp);
          Serial.print("/");
          Serial.print(s->minTemp);
          Serial.print("/");
          Serial.print(s->maxTemp);
          Serial.print(" F, Location: ");
          Serial.println(s->name);
        }
        else {
          // show extended stats
          Serial << " Location: " << s->name << " (crcerrors: " << s->crcerrors << " ) (lis: "
<< s->liLastConversion << ")" << endl;
          Serial << "    Last: " << s->lastTemp << " @ ";
          serialPrintDateTime(s->lastTimeStamp);
          Serial << endl;
          Serial << "    Min: " << s->minTemp << " @ ";
          serialPrintDateTime(s->minTimeStamp);
          Serial << endl;
          Serial << "    Max: " << s->maxTemp << " @ ";
          serialPrintDateTime(s->maxTimeStamp);
          Serial << endl;
        }
      }
      else {
        Serial.println(" is pending first read.");
      }
    }
    // Display light sensor data now
    Serial.println();
    displayAnalogLightSensorData();
    displayDigitalLightSensorData();
```

```
}
else {
  // streaming
  //        Serial << "@" << endl;
  //     serialPrintDateTime(now());
  Serial << millis() << "," << now();
  Serial.print(",");
  /*     Serial << "," << millis() / 1000 << endl;
       Serial << globalMin << ",";
       serialPrintDateTime(globalMinTimeStamp);
       Serial << "," << globalMinName << endl;
       Serial << globalMax << ",";
       serialPrintDateTime(globalMaxTimeStamp);
       Serial << "," << globalMaxName << endl;
       */
  for (s = tempSensors; s != NULL; s = (DS18B20 *) s->hh.next) {
    //      Serial.print(OneWireaddrtostring(s->addr, false));
    //      Serial.print(",");
    Serial.print(s->lastTemp);
    if (s->hh.next != NULL) {
      Serial.print(",");
    }
    /*      serialPrintDateTime(s->lastTimeStamp);
     Serial.print(",");
     Serial.print(s->minTemp);
     Serial.print(",");
     serialPrintDateTime(s->minTimeStamp);
     Serial.print(",");
     Serial.print(s->maxTemp);
     Serial.print(",");
     serialPrintDateTime(s->maxTimeStamp);
     Serial.print(",");
     Serial.println(s->name); */
  }

  Serial.print(",LA,");
  Serial.print(luxAnalogReadings[luxAnalogRawIndex]);
  Serial.print(",");
  Serial.print(luxAnalogRawAverage);
  Serial.print(",");
```

```
    Serial.print(pow(10, luxAnalogLog));
    Serial.print(",");
    Serial.print(pow(10, luxAnalogLogSmoothed));

    if (tslFound) {
      Serial.print(",LD,");
      Serial.print("IR,"); Serial.print(luxDigitalIR);  Serial.print(",");
      Serial.print("F,"); Serial.print(luxDigitalFull); Serial.print(",");
      Serial.print("V,"); Serial.print(luxDigitalFull - luxDigitalIR); Serial.print(",");
      Serial.print("L,"); Serial.print(luxDigitalCalculated);
    }
  }
  Serial.println();
}

boolean update_sensor(byte addr[], float temp) {
  struct DS18B20 *s;

  s = find_sensor(addr);
  if (s != NULL) {
    if (!s->active) {
      s->active = true;
    }

    s->lastTemp = temp;
    s->lastTimeStamp = now();

    if (temp < s->minTemp) {
      s->minTemp = temp;
      s->minTimeStamp = now();
    }

    if (temp < globalMin) {
      globalMin = temp;
      globalMinName = s->name;
      globalMinTimeStamp = now();
    }

    if (temp > s->maxTemp) {
      s->maxTemp = temp;
```

```
      s->maxTimeStamp = now();
    }

    if (temp > globalMax) {
      globalMax = temp;
      globalMaxName = s->name;
      globalMaxTimeStamp = now();
    }

    return true;
  }
  else {
#ifdef DEBUG
    Serial.print("update_sensor() failed for addr ");
    Serial.print(OneWireaddrtostring(addr, false));
    Serial.print(" ");
    Serial.print(temp);
    Serial.println(" F");
#endif
    return false;
  }
}

int count_sensors() {
  struct DS18B20 *s;
  int count = 0;
  for (s = tempSensors; s != NULL; s = (DS18B20 *) s->hh.next, count++) {
  }
  return count;
}

void clear_sensor_stats() {
  struct DS18B20 *s;

  for (s = tempSensors; s != NULL; s = (DS18B20 *) s->hh.next) {
    s->minTemp = DEFAULT_MIN_TEMP;
    s->minTimeStamp = (time_t) 0;
    s->maxTemp = DEFAULT_MAX_TEMP;
    s->maxTimeStamp = (time_t) 0;
  }
```

```
}

void clear_global_sensor_stats() {
  globalMin = DEFAULT_MIN_TEMP;
  globalMinName = NULL;
  globalMinTimeStamp = (time_t) 0;
  globalMax = DEFAULT_MAX_TEMP;
  globalMaxName = NULL;
  globalMaxTimeStamp = (time_t) 0;
}

void doLuxReadingAnalog() {
  float rawRange = 1024; // 3.3v
  float logRange = 5.0;  // 3.3 = 10^5 lux

  luxAnalogRawTotal = luxAnalogRawTotal - luxAnalogReadings[luxAnalogRawIndex];
  luxAnalogReadings[luxAnalogRawIndex] = analogRead(ANALOG_LUX_SENSOR_PIN);
  luxAnalogRawTotal = luxAnalogRawTotal + luxAnalogReadings[luxAnalogRawIndex];
  luxAnalogRawAverage = luxAnalogRawTotal / luxAnalogNumReadings;
  luxAnalogLog = luxAnalogRawAverage * logRange / rawRange;
  luxAnalogLogSmoothed = luxAnalogRawAverage * logRange / rawRange;

  luxAnalogRawIndex = luxAnalogRawIndex + 1;
  if (luxAnalogRawIndex >= luxAnalogNumReadings){
    luxAnalogRawIndex = 0;
  }

}

void displayAnalogLightSensorData() {
  Serial.print("Light [analog] : ");
  Serial.print("Raw: ");
  Serial.print(luxAnalogReadings[luxAnalogRawIndex]);
  Serial.print("  Smoothed: ");
  Serial.print(luxAnalogRawAverage);
  Serial.print("  Log: ");
  Serial.print(pow(10, luxAnalogLog));
  Serial.print("  Smoothed: ");
  Serial.println(pow(10, luxAnalogLogSmoothed));
}
```

```
// Updates global variables to be used in displayDigitalLightSensorDataData()
void doLuxReadingDigital() {
  if (!tslFound) {
    Serial.println("No TSL2591 device has been found.");
    return;
  }

  // You can change the gain on the fly, to adapt to brighter/dimmer light situations
  //tsl.setGain(TSL2591_GAIN_LOW);    // 1x gain (bright light)
  tsl.setGain(TSL2591_GAIN_MED);      // 25x gain
  //tsl.setGain(TSL2591_GAIN_HIGH);   // 428x gain

  // Changing the integration time gives you a longer time over which to sense light
  // longer timelines are slower, but are good in very low light situtations!
  tsl.setTiming(TSL2591_INTEGRATIONTIME_100MS);  // shortest integration time (bright light)
  //tsl.setTiming(TSL2591_INTEGRATIONTIME_200MS);
  //tsl.setTiming(TSL2591_INTEGRATIONTIME_300MS);
  //tsl.setTiming(TSL2591_INTEGRATIONTIME_400MS);
  //tsl.setTiming(TSL2591_INTEGRATIONTIME_500MS);
  //tsl.setTiming(TSL2591_INTEGRATIONTIME_600MS);  // longest integration time (dim light)

  luxDigitalFullLuminosity = tsl.getFullLuminosity();
  luxDigitalIR = luxDigitalFullLuminosity >> 16;
  luxDigitalFull = luxDigitalFullLuminosity & 0xFFFF;
  luxDigitalCalculated = tsl.calculateLux(luxDigitalFull, luxDigitalIR);
}

void displayDigitalLightSensorData() {
  if (!tslFound) {
    Serial.println("No TSL2591 device has been found.");
    return;
  }

  Serial.print("Light [digital]: ");
  Serial.print("IR: "); Serial.print(luxDigitalIR);  Serial.print("  ");
  Serial.print("Full: "); Serial.print(luxDigitalFull); Serial.print("  ");
  Serial.print("Visible: "); Serial.print(luxDigitalFull - luxDigitalIR); Serial.print("  ");
  Serial.print("Lux: "); Serial.print(luxDigitalCalculated);
  Serial.println();
```

```
}

boolean findDS18B20Devices(OneWire & ow) {
  byte addr[8];
  unsigned int deviceCount = 0;

#ifdef DEBUG
  Serial.println("Searching bus...");
#endif

  //find a device
  while (ow.search(addr)) {
    if (OneWire::crc8( addr, 7) != addr[7]) {
      Serial.println("Bad crc!!!");
      continue;
    }

    if (addr[0] != DS18B20_ID) {
#ifdef DEBUG
      Serial.print("Unknown device: ");
#endif
      Serial.println(OneWireaddrtostring(addr, false));
      continue;
    }

#ifdef DEBUG
    Serial.print("Found a device:");
    Serial.println(OneWireaddrtostring(addr, false));
#endif

    deviceCount++;
    if (find_sensor(addr) == NULL) {
#ifdef DEBUG
      Serial.print("Adding new sensor to list: ");
#endif
      add_sensor(addr);
    }
#ifdef DEBUG
    else {
      Serial.print("We already know about this sensor: ");
```

```
    }
#endif
    Serial.println(OneWireaddrtostring(addr, false));
    Serial.println();
  }

#ifdef DEBUG
  Serial.println("No more devices found... resetting search.");
  Serial.println();
#endif
  ow.reset_search();
}

boolean requestTemperatureConversion(OneWire ow, DS18B20 *sensor) {
  ow.reset();
  ow.select(sensor->addr);
  ow.write(0x44, 1);

  return true;
}

float retrieveTemperature(OneWire ow, DS18B20 *sensor) {
  byte data[12];
  float temp;

  ow.reset();
  ow.select(sensor->addr);
  ow.write(0xBE);
  for (int i = 0; i < 9; i++) {
    data[i] = ow.read();
  }

  temp = ( (data[1] << 8) + data[0] ) * 0.0625; // tempc
  temp = (temp * 1.8) + 32;  // tempf

  if (OneWire::crc8( data, 8) != data[8]) {
    temp = -9999;
  }

  return temp;
```

```
}

void toggleLED() {
  switch (ledStatus) {
    case 1:
      digitalWrite(LED_PIN, LOW);
      ledStatus = 0;
      break;
    default:
      digitalWrite(LED_PIN, HIGH);
      ledStatus = 1;
      break;
  }
}

void showActivityChar() {
  lcd.setCursor(18, 0);
  switch (activityCharIndex) {
    case 1:
      activityCharIndex--;
      lcd.print(count_sensors());
      lcd.print(" ");
      break;
    default:
      activityCharIndex++;
      lcd.print(count_sensors());
      lcd.print("*");
      break;
  }
}

String OneWireaddrtostring(byte addr[], boolean lcd) {
  String toReturn;

  for ( int i = 0; i < 8; i++) {
    if (addr[i] < 16) {
      toReturn += "0";
    }
    toReturn += String(addr[i], HEX);
    if (i < 7) {
```

```
      if (!lcd) {
        // don't print semicolons on the lcd
        // we don't have enough room
        toReturn += ":";
      }
    }
  }

  return toReturn;
}

void update_lcd() {
  struct DS18B20 *s;
  unsigned int count;
  unsigned int maxCount = 2;

  s = tempSensors;
  if (lcdDisplayAddresses) {
    maxCount = 3;
  }

  if (lcdClearMetro.check() == 1) {
    lcd.clear();
    lcdClearMetro.reset();
  }

  if (lcdDisplayAddresses) {
    lcd.setCursor(0, 0);
    lcd.print("Uptime: ");
    lcd.print(millis() / 1000);
    lcd.print("s ");
  }

  for (count = 0; count < maxCount; count++, s = (DS18B20 *) s->hh.next) {
    if (s == NULL) {
      break;
    }

    if (!lcdDisplayAddresses) {
      if (!s->active) {
```

```
        continue;
      }
      lcd.setCursor(count * 10, 0);
      lcd.print(s->name);
      lcd.setCursor(count * 10, 1);
      lcd.print(s->lastTemp);
      lcd.print((char)223);
      lcd.print("F");
      lcd.setCursor(count * 10, 2);
      lcd.print(s->minTemp);
      lcd.print((char)223);
      lcd.print("F");
      lcd.setCursor(count * 10, 3);
      lcd.print(s->maxTemp);
      lcd.print((char)223);
      lcd.print("F");
    }
    else {
      // display addresses instead of temps
      lcd.setCursor(0, count + 1);
      lcd.print("    ");
      lcd.print(OneWireaddrtostring(s->addr, true));
      lcd.setCursor(0, count + 1);
      lcd.print(s->name);
      lcd.print(":");
    }
  }
}

void serialPrintDateTime(time_t timeStamp) {
  time_t timeStampAdjusted = timeStamp + (DEFAULT_TIME_ZONE_OFFSET * 60 * 60);
  Serial << month(timeStampAdjusted) << "/" << day(timeStampAdjusted) << "/" <<
year(timeStampAdjusted) << " " << hour(timeStampAdjusted) << ":";
  if (minute(timeStampAdjusted) < 10) Serial << "0";
  Serial << minute(timeStampAdjusted) << ":";
  if (second(timeStampAdjusted) < 10) Serial << "0";
  Serial << second(timeStampAdjusted);
}

void processTimeSyncMessage() {
```

```
  int count = 0;
  char buf[11];
  boolean status = false;  // did we get good data
  printPROGMEMString(message_timesync_waiting); // "Waiting for time data in @time_t
format..."
  Serial.println();
  Serial.flush();
  while (count < 11) {
    if (Serial.available()) {  // receive all 11 bytes into "buf"
      buf[count++] = Serial.read();
    }
  }
  if (buf[0] == '@') {
    time_t pctime = 0;
    for (int i = 1; i < 11; i++) {
      char c = buf[i];
      if (c >= '0' && c <= '9') {
        pctime = (10 * pctime) + (c - '0') ; // convert digits to a number
      }
    }
    pctime += 10;
    setTime(pctime);   // Sync clock to the time received
    status = true;
  }
  if (status) {
    // "Sync message received and time updated."
    printPROGMEMString(message_timesync_updated);
    Serial.println();
  }
  else {
    // "Invalid sync message received."
    printPROGMEMString(message_timesync_invalid);
    Serial.println();
  }
}

void jumpToBootloader() {
  unsigned int counter = 10;
  printPROGMEMString(message_bootloader_jump_1); // "Jumping to bootloader in "
  Serial << counter << " seconds...";
```

```
  Serial.println();
  printPROGMEMString(message_bootloader_jump_2); // "Make sure you close your serial
console!!!"
  Serial.println(); Serial.println();
  printPROGMEMString(message_pressanykeytoabort); // "PRESS ANY KEY TO ABORT!!!"
  Serial.println(); Serial.println();
  Serial.flush();
  lcd.clear();
  while (counter > 0) {
    lcd.setCursor(0, 0);
    lcd.print(counter);
    Serial << counter << " ";
    if (Serial.available()) {
      Serial.flush();
      Serial.println(); Serial.println();
      printPROGMEMString(message_aborted); // "Aborted!"
      Serial.println(); Serial.println(); Serial.println();
      return;
    }
    delay(1000);
    counter--;
  }
  printPROGMEMString(message_bootloader_jump_jumping); // "Jumping!"
  Serial.println();
  cli();
  // disable watchdog, if enabled
  // disable all peripherals
  UDCON = 1;
  USBCON = (1 << FRZCLK); // disable USB
  UCSR1B = 0;
  delay(5);
#if defined(__AVR_AT90USB162__)                // Teensy 1.0
  EIMSK = 0; PCICR = 0; SPCR = 0; ACSR = 0; EECR = 0;
  TIMSK0 = 0; TIMSK1 = 0; UCSR1B = 0;
  DDRB = 0; DDRC = 0; DDRD = 0;
  PORTB = 0; PORTC = 0; PORTD = 0;
  asm volatile("jmp 0x3E00");
#elif defined(__AVR_ATmega32U4__)              // Teensy 2.0
  EIMSK = 0; PCICR = 0; SPCR = 0; ACSR = 0; EECR = 0; ADCSRA = 0;
  TIMSK0 = 0; TIMSK1 = 0; TIMSK3 = 0; TIMSK4 = 0; UCSR1B = 0; TWCR = 0;
```

```
  DDRB = 0; DDRC = 0; DDRD = 0; DDRE = 0; DDRF = 0; TWCR = 0;
  PORTB = 0; PORTC = 0; PORTD = 0; PORTE = 0; PORTF = 0;
  asm volatile("jmp 0x7E00");
#elif defined(__AVR_AT90USB646__)              // Teensy++ 1.0
  EIMSK = 0; PCICR = 0; SPCR = 0; ACSR = 0; EECR = 0; ADCSRA = 0;
  TIMSK0 = 0; TIMSK1 = 0; TIMSK2 = 0; TIMSK3 = 0; UCSR1B = 0; TWCR = 0;
  DDRA = 0; DDRB = 0; DDRC = 0; DDRD = 0; DDRE = 0; DDRF = 0;
  PORTA = 0; PORTB = 0; PORTC = 0; PORTD = 0; PORTE = 0; PORTF = 0;
  asm volatile("jmp 0xFC00");
#elif defined(__AVR_AT90USB1286__)             // Teensy++ 2.0
  EIMSK = 0; PCICR = 0; SPCR = 0; ACSR = 0; EECR = 0; ADCSRA = 0;
  TIMSK0 = 0; TIMSK1 = 0; TIMSK2 = 0; TIMSK3 = 0; UCSR1B = 0; TWCR = 0;
  DDRA = 0; DDRB = 0; DDRC = 0; DDRD = 0; DDRE = 0; DDRF = 0;
  PORTA = 0; PORTB = 0; PORTC = 0; PORTD = 0; PORTE = 0; PORTF = 0;
  asm volatile("jmp 0x1FC00");
#endif
}

void printCLIOptions() {
//  char buf[CLI_STRING_BUFFER_LENGTH];

  for (int i = 0; i < NUM_CLI_LINES; i++) {
//    strcpy_P(buf, (char*)pgm_read_word(&(cli_string_table[i])));
//    Serial.println(buf);
    printPROGMEMString((char*) pgm_read_word(&(cli_string_table[i])));
    Serial.println();
  }

  Serial.println();
}

// 20150822 MSHARP - Added function
void printPROGMEMString(const char* PMSTRING) {
  int i;
  int len = strlen_P(PMSTRING);
  char nextCharacter;
  for (i = 0; i < len; i++) {
    nextCharacter = pgm_read_byte_near(PMSTRING + i);
    Serial.print(nextCharacter);
  }
```

```
}

void setup() {
  // Initialize the display and tell the world we're starting to work
  lcd.begin(20, 4);
  lcd.setCursor(0, 0);
  lcd.print("Setting up...");

  pinMode(LED_PIN, OUTPUT);
  ledMetro.reset();
  Serial.begin(9600);

  // displaying activity char so we know we've started the initial 1-wire search
  showActivityChar();

  // Give human 10 seconds to connect via serial to watch for debug information
#ifdef DEBUG
  for (int i = 0; i < 10; i++) {
    Serial << i << " ";
    delay(1000);
  }
  Serial << endl;
#endif

  findDS18B20Devices(ds);

  // Setup Adafruit i2c TSL sensor
  if (tsl.begin()) {
    Serial.println("Found a TSL2591 sensor!!!");
    tslFound = true;
  }

  // Give human 10 seconds to view debug information from device scan
#ifdef DEBUG
  for (int i = 0; i < 10; i++) {
    Serial << i << " ";
    delay(1000);
  }
  Serial << endl;
```

```
#endif

  // Initialize analog lux reading smoothing array
  for (int thisReading = 0; thisReading < luxAnalogNumReadings; thisReading++) {
    luxAnalogReadings[thisReading] = 0.0;
  }
} // end setup

void loop() {
  loopIteration++;
  loopStartMillis = millis();

  float tmpTemp = 0;
  struct DS18B20 *s;

  if (activityCharMetro.check() == 1) {
    showActivityChar();
    activityCharMetro.reset();
  }

  // manage the sensors
  for (s = tempSensors; s != NULL; s = (DS18B20 *) s->hh.next) {
    if (s->converting) {
      if (s->conversionTimer.check() == 1) {
        tmpTemp = retrieveTemperature(ds, s);
        if (tmpTemp != -9999) {
          update_sensor(s->addr, tmpTemp);
        }
        else {
          s->crcerrors++;
        }
        s->liLastConversion = loopIteration - s->startConversionLI;
        s->converting = false;
        tmpTemp = 0;
      }
    }
    else {
      requestTemperatureConversion(ds, s);
      s->startConversionLI = loopIteration;
      s->conversionTimer.reset();
```

```
      s->converting = true;
    }

  } // for tempSensors

  // manage analog lux sensor
  if (lightAnalogSensor.check() == 1) {
    doLuxReadingAnalog();
    lightAnalogSensor.reset();
  }



  // process command line input
  if (Serial.available() > 0) {
    char c = Serial.read();
    switch (c) {
      case 'B':
        jumpToBootloader();
        break;
      case 'c':
        if (consoleMode == streaming) {
#ifdef DEBUG
          Serial.println("Setting console mode to standard.");
#endif
          consoleMode = standard;
        }
        else if (consoleMode == standard) {
#ifdef DEBUG
          Serial.println("Setting console mode to extended.");
#endif
          consoleMode = extended;
        }
        else if (consoleMode == extended) {
#ifdef DEBUG
          Serial.println("Setting console mode to streaming.");
#endif
          consoleMode = streaming;
        }
        break;
      case 'C':
```

```
        clear_sensor_stats();
        break;
      case 'd':
      case 'D':
        if (lcdDisplayAddresses) {
#ifdef DEBUG
          Serial.println("Switching LCD to display sensor values");
#endif
          lcdDisplayAddresses = false;
        }
        else {
#ifdef DEBUG
          Serial.println("Switching LCD to display sensor addresses");
#endif
          lcdDisplayAddresses = true;
        }
        Serial.println();
        Serial.flush();
        lcd.clear();
        break;
      case 'f':
      case 'F':
        Serial << "Free memory: " << freeMemory() << " bytes." << endl;
#ifdef DEBUG
        Serial << sizeof(DS18B20) * count_sensors() << " bytes for " << count_sensors() << "
DS18B20 sensors" << endl;
        Serial << sizeof(LiquidCrystal) << " bytes for LiquidCrystal object" << endl;
        Serial << sizeof(Metro) * NUM_METROS << " bytes for " << NUM_METROS << " Metro
objects" << endl;
        Serial << sizeof(OneWire) << " bytes for OneWire object" << endl;
#endif
        Serial << endl;
        break;
      case 'G':
        clear_global_sensor_stats();
        break;
      case 'l':
        displayAnalogLightSensorData();
        break;
      case 'L':
```

```
        displayDigitalLightSensorData();
        break;
      case 's':
      case 'S':
#ifdef DEBUG
        Serial.println("Setting console mode to streaming.");
#endif
        consoleMode = streaming;
        break;
      case 'T':
#ifdef DEBUG
        Serial.println("Processing time sync request...");
#endif
        processTimeSyncMessage();
#ifdef DEBUG
        Serial.println("Done!");
#endif
        Serial.println();
        break;
      case 'h':
      case 'H':
      case '?':
        printCLIOptions();
        // consoleUpdateMetro.autoreset(false); // 20150822 MSHARP commented out because
autoreset is now private method
        consoleUpdateMetro.reset();
        helpPauseMetro.reset();
        consolePaused = true;
        break;
      case 'v':
      case 'V':
        printPROGMEMString(version_line_1);
        Serial.print(" ");
        Serial.println(SKETCHVERSION);
        Serial.println();
        break;
      default:
        Serial.print("Key: 0x");
        Serial.println(c, HEX);
        printCLIOptions();
```

```
    } // testing c
  }

  if (helpPauseMetro.check() == 1) {
    consolePaused = false;
    // consoleUpdateMetro.autoreset(true); // 20150822 MSHARP commented out because autoreset
is now private method
    consoleUpdateMetro.reset();
  }

  if ((!consolePaused) && (consoleUpdateMetro.check() == 1)) {
    update_console();
  }

  if (lcdUpdateMetro.check() == 1) {
    update_lcd();
  }

  if (ledMetro.check() == 1) {
    toggleLED();
    ledMetro.reset();
  }

  lastLoopDuration = millis() - loopStartMillis;
} // end loop
```

Couldn't find my previous Python script so I had to crank out a new one quickly.

```
# Python 3.12 venv setup
mkdir -p ~/dev/python/python-teensyclimate-interface
cd ~/dev/python/python-teensyclimate-interface
python3.12 -m venv ven
source ./venv/bin/activate
pip install pyserial
```

```
# file:teensyclimate-log.py
# MSHARP 20240115
# 1. Open serial port
```

```
# 2. Write time sycn string to serial port
# 3. Start logging data to console
#    Intent is to SSH into remote machine connected to microcontroller
#    and log the session output to another computer, say via PuTTy

import serial
import time

# open serial port
teensy = serial.Serial('/dev/ttyACM0', 115200, timeout=3)

# send time sync command with time data
timeSyncString = "@"+str(time.time())[0:10]
print(str(f"Writing time sync string {timeSyncString} to serial port..."))
teensy.write(f'T'.encode())
teensy.write(f'{timeSyncString}'.encode())

print(f'Response:')

# loop to read responses from serial port
while True:
    try:
        line = teensy.readline()
        if (line != ""):
            print(line.rstrip().decode())
    except KeyboardInterrupt:
        print()
        print("CTRL-C received")
        break;

print("Shutting down...")
teensy.close()
```

#end

radsecproxy

radsecproxy Github Project Page

radsecproxy.conf man page

Configuration Example

Configuration Options

```
# Note that some block option values may reference a block by name, in which case the block
name must be previously defined. Hence the order of the blocks may be significant.
# Recommended block order:
#   tls
#   rewrite
#   client
#   server
#   realm



# The rewrite actions are performed in this sequence:
# 1. RemoveAttribute (or WhitelistAttribute)
# 2. ModifyAttribute
# 3. SupplementAttribute
# 4. AddAttribute

rewrite name {
AddAttribute attribute:value
AddVendorAttribute vendor:subattribute:value
SupplementAttribute attribute:value
SupplementVendorAttribute vendor:subattribute:value
ModifyAttribute attribute:/regex/replace/
ModifyVendorAttribute vendor:subattribute:/regex/replace/
RemoveAttribute attribute
RemoveVendorAttribute vendor[:subattribute]
WhitelistMode (on|off)
```

```
	WhitelistAttribute attribute
	WhitelistVendorAttribute vendor[:subattribute]
}

tls name {
	CACertificateFile file
	CACertificatePath path
	CertificateFile file
	CertificateKeyFile file
	CertificateKeyPassword password
	PolicyOID oid
	CRLCheck (on|off)
	CacheExpiry seconds
}

client (name|fqdn|(address[/length])) {
	Host (fqdn|(address[/length])) # multiple lines allowed
	IPv4Only (on|off)
	IPv6Only (on|off)
	Type type (UDP|TCP|TLS|DTLS)
	Secret secret
	TLS tls
	CertificateNameCheck (on|off)
	matchCertificateAttribute ( CN | SubjectAltName:URI | SubjectAltName:DNS ) :/regexp/
	MatchCertificateAttribute SubjectAltName:IP:address
	DuplicateInterval seconds
	AddTTL 1-255
	TCPKeepalive (on|off)
	FticksVISCOUNTRY cc
	FticksVISINST institution
	RewriteIn rewrite
	RewriteOut rewrite
	RewriteAttribute User-Name:/regex/replace/
}

server (name|((fqdn|address)[:port])) {
	Host (fqdn|address)[:port]
	Port port
	DynamicLookupCommand command
	StatusServer (on|off|minimal|auto)
```

```
RetryCount count
RetryInterfval interval
RewriteOut rewrite
RewriteIn rewrite
LoopPrevention (on|off)
IPv4Only (on|off)
IPv6Only (on|off)
Type type
Secret secret
TLS tls
CertificateNameCheck (on|off)
matchCertificateAttribute ( CN | SubjectAltName:URI | SubjectAltName:DNS ) :/regexp/
MatchCertificateAttribute SubjectAltName:IP:address
AddTTL 1-255
TCPKeepalive (on|off)
}

realm (*|realm|/regex/) {
Server server
AccountingServer server
AccountingResponse (on|off)
ReplyMessage message
}
```

Configuration framework

Create the folders and files first:

```
mkdir /etc/radsecproxy.d

touch /etc/radsecproxy.d/tls.conf
touch /etc/radsecproxy.d/rewrites.conf
touch /etc/radsecproxy.d/clients.conf
touch /etc/radsecproxy.d/servers.conf
touch /etc/radsecproxy.d/realms.conf
```

```
chown -R radsecproxy:radsecproxy /etc/radsecproxy.*

chmod 770 /etc/radsecproxy.d
chmod 660 /etc/radsecproxy.conf /etc/radsecproxy.d/*.conf
```

Now populate the files as needed:

```
# /etc/radsecproxy.conf

IPv4Only                on

ListenUDP               *:1812
ListenUDP               *:1813
ListenTLS               *:2083

# For testing later reduce to 3
LogLevel                4

#LogDestination         file:///var/log/radsecproxy.log
LogDestination          x-syslog:///
LoopPrevention          on

Include /etc/radsecproxy.d/tls.conf
Include /etc/radsecproxy.d/rewrites.conf
Include /etc/radsecproxy.d/clients.conf
Include /etc/radsecproxy.d/servers.conf
Include /etc/radsecproxy.d/realms.conf
```

```
# file:tls.conf

tls default {
    CACertificateFile  /etc/radsecproxy.d/certs/trusted-roots.crt
    CertificateFile    /etc/radsecproxy.d/certs/certificate.crt
    CertificateKeyFile /etc/radsecproxy.d/certs/certificate.private.key

    CRLCheck off
    #CacheExpiry 3600
    #policyOID               1.3.6.1.5.5.7.3.2
}
```

```
# file:rewrites.conf

# examples of possible rewrites

# @azuremfa -> @mydomain.com : for forcing Azure MFA
rewrite azuremfa {
        modifyAttribute 1:/^(.*)@azure$/\1@mydomain.com/
        modifyAttribute 1:/^(.*)@azuremfa$/\1@mydomain.com/
}

# @azurenomfa -> @mydomain.com : for forcing logins that don't require MFA
rewrite azurenomfa {
modifyAttribute 1:/^(.*)@azurenomfa$/\1@mydomain.com/
}

# @router -> @mydomain.com : for logging into devices with MFA required
rewrite azuremfa-router {
        modifyAttribute 1:/^(.*)@router$/\1@mydomain.com/
}

# @duo -> @mydomain.com : for forcing DUO MFA
rewrite duomfa {
        modifyAttribute 1:/^(.*)@duo$/\1@mydomain.com/
modifyAttribute 1:/^(.*)@duomfa$/\1@mydomain.com/
}

# @oldBusinessName -> [norealm]] : for forcing login to legacy Active Directory
rewrite oldBusinessName {
modifyAttribute 1:/^(.*)@oldBusinessName$/\1/
}

# adding a radsecproxy identifying attribute for incoming client requests
rewrite tagrsp1 {
RemoveVendorAttribute 14988:11
AddVendorAttribute 14988:11:"rsp1"
}

# [username]@legacyRadius -> [username] : for forcing logins via a legacy radius server
# logins should have been presented to us a username@legacyRadius
# we need to pass them the legacy radius server with no @legacyRadius
```

```
rewrite legacyRadius {
        modifyAttribute 1:/^(.*)@legacyRadius$/\1/
}
```

```
# file:clients.conf

client office1 {
host 1.1.1.1
host 2.2.2.2
host 10.1.2.1
#rewriteIn tagrsp1
secret aBetterPasswordThanThis
type udp
}

client office2 {
host 3.3.3.3
host 10.3.2.1
#rewriteIn tagrsp1
secret aBetterPasswordThanThis
type udp
}

client catchallIPv4UDP {
host 0.0.0.0/0
#rewriteIn tagrsp1
secret aBetterPasswordThanThis
type udp
}

client catchallIPv4TLS {
host 0.0.0.0/0
#rewriteIn tagrsp1
type TLS
}
```

```
# file:servers.conf

# Azure NPS 001 - No MFA
server azureNPS01 {
```

```
        host 192.168.1.11:1812
        type udp
        secret aBetterPasswordThanThis
RetryCount 0
        RetryInterval 3
        RewriteOut azurenomfa
}

# Azure NPS 002 - MFA required
server azureNPS02MFA {
        host 192.168.1.12:1812
        type udp
        secret aBetterPasswordThanThis
RetryCount 0
        RetryInterval 30
        RewriteOut azuremfa
}

# Azure NPS 002 - MFA Required - with router specific rewrite
server azureNPS02MFA-router {
        host 192.168.1.12:1812
        type udp
        secret aBetterPasswordThanThis
RetryCount 0
        RetryInterval 30
        RewriteOut azuremfa-router
}

# DUO MFA
server duoauthproxy {
        host 127.0.0.1:31812
        type udp
        secret aBetterPasswordThanThis
RetryCount 0
        RetryInterval 30
        RewriteOut duomfa
}

# Legacy Active Directory
server legacyAD {
```

```
        host 192.168.2.11:1812
        type udp
        secret aBetterPasswordThanThis
RetryCount 0
        RetryInterval 30
        RewriteOut legacyRadius
}

# Legacy Radius Server
server legacyRadius01 {
        host 8.9.10.11:1812
        type udp
        secret aBetterPasswordThanThis
RetryCount 0
        RetryInterval 5
        RewriteOut legacyRadius
}
```

```
# file:realms.conf

# Force Azure MFA
realm azure {
        server azureNPS02MFA
        AccountingResponse on
}

# Force Azure MFA
realm azuremfa {
        server azureNPS02MFA
        AccountingResponse on
}

# Force Azure No MFA
realm azurenomfa {
        server azureNPS01
        AccountingResponse on
}

# Force DUO MFA
realm duo {
```

```
server duoauthproxy
AccountingResponse on
}

# Force DUO MFA
realm duomfa {
server duoauthproxy
AccountingResponse on
}

# Force legacy Active Directory
realm mydomain.com {
        server legacyRadius
        AccountingResponse on
}

# Force router management login - service against legacy radius server
realm router {
        server legacyRadius
        AccountingResponse on
}

# For legacy Active Directory
realm old {
        server legacyRadius
        AccountingResponse on
}

# All other realms
realm * {
        server legacyRadius
        AccountingResponse on
#       ReplyMessage "Contact tech support..."
}

# All other realms - deny all other requests
# Obviously you can only have one wildcard matcher, this is just another option
realm * {
        ReplyMessage "Radius request denied by proxy. Contact tech support."
}
```

Software

Below are links to various software I use. Software is free / open-source unless otherwise noted.

Unclassified

- draw.io - diagram / flow charting
- PlantUML plugin for Visual Studio Code (jebbs) - building sequence diagrams
- SimpleMind - Mind Mapping - $$$
- WinMerge - an Open Source differencing and merging tool for Windows. Use for comparing both folders and files.

Cygwin - Additional components to install

- aria2 - Download utility for HTTP/HTTPS, FTP, BitTorrent and Metalink
- autossh
- openssh
- unzip
- vim
- zip

File Encryption

- AES Crypt

File Management

- 7-Zip
- grepWin

File Sync

- SyncFolders - Synchronize or backup your files and folders
- Cyberduck - libre server and cloud storage browser

Database Clients

- HeidiSQL - client for MariaDB, MySQL, Microsoft SQL, PostgreSQL and SQLite

Disk Usage Utilities

- JDiskReport
- Filelight
- Space Sniffer
- WinDirStat

Graphics Editing

- paint.net
- the gimp

Linux Performance Monitoring Tools

- atop - all the info
- btop
- htop
- iftop
- iotop
- glances
- s-tui
- wavemon - ncurses-based monitoring application for wireless devices

Network tools - clients

- MobaXterm - RDP/SSH/Telnet client - free / $$$
- NTRadPing - radius client
- PuTTY - SSH/Telnet client
- TightVNC - VNC client/server
- UltraVNC - VNC client/server
- WinSCP

Network tools - configuration

- Simple IP Config - Windows network interface configuration tool - eases changing interface IP address configurations

Network tools - packet capture

- Microsoft Network Monitor 3.4 (archive)
- Wireshark - Packet capture software

Network tools - scanning

- Advanced IP Scanner
- Angry IP Scanner
- Nmap

Screen Capture / Annotation

- Greenshot
- ShareX - Screen capture, file sharing and productivity tool

Text / Code Editors

- neovim - hyperextensible Vim-based text editor
- notepad++
  - notepadqq is a notepad++ like editor for the Linux desktop.
- vim - a highly configurable text editor built to make creating and changing any kind of text very efficient
- visidata - VisiData is an interactive multitool for tabular data (CSV included) written in Python
- Visual Studio Code

Markdown Editors

- Editor.md - Open source online Markdown editor for use in embedded JS applications
- Obsidian - Open source Evernote replacement you sync yourself
  - Better than Evernote without the syncing, not as good as Notion.so, but free
- Typora - Not FOSS, but inexpensive and really nice Markdown editor

Programming / Scripting

- Node.js
- Python
- rust

Video Capture and Editing

- DaVinci Resolve
- OBS

- OpenShot

VPN Clients

- OpenVPN Connect - Windows / macOS / Linux
- WireGuard

Windows Registry Tools

- Nirsoft RegistryChangesView

Windows Shell Add-Ons

- Nilesoft Shell - I've stopped using this due to certain issues.

Ubiquiti UniFi

Device CLI

Yes, the UniFi switches actually have a CLI... who knew! No you don't have to wait three minutes for the Unifi controller to update... □□

You can make configuration changes from the cli, but they will not be persistent across reboots or resyncs with the controller.

The commands below are from a USW-48-PoE (Standard). Other models may vary.

```
cli
# or
telnet localhost
```

```
show running-config
```

```
show mac address-table
show mac address-table interfaces GigabitEthernet 1
show mac address-table vlan 2
```

```
# wouldn't it be nice if Ubiquiti would actually share this information from the
controller!?!?!
show lldp neighbor
```

```
show interfaces GigabitEthernet 1 status
show interfaces GigabitEthernet 1-8 status
show interfaces GigabitEthernet 1-26 status
show interfaces GigabitEthernet 1-52 status

# example output
USW-48-G2# show interfaces GigabitEthernet 1-8
```

```
  <cr>
  protected  Configure an interface to be a protected port
  status     Port status information
JRN1A-SW2-USW-48-G2# show interfaces GigabitEthernet 1-52 status
Port  Name                 Status      Vlan  Duplex  Speed    Type
gi1   Port 1               notconnect  11    auto    auto     Copper
gi2   Port 2               connected   11    a-full  a-100M   Copper
gi3   Port 3               connected   11    a-full  a-1000M  Copper
gi4   Port 4               notconnect  11    auto    auto     Copper
gi5   Port 5               connected   11    a-full  a-1000M  Copper
gi6   Port 6               notconnect  11    auto    auto     Copper
gi7   Port 7               notconnect  11    auto    auto     Copper
gi8   Port 8               notconnect  11    auto    auto     Copper
```

```
show vlan
show vlan 2 interfaces GigabitEthernet 1-52 membership
```

```
# show 802.1x configuration for a port or multiple ports
show authentication interfaces GigabitEthernet 1
show authentication interfaces GigabitEthernet 1-24
show authentication interfaces GigabitEthernet 1-48
```

```
# show 802.1x authentication sessions
show authentication sessions

# example output with MAC Addresses redacted
USW-48-G2# show authentication  sessions
Total Session Number: 13

Interface  MAC Address       Type    Status       Session ID
---------- ----------------- ------- ------------ ----------------
gi3        XX:YY:ZZ:11:22:33 mac     Guest        0000000100015C5C
gi5        XX:YY:ZZ:11:22:33 mac     Guest        0000000200015C5C
gi26       XX:YY:ZZ:11:22:33 mac     Guest        0000000300015D10
gi40       XX:YY:ZZ:11:22:33 mac     Guest        0000000500015F0E
gi39       XX:YY:ZZ:11:22:33 mac     Guest        0000000700016256
```

```
gi34       XX:YY:ZZ:11:22:33 mac     Guest        0000000A00016DAA
gi18       XX:YY:ZZ:11:22:33 mac     Guest        0000000D00017930
gi2        XX:YY:ZZ:11:22:33 dot1x   Authorized   0000000E000180B0
gi43       XX:YY:ZZ:11:22:33 dot1x   Authorized   0000000F000188EE
gi20       XX:YY:ZZ:11:22:33 dot1x   Authorized   00000012003531F1
gi24       XX:YY:ZZ:11:22:33 dot1x   Authorized   00000013004F754C
gi28       XX:YY:ZZ:11:22:33 dot1x   Authorized   00000016006F8AB7
gi31       XX:YY:ZZ:11:22:33 dot1x   Authorized   0000001700731DA1
```

```
# configure
(config)# interface GigabitEthernet 34
(config-if)# shutdown
(config-if)# no shutdown
```

Random Stuff

List users in the UniFi Controller database

```
# show list of users in the unifi mongodb database
mongo --port 27117 ace --eval "db.admin.find().forEach(printjson);"
```

Change password for a UniFi Controller user

```
# change <UserName> to an actual user on the unifi controller
# the command will reset that user password to 'password'
mongo --port 27117 ace --eval 'db.admin.update( { "name" : "<UserName>" }, { $set : {
"x_shadow" :
"$6$GgQYRQnUs4wYkRd$7g6mig.les9salut9CZjUrG/UqqF6R/2RiCaCQEpEzz/7UtAtzeeQsVDnacAW1el2KH/jvUuJ4
Eh08xy.KGl0/" } } )'
```

[Decrypt a UniFi Controller backup]

Force dhcp to renew ip address

1. Get the PID of the udhcpc process
2. Send that process the USR1 signal which tells udhcpc to renew its IP address □□

```
# find the PID of the udhcpc process
ps | grep udhcpc
```

```
# output of the above command
 4052 admin     3480 S    /sbin/udhcpc -f -i eth0 -V ubnt -A 10 -s /etc/udhcpc/udhcpc -p
/var/run/udhcpc.eth0.pid
 6648 admin     3504 R    sh /usr/etc/syswrapper.sh ssh-trace-cmd -c ps | grep udhcpc -n 4 -i
 6650 admin     3480 R    grep udhcpc
```

```
# instruct udhcpc to renew its IP address by sending it's process the USR1 signal
kill -USR1 4052
```

Finding the site a device is connected to

```
# if you're using the Unifi OS Server, you need to enter the docker container first
uosserver shell

# Connect using the traditional mongo shell on UniFi's default port
mongo --port 27117

# find the site_id of a device
db.device.find({mac: "xx:xx:xx:xx:xx:xx"}, {site_id: 1, name: 2})

# find the description of the site
db.site.find({_id: ObjectId("YOUR_SITE_ID_HERE")}, {name: 1, desc: 1})
```

Manually removing items from mongo

```
# if you're using the Unifi OS Server, you need to enter the docker container first
uosserver shell

# Connect using the traditional mongo shell on UniFi's default port
mongo --port 27117
```

```
// ============================================================
// Once inside the shell, switch to the primary UniFi database
use ace



// ============================
// Force forget a ghost device

// For modern UniFi setups (MongoDB 5.0+)
db.device.deleteMany({"mac":"xx:xx:xx:xx:xx:xx"})
db.user.deleteMany({"mac":"xx:xx:xx:xx:xx:xx"})



// ======================
// Delete an Admin Profile

db.admin.deleteMany({"email":"admin@example.com"})



exit

// =============
// apply changes

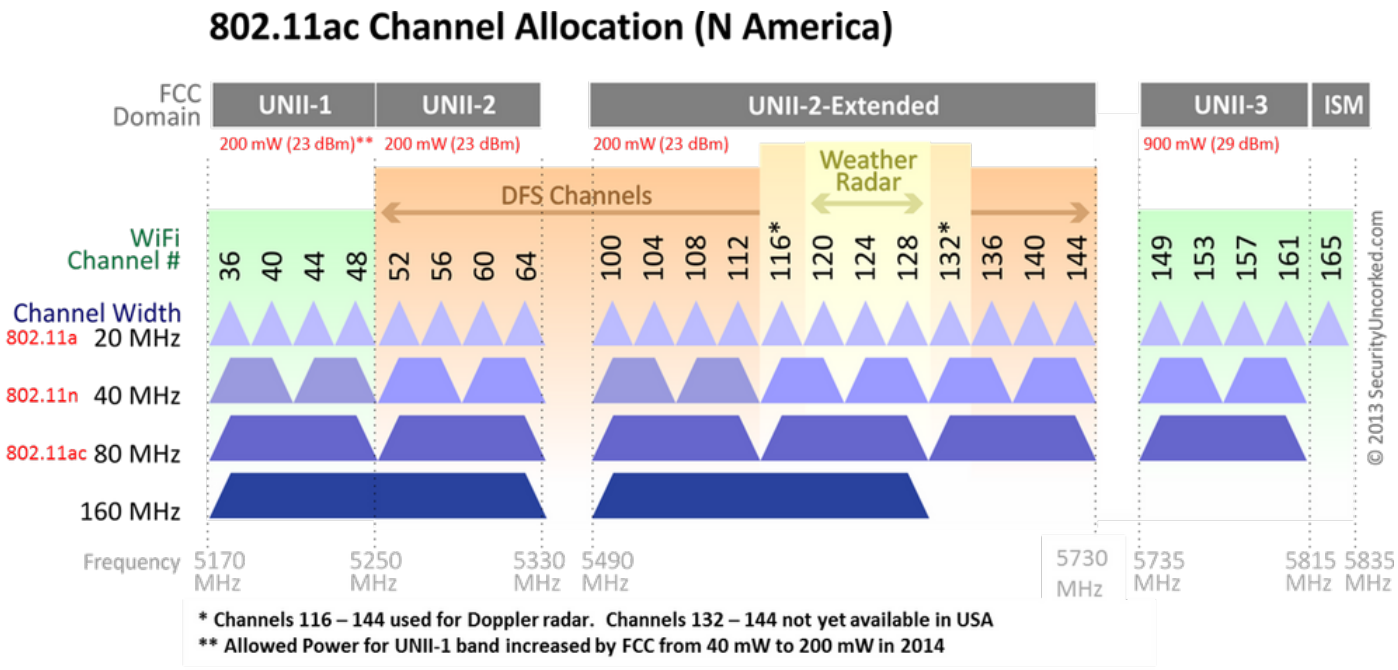
// On Linux / Ubuntu
sudo systemctl restart unifi

// On Cloud Key / UDM
init 6 or restart via the OS Settings Console
```

:end

WiFi Regulatory Notes

More notes to come. For now, these are random useful illustrations from around the web.



		Center Freq (MHz)	802.11a (20 MHz)	802.11n (40 MHz)	802.11ac (80 MHz)	802.11ac (160 MHz)
All	UNII-1	5180	36	38 [36-40]	42 [36-48]	50 [36-64]
		5200	40			
		5220	44	46 [44-48]		
		5240	48			
DFS	UNII-2	5260	52	54 [52-56]	58 [52-64]	
		5280	56			
		5300	60	62 [60-64]		
		5320	64			
	UNII-2e	5500	100	102 [100-104]	106 [100-112]	114 [100-128]
		5520	104			
		5540	108	110 [108-112]		
		5560	112			
		5580	116	118 [116-120]	122 [116-128]	
		5600	120			
		5620	124	126 [124-128]		
		5640	128			
		5660	132	134 [132-136]	138 [132-144]	
		5680	136			
		5700	140	142 [140-144]		
		5720	144			
All	UNII-3	5745	149	151 [149-153]	155 [149-161]	
		5765	153			
		5785	157	159 [157-161]		
		5805	161			
	ISM	5825	165			

Total # of Channels

Full DFS Support:	25	12	6	2 (3 with 80+80)
No Channel 144:	24	11	5	2
No DFS Support:	9	4	2	0

-end

WireGuard

Generic client template

```
[Interface]
PrivateKey = xxx
Address = 192.168.170.X/24

[Peer]
PublicKey = yyy
PreSharedKey = zzz
AllowedIPs = 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
Endpoint = router.domain.com:13231
PersistentKeepalive = 25
```

Ubuntu client setup using systemd

```
# create and store the local private key
wg genkey | sudo tee /etc/wireguard/private.key
sudo chmod go= /etc/wireguard/private.key

# create and store the related public key
sudo cat /etc/wireguard/private.key | wg pubkey | sudo tee /etc/wireguard/public.key

# create and store an additional preshared key
wg genpsk | sudo tee /etc/wireguard/psk
```

```
WGPRIVKEY=`cat /etc/wireguard/private.key`
WGPSK=`cat /etc/wireguard/psk`

cat << EOF >> /etc/wireguard/wg0.conf
[Interface]
PrivateKey = ${WGPRIVKEY}
Address =
```

```
[Peer]
PublicKey =
PreSharedKey = ${WGPSK}
AllowedIPs = 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
Endpoint = router.domain.com:13231
PersistentKeepalive = 25
EOF
```

```
sudo vi /etc/wireguard/wg0.conf
```

```
# enable and start the wg0 interface using the wg-quick service
sudo systemctl enable wg-quick@wg0.service
sudo systemctl start wg-quick@wg0.service
```

Using post-up and post-down scripts in a WireGuard configuration

You can use PostUp and PostDown scripts to run PowerShell commands to manage Name Resolution Policy Table (NRPT) rules when a WireGuard tunnel connects and disconnects.

```
[Interface]
PostUp = powershell.exe -Command "& { Add-DnsClientNrptRule -Comment 'wg-tunnel-xxx' -
Namespace '.xxx.net' -NameServers 172.16.16.254 }"
PostDown = powershell.exe -Command "& { Get-DnsClientNrptRule | where Comment -eq 'wg-tunnel-
xxx' | foreach { Remove-DnsClientNrptRule -Name $_.Name -Force } }"
```

You will need to enable the ability to run scripts through the addition of the DangerousScriptExecution key:

```
# run the following command using PowerShell running as Administrator
reg add HKLM\Software\WireGuard /v DangerousScriptExecution /t REG_DWORD /d 1 /f
```

Reference: adminregistry.md

ProFTPd

ProFTPd and SFTP

Ubuntu 22.04

1. Install proftpd-basic
2. Edit /etc/proftpd/modules.conf and enable mod_sftp and mod_sftp_pam
3. Edit /etc/proftpd/sftp.conf (see example below)
4. Edit /etc/proftpd/proftpd.conf and enable the sftp.conf include
5. Create sftproot folders in user home directories as needed (based on the configuration shown below)
6. You should be ready to rock and roll!

```
# mod_sftp is part of proftpd-basic, not proftpd-core
apt install proftpd-basic
```

```
<IfModule mod_sftp.c>
SFTPEngine     on
Port           2222
SFTPLog        /var/log/proftpd/sftp.log
#
# Configure all host keys, using the same host key
# files that OpenSSH uses.
#
SFTPHostKey /etc/ssh/ssh_host_rsa_key
SFTPHostKey /etc/ssh/ssh_host_dsa_key
SFTPHostKey /etc/ssh/ssh_host_rsa_key
SFTPHostKey /etc/ssh/ssh_host_ecdsa_key
SFTPHostKey /etc/ssh/ssh_host_ed25519_key

#SFTPAuthMethods publickey
#SFTPAuthMethods keyboard-interactive
SFTPAuthMethods publickey keyboard-interactive
SFTPAuthorizedUserKeys file:/etc/proftpd/authorized_keys/%u

#
```

```
# Use either
#    ssh-keygen -e -f ~user/.ssh/id_rsa.pub >/etc/proftpd/authorized_keys/user
# or
#    ssh-keygen -e -f ~user/.ssh/authorized_keys >/etc/proftpd/authorized_keys/user
# to convert users public keys in RFC4716 format.
#
#
# Enable compression
#
#SFTPCompression delayed

# Other desired configuration options
DefaultRoot ~/sftproot
RequireValidShell off
SFTPPAMOptions NoTTY NoInfoMsgs NoRadioMsgs
Umask  006 007

# Limit logins to only a specific group
<Limit LOGIN>
  AllowGroup sftponly
  DenyAll
</Limit>

</IfModule>
```

#end

Bind9

Recursion without recursion to enable forward zones

Problem

I need to add a forward zone to an authoritative bind9 server, but forward zones don't appear to work with recursion disabled, and I don't want to turn this server into an open resolver.

Solution

Turn the server into an open resolver, but delete all of the root hints so it doesn't have anywhere to lookup from, and make sure you don't have forwarders specified under options {}

```
options {
    ...
    allow-recursion { ::/0; 0.0.0.0/0; };
};

zone "some.forward.zone.net"  {
   type forward;
   forward only;
   forwarders { 192.168.168.168; };
};

zone "." {
        type hint;
        file "/dev/null";
};
```

Source

#end

GitLab

Upgrade paths

Click here to view the required upgrade paths. Use the commands below to perform the individual upgrades.

Upgrade to a specific version using the official repositories

source

Use the first command

```
# Ubuntu/Debian
sudo apt-cache madison gitlab-ee
```

```
# Ubuntu/Debian
sudo apt install gitlab-ee=<version>
```

Example staged upgrade:

```
apt install gitlab-ee=14.10.5-ee.0
apt install gitlab-ee=15.0.5-ee.0
apt install gitlab-ee=15.1.6-ee.0
apt install gitlab-ee=15.4.6-ee.0
apt install gitlab-ee=15.11.12-ee.0
```

PoE Detection and Negotiation

Reference

Virtual disk format conversions

qemu-img utility

You can download the qemu-img utility for Windows from here.

OVA to VHDX

```
qemu-img.exe -p convert source.vmdk -O vhdx -o subformat=fixed destination.vhdx
```

#end

WiFi Standards

WiFi Standards Comparison

	WiFi 4	WiFi 5	WiFi 6	WiFi 6e	WiFi 7
Launch date		2013	2019	2021	2024
Standard name		802.11ac	802.11ax	802.11ax	802.11be
Notable additions		MU-MIMO	OFDMA TWT WPA3 Encryption + modulation +		MLO +Modulation +Max channel width
Power and battery life		-	Supports TWT		
Security protocols		WPA, WPA2	WPA, WPA2, **WPA3 (OWE)**		
Modulation		up to 256-QAM	up to 1024-QAM		up to 4096-QAM
Beamforming		up to four antennas	up to eight antennas		
Subcarriers (Resource units)		-	Supports **OFDMA**	**OFDMA**	**OFDMA**
Multi-RU		-	No	No	**Yes**
MIMO	Added				
MU-MIMO		up to 4x4 downlink unidirectional	up to 8x8 bidirectional	up to 8x8 bidirectional	up to 16×16 bidirectional
MLO (Multi-Link Operation)		-	-	-	Yes
Frequency bands		2.4 GHz, 5 GHz	2.4 GHz, 5 GHz	2.4 GHz, 5 GHz, 6 GHz	2.4 GHz, 5 GHz, 6 GHz
Channel width		20, 40, 80, 80+80, 160 MHz	Same	Same	Up to 320 MHz
BSS coloring		-	Yes		

Latency			Reduced if all devices using the network are WiFi 6 and above, due to the above features.		

MIMO (multiple-input and multiple-output)

Aka single-user MIMO.

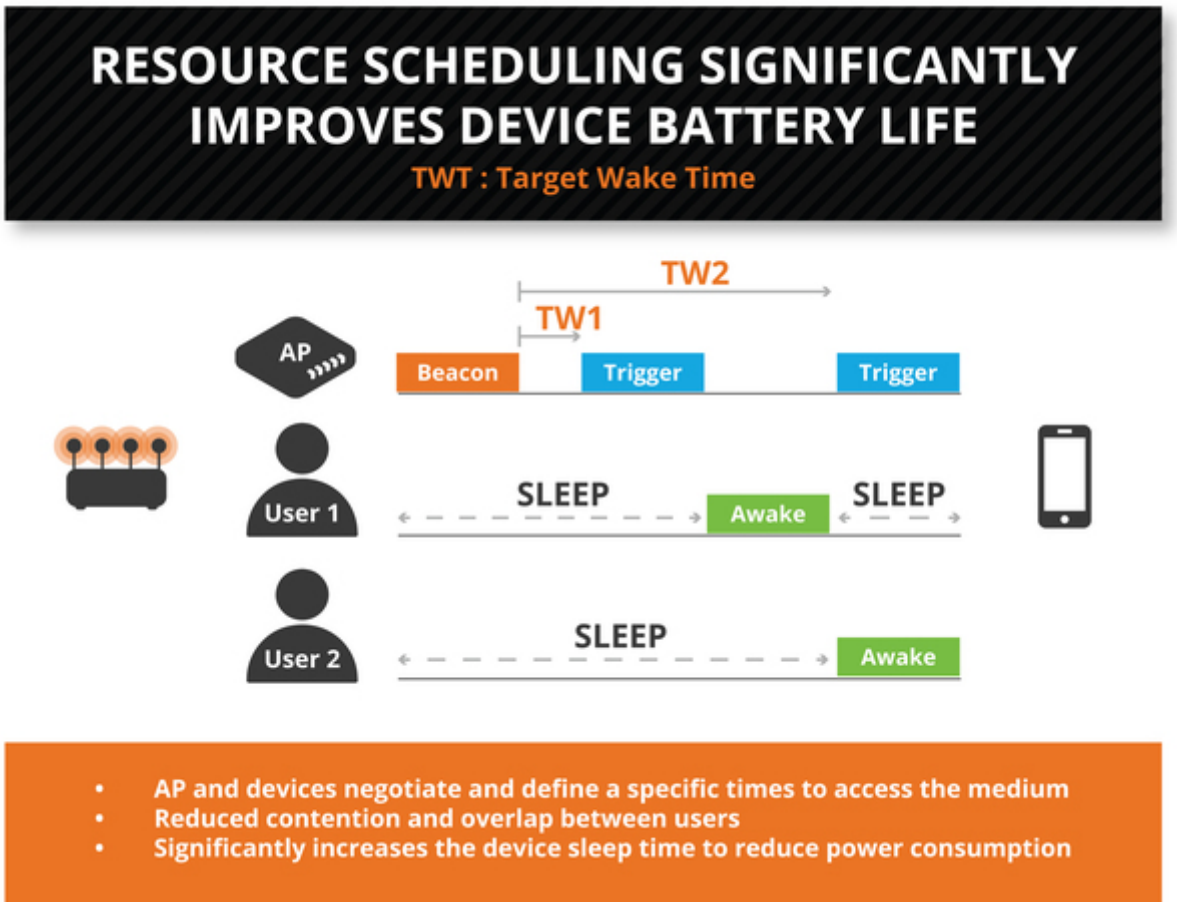
Leverages spatial multiplexing, or space-division multiplexing, used to transmit independent channels separated in space.

In wireless communications, each spatial stream is serviced by...

MU-MIMO (multi-user MIMO)

Target Wake Time (TWT)

[Source]
TWT enables devices to determine when and how frequently they will wake up to send or receive data. Essentially, this allows 802.11ax access points to effectively increase device sleep time and significantly conserve battery life, a feature that is particularly important for the IoT. In addition to saving power on the client device side, Target Wake Time enables wireless access points and devices to negotiate and define specific times to access the medium. This helps optimize spectral efficiency by reducing contention and overlap between users.



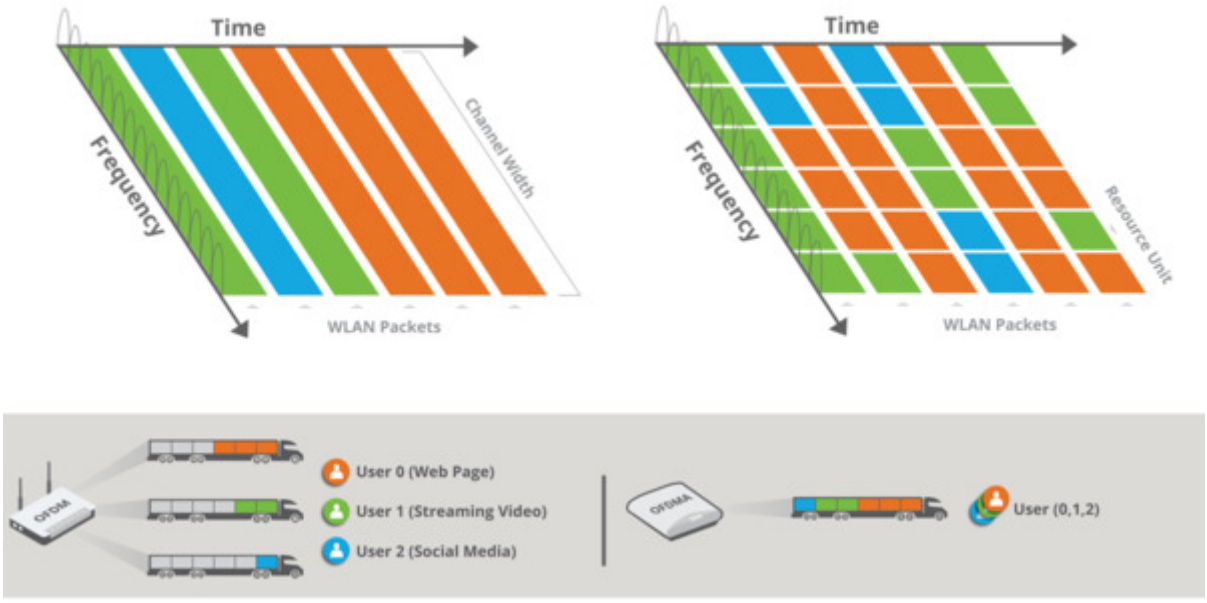
The Target Wake Time mechanism first appeared in the IEEE 802.11ah “Wi-Fi HaLow” standard. Published in 2017, the low-power standard is specifically designed to support the large-scale deployment of IoT infrastructure – such as stations and sensors – that intelligently coordinate signal sharing. The TWT feature further evolved with the IEEE 802.11ax standard, as stations and sensors are now only required to wake and communicate with the specific Beacon(s) transmitting instructions for the TWT Broadcast sessions they belong to. This allows the wireless IEEE 802.11ax standard to optimize power saving for many devices, with more reliable, deterministic and LTE-like performance. As Maddalena Nurchis and Boris Bellalta of the Universitat Pompeu Fabra in Barcelona noted in a recent paper, TWT also “opens the door” to fully maximizing new MU capabilities in 802.11ax by supporting the scheduling of both MU-DL and MU-UL transmissions. In addition, TWT can be used to collect information from stations, such as channel sounding and buffers occupancy in pre-defined periods. Last, but certainly not least, TWT can potentially help multiple WLANs in dense deployment scenarios reach consensus on non-overlapping schedules to further improve Overlapping Basic Service Set (OBSS) co-existence.

WPA3 Opportunistic Wireless Encryption (OWE)

[Source]

Multi-User, Multiple Input, Multiple Outputs (MU-MIMO)

Orthogonal Frequency-Division Multiple Access (OFDMA)



The Benefits of OFDMA for Wi-Fi 6 - A technology brief highlighting Qualcomm Technologies' competitive advantage

OFDM subdivides the Wi-Fi channel into smaller frequency allocations called resource units. By partitioning the channel, parallel transmissions of smaller frames to multiple users occur simultaneously. For example, a traditional 20 MHz channel might be partitioned into as many as nine smaller channels. Using OFDMA, a Wi-Fi 6 AP could simultaneously transmit smaller frames to nine Wi-Fi 6 clients.

The Wi-Fi Alliance Whitepaper further explains the difference between uplink and downlink OFDMA.

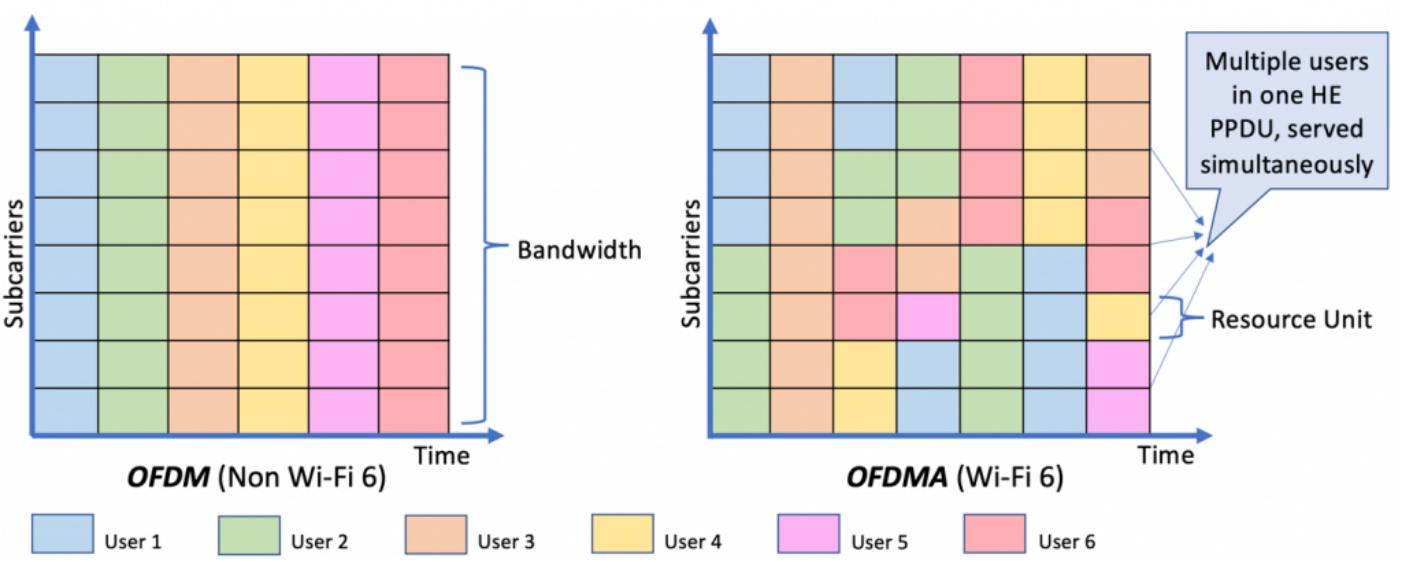
Uplink OFDMA is one of the key features introduced by Wi-Fi 6 and is among the most significant differences relative to 802.11ac. Uplink OFDMA allows data frames to be transmitted simultaneously by
multiple stations. This amortizes preamble overhead and medium contention overhead, which leads to
high aggregated network throughput. Uplink OFDMA can provide additional gains by permitting greater
transmit power level per device, subject to regulatory requirements, and thus signal coverage on the uplink,
since the transmit power of each client device can be concentrated on smaller allocated resource units.

Downlink OFDMA allows multiple data frames to be transmitted in a single data unit to multiple stations,
thus amortizing preamble overhead and medium contention overhead, leading to higher aggregated
network throughput. Downlink OFDMA can further optimize aggregate throughput by balancing the allocation of power between users at high versus low signal-to-noise ratios, subject to total power constraints and regulatory requirements.

Resource Units

[Cisco Blogs: Wi-Fi 6 OFDMA: Resource unit (RU) allocations and mappings]

In Wi-Fi 6, subcarrier spacing is 78.125 KHz.



Multi-Link Operation (MLO)

Wi-Fi 6 devices can only connect to one band at a time, either 2.4 GHz or 5 GHz.
Wi-Fi 7 devices can use MLO to connect to multiple bands at once, including 2.4 GHz, 5 GHz, and 6 GHz.
MLO allows devices to send and receive data on multiple bands at the same time. This can help reduce latency and improve reliability. MLO can also be used for load balancing by switching between bands to minimize contention.

Frequency bands

USB Pendrive

YUMI

YUMI exFAT - a Multiboot USB bootable software. Easily copy multiple ISO images to a bootable USB flash drive and select which ISO to load at boot

How to Restore USB flash drive using diskpart in Windows

Source

1. Open a command Prompt as administrator (**cmd.exe**)
2. Type **diskpart**
3. Next type **list disk**
4. Type **Select Disk X** (where **X** is the disk number of your drive).
5. Type **clean**

   * If Error, See Note Below
6. Next type **create partition primary**
7. Type **Format fs=exfat quick**
   Note: (for 32GB and smaller drives, use fs=**fat32**) instead.
8. Then type **active**
9. Next type **assign**
10. Finally, type **exit** to quit

Trend Micro Vision One Search Auto-Clicker Auto-Refresher

Use case

I need to auto-refresh the Trend Vision One Quarantine list every minute since they haven't added that into their platform. I tried using an auto-clicker / auto-refresh browser extension, but it would only hard refresh the page or click on anchor links via XPath, not buttons.

I eventually realized I could just write some JavaScript that would do the same thing. Below is the result. I found the buttonXpath is different depending on your browser, so if you use this in anything other than Chrome, you'll probably need to update that variable before using.

Code

```
# start the clicking

function clickButtonByXpath(xpathExpression) {
  try {
    const result = document.evaluate(xpathExpression, document, null,
XPathResult.FIRST_ORDERED_NODE_TYPE, null);
    const element = result.singleNodeValue;

    if (element) {
      element.click();
      console.log(`Button found and clicked: ${xpathExpression}`);
    } else {
      console.warn(`Button not found for XPath: ${xpathExpression}`);
    }
  } catch (error) {
```

```
    console.error(`Error clicking button via XPath: ${error}`);
  }
}

// Define the XPath expression for your button
const buttonXpath =
'//*[@id="root"]/div[2]/div/div/main/div/div/div[2]/div/div[3]/div[2]/button'; // Replace with
your actual XPath

// Set an interval to call the function every 60 seconds (60000 milliseconds)
const buttonClickInterval = setInterval(() => {
  clickButtonByXpath(buttonXpath);
}, 60000);
```

```
# stop the button clicking
clearInterval(buttonClickInterval);
```

-end