How to securely isolate and execute SIPArmyKnife from Kali Linux

Version 0.1, Last Updated: 21 Aug 2020

This site is dedicated to sharing information about the practice, ideas, concepts and patterns regarding computer security.

Table of Contents

1. Introduction

The motivation behind this paper is to explore using the tool SIP Army Knife that comes with Kali Linux.

What is a Fuzzer/Fuzzing:

"Fuzzing or fuzz testing is an automated software testing technique that involves providing invalid, unexpected, or random data as inputs to a computer program. The program is then monitored for exceptions such as crashes, failing built-in code assertions, or potential memory leaks. Typically, fuzzers are used to test programs that take structured inputs. This structure is specified, e.g., in a file format or protocol and distinguishes valid from invalid input. An effective fuzzer generates semi-valid inputs that are "valid enough" in that they are not directly rejected by the parser, but do create unexpected behaviors deeper in the program and are "invalid enough" to expose corner cases that have not been properly dealt with.

For the purpose of security, input that crosses a trust boundary is often the most interesting. For example, it is more important to fuzz code that handles the upload of a file by any user than it is to fuzz the code that parses a configuration file that is accessible only to a privileged user." ***source***: Fuzzing

What is this tool:

"SIP Army Knife is a fuzzer that searches for cross site scripting, SQL injection, log injection, format strings, buffer overflows, and more." ***source***: Kali Linux

The tool being reviewed is a Perl script, `/usr/bin/siparmyknife`. I am excited to get started with this tool.

2. Requirements

2.1. Writing Conventions

If you see the following $ symbol on a command line to execute, what that means is that the command is executed as a regular user; meaning an account that does not have administrative privileges. Ignore the leading $ and execute the rest of the command.

```
$ command to execute as a regular user
```

If you see a command line lead with the # symbol, then that means that the command is executed as the root user. This implies you need to elevate to the root user before running the command, e.g. with: `sudo su - root`.

```
# command to execute as the root user
```

2.2. VirtualBox

Go to: https://www.virtualbox.org/wiki/Downloads and download VirtualBox.

The author is running on Ubuntu 18.04, so following to this URL: https://www.virtualbox.org/wiki/Linux_Downloads

For Ubuntu, double click on the .deb file, i.e. virtualbox-5.2_5.2.0-118431-Ubuntu-zesty_amd64.deb, and install VirtualBox on your local workstation.

2.2.1. Clean VirtualBox Networking

This section is here in case you already had virtualbox installed from before. The intent is to clean up the previous networking. If you do not need to do this, skip to Add VirtualBox Networking

Run these two commands from a Terminal:

```
$ VBoxManage list natnetworks
$ VBoxManage list dhcpservers
```

Output (example):

```
NetworkName:    192.168.139-NAT
IP:             192.168.139.1
Network:        192.168.139.0/24
IPv6 Enabled:   No
IPv6 Prefix:    fd17:625c:f037:2::/64
DHCP Enabled:   Yes
Enabled:        Yes
loopback mappings (ipv4)
        127.0.0.1=2


NetworkName:    192.168.139-NAT
Dhcpd IP:       192.168.139.3
LowerIPAddress: 192.168.139.101
UpperIPAddress: 192.168.139.254
NetworkMask:    255.255.255.0
Enabled:        Yes
Global Configuration:
    minLeaseTime:     default
    defaultLeaseTime: default
    maxLeaseTime:     default
    Forced options:   None
    Suppressed opts.: None
        1/legacy: 255.255.255.0
Groups:               None
Individual Configs:   None

NetworkName:    HostInterfaceNetworking-vboxnet0
Dhcpd IP:       172.20.0.3
LowerIPAddress: 172.20.0.101
UpperIPAddress: 172.20.0.254
NetworkMask:    255.255.255.0
Enabled:        Yes
Global Configuration:
    minLeaseTime:     default
    defaultLeaseTime: default
    maxLeaseTime:     default
    Forced options:   None
    Suppressed opts.: None
        1/legacy: 255.255.255.0
Groups:               None
Individual Configs:   None
```

Now, delete ALL of the pre-installed VirtualBox networks (one at a time following the syntax below):

```
VBoxManage natnetwork remove --netname <NetworkName_from_above>
VBoxManage natnetwork remove --netname 192.168.139-NAT
```

Repeat as many times as necessary to delete all of them.

Now, delete ALL of the pre-installed DHCP services:

```
VBoxManage dhcpserver remove --netname <DHCP_Server_NetworkName_from_above>
VBoxManage dhcpserver remove --netname 192.168.139-NAT
```

Repeat as many times as necessary to delete all of them.

2.2.2. Add VirtualBox Networking

Now, add the new VirtualBox networks so the Kali Linux guides work.

```
VBoxManage natnetwork add \
    --netname 192.168.139-NAT \
    --network "192.168.139.0/24" \
    --enable --dhcp on

VBoxManage dhcpserver add \
    --netname 192.168.139-NAT \
    --ip 192.168.139.3 \
    --lowerip 192.168.139.101 \
    --upperip 192.168.139.254 \
    --netmask 255.255.255.0 \
    --enable

VBoxManage hostonlyif create

VBoxManage hostonlyif ipconfig vboxnet0 \
    --ip 172.20.0.1 \
    --netmask 255.255.255.0

VBoxManage dhcpserver add \
    --ifname vboxnet0 \
    --ip 172.20.0.3 \
    --lowerip 172.20.0.101 \
    --upperip 172.20.0.254 \
    --netmask 255.255.255.0

VBoxManage dhcpserver modify \
    --ifname vboxnet0 \
    --enable
```

VirtualBox install complete.

2.3. Vagrant

Go to: https://www.vagrantup.com/downloads.html, follow the appropriate link to your OS and 32 or 64 bit version representing your local workstation. Download.

For Ubuntu, double click on the .deb file, i.e. vagrant_2.0.1_x86_64.deb, and install Vagrant on your local system.

2.4. Kali Linux and Damn Vulnerable Web Application (DVWA)

The author highly recommends to create a directory structure that is easy to navigate and find your code. As an example, you could use something similar to:

```
${HOME}/Source_Code/Education/vagrant-machines/kali-linux-vm/
```

Go ahead and make this structure with the following command (inside a Terminal):

```
$ mkdir -p ${HOME}/Source_Code/Education/vagrant-machines/kali-linux-vm/
```

From a Terminal, change directory to:

```
$ cd ${HOME}/Source_Code/Education/vagrant-machines/kali-linux-vm/
```

2.4.1. Vagrantfile

Inside of the kali-linux-vm directory, populate a new file with the exact name, “Vagrantfile”. Case matters, uppercase the “V”. This file will contain both virtual machines for Kali Linux as well as setting up the DVWA virtual machine.

Aggregating both virtual machines into one file has saved the author a lot of time. The coolness here is setting up the variables at the top of the Vagrantfile mimicing shell scripting inside of a virtual machine (passed in with provision: shell). I tested using: `apt-get update && apt-get upgrade -y`, but opted to take it out since it took over 45 minutes on my slower (old) hardware. See comment about downloading this file immediately preceding the code block.

```ruby
# -*- mode: ruby -*-
# vi: set ft=ruby :

$os_update = <<SCRIPT
apt-get update
SCRIPT

VAGRANTFILE_API_VERSION = "2"


Vagrant.configure(VAGRANTFILE_API_VERSION) do |config|
    config.vm.define "kali-linux-vagrant" do |conf|
        conf.vm.box = "kalilinux/rolling"

        # For Linux systems with the Wireless network, uncomment the line:
        conf.vm.network "public_network", bridge: "wlo1", auto_config: true

        # For macbook/OSx systems, uncomment the line and comment out the Linux Wireless network:
        #conf.vm.network "public_network", bridge: "en0: Wi-Fi (AirPort)", auto_config: true

        conf.vm.hostname = "kali-linux-vagrant"
        conf.vm.provider "virtualbox" do |vb|
            vb.gui = true
            vb.memory = "4096"
            vb.cpus = "2"
            vb.customize ["modifyvm", :id, "--vram", "32"]
            vb.customize ["modifyvm", :id, "--accelerate3d", "off"]
            vb.customize ["modifyvm", :id, "--ostype", "Debian_64"]
            vb.customize ["modifyvm", :id, "--boot1", "dvd"]
            vb.customize ["modifyvm", :id, "--boot2", "disk"]
            vb.customize ["modifyvm", :id, "--audio", "none"]
            vb.customize ["modifyvm", :id, "--clipboard", "hosttoguest"]
            vb.customize ["modifyvm", :id, "--draganddrop", "hosttoguest"]
            vb.customize ["modifyvm", :id, "--paravirtprovider", "kvm"]
        end
        conf.vm.provision "shell", inline: $os_update
    end

    config.vm.define "dvwa-vagrant" do |conf|

        conf.vm.box = "ubuntu/xenial64"

        conf.vm.hostname = "dvwa-vagrant"

        # For Linux systems with the Wireless network, uncomment the line:
        conf.vm.network "public_network", bridge: "wlo1", auto_config: true

        # For macbook/OSx systems, uncomment the line and comment out the Linux Wireless network:
        #conf.vm.network "public_network", bridge: "en0: Wi-Fi (AirPort)", auto_config: true

        config.vm.network "forwarded_port", guest: 80, host: 8080, auto_correct: true
        config.vm.network "forwarded_port", guest: 3306, host: 3306, auto_correct: true

        conf.vm.provider "virtualbox" do |vb|
            vb.memory = "1024"
            vb.cpus = "2"
            vb.gui = false
            vb.customize ["modifyvm", :id, "--vram", "32"]
            vb.customize ["modifyvm", :id, "--accelerate3d", "off"]
            vb.customize ["modifyvm", :id, "--ostype", "Ubuntu_64"]
            vb.customize ["modifyvm", :id, "--boot1", "dvd"]
            vb.customize ["modifyvm", :id, "--boot2", "disk"]
            vb.customize ["modifyvm", :id, "--audio", "none"]
            vb.customize ["modifyvm", :id, "--clipboard", "hosttoguest"]
            vb.customize ["modifyvm", :id, "--draganddrop", "hosttoguest"]
            vb.customize ["modifyvm", :id, "--paravirtprovider", "kvm"]
        end
        conf.vm.provision "shell", inline: $os_update
        conf.vm.provision :shell, path: "bootstrap.sh"
    end
end
```

Save and write this file.

You can also download from:

```
$ curl -o Vagrantfile  http://securityhardening.com/files/Vagrantfile_20200928.txt
```

2.4.2. bootstrap.sh

Inside of the kali-linux-vm directory, populate a new file with the exact name, `bootstrap.sh`. Case matters, all lowercase. See comment about downloading this file immediately preceding the code block. `bootstrap.sh` (include the shebang in your file: the first line with `#!/usr/bin/env bash`):

```
#!/usr/bin/env bash
PHP_FPM_PATH_INI='/etc/php/7.0/fpm/php.ini'
PHP_FPM_POOL_CONF='/etc/php/7.0/fpm/pool.d/www.conf'
MYSQL_ROOT_PW='Assword12345'
MYSQL_dvwa_user='dvwa'
MYSQL_dvwa_password='sunshine'
DVWA_admin_password='admin'
recaptcha_public_key='u8392ihj32kl8hujalkshuil32'
recaptcha_private_key='89ry8932873832lih32ilj32'


install_base() {
    add-apt-repository -y ppa:nginx/stable
    sudo apt-get update
    sudo apt-get dist-upgrade -y
    sudo apt-get install -y \
        nginx \
        mariadb-server \
        mariadb-client \
        php \
        php-common \
        php-cgi \
        php-fpm \
        php-gd \
        php-cli \
        php-pear \
        php-mcrypt \
        php-mysql \
        php-gd \
        git \
        vim
}

config_mysql(){
    mysqladmin -u root password "${MYSQL_ROOT_PW}"
## Config the mysql config file for root so it doesn't prompt for password.
## Also sets pw in plain text for easy access.
## Don't forget to change the password here!!

cat <<EOF > /root/.my.cnf
[client]
user="root"
password="${MYSQL_ROOT_PW}"
EOF
    mysql -BNe "drop database if exists dvwa;"
    mysql -BNe "CREATE DATABASE dvwa;"
    mysql -BNe "GRANT ALL ON *.* TO '"${MYSQL_dvwa_user}"'@'localhost' IDENTIFIED BY '"${MYSQL_dvwa_password}"';"

    systemctl enable mysql
    systemctl restart mysql
    sleep 2
}


config_php(){
    ## Config PHP FPM INI to disable some security settings:

    sed -i 's/^;cgi.fix_pathinfo.*$/cgi.fix_pathinfo = 0/g' ${PHP_FPM_PATH_INI}
    sed -i 's/allow_url_include = Off/allow_url_include = On/g' ${PHP_FPM_PATH_INI}
    sed -i 's/allow_url_fopen = Off/allow_url_fopen = On/g' ${PHP_FPM_PATH_INI}
    sed -i 's/safe_mode = On/safe_mode = Off/g' ${PHP_FPM_PATH_INI}
    echo "magic_quotes_gpc = Off" >> ${PHP_FPM_PATH_INI}
    sed -i 's/display_errors = Off/display_errors = On/g' ${PHP_FPM_PATH_INI}

    ## explicitly set pool options
    ## (these are defaults in ubuntu 16.04 so i'm commenting them out.
    ## If they are not defaults for you try uncommenting these)
    #sed -i 's/^;security.limit_extensions.*$/security.limit_extensions = \
    #.php .php3 .php4 .php5 .php7/g' /etc/php/7.0/fpm/pool.d/www.conf
    #sed -i 's/^listen.owner.*$/listen.owner = www-data/g' /etc/php/7.0/fpm/pool.d/www.conf
    #sed -i 's/^listen.group.*$/listen.group = www-data/g' /etc/php/7.0/fpm/pool.d/www.conf
    #sed -i 's/^;listen.mode.*$/listen.mode = 0660/g' /etc/php/7.0/fpm/pool.d/www.conf


    systemctl restart php7.0-fpm
}


config_nginx(){

cat << 'EOF' > /etc/nginx/sites-enabled/default
server
{
    listen  80;
    root /var/www/html;
    index index.php index.html index.htm;
    #server_name localhost
    location "/"
    {
        index index.php index.html index.htm;
        #try_files $uri $uri/ =404;
    }

    location ~ \.php$
    {
        include /etc/nginx/fastcgi_params;
        fastcgi_pass unix:/var/run/php/php7.0-fpm.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $request_filename;
    }
}
EOF

    systemctl restart nginx
}


install_dvwa(){

    if [[ ! -d "/var/www/html" ]];
    then
          mkdir -p /var/www;
          ln -s /usr/share/nginx/html /var/www/html;
          chown -R www-data. /var/www/html;
    fi

    cd /var/www/html
    rm -rf /var/www/html/.[!.]*
    rm -rf /var/www/html/*
    git clone https://github.com/ethicalhack3r/DVWA.git ./
    chown -R www-data. ./
    cp config/config.inc.php.dist config/config.inc.php

    ### chmod uploads and log file to be writable by nobody
    chmod 777 ./hackable/uploads/
    chmod 777 ./external/phpids/0.6/lib/IDS/tmp/phpids_log.txt

    ## change the values in the config to match our setup (these are what you need to update!
    sed -i '/db_user/ s/root/'${MYSQL_dvwa_user}'/' /var/www/html/config/config.inc.php
    sed -i '/db_password/ s/p@ssw0rd/'${MYSQL_dvwa_password}'/' /var/www/html/config/config.inc.php
    sed -i "/recaptcha_public_key/ s/''/'"${recaptcha_public_key}"'/" /var/www/html/config/config.inc.php
    sed -i "/recaptcha_private_key/ s/''/'"${recaptcha_private_key}"'/" /var/www/html/config/config.inc.php

}
```

```
update_mysql_user_pws(){
## The mysql passwords are set via /usr/share/nginx/html/dvwa/includes/DBMS/MySQL.php.
#  If you edit this every time they are reset it will reset to those.
#  Otherwise you can do a sql update statement to update them all (they are just md5's of the string.
#  The issue is the users table doesn't get created until you click that button T_T to init.

#mysql -BNe "UPDATE dvwa.users SET password = md5('YOUR_MYSQL_PW_HERE') WHERE user = 'admin';"
#mysql -BNe "UPDATE dvwa.users SET password = md5('YOUR_MYSQL_PW_HERE') WHERE user = 'gordonb';"
#mysql -BNe "UPDATE dvwa.users SET password = md5('YOUR_MYSQL_PW_HERE') WHERE user = '1337';"
#mysql -BNe "UPDATE dvwa.users SET password = md5('YOUR_MYSQL_PW_HERE') WHERE user = 'pablo';"
#mysql -BNe "UPDATE dvwa.users SET password = md5('YOUR_MYSQL_PW_HERE') WHERE user = 'smithy';"

sed -i '/admin/ s/password/'${DVWA_admin_password}'/g'  /var/www/html/dvwa/includes/DBMS/MySQL.php
sed -i '/gordonb/ s/abc123/'${DVWA_admin_password}'/g'  /var/www/html/dvwa/includes/DBMS/MySQL.php
sed -i '/1337/ s/charley/'${DVWA_admin_password}'/g'  /var/www/html/dvwa/includes/DBMS/MySQL.php
sed -i '/pablo/ s/letmein/'${DVWA_admin_password}'/g'  /var/www/html/dvwa/includes/DBMS/MySQL.php
sed -i '/smithy/ s/password/'${DVWA_admin_password}'/g'  /var/www/html/dvwa/includes/DBMS/MySQL.php
}


install_base
config_mysql
install_dvwa
update_mysql_user_pws
config_php
config_nginx
```

Save and write this file.

If you have issues with copying and pasting the above file because code blocks in PDFs always copy correctly [NOT!], you could use curl, i.e. Make sure the bootstrap.sh file ends up in the same directory as the Vagrantfile.

```
$ curl -o bootstrap.sh  http://securityhardening.com/files/bootstrap_sh_20200928.txt
```

From a Terminal, change directory to:

```
$ cd ${HOME}/Source_Code/Education/vagrant-machines/kali-linux-vm/
```

Then run (inside the directory kali-linux-vm):

```
$ vagrant up
```

This will download the appropriate images and start the virtual machines. Once running, through the VirtuaBox GUI, login as root. Password is “toor”, root backwards. Edit the following file: `/etc/ssh/sshd_config`

And change the line: `#PermitRootLogin prothibit-password` To: `PermitRootLogin yes` Meaning strip the comment out on the beginning of the line and alter `prohibit-password` to `yes`.

Then restart the ssh daemon:

```
# kill -HUP $(pgrep sshd)
```

Notice, you are on a Bridged adapter, this will open the instance to allow root to ssh in with the most unsecure password in the world. Only make this change (allowing root to login via SSH) if you require root SSH access. You can change the root user’s password, which is highly recommended.

For the DVWA instance, I would first run ‘vagrant status’ to capture the name that vagrant is using for the running instance.

```
# vagrant status
```

Choose

```
Current machine states:
kali-linux-vagrant running (virtualbox)
dvwa-vagrant running (virtualbox)
```

This environment represents multiple VMs. The VMs are all listed above with their current state. For more information about a specific VM, run `vagrant status NAME`.

From there, log into the DVWA instance with:

```
$ vagrant ssh dvwa-vagrant
```

And then get the current IP address.

```
$ ip a
```

Choose the second network adapter, it should look like:

```
ubuntu@dvwa:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 02:53:17:3c:de:80 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::53:17ff:fe3c:de80/64 scope link
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:f0:77:2d brd ff:ff:ff:ff:ff:ff
    inet 172.20.156.76/24 brd 172.20.156.255 scope global enp0s8
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fef0:772d/64 scope link
        valid_lft forever preferred_lft forever
```

The test network used for this paper uses 172.20.156.0/24 as the network range [shown here in section 3]. Therefore, the adapter, enp0s8 is what he is looking for. The IP to use as a target is 172.20.156.76. Write down your value.

3. SIP Army Knife

In order to resolve missing dependencies, you will need to perform a Distribution upgrade on the Kali Linux system.

On your Host system, running VirtualBox and Vagrant, open a teminal and run:

```
$ vagrant status
```

Then SSH into the Kali Linux namespace

```
$ vagrant ssh kali-linux-vagrant
```

Elevate to the root user

```
$ sudo su -
```

Create a new user and assign a new password so we are not logging in as the root user The default password for root is `toor` in case you need it for Kali Linux.

```
# useradd kali

# passwd kali
```

Get updates first

```
# apt-get updates -y
```

Don't tack on the `-y` to answer yes by default. Manually take the default choice for questions with this process. If you see an error with the package `libnss` do a reboot at that point and re-run the line with dist-upgrade again.

```
apt-get upgrade -f
apt-get dist-upgrade -f
dpkg --configure -a
```

Reboot the system to use the new Kernel and glibc libraries.

```
# reboot
```

Log back in and elevate to the root user. Now install SIP Army Knife with this command

```
# apt-get install -y siparmyknife
```

In case my IP address Changed for DVWA, I am going to run this command from my Host

```
$ vagrant ssh dvwa-vagrant -c "ip a"
```

My attack sequence, from inside of the Kali Linux VM should look like this:

```
# /usr/bin/siparmyknife -h 172.20.156.142
```

Output:

```
CANT OPEN SOCKET!!!
IO::Socket::INET: connect: Connection refused
```

Debugging:

```
root@kali-linux-vagrant:~# nc -v 172.20.156.142 80
172.20.156.142: inverse host lookup failed: Unknown host
(UNKNOWN) [172.20.156.142] 80 (http) open
```

So, something flaky is going on here with Kali Linux in regards to versioning or dependencies [my assessment].

I did Google for the error code and found a lot of conjecture or theories.

Stopping here.

4. Conclusion

Before I start with the conclusion, let me state that I have some bias towards Kali Linux. It is an amazing tool. I have been and remain passionate about this tooling since I first discovered it around seven years ago.

The subject of this paper was originally going to be about `Powerfuzzer`. I spent hours researching notes on the tool and getting everything ready. When I went to install `Powerfuzzer` into my base Virtual Machine for Kali Linux, it did not show up as a tool to install. I did some research and indeed it is listed on the Kali Linux Tools website, but its not available. At this point, I cut my losses and switched tools to SIP Army Knife.

The tool did show up when I ran: `apt-cache search armyknife`. But when I went to install the tool, it failed with a conflict for something related to the beautiful user interface. I decided to upgrade the distribution. That then allowed me to install the tool.

However, when I went to run the tool, it gives me an error about the socket connection being refused.

The finale of all of this is, Kali Linux is still an amazing set of tools for any security professional. If you have the ability [skills] to lean in and donate your time to one or more project(s) in order to update the tooling to work on the newer versions of base packages, like Perl or Python [my implication is that there are dependency errors that need to be resolved for these tools to continue to work as the Software Development Lifecycle moves forward for the base of Kali Linux]; please do so. There are 8,760 hours in a year. If you would dontate 10 hours to updating projects like this; you are giving back to the community and helping ensure a rich set of tools that everyone can use.

5. Appendix

References

https://tools.kali.org/vulnerability-analysis/siparmyknife

https://packetstormsecurity.com/files/107301/SIP-Army-Knife-Fuzzer-1123

https://gitlab.com/kalilinux/packages/siparmyknife

https://en.wikipedia.org/wiki/Fuzzing

Kali Linux Upgrade Sequence

Elevate to the root user

```
$ sudo su -
```

Get updates first

```
# apt-get updates -y
```

Do NOT tack on the `-y` to answer yes by default. Manually take the default choice for questions with this process. If you see an error with the package `libnss` do a reboot at that point and re-run the line with dist-upgrade again.

```
apt-get upgrade -f
apt-get dist-upgrade -f
dpkg --configure -a
```

Reboot the system to use the new Kernel and glibc libraries.

```
# reboot
```

Source Code at the time of writting

This is what the Software looked like during the author's evaluation:

```
#!/usr/bin/perl
#SIP VoIP Protocol Fuzzer
#Created: Blake Cornell

use strict;
#use warnings; LOTS OF WARNING ____ SOLVE THIS AND INCREASE EFFICIENTY

use IO::Select;
use IO::Socket;
use IO::Socket::INET;
use Getopt::Long;
use Pod::Usage;
use Time::HiRes qw( alarm );
use Digest::MD4 qw( md4_hex );
use Digest::MD5 qw( md5_hex );
use Digest::CRC qw( crc32 crc16 );
use HTML::Entities;

my @timeoutDetection = ();
my @md5Requests = ();
my @md4Requests = ();
my @crc32Requests = ();
my @crc16Requests = ();
my $packetCount = 0;
my $socketType='';
my $result = GetOptions('host|h=s' => \(my $host = ''),
      'dport|p=s' => \(my $dport = ''),
      'sport|p=s' => \(my $sport = ''),
      'verbose|v' => \(my $verbose),
      'veryverbose|vv' => \(my $veryVerbose),
      'connection|c' => \(my $connection), #to listen to response or not
      'density|d=s' => \(my $density = 0), #determines how many mutations to use
      'timeout|t=s' => \(my $timeout = .1),
      'count' => \(my $countTests = 0), #counts the number of packets to test
      'md4' => \(my $md4), #can cause timeouts
      'md5' => \(my $md5), #can cause timeouts
      'crc32' => \(my $crc32), #can cause timeouts
      'crc16' => \(my $crc16), #can cause timeouts
      'start=s' => \(my $startPosition), #if set, then start at this position
      'stringFormats' => \(my $stringFormats),
      'stringOverflows' => \(my $stringOverflows),
      'integerFormats' => \(my $integerFormats),
      'injectHeaders' => \(my $injectHeaders),
      'xss' => \(my $xss),
      'sqli' => \(my $sqli),
      'callId' => \(my $callId), #call id is incremented
      'detectVersion' => \(my $detectVersion),
      'getOptions' => \(my $getOptions),
      'help' => \(my $help),
      'proto=s' => \(my $proto),
      'sproto=s' => \(my $sproto),
      'source|s=s' => \(my $source = '')) or pod2usage(2); #sip source IP

print "\n\n";
if($help) { displayHelp(); };
if(!$host) { print "-h, Enter host\n"; exit 1; }
if(!$dport) { $dport = 5060; }
if(!$sport) {
  $sport = 12345;
  if($verbose) {
    print "Source Spoof Port default setting: " . $sport . "\n";
  }
}else{
  if($verbose) {
    print "Source Spoof Port user setting: " . $sport . "\n";
  }
}
if(!$connection) { $connection = 1; }else{ $connection = 0; }
if(!$density) { $density = 0; }
if(!$source) { $source = $host; }



$proto = uc($proto);
if(!$proto || ($proto != 'TCP' && $proto != 'UDP')) {
  $proto = "TCP";
  if($verbose) { print "Destination Protocol/Layer 4 Protocol default setting: " . $proto . "\n" };
}else{
  if($verbose) { print "Destination Protocol/Layer 4 Protocol user setting: " . $proto . "\n" };
}


$sproto = uc($sproto);
if(!$sproto || ($sproto != 'TCP' && $sproto != 'UDP')) {
  $sproto = "TCP";
  if($verbose) { print "Source Protocol/Layer 4 Protocol default setting: " . $sproto . "\n" };
}else{
  if($verbose) { print "Source Protocol/Layer 4 Protocol user setting: " . $sproto . "\n" };
}

print "\n\n\n";

if($getOptions) { print getOptions($host,$source,$sport); exit; }

my @requestTypes01=('BYE','OPTIONS','PRACK','PUBLISH','INFO','MESSAGE','UPDATED','REFER','SUBSCRIBE','NOTIFY');
#my
```

```
@requestTypes01=('REGISTER','INVITE','BYE','OPTIONS','PRACK','PUBLISH','INFO','MESSAGE','UPDATED','REFER','SUBSCRIB
E','NOTIFY');

if($detectVersion || $verbose) {
  my $response = getOptions($host,$source,$sport);
  if(lc($response) =~ m/\nserver: (.[^\n]+)\n/s) {
    print 'Server Header Detected: '.$1."\n";
    print "This is probably A PBX. Adjusting SIP methods accordingly.\n";

@requestTypes01=('REGISTER','INVITE','BYE','OPTIONS','PRACK','PUBLISH','INFO','MESSAGE','UPDATED','REFER','SUBSCRIB
E','NOTIFY');
  }elsif(lc($response) =~ m/\nuser-agent: (.[^\n]+)\n/s) {
    print 'Phone Version Detected: '.$1."\n";

@requestTypes01=('INVITE','BYE','OPTIONS','PRACK','PUBLISH','INFO','MESSAGE','UPDATED','REFER','SUBSCRIBE','NOTIFY'
);
    print "This is probably an end user device. Adjusting SIP methods accordingly.\n";
  }else{
    print "No Server nor User-Agent header.\n";
  }
}

my @packTypes = ('c');
my @strOverflows = ("A",pack("H*",(0x61.0x00)),pack("H*",(0x0d.0x0a)),pack("H*",(0x1b)),pack("H*",(0x00)));

my $strOverflowLen = $#strOverflows;
for(my $i=0;$i<=$strOverflowLen;$i++) {
  push(@strOverflows,$strOverflows[$i]x32768);
  push(@strOverflows,$strOverflows[$i]x16384);
  push(@strOverflows,$strOverflows[$i]x8192);
  push(@strOverflows,$strOverflows[$i]x4096);
  push(@strOverflows,$strOverflows[$i]x2048);
  push(@strOverflows,$strOverflows[$i]x1024);
  push(@strOverflows,$strOverflows[$i]x512);
  push(@strOverflows,$strOverflows[$i]x256);
  push(@strOverflows,$strOverflows[$i]x128);
}

my @strFormats = ('%c','%d','%i','%e','%E','%f','%g','%G','%o','%u','%x','%X','%p','%s','%n');
my $strFrmtLen = $#strFormats;
for(my $i=0;$i<=$strFrmtLen;$i++) {
    push(@strOverflows,$strFormats[$i]x32768);
    push(@strOverflows,$strFormats[$i]x16384);
    push(@strOverflows,$strFormats[$i]x8192);
    push(@strOverflows,$strFormats[$i]x4096);
    push(@strOverflows,$strFormats[$i]x2048);
    push(@strOverflows,$strFormats[$i]x1024);
    push(@strOverflows,$strFormats[$i]x512);
    push(@strOverflows,$strFormats[$i]x256);
    push(@strOverflows,$strFormats[$i]x128);
    push(@strOverflows,substr($strFormats[$i],0,1)."9999999".substr($strFormats[$i],1,1));
    push(@strOverflows,substr($strFormats[$i],0,1).".4097".substr($strFormats[$i],1,1));
    push(@strOverflows,substr($strFormats[$i],0,1).".9999".substr($strFormats[$i],1,1));
    push(@strOverflows,substr($strFormats[$i],0,1)."-.0".substr($strFormats[$i],1,1));
}

my @chars = (" ",":","<",">","@",".","/",";","=","-","\n","\t","\r"); #CUT THIS UP W/DENSITY
for(my $i=0;$i<=$#chars;$i++) {
    push(@strOverflows,$chars[$i]x32000);
}
push(@strOverflows,"SHOULDNOTBEINLOGS "x100);

my @intFormats = (1,0.0,.0,"0.".("0"x32000)."1",("0"x32000)."1");
my $intFormatLen = $#intFormats;
for(my $i=0; $i<=$#chars;$i++) {
  push(@intFormats,$intFormats[$i]*-1);
}

my @xssInjections=(  '<script>alert(PAYLOAD)</script>',
      '"><script>alert(PAYLOAD)</script>',"'><script>alert(PAYLOAD)</script>",
      '" style="javascript:alert(PAYLOAD);"',"' style='javascript:alert(PAYLOAD);'",
      '" onload="alert(PAYLOAD);"',"' onload='alert(PAYLOAD);'",
      '<IMG SRC="javascript:alert(\'PAYLOAD\');">',"<IMG SRC='javascript:alert(\"PAYLOAD\");'>",
      '<IMG """><SCRIPT>alert("PAYLOAD")</SCRIPT>">',"<IMG '''><SCRIPT>alert('PAYLOAD')</SCRIPT>'>");
```

```
my @headerInjections=("\nSipFuzzerHeader: value\n");


#my @logInjection=("[DATE] NOTICE[4069] chan_sip.c: Registration from '2 <sip:voip0day@$source>' failed for
'$source' - No matching peer found");

#path needs to be updated:
#my @pbxWare=("");
#my @trixBox=("");
#my @fonality=("");
#my @switchVox=("");
#my @cisco=("");
#my @avaya=("");
#my @aastra=("");
#my @asteriskRealtime=("");
#my @startFishPBX=("");


my @sqlInjections=("' or 1=1","' or 1=1',"' or 1=0","' or 1=0');

if($density >= 10) {
  push(@requestTypes01,'ACK','CANCEL');
}


my @headerTypes=('Via','Max-Forwards','Contact','To','From','User-Agent','Call-ID','To','From','User-Agent','Call-
ID','CSeq','Content-Type','Content-Length'); #Route, Record-Route,

my %responseCodes = (100=>'Trying',
      180=>'Ringing',
      181=>'Call Is Being Forwarded',
      182=>'Queued',
      183=>'Session Progress',
      200=>'Ok',
      202=>'Accepted: Cannot Process',
      300=>'Multiple Choices',
      301=>'Moved Permanetly',
      302=>'Moved Temporarily',
      305=>'Use Proxy',
      380=>'Alternative Service',
      400=>'Bad Request',
      401=>'Unauthorized',
      402=>'Payment Required',
      403=>'Forbidden',
      404=>'Not Found',
      405=>'Method Not Allowed',
      406=>'Not Acceptable',
      407=>'Proxy Authentication Required',
      408=>'Request Timeout',
      409=>'Conflict',
      410=>'Gone',
      412=>'Consitional Request Failed',
      413=>'Request Entity Too Large',
      414=>'Request-URI Too Long',
      415=>'Unsupported Media Type',
      416=>'Unsupported URI Scheme',
      417=>'Unknown Resource-Priority',
      420=>'Bad Extension',
      421=>'Extension Required',
      422=>'Session Interval Too Small',
      423=>'Inverval Too Brief',
      424=>'Bad Location Information',
      428=>'Use Identity Header',
      429=>'Provide Referrer Identity',
      433=>'Anominity DIssalowed',
      436=>'Bad Identity-Info',
      437=>'Unsupported Certificate',
      438=>'Invalid Identity Header',
      480=>'Temporarily Unavailable',
      481=>'Call/Transaction Does Not Exist',
      482=>'Loop Detected',
      483=>'Too Many Hops',
      484=>'Address Incomplete',
      485=>'Ambiguous',
      486=>'Busy Here',
      487=>'Request Terminated',
      488=>'Not Acceptable Here',
      489=>'Bad Event',
      491=>'Request Pending',
      493=>'Undecipherable S/MIME Body',
      494=>'Security Agreement Required',
      500=>'Internal Server Error',
      501=>'Not Implimented: REQUEST METHOD',
      502=>'Bad Gateway',
      503=>'Service Unavailable',
      504=>'Server Time-Out',
      505=>'Version Not Supported: SIP PROTOCOL',
      513=>'Message Too Large',
      580=>'Precondition Failure',
      600=>'Busy Everywhere',
      603=>'Decline',
      604=>'Does Not Exist Anywhere',
      606=>'Not Acceptable'
      );

sub sendFiniteMutation {
    my($pack,$index,$injectString,@args)=@_;

    deliverPacket(substr($pack,0,$index).$injectString,$proto);
    deliverPacket(substr($pack,0,$index).$injectString.substr($pack,$index),$proto);
    deliverPacket(substr($pack,0,$index).$injectString.substr($pack,$index-1),$proto);
    deliverPacket(substr($pack,0,$index).$injectString.substr($pack,$index+1),$proto);
    deliverPacket(substr($pack,0,$index-1).$injectString,$proto);
    deliverPacket(substr($pack,0,$index-1).$injectString.substr($pack,$index),$proto);
    deliverPacket(substr($pack,0,$index-1).$injectString.substr($pack,$index-1),$proto);
    deliverPacket(substr($pack,0,$index-1).$injectString.substr($pack,$index+1),$proto);
    deliverPacket(substr($pack,0,$index+1).$injectString,$proto);
    deliverPacket(substr($pack,0,$index+1).$injectString.substr($pack,$index),$proto);
    deliverPacket(substr($pack,0,$index+1).$injectString.substr($pack,$index-1),$proto);
    deliverPacket(substr($pack,0,$index+1).$injectString.substr($pack,$index+1),$proto);

}

my %sdpDirectives = (
      'a'=>'Session Attribute',
      'b'=>'Bandwidth Information',
      'c'=>'Connection Information',
      'e'=>'Email Address',
      'i'=>'Session Information/Media Title',
      'k'=>'Encryption Key',
      'o'=>'Owner/Creator and Session Identifier',
      'p'=>'Phone Number',
      's'=>'Session Name',
      'u'=>'URI of Description',
      'v'=>'Protocol Version',
      'z'=>'Time Zone Adjustments',
      't'=>'Time The Session Has Been Active',
      'r'=>'Repeat Times');

my $currentCallId = "0000000000";#0000000000;
if($startPosition) {
  $currentCallId = ((split('_',$startPosition))[1]);
}

my @spoofHosts = ($source,$host,'127.0.0.1');
my @sourceNames = ('bob','doesNotExistSource');
my @destNames = ('alice','doesNotExistDest');


foreach my $spoofHost(@spoofHosts) {

my $content = '';

my $lastPacketCount = 0;
 foreach my $requestType (@requestTypes01) {

#if($requestType == "REGISTER") {
#  $content = '';
#}else{
  $content = 'v=0
```

```
o=- 6 2 IN IP4 '.$spoofHost.'
s=A B C
c=IN IP4 '.$spoofHost.'
t=0 0
m=audio 15508 RTP/AVP 107 119 100 106 0 105 98 8 3 101
a=alt:1 3 : Sf+epwJ/ N2AgCbzU '.$spoofHost.' 15508
a=alt:2 2 : 16gbQBzu 3rvjxmQo '.$spoofHost.' 15508
a=alt:3 1 : CAKEYBiS GGAivEwQ '.$spoofHost.' 15508
a=fmtp:101 0-15
a=rtpmap:107 BV32/16000
a=rtpmap:119 BV32-FEC/16000
a=rtpmap:100 SPEEX/16000
a=rtpmap:106 SPEEX-FEC/16000
a=rtpmap:105 SPEEX-FEC/8000
a=rtpmap:98 iLBC/8000
a=rtpmap:101 telephone-event/8000
a=sendrecv';

#}


my $pack = $requestType.' sip:bob@'.$host.' SIP/2.0
Via: SIP/2.0/'.$proto.' '.$source.':'.$sport.';branch=z9hG4bK-d8754z-b538815be3603112-1---d8754z-;rport='.$sport.'
Max-Forwards: 70
To: bob <sip:bob@'.$host.'>
From: "alice" <sip:alice@'.$source.'>;tag=102
Contact: <sip:bob@'.$source.'>
User-Agent: A_B_C
Call-ID: '.$currentCallId.'
CSeq: 1 '.$requestType.'
Content-Type: application/sdp
Content-Length: '.length($content).'

'.$content;


    foreach my $char (@chars) {
  my $offset = 0;
  my $index = 0;
  my $lastTimestamp = time;
  while($index != -1) {
      my $currentPosition = ($requestType.'_'.$currentCallId.'_'.$char.'_'.$index.'_'.$offset);
      if(!$startPosition || $startPosition eq $currentPosition) {
    $startPosition = undef;
    if($verbose || $countTests) {
      print '['.currentTime().'] '.time.': ';
      print $currentPosition.": ";
      print time-$lastTimestamp." Seconds, ".($packetCount-$lastPacketCount)." Packet Count, ";
      if(time == $lastTimestamp) { $lastTimestamp--; }
      print (($packetCount-$lastPacketCount)/(time-$lastTimestamp));
      print " Connections Per Second\n";

      $lastTimestamp = time;
      $lastPacketCount = $packetCount;
    }

    my $fuzz='';
    if($density >= 1) {
      deliverPacket(substr($pack,0,$index));#first to index
      deliverPacket(substr($pack,0,$index-1));
      deliverPacket(substr($pack,0,$index+1));
    }

    my $maxCharset = 127;
    if($density >= 4) { $maxCharset = 255; };
    for(my $injectChars=0;$injectChars<=$maxCharset;$injectChars++) {
        foreach my $packType (@packTypes) {
      if($density >= 1) {
          sendFiniteMutation($pack,$index,pack($packType,$injectChars));
      }
        }
    }
    if($density >= 1 || $stringFormats) {
      foreach my $strFormat (@strFormats) {
          sendFiniteMutation($pack,$index,$strFormat);
      }
    }
    if($density >= 1 || $stringOverflows) {
      foreach my $strOverflow (@strOverflows) {
          sendFiniteMutation($pack,$index,$strOverflow);
      }
    }
    if($density >= 1 || $integerFormats) {
      foreach my $intFormat (@intFormats) {
          sendFiniteMutation($pack,$index,$intFormat);
      }
    }
    if($density >= 2 || $xss) {
      foreach my $xssInject (@xssInjections) {
          $currentPosition = HTML::Entities::encode($currentPosition);
          $currentPosition = HTML::Entities::encode($currentPosition,' ');

          $xssInject=~s/PAYLOAD/\'$currentPosition\'/;
          sendFiniteMutation($pack,$index,$xssInject);
      }
    }
    if($density >= 2 || $injectHeaders) {
      foreach my $header (@headerInjections) {
        sendFiniteMutation($pack,$index,$header);
          #send variable that enables a regex via the response.
      }
    }
    if($density >= 2 || $sqli) {
      foreach my $sqlInject (@sqlInjections) {
          sendFiniteMutation($pack,$index,$sqlInject);
      }
    }
          deliverPacket(substr($pack,0,$index-1).substr($pack,$index+1));
      }else{
    if($verbose) { print $requestType.'_'.$char.'_'.$index.'_'.$offset."\n"; }
      }

      $offset = $index+1;
      $index = index($pack,$char,$offset);
  }
    }
 }
}
if($verbose) { print "Packet Count: ".$packetCount."\n"; }
exit;

sub sendFiniteMutation {
    my($pack,$index,$injectString,@args)=@_;

    deliverPacket(substr($pack,0,$index).$injectString);
    deliverPacket(substr($pack,0,$index).$injectString.substr($pack,$index));
    deliverPacket(substr($pack,0,$index).$injectString.substr($pack,$index-1));
    deliverPacket(substr($pack,0,$index).$injectString.substr($pack,$index+1));
    deliverPacket(substr($pack,0,$index-1).$injectString);
    deliverPacket(substr($pack,0,$index-1).$injectString.substr($pack,$index));
    deliverPacket(substr($pack,0,$index-1).$injectString.substr($pack,$index-1));
    deliverPacket(substr($pack,0,$index-1).$injectString.substr($pack,$index+1));
    deliverPacket(substr($pack,0,$index+1).$injectString);
    deliverPacket(substr($pack,0,$index+1).$injectString.substr($pack,$index));
    deliverPacket(substr($pack,0,$index+1).$injectString.substr($pack,$index-1));
    deliverPacket(substr($pack,0,$index+1).$injectString.substr($pack,$index+1));

}

sub checkCollision {
  my($data,@args)=@_;
  if($md5) {
    my $hash = md5_hex($data);
    if(grep(/$hash/,@md5Requests)) {
      return 0;
    }
    push(@md5Requests,$hash);
    return 1;
  }elsif($md4) {
    my $hash = md4_hex($data);
    if(grep(/$hash/,@md4Requests)) {
      return 0;
    }
    push(@md4Requests,$hash);
    return 1;
  }elsif($crc32) {
    my $hash = crc32($data);
    if(grep(/$hash/,@crc32Requests)) {
      return 0;
    }
    push(@crc32Requests,$hash);
    return 1;
  }elsif($crc16) {
    my $hash = crc16($data);
    if(grep(/$hash/,@crc16Requests)) {
      return 0;
    }
    push(@crc16Requests,$hash);
    return 1;
  }
  return 1
}

sub deliverPacket {
  my($data,$proto,@args)=@_;
  if(checkCollision($data)) {
    $packetCount++;
    if($countTests) {
      return 0;
    }
    return sendSocket($data,$host,$dport,$proto);
  }else{
    if($verbose) { print "\nSkipping Collition Packet\n"; }
    #if($verbose) { print "MD5 COLLISION!!!!!!!!!\n"; }
    #if($verbose) { print "."; }
  }
  if($callId) { $currentCallId++; };
}

sub timeoutDetection {
  my($packetNumber,@args)=@_;
}

sub sendSocket {
        my($msg,$ipaddr,$dport,$proto,@args)=@_;
  my $response='';
  my $sock = new IO::Socket::INET->new(
          #LocalPort=>$sport,
          Proto=>$proto,
          PeerPort=>$dport,
          PeerAddr=>$ipaddr) or die "CANT OPEN SOCKET!!!\n$@\n";
#      $sock->send($msg);
      print $sock $msg;

    if($connection) {
  my $MAXLEN = 1024;
  my $TIMEOUT = .1;
  if(defined($timeout) & $timeout ne '' && $timeout != 0) { #timeout of 0 hangs
    $TIMEOUT=$timeout;
  }
  eval {
    local $SIG{ALRM} = sub { die "alarm time out"; };
    alarm $TIMEOUT;
    $sock->recv($response,65535) or next;

      my $retVal = parseResponse($response,$msg);
      if($retVal == 200) {
        if(defined($veryVerbose)) {
          print "\n\n\n\n\n".$msg."\n\n\n";
        }
#send ack then bye when recieved 200, make sure there is no SDP info -> content-length=0
#once sent bye wait for 200 response.
        print "\t\tSENDING RECURSIVE REQUEST\t200\n";
        sendSocket($msg,$ipaddr,$dport,$timeout,$proto);
        print "\t\tCLEANING RECURSIVE REQUEST\n";
      }elsif($retVal == 100) { #wait on 100 for another packet., should be a 200
  #      print "\t\tSENDINT RECURSIVE REQUEST\t100\n";
  #      sendSocket($msg,$ipaddr,$dport,$timeout);
  #      print "\t\tCLEANING RECURSIVE REQUEST\n";
      }elsif($retVal == 401) {
  #      print "\t\tSENDINT RECURSIVE REQUEST\t401\n";
  #      sendSocket($msg,$ipaddr,$dport,$timeout);
  #      print "\t\tCLEANING RECURSIVE REQUEST\n";
      }
      return $response;
      #return($respaddr,$dport);
  };
  $sock->close();
    }
}

sub parseResponse {
  my($msg,$request,@args)=@_;
  my @lines=split("\n",$msg);
  my @words = split(" ",$lines[0]);

#######RULE 1: Abnormal Response Code
#  if($words[1] != 404 && $words[1] != 501 && $words[1] != 503 && $words[1] != 488) {
#    if(!$verbose) { print $words[1]."\t".$responseCodes{$words[1]}."\n"; }
#  }
#######END RULE 1
print "\t\t".$words[1]."\t".$responseCodes{$words[1]}." ......... ";
print substr($words[1],0,3);

  if(defined($veryVerbose)) {
    print $msg."\n\n\n";
  }
  if($density >= 2 || $injectHeaders) {
    if($msg =~ /SipFuzzerHeader: value/) {
      print "\t\t\tHEADER INJECTION\n";
      print "\t\t\tHEADER INJECTION\n";
      print "\t\t\tHEADER INJECTION\n";
      print "\t\t\tHEADER INJECTION\n";
      print $request."\n\n".$msg."\n\n";
    }
  }
  if(substr($words[1],0,1)==1) {#PROVISIONAL
    if(substr($words[1],1,2)== 00) {
      return 100;
    }
  }elsif(substr($words[1],0,1)==2) {#SUCCESS
    if(substr($words[1],1,2)== 00) {
      #if(defined($veryVerbose)) {
        print "\t\t200\n\n";
      #}
      return 200;
    }else{
      #if(defined($veryVerbose)) {
        print "\t\t2XX\n\n";
      #}
      return substr($words[1],0,3);
    }
  }elsif(substr($words[1],0,1)==3) {#REDIRECTION
  }elsif(substr($words[1],0,1)==4) {#CLIENT ERROR
    if(substr($words[1],1,2) == 01) {
      foreach my $line(@lines) {
        if($line =~ /^WWW-Authenticate/) {
          return 401;
        }
      }
    }elsif(substr($words[1],1,2) == 04) {
      return 404;
    }
  }elsif(substr($words[1],0,1)==5) {#SERVER ERROR
  }elsif(substr($words[1],0,1)==6) {#GLOBAL FAILURE
  }
}

sub displayHelp {
  print "THIS IS THERE TO PUT THE HELP REFERENCE\n";
  exit;
}

#http://perl.about.com/od/perltutorials/a/perllocaltime_2.htm
sub currentTime {
  my @months = qw(Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec);
  my @weekDays = qw(Sun Mon Tue Wed Thu Fri Sat Sun);
  my($second, $minute, $hour, $dayOfMonth, $month, $yearOffset, $dayOfWeek, $dayOfYear, $daylightSavings) =
localtime();
  my $year = 1900 + $yearOffset;
  my $theTime = "$hour:$minute:$second, $weekDays[$dayOfWeek] $months[$month] $dayOfMonth, $year";
  return $theTime;
}

sub getOptions {
  my($dhost,$source,$sport,$proto,@args)=@_;

my $request = 'OPTIONS sip:bob@'.$dhost.' SIP/2.0
Via: SIP/2.0/UDP '.$source.':'.$sport.'
From: sip:alice@'.$source.';tag=55a66b
To: sip:bob@'.$dhost.'
Call-ID: 70710@'.$source.'
CSeq: 1 OPTIONS';

  return deliverPacket($request, $proto);
}
```