Dagah User Manual
1.1.7

About Shevirah:
Shevirah is a U.S. company founded in 2015 by cybersecurity expert Georgia Weidman. We specialize in products for automated mobile and IoT device vulnerability assessment, penetration testing, and mobile security awareness training. Our capabilities compliment traditional Mobile Threat Defense (MTD), Enterprise Mobile Management (EMM), Mobile Device Management (MDM), or mobile app inspection tools. Shevirah's name comes from the Hebrew word for "shattering or breaking of vessels", reflecting the goals of cybersecurity assessment and testing. Shevirah's platform includes the patent pending Dagah software. We are headquartered in the greater Washington, D.C. area. Learn more, or request a free trial, at Shevirah.com.

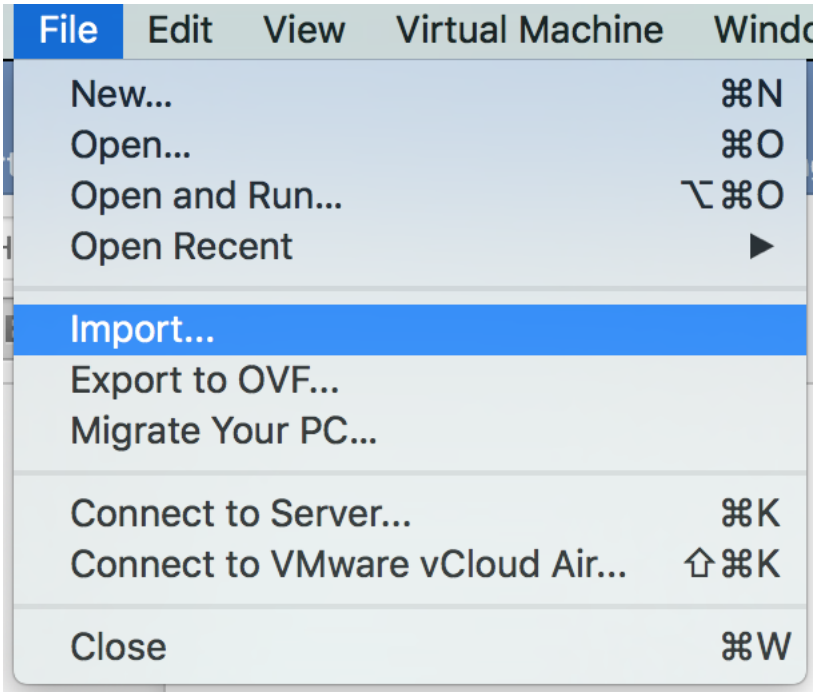
Installation

Virtual Machine

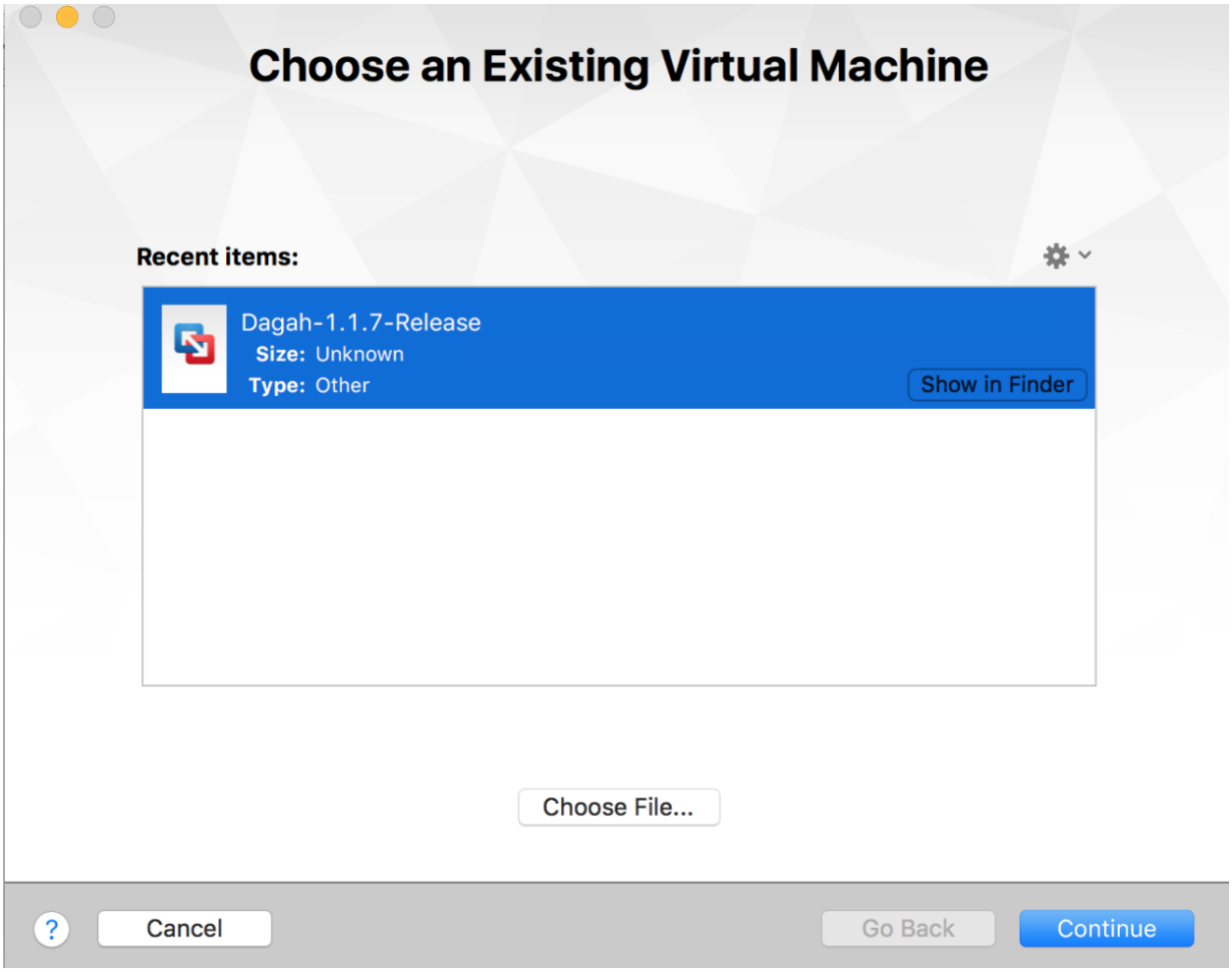
The virtual machine OVA can be imported into Vmware or Virtualbox. We will use Virtualbox in this guide.

Installing in Vmware:

To import the OVA into Vmware open Vmware and go to File->Import

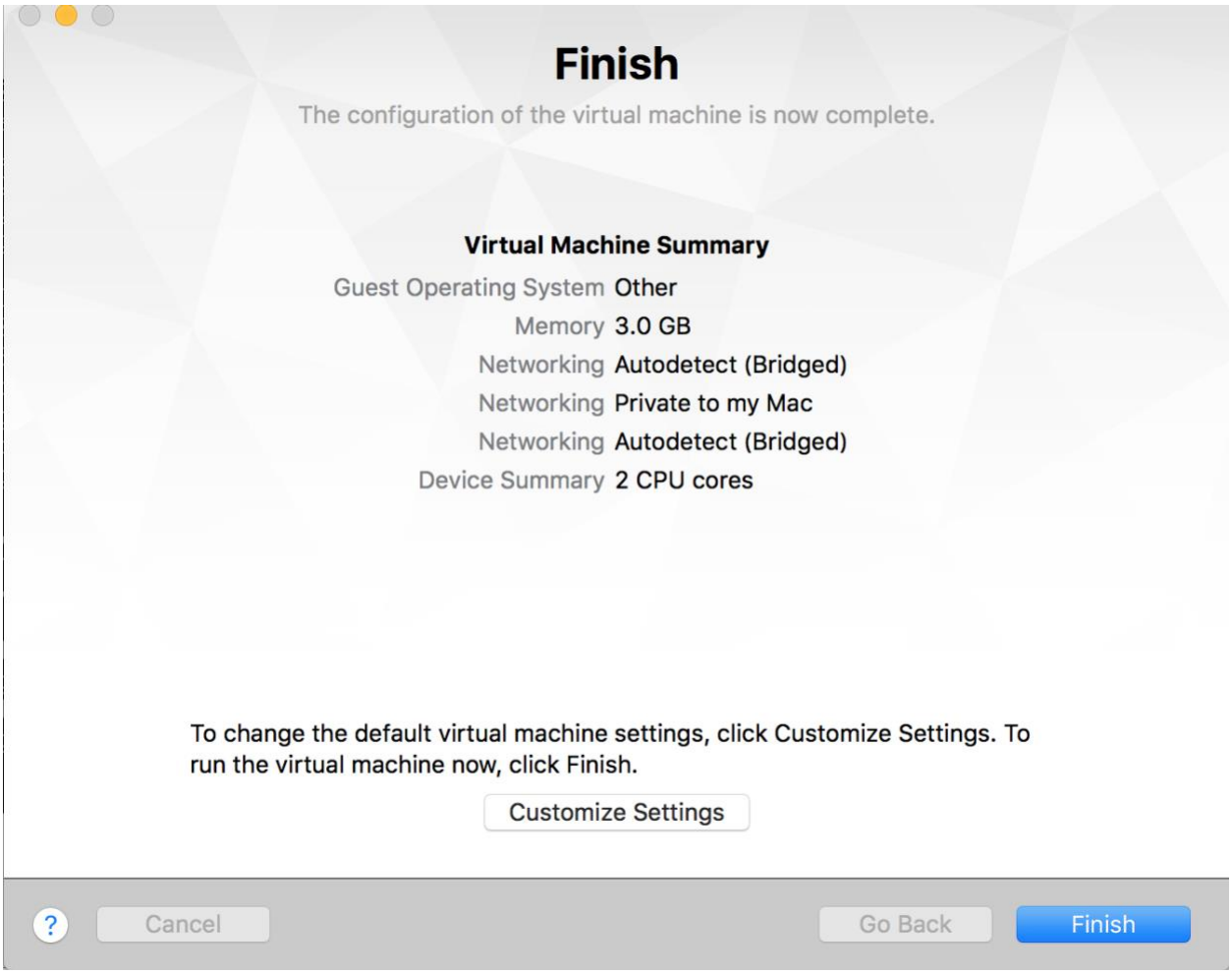


Select the OVA in the “Choose an Existing Virtual Machine” window.



Click continue. You may encounter an error about the OVF specification like the one shown below. This is a common issue with importing OVA files. Just click Retry and the import should continue without issue.

Once the import has finished click Finish and start the virtual machine.



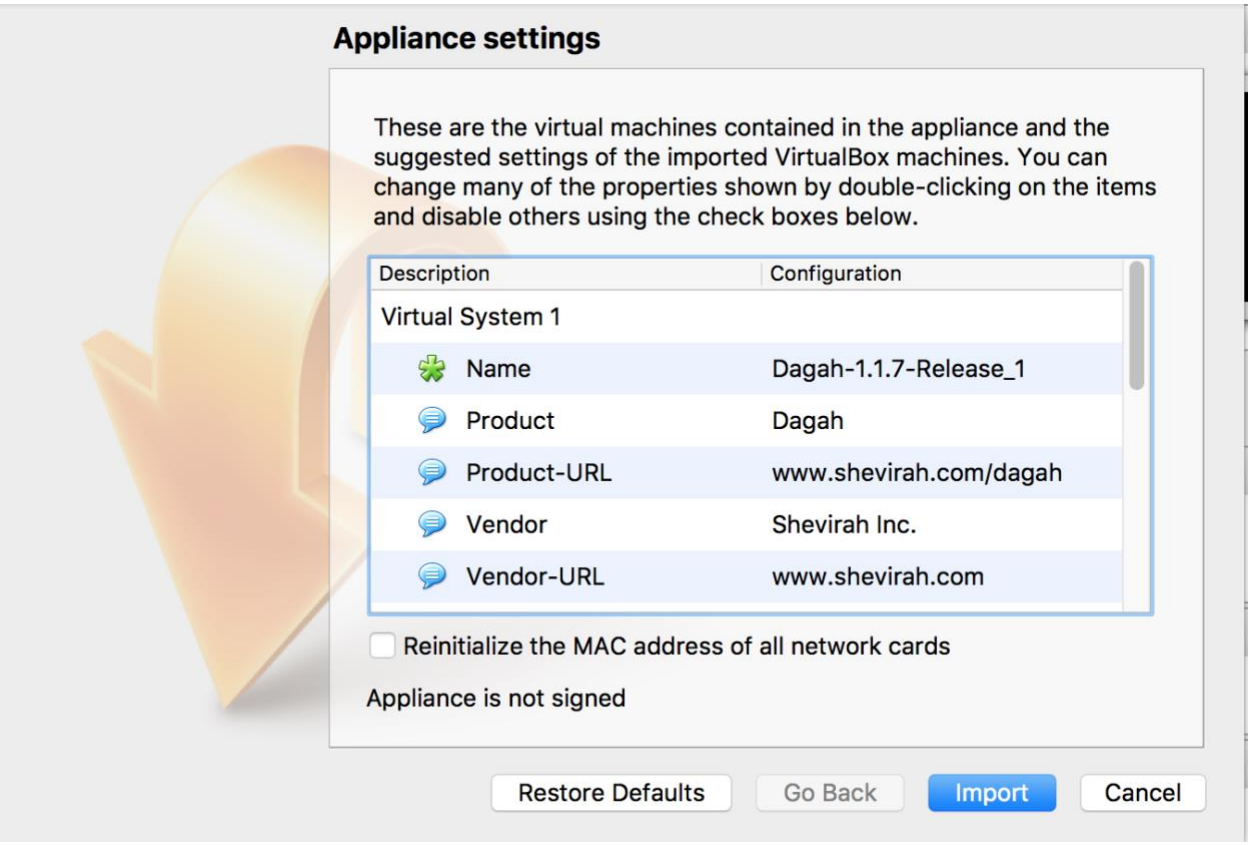
Installing in Virtual Box:

Note: If you are setting up a virtual machine for the first time, make sure that the network settings are correct.

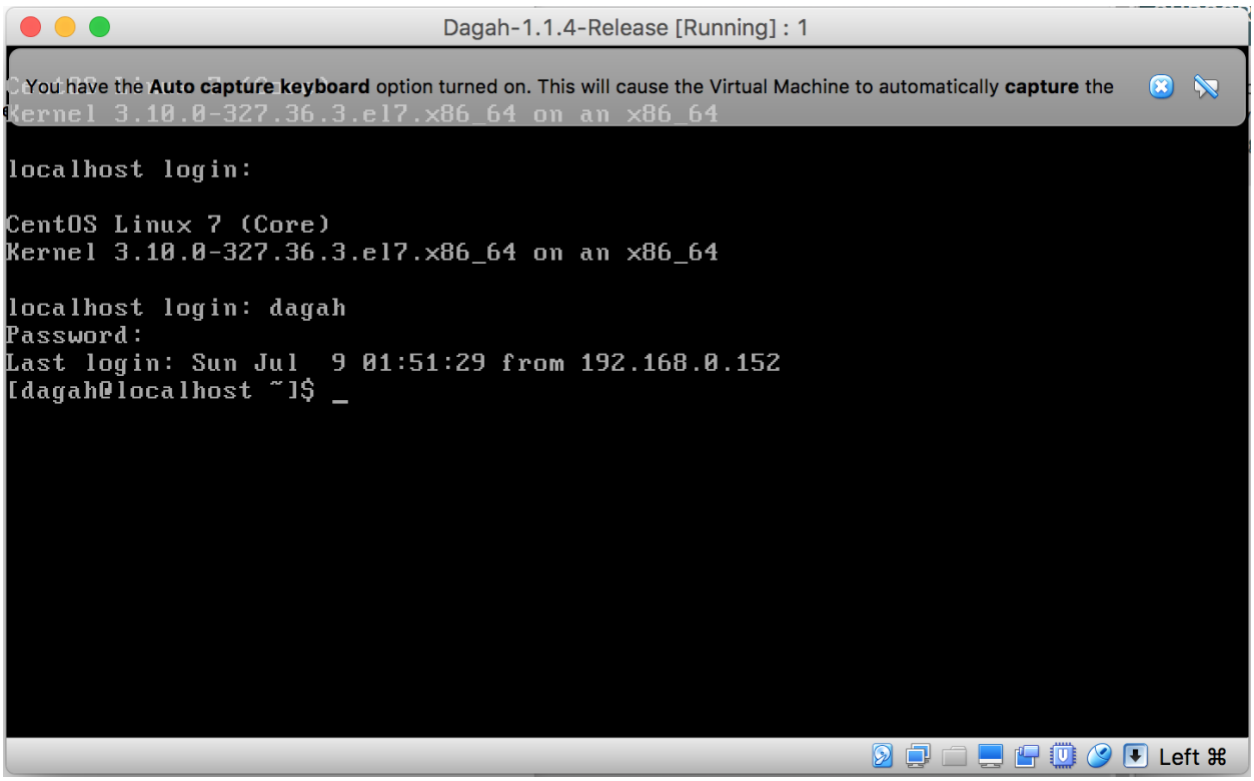
Sometimes the defaults can be wrong.

Oracle, for example, has a nice explanation of the various option
https://blogs.oracle.com/scoter/networking-in-virtualbox-v2

Click on the .ova file you downloaded from the Shevirah website. It will open the import dialog in Virtual Box.



The virtual machine has DHCP enabled. You will need to find the IP address based on your local network. Log into the VM directly with the credentials **dagah:dagah**.



Once logged in run the command ifconfig and find the IP address on the local network. Yours will be different than mine as the Dagah VM uses DHCP.

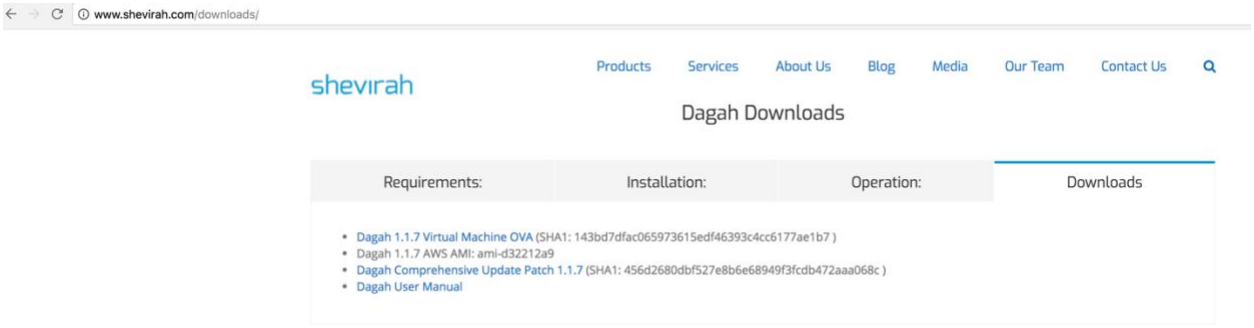
```
Password:
Last login: Sun Jul  9 01:51:29 from 192.168.0.152
[dagah@localhost ~]$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    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
qlen 1000
    link/ether 08:00:27:76:73:6d brd ff:ff:ff:ff:ff:ff
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
qlen 1000
    link/ether 08:00:27:3e:d8:c3 brd ff:ff:ff:ff:ff:ff
    inet 192.168.33.10/24 brd 192.168.33.255 scope global enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe3e:d8c3/64 scope link
       valid_lft forever preferred_lft forever
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
qlen 1000
    link/ether 08:00:27:a5:49:c0 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.189/24 brd 192.168.0.255 scope global dynamic enp0s9
       valid_lft 85455sec preferred_lft 85455sec
[dagah@localhost ~]$ _
```

Use this IP address to log in to the web interface via HTTP. You can also log in via SSH to use the command line interface.

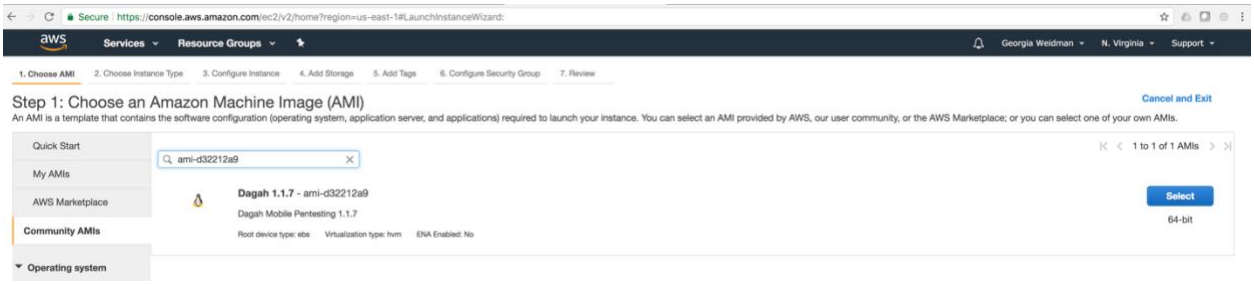
```
Georgias-MBP:Downloads georgiaweidman$ ssh dagah@192.168.0.189
dagah@192.168.0.189's password:
Last login: Tue Jul 18 03:14:14 2017 from 192.168.0.152
[dagah@localhost ~]$
```

AMI

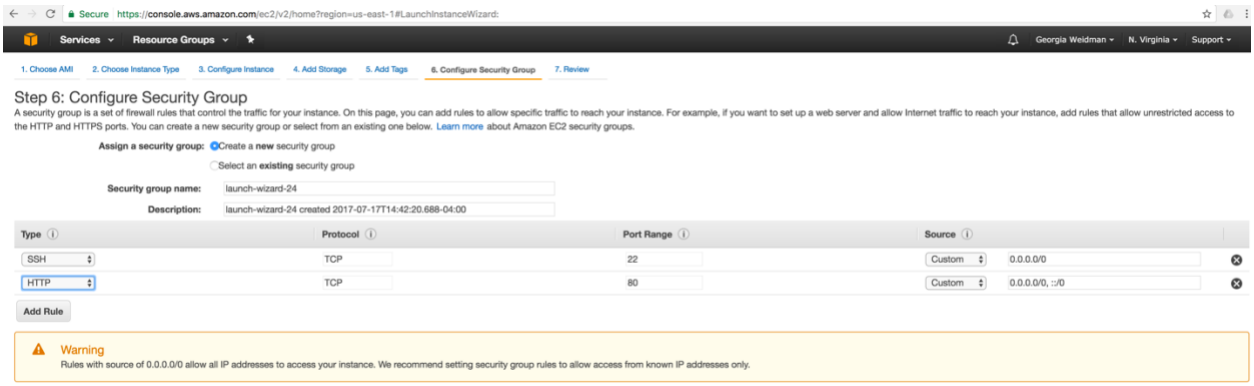
Alternatively, there is an AMI published on Amazon that you can import into your AWS account. The current AMI identifier is published at www.shevirah.com/downloads (ami-d32212a9 at the time of this writing).



In your AWS EC2 account go to Launch Instance and choose “Community AMIs” on the left. Search for the AMI ID from the Shevirah website.



Select the Dagah AMI and launch it as you would any other Amazon instance. For the Security Group, Dagah needs HTTP (80) and SSH (22) open.



The IP address you will use in the Setup section below is the external IP address assigned by EC2.

Setup

The Command Line

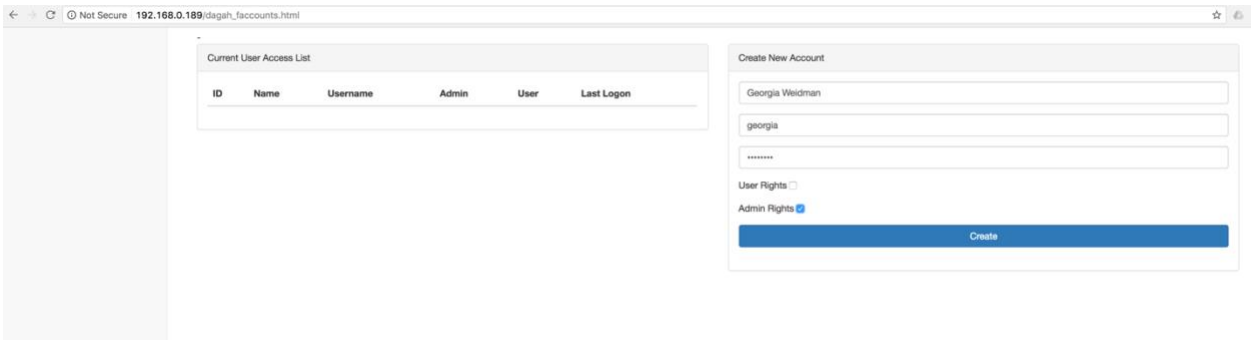
The Dagah command line interface is at /home/dagah/dagah. The GUI is a wrapper for the command line and thus the command line can do everything the GUI can do and then some.

You can run the command line interface with **python dagah2.pyc**

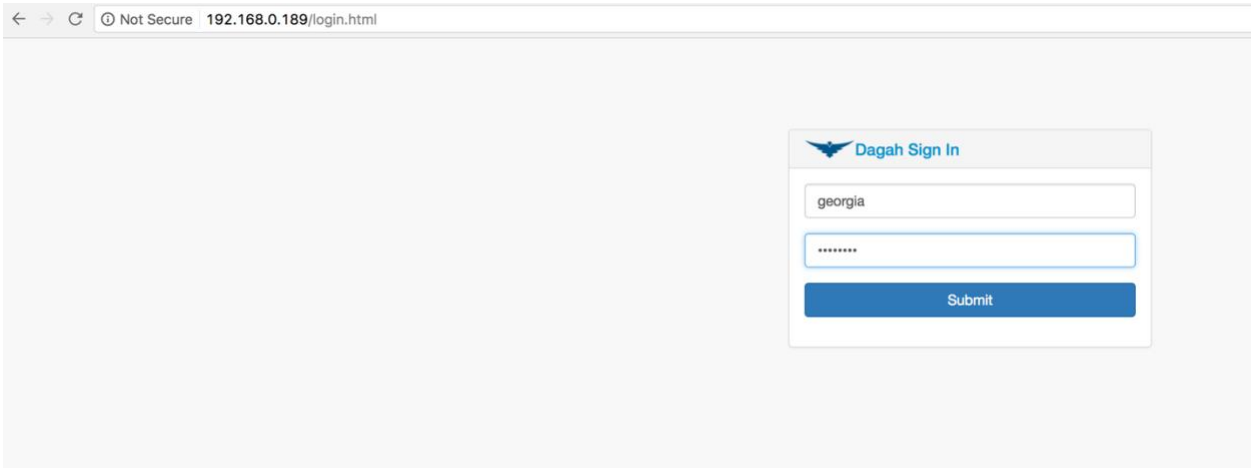
The configuration options for the command line are at /home/dagah/dagah/config. Open the file in your favorite editor to make changes. The only option you are required to set is the IPADDRESS (automatic setting at boot coming soon). We will look at editing some other option as they come up in this walkthrough. For now save the config file with IPADDRESS=192.168.0.189 (change to your IP address). Otherwise you are ready to go.

The GUI

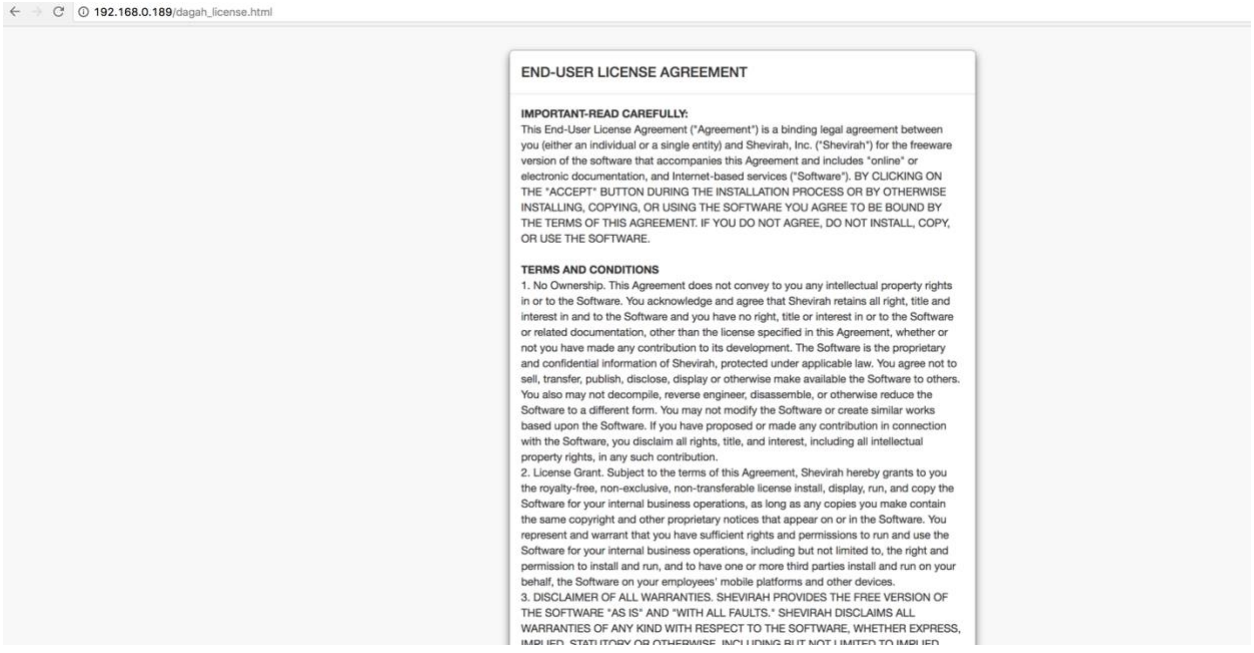
Alternatively, you can use the Dagah GUI. Browse to http://192.168.0.189 (change to your IP address). You will be prompted to create a user account.



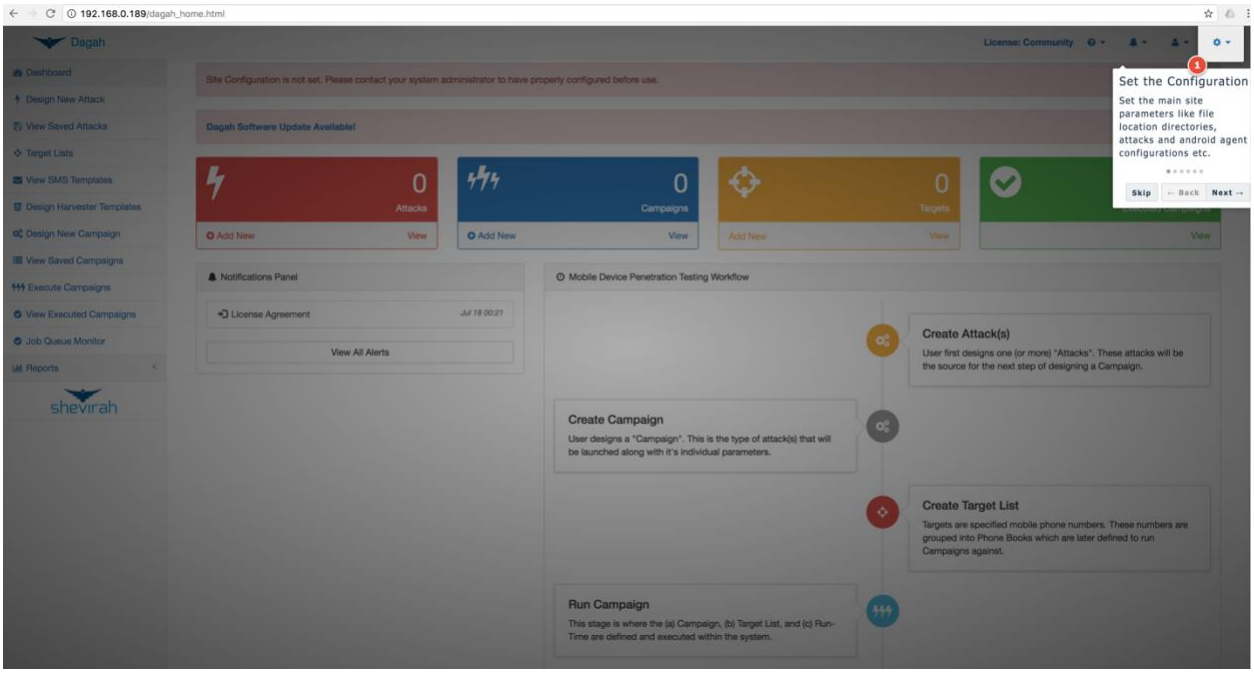
Log in as the user you just created.



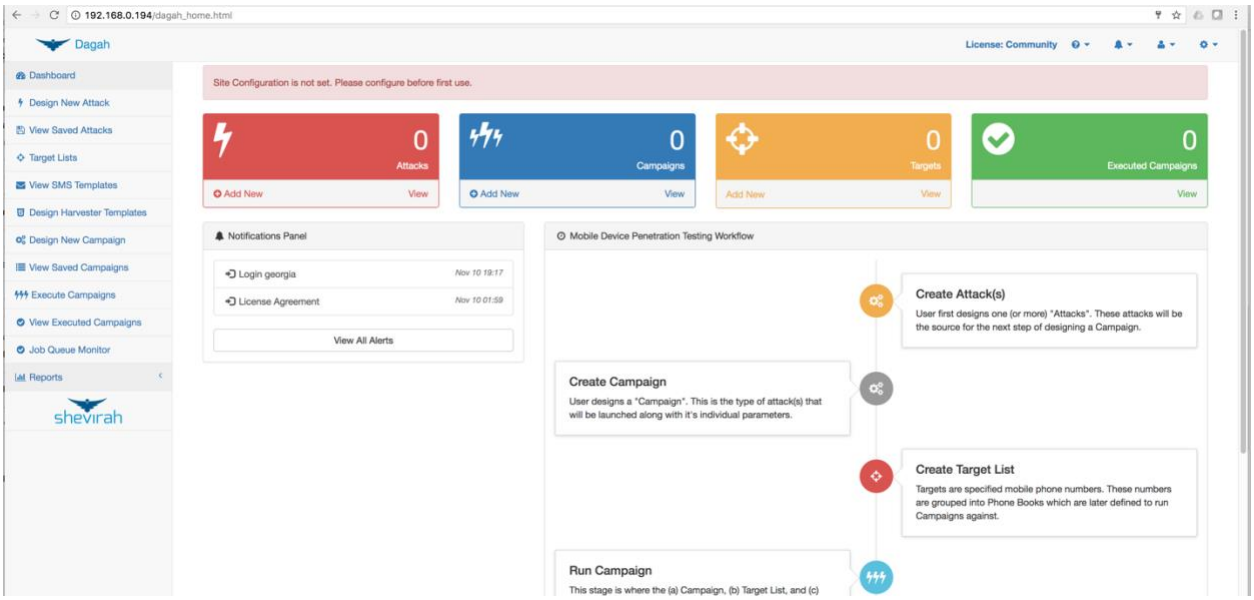
The first time you log in you will be prompted to read and accept a license agreement.



The first time you log in a little box walks you through the steps of using Dagah. You can walk through it with the Next-> button or skip through it with the Skip button if you'd rather read it all here.

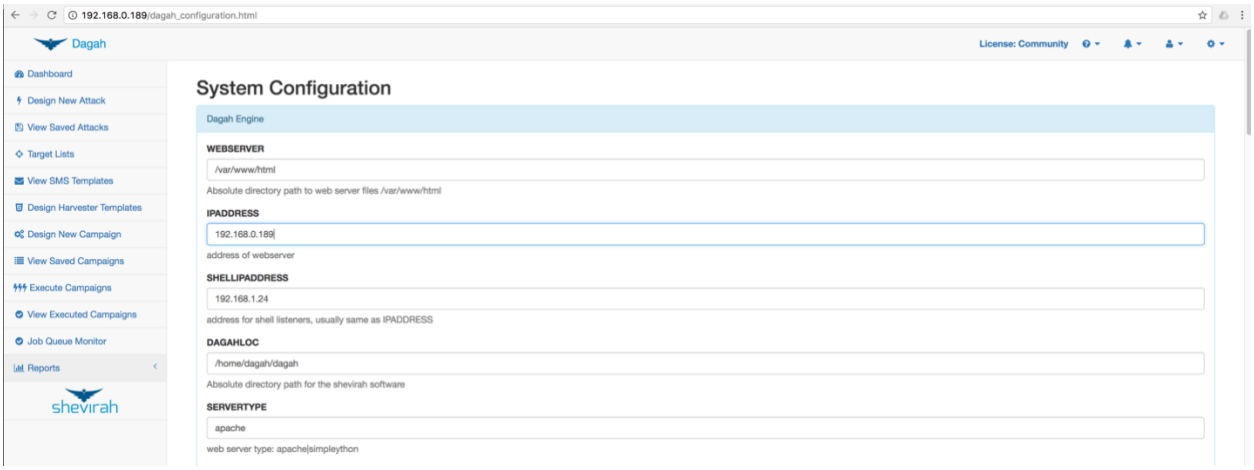


You will see a message that the Site Configurator is not set. Click the gear icon on the top right and select Site Configuration.



The only option you need to set to use Dagah is the IP address (auto configuration of IP coming soon).

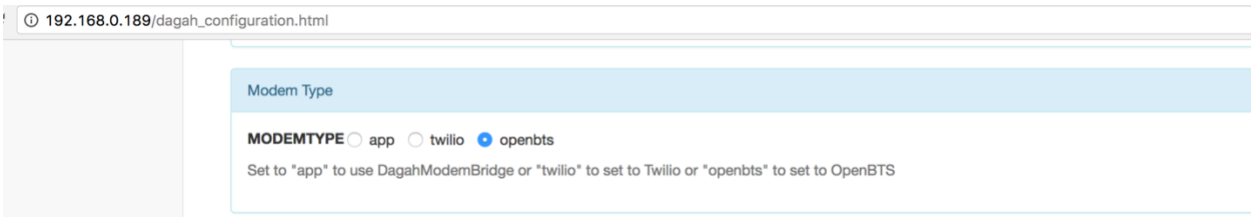
Set the IP Address field to 192.168.0.189 (change to your IP).



Scroll to the bottom of the page, and save the configuration. You will be prompted to log in again. You are now ready to use the Dagah GUI. We will discuss some other configuration options as we walk through some Dagah use cases.

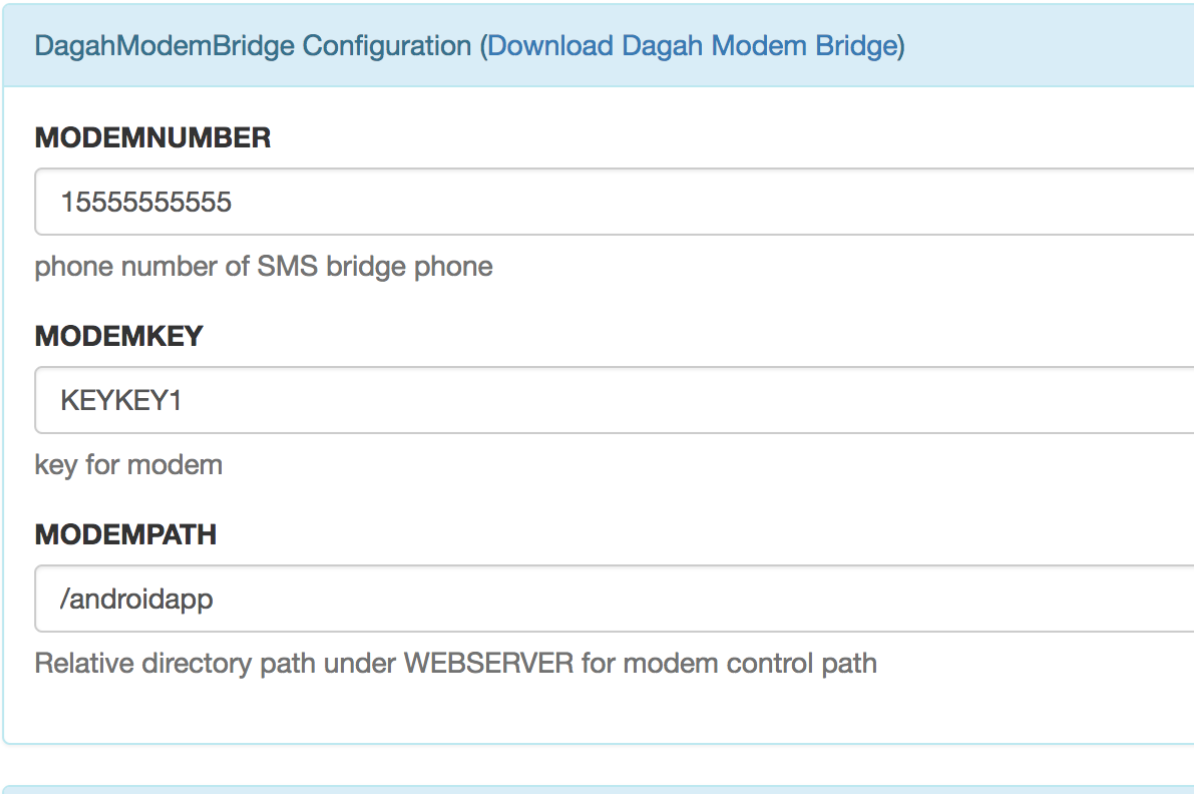
Mobile Modems

Dagah allows you to send attacks over other communication methods besides traditional TCP/IP, as mobile devices and IoT speak a variety of communication methods mobile modem, Zigbee, Bluetooth, etc. as the case may be. If you are going to use a delivery method that requires a mobile modem, attach one through the configuration page.



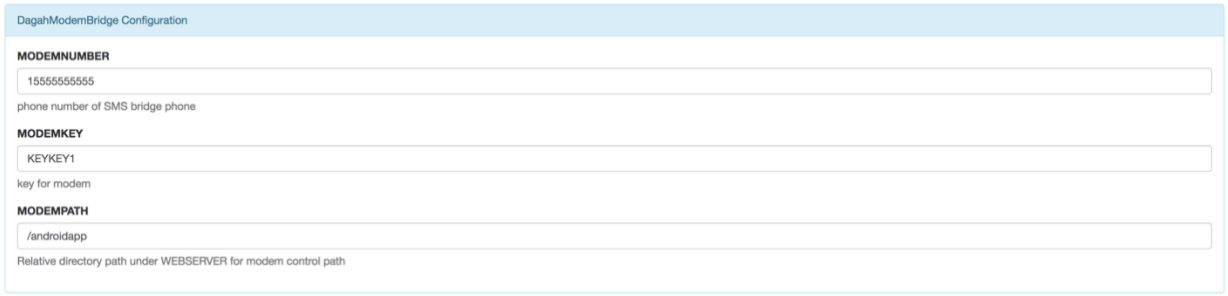
App

The Dagah Mobile Modem App for Android allows you to use the mobile modem in the device to send attacks over SMS, NFC, and Bluetooth. To install the application on your Android device browse to http://192.168.0.189/DagahModemBridge.apk (change IP address to yours). You can also directly download the DagahModemBridge.apk from the link in the DagahModemBridge Configuration section of the Site Configuration page shown below.

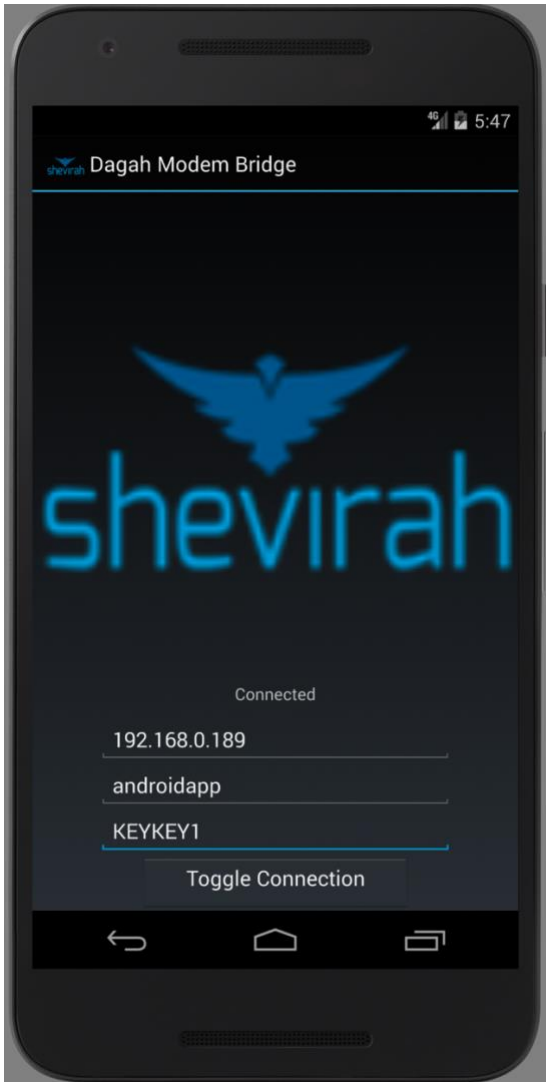


Install the app. You will be presented with a screen the one shown below.

The IP address should be set to the IP of the VM. The path is where it checks in on the webserver and the key is going to be phased out for Virgil security crypto soon. Path and key should match the configuration options on the server configuration page (or config file for command line).



The default path is /androidapp and the default key is KEYKEY1. After filling in the information click the Toggle Connection button. If the app is able to connect to the server you will see Connected on the app.

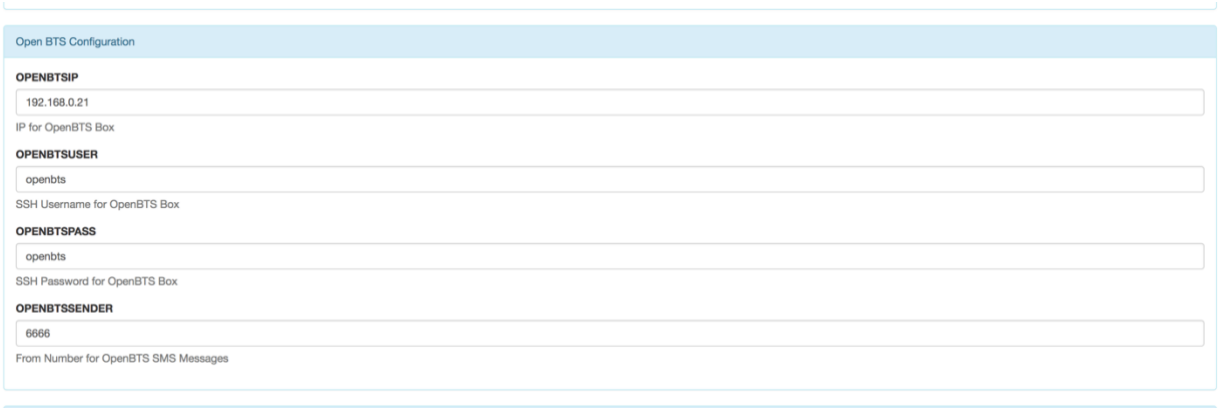


The app will periodically check in with the server for commands. To disconnect click Toggle Connection again and the Connected message will disappear.

OpenBTS

Dagah can attach to an OpenBTS based system to simulate a rogue cell tower. To use OpenBTS toggle the Modem Type radio button in the Site Configurator to openbts.

The OpenBTS system needs to have SSH enabled for Dagah to log in. Fill out the OpenBTS Configuration section on the Configuration page with the IP address, username, and password to log in to the OpenBTS box.



The OPENBTSSENDER option is the sender phone number for SMS messages. OpenBTS allows you to spoof this field.

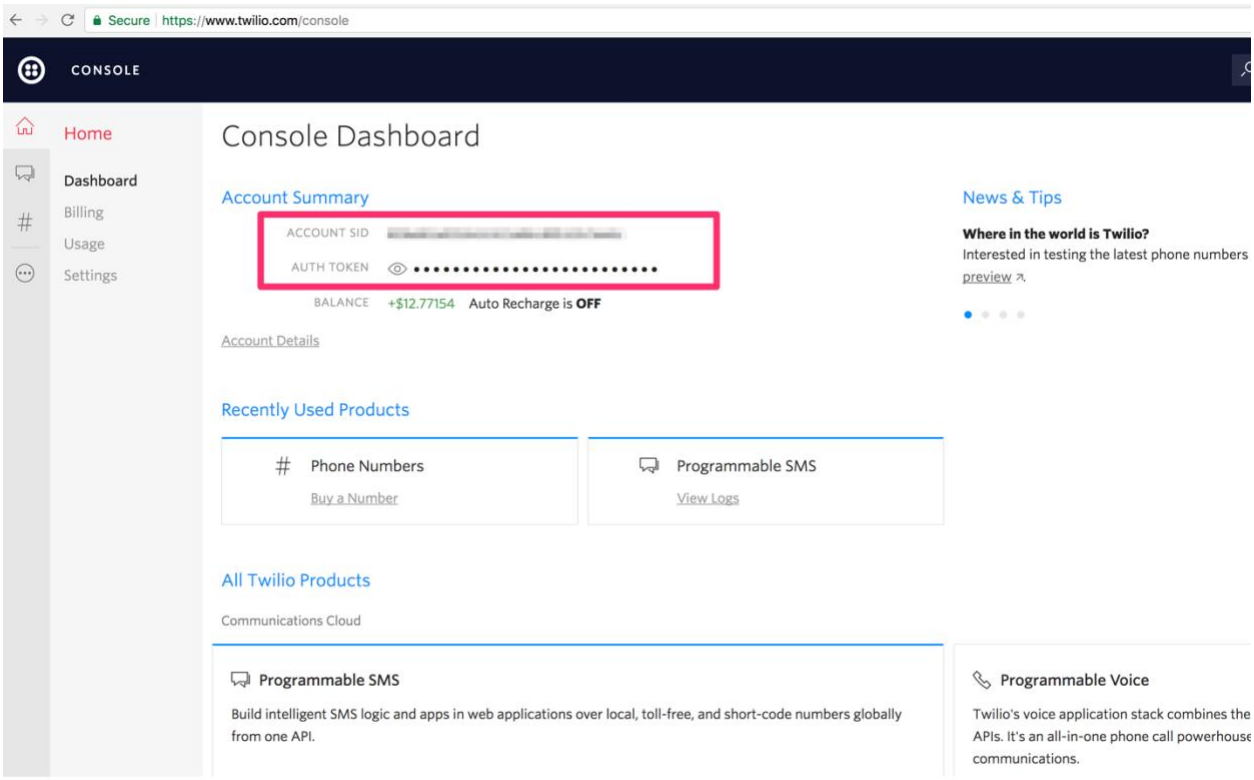
Twilio

Dagah can also hook up to the Twilio service to send text messages. To use Twilio toggle the Modem Type radio button in the Site Configurator to twilio.

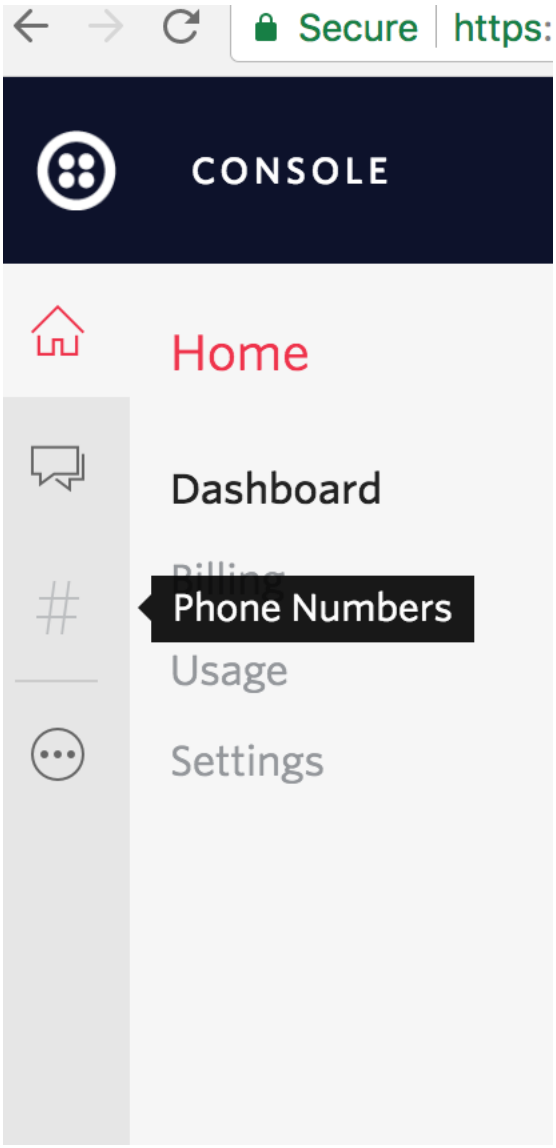
Sign up for a Twilio account at twilio.com.

You will need your Account SID and Auth Token from Twilio. You will also need a Twilio number for the sender number.

Log into your Twilio account. At the Console Dashboard you will see your Account SID (blurred for my account details in the screenshot below). Right beneath it is your Auth Token. Click the eye symbol to make it readable.

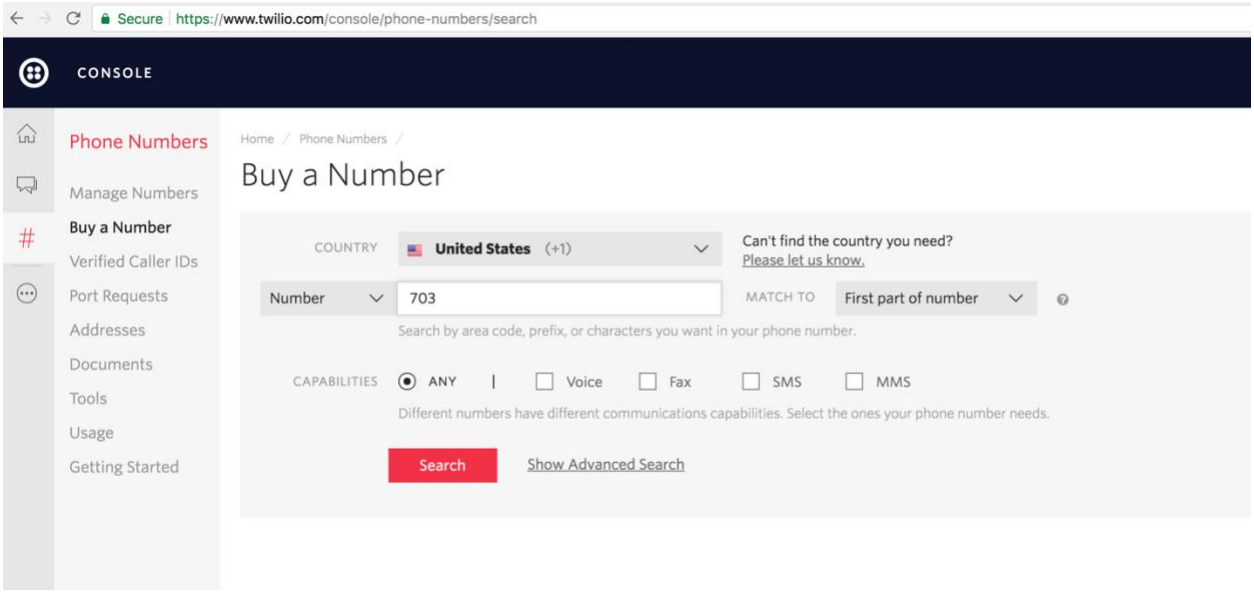


To get a Twilio number to use with Dagah click the # sign on the left hand menu of the Console Dashboard.

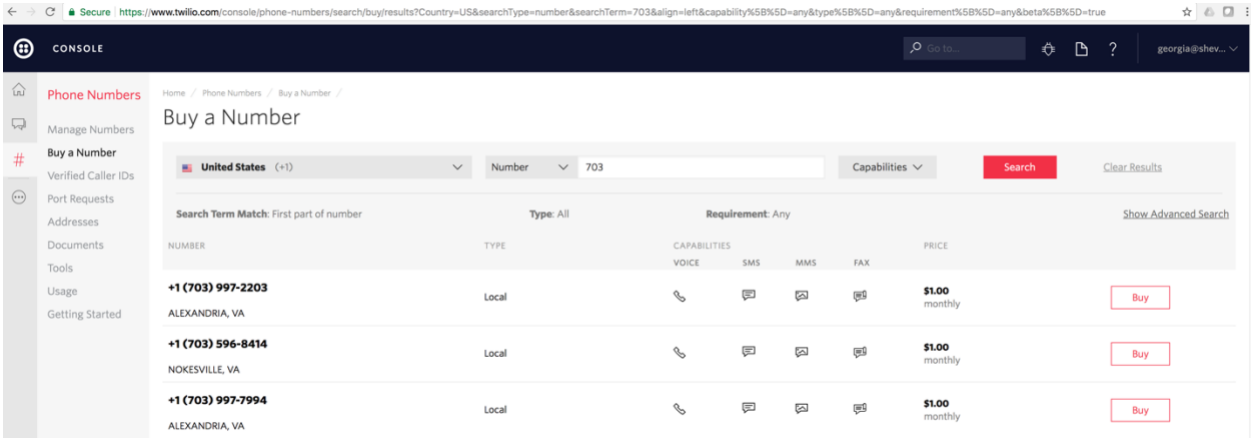


Next click the red + to buy a Twilio number.

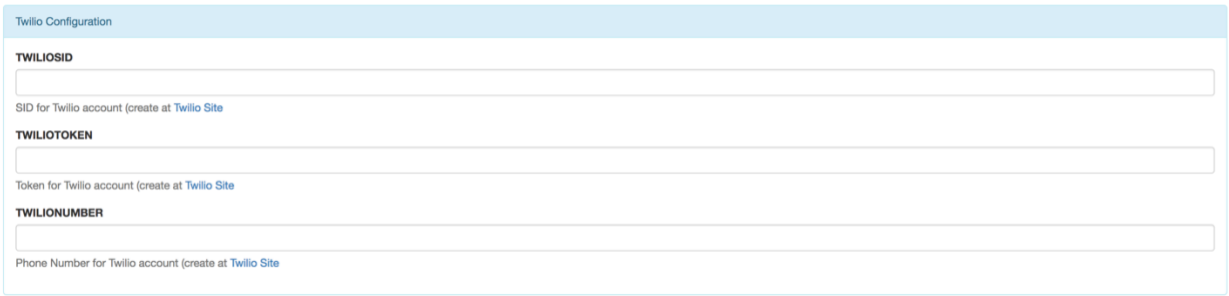
Based on the type of phishing attack you want to perform, the location of your client, etc. you may want to look for a specific area code, number similar to a number known to your target etc. with the Search function.



Choose a number from the resulting list and click Buy on the right hand side.



Enter your new number along with your Sid and Token in the Twilio Configuration section of the Site Configurator.

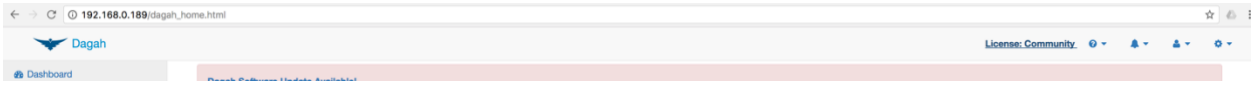


(More modem options coming soon)

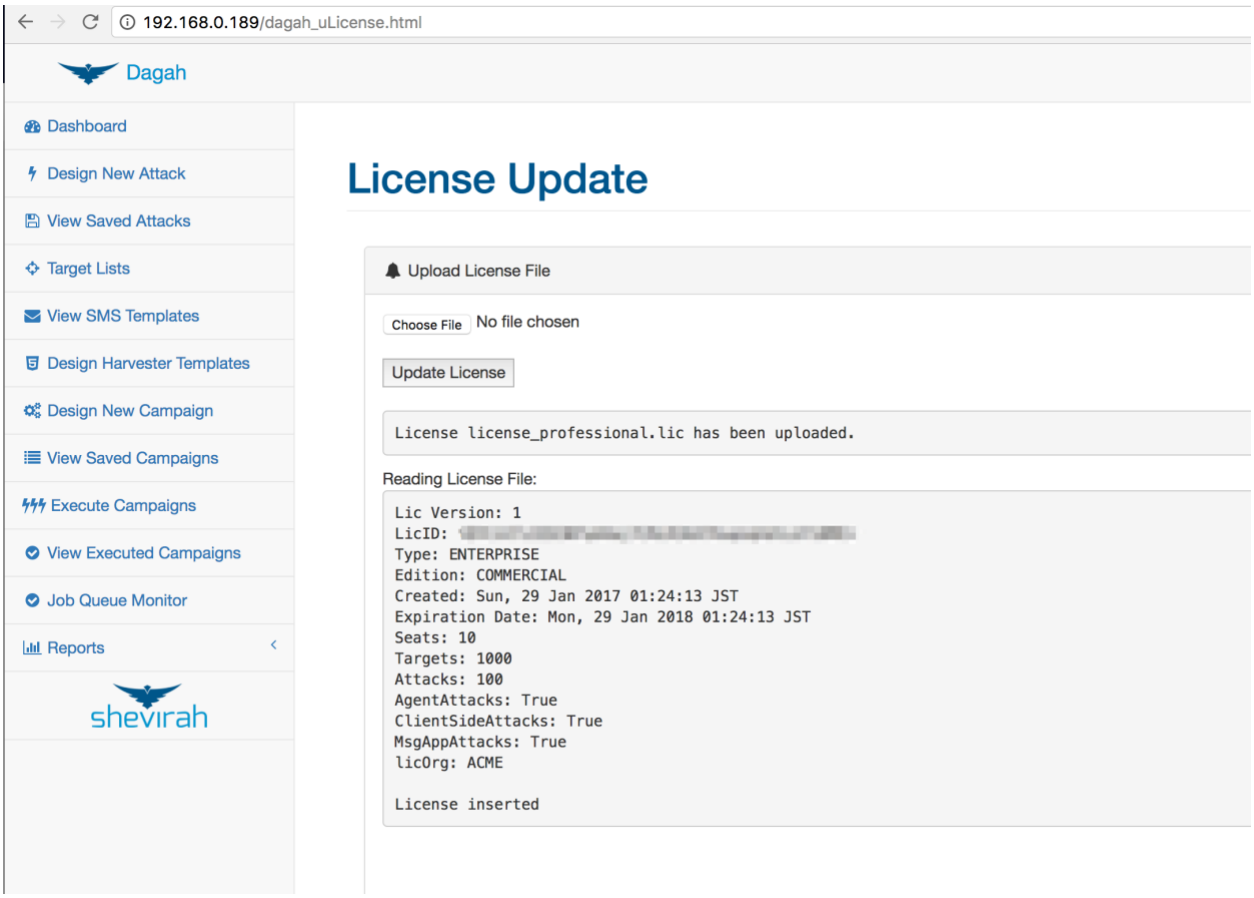
Licensing:

By default, Dagah is the Community version. It is free but limited in capabilities. Good for broke researchers and people who want to get to know the product before buying a professional license.

To insert a different license, click on License:Community at the top right.



Click Choose License and select your license file from the file browser dialog. It should be a .lic file. Click Update License.



When you log in again you should see License:PROFESSIONAL or License:ENTERPRISE at the top right based on the license you have.

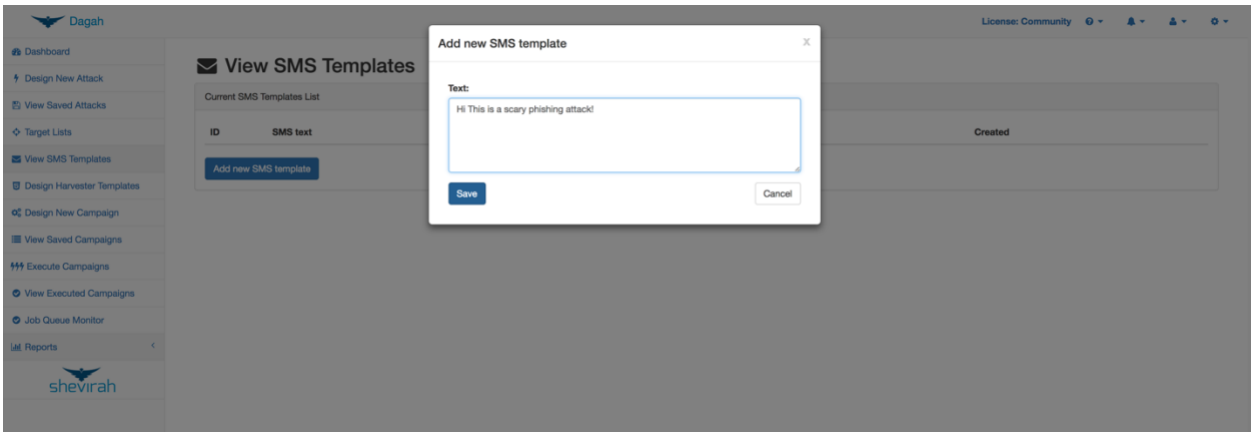
Attack Methods

SMS

Dagah can send text messages using the Dagah Modem App, Twilio, or OpenBTS.

In addition to entering text for the text message, you can save templates of common attacks.

To create an SMS template go to View SMS Templates in the menu on the left side of Dagah. Click Add new SMS template and enter the text.



Near Field Communication (NFC)

Dagah can deliver attacks via NFC tags using the DagahModemBridge App.

QR Code

Dagah can create QR Code representation of attack links. A modem is not necessary. The user can deliver the QR code to targets however they like. For example, on a pilot engagement Shevirah's team made a poster that appeared to be for a discount code for a restaurant in the same building as the target organization. The poster with the QR code prompting employees to download an agent version of the restaurant's mobile app (more on that later) was hung in the company's break room.

Email

Currently supports Gmail.

External

For customers who are only interested in post exploitation and return of investment of currently deployed mobile security controls testing, external delivery will set up the attack but leave delivering post exploitation agents to the user.

Messaging Apps

Dagah can also hook up to messaging apps to deliver attacks. Connecting message apps is currently only supported on the command line (GUI support coming soon).

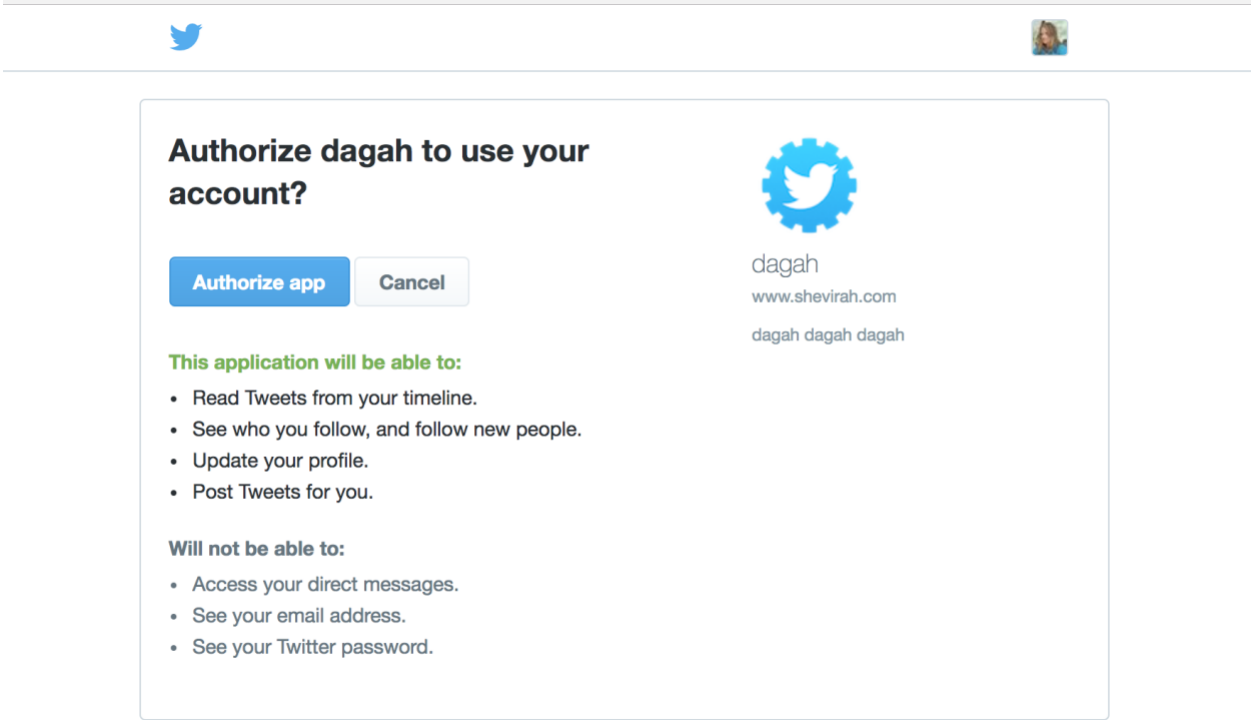
Twitter

To connect to twitter run the command python dagah2.pyc Connect twitter

```
[dagah@localhost dagah]$ python dagah2.pyc Connect twitter
Lic:
Version: 1
LicID: xxxxx
Type: ENTERPRISE
Edition: COMMERCIAL
Created: Sun, 29 Jan 2017 01:24:13 JST
Expiration Date: Mon, 29 Jan 2018 01:24:13 JST
Seats: 10
Targets: 1000
Attacks: 100
AgentAttacks: True
ClientSideAttacks: True
MsgAppAttacks: True
licOrg: ACME

Dagah License Unchanged
Lic: free = 0
Dagah 2 --scripted configurations and XML input output
Go to the following link in your browser:
https://api.twitter.com/oauth/authorize?oauth_token=XXX
Have you authorized me? (y/n)
```

You will be presented with a link to enter into your browser.



You will be asked to authorize the app. When you click the button, you will be redirected to a page that will give you a PIN to enter into the command line.

Enter the PIN at the command line.

```
Have you authorized me? (y/n) y
What is the PIN? XXXXX
Save these to your config file:
TWITTERACCESSTOKEN  = XXXXXXXXXXXXXXXX
TWITTERACCESSTOKENSECRET = XXXXXXXXXXXXXXX
```

Save the TWITTERACCESSTOKEN and TWITTERACCESSTOKENSECRET in the config file in /home/dagah/dagah (of the GUI Site Configuration page).

Whatsapp

Currently down due to changes in the Whatsapp API. Functionality to be restored in the next release.

Attacks:

Basic

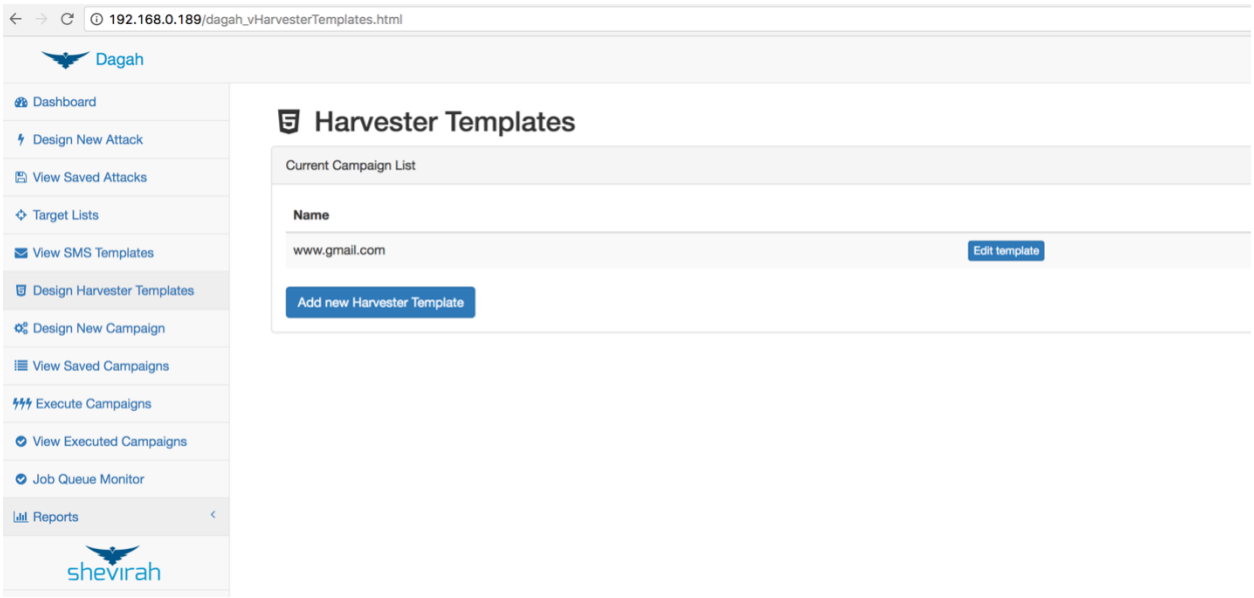
If you just want to see who clicked, scanned, etc. a kind of attack and are not interested in post exploitation at this point, use basic.

Harvester

Harvester is common in email phishing. This is the same concept where the target will be presented with a login page that will harvest the submitted credentials. You can clone a website or make a template.

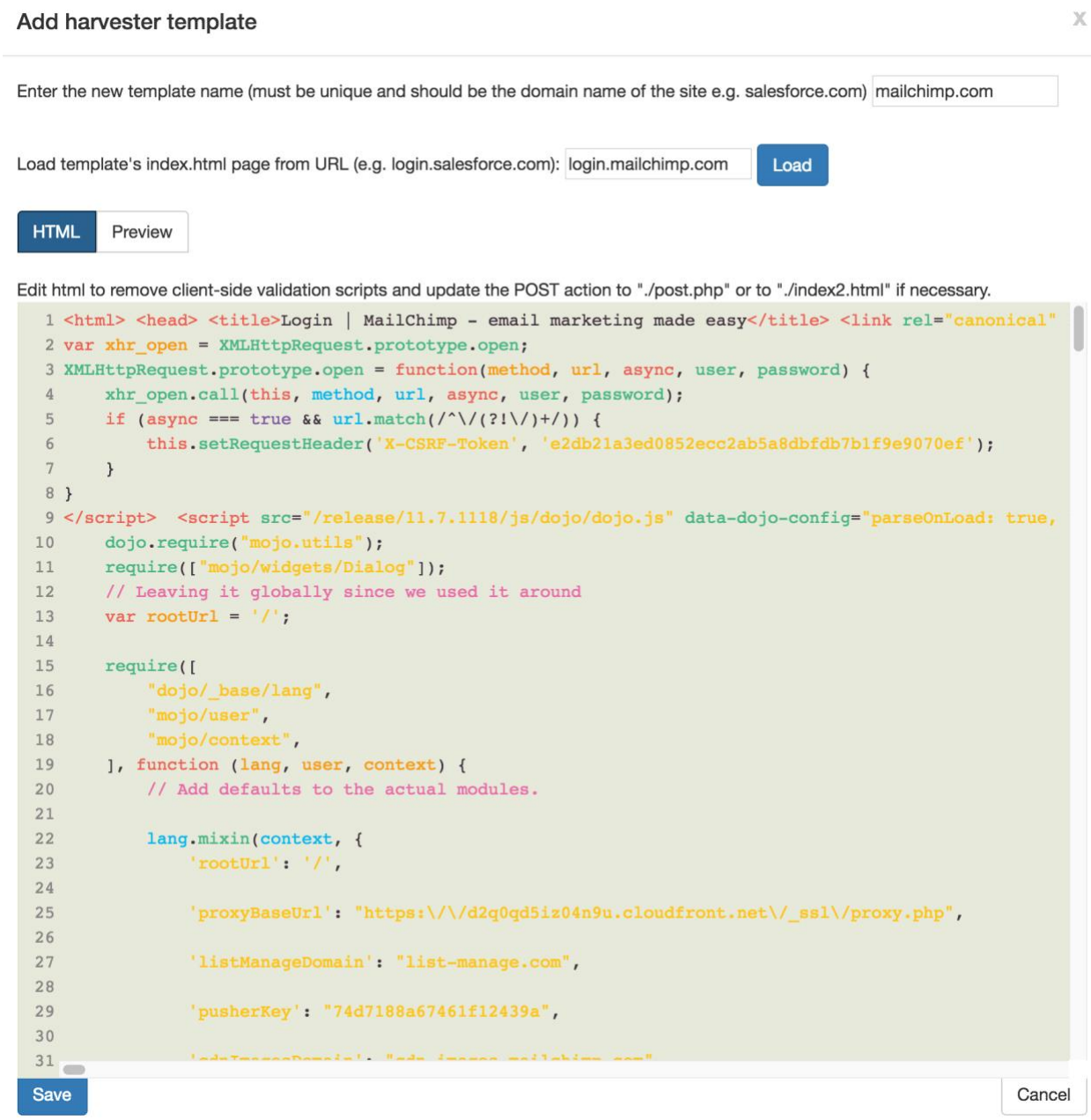
Harvester Templates

To save a harvester template click Design Harvester Templates on the left hand menu of the Dagah GUI.



There is one (more to come) built in template for www.gmail.com made up of separate username and password pages.

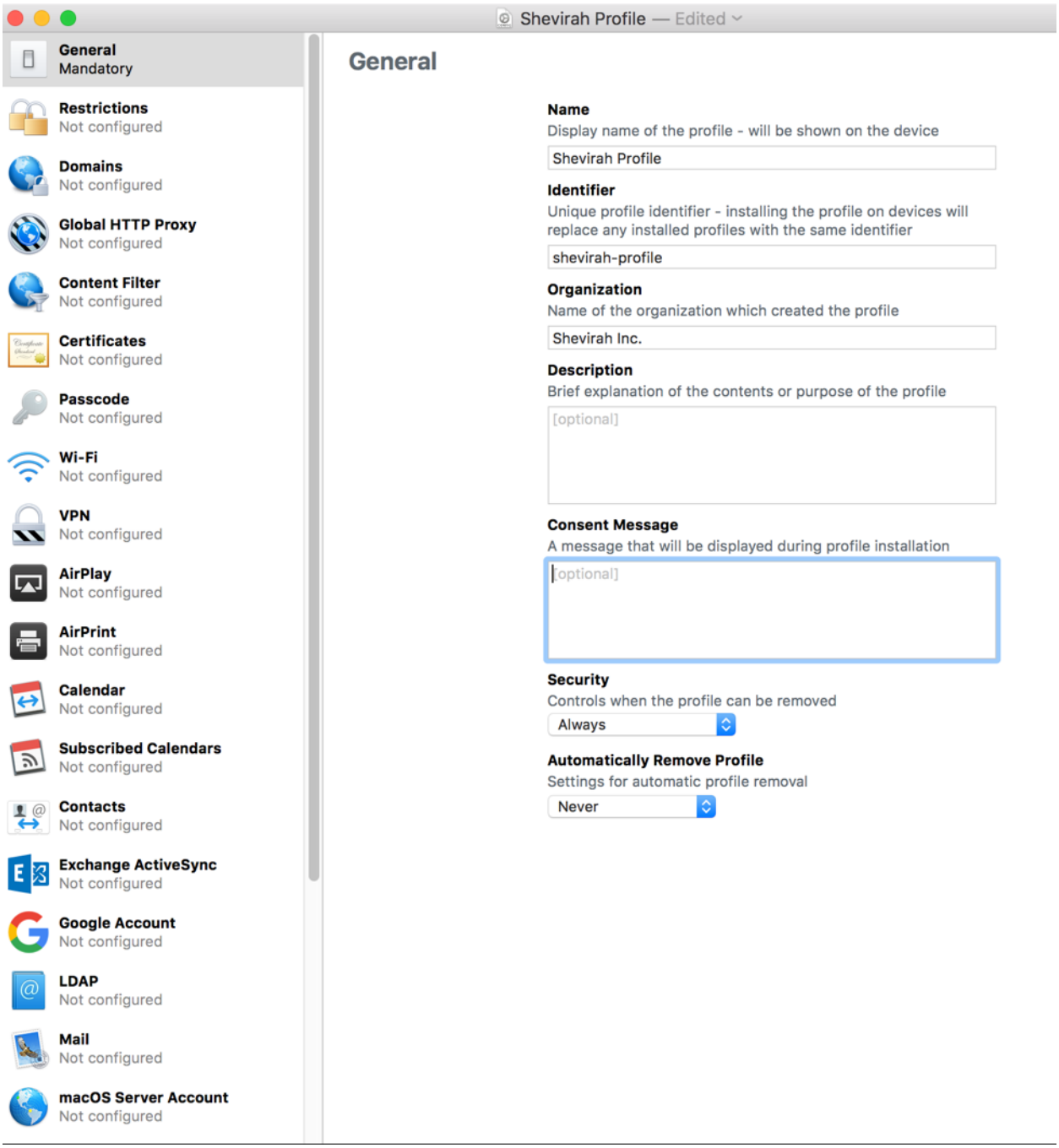
To add a template click Add new Harvester Template. Name the template and the url to clone as a base to edit.



You can make changes to the HTML and Preview it before saving.

Profiles

Dagah will also let you deliver iOS configuration profiles created with Apple Configurator 2.



Profiles can be used to change security settings and install apps outside of the Apple App Store. Shevirah published a whitepaper on attacking iOS with malicious profiles (get paper up and linked here).

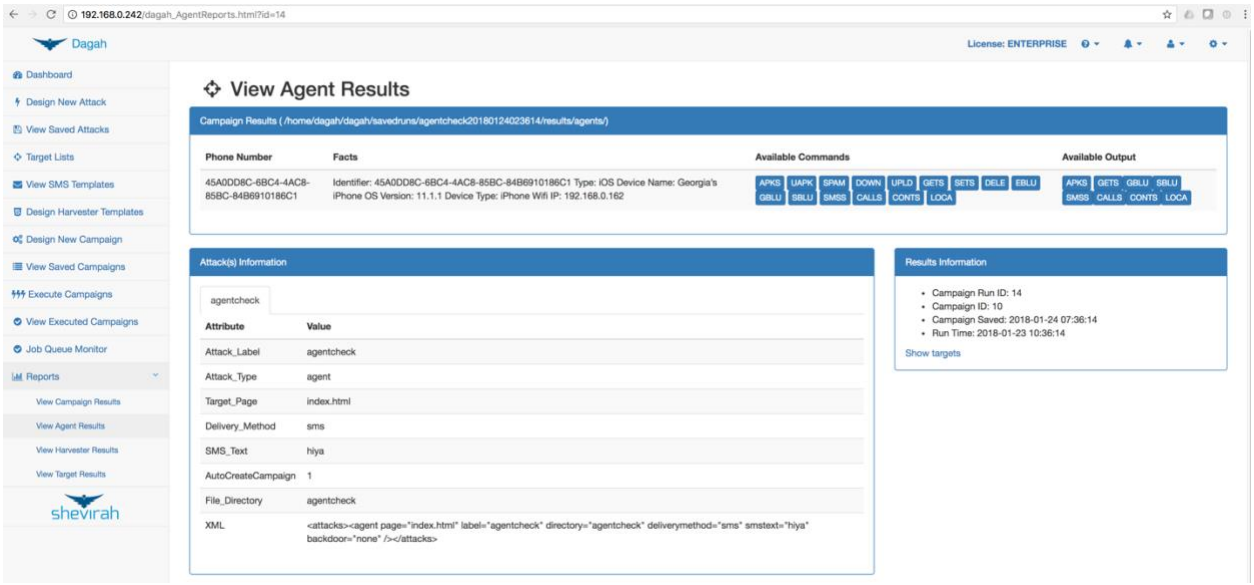
Agents

Agents simulate malicious applications. There are agents for Android and iOS currently. They can be used for post exploitation such as information gathering, pivoting, remote control of the device, and testing return on investment on security controls.

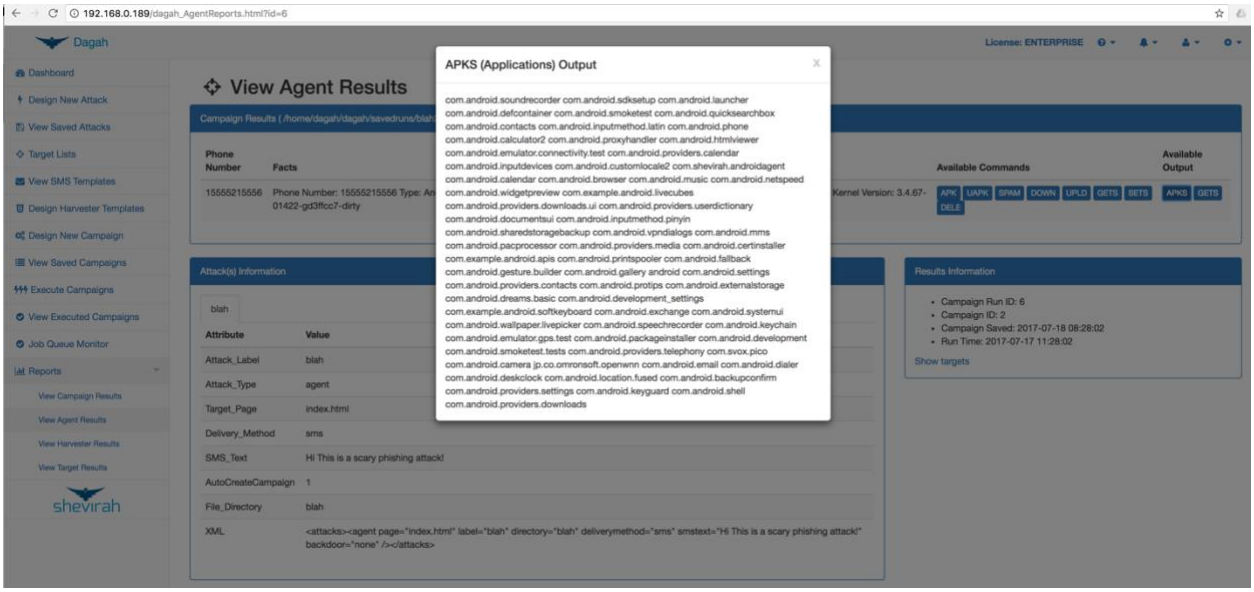
Remote/Client Side vulnerabilities (coming soon)

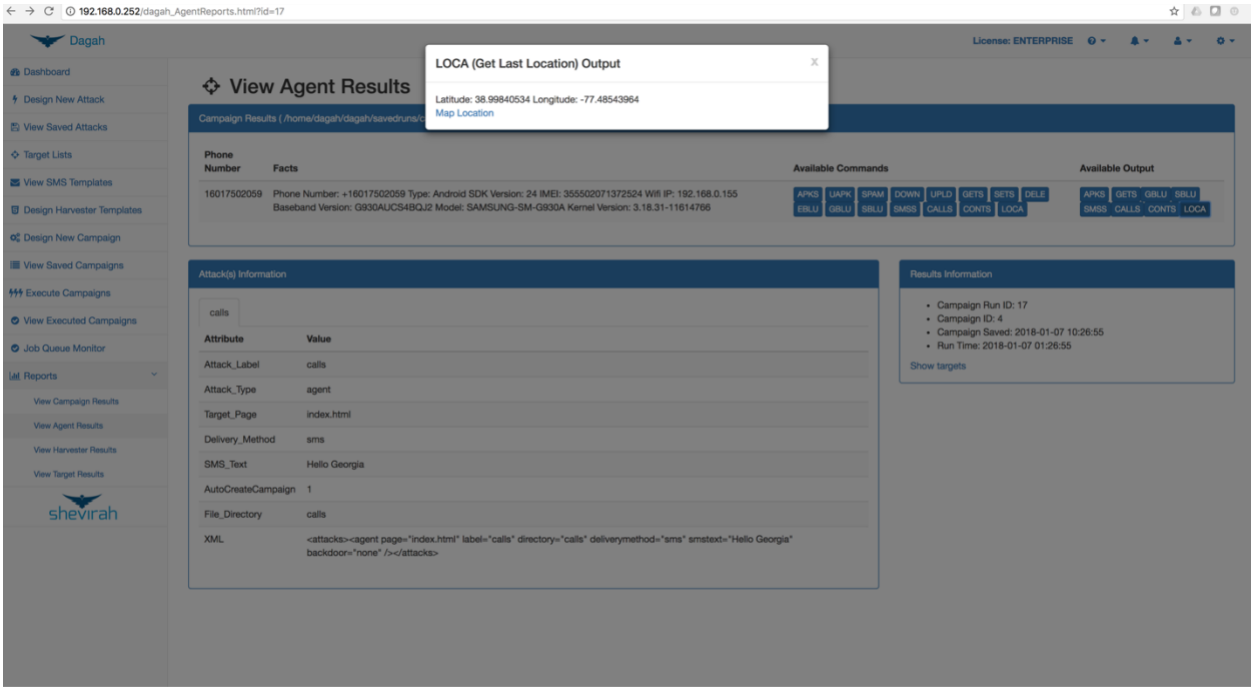
Agent Post Exploitation

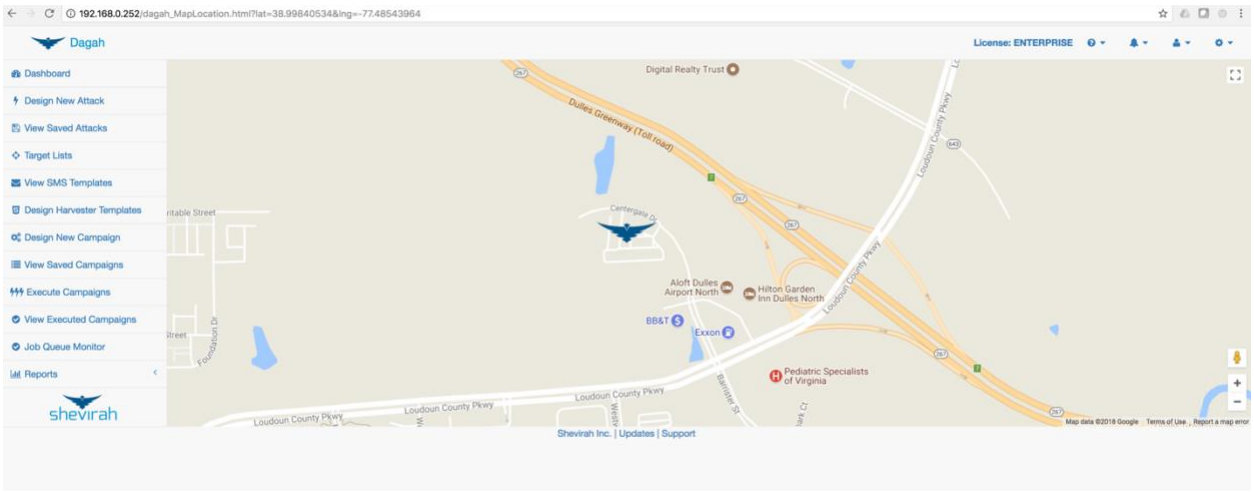
Once an agent is in place it can be used for post exploitation. There is much more of this coming soon and some in the command line I just haven't ported into the GUI yet. After an agent campaign in the GUI under reports go to View Agent Results and choose the Campaign run (more on campaigns and runs in the GUI example later).



Click a button to run the associated post exploitation command. If the command has output, the associated buttons are of the right side.


192.168.0.189/dagah_AgentReports.html?id=6
Dagah
License: ENTERPRISE
Dashboard
Design New Attack
View Saved Attacks
Target Lists
View SMS Templates
Design Harvester Templates
Design New Campaign
View Saved Campaigns
Execute Campaigns
View Executed Campaigns
Job Queue Monitor
Reports
View Campaign Results
View Agent Results
View Harvester Results
View Target Results
shevirah
View Agent Results
Campaign Results (/home/dagah/dagah/savedruns/blah
Phone Number
Facts
Available Commands
Available Output
15555215556
Phone Number: 15555215556 Type: Android SDK Version: 24 IMEI: 000000000000000 Wifi IP: 192.168.0.155 Baseband Version: 01422-gd3ffcc7-dirty
Kernel Version: 3.4.67-
APK UAPK SPAM DOWN UPLD GETS SETS DELE
APKS GETS
APKS (Applications) Output
com.android.soundrecorder com.android.sdksetup com.android.launcher
com.android.defcontainer com.android.smoketest com.android.quicksearchbox
com.android.contacts com.android.inputmethod.latin com.android.phone
com.android.calculator2 com.android.proxyhandler com.android.htmlviewer
com.android.emulator.connectivity.test com.android.providers.calendar
com.android.inputdevices com.android.customlocale2 com.shevirah.androidagent
com.android.calendar com.android.browser com.android.music com.android.netspeed
com.android.widgetpreview com.example.android.livecubes
com.android.providers.downloads.ui com.android.providers.userdictionary
com.android.documentsui com.android.inputmethod.pinyin
com.android.sharedstoragebackup com.android.vpndialogs com.android.mms
com.android.pacprocessor com.android.providers.media com.android.certinstaller
com.example.android.apis com.android.printspooler com.android.fallback
com.android.gesture.builder com.android.gallery android com.android.settings
com.android.providers.contacts com.android.protips com.android.externalstorage
com.android.dreams.basic com.android.development_settings
com.example.android.softkeyboard com.android.exchange com.android.systemui
com.android.wallpaper.livepicker com.android.speechrecorder com.android.keychain
com.android.emulator.gps.test com.android.packageinstaller com.android.development
com.android.smoketest.tests com.android.providers.telephony com.svox.pico
com.android.camera jp.co.omronsoft.openwnn com.android.email com.android.dialer
com.android.deskclock com.android.location.fused com.android.backupconfirm
com.android.providers.settings com.android.keyguard com.android.shell
com.android.providers.downloads
Attack(s) Information
blah
Attribute
Value
Attack_Label
blah
Attack_Type
agent
Target_Page
index.html
Delivery_Method
sms
SMS_Text
Hi This is a scary phishing attack!
AutoCreateCampaign
1
File_Directory
blah
XML
<attacks><agent page="index.html" label="blah" directory="blah" deliverymethod="sms" smstext="Hi This is a scary phishing attack!" backdoor="none" /></attacks>
Results Information
Campaign Run ID: 6
Campaign ID: 2
Campaign Saved: 2017-07-18 08:28:02
Run Time: 2017-07-17 11:28:02
Show targets


192.168.0.252/dagah_AgentReports.html?id=17
Dagah
License: ENTERPRISE
Dashboard
Design New Attack
View Saved Attacks
Target Lists
View SMS Templates
Design Harvester Templates
Design New Campaign
View Saved Campaigns
Execute Campaigns
View Executed Campaigns
Job Queue Monitor
Reports
View Campaign Results
View Agent Results
View Harvester Results
View Target Results
shevirah
View Agent Results
LOCA (Get Last Location) Output
Latitude: 38.99840534 Longitude: -77.48543964
Map Location
Campaign Results (/home/dagah/dagah/savedruns/c
Phone Number
Facts
Available Commands
Available Output
16017502059
Phone Number: +16017502059 Type: Android SDK Version: 24 IMEI: 355502071372524 Wifi IP: 192.168.0.155 Baseband Version: G930AUCS4BQJ2 Model: SAMSUNG-SM-G930A Kernel Version: 3.18.31-11614766
APKS UAPK SPAM DOWN UPLD GETS SETS DELE EBLU GBLU SBLU SMSS CALLS CONTS LOCA
APKS GETS GBLU SBLU SMSS CALLS CONTS LOCA
Attack(s) Information
calls
Attribute
Value
Attack_Label
calls
Attack_Type
agent
Target_Page
index.html
Delivery_Method
sms
SMS_Text
Hello Georgia
AutoCreateCampaign
1
File_Directory
calls
XML
<attacks><agent page="index.html" label="calls" directory="calls" deliverymethod="sms" smstext="Hello Georgia" backdoor="none" /></attacks>
Results Information
Campaign Run ID: 17
Campaign ID: 4
Campaign Saved: 2018-01-07 10:26:55
Run Time: 2018-01-07 01:26:55
Show targets



Using the Command Line

Here's an example of using the command line interface to run an agent attack. To get basic help information just run python dagah2.pyc.

```
[dagah@localhost dagah]$ python dagah2.pyc
Lic:
Version: 1
LicIDXXXX
Type: ENTERPRISE
Edition: COMMERCIAL
Created: Sun, 29 Jan 2017 01:24:13 JST
Expiration Date: Mon, 29 Jan 2018 01:24:13 JST
Seats: 10
Targets: 1000
Attacks: 100
AgentAttacks: True
ClientSideAttacks: True
MsgAppAttacks: True
licOrg: ACME

Dagah License Unchanged
Lic: free = 0
Dagah 2 --scripted configurations and XML input output
Options:
        Campaign <campaignconfigfile>.xml               (create a saved campaign of phishing attacks)
             list                               (list saved campaign details)
        Run     <numbersfile>.txt <campaign> [savefilename]  (run saved campaign against target phone
numbers)
        Report   <savedrunfile>                 (get results of last run)
        Delete Campaign|Run <campaignname|runlabel          (delete campaign/run)
[dagah@localhost dagah]$
```

The Dagah command line runs campaigns made of XML representing the different attacks. In the professional version of Dagah a campaign can include multiple attacks. Some examples of the XML for different campaigns are shown below.

Basic example:

```
<campaign name="testbasicopenbts">
<attacks>
<basic deliverymethod="sms" directory="/testbasicqrcattack" label="testbasicqrcattack"
page="/index.html" webpagetext="test"/>
</attacks>
</campaign>
```

Harvester example:

```
<!-- Sample Campaign for a Phishing Harvester Attack
  Campaign Arguments: name: A unique label for the campaign,
    alphanumeric without spaces -->
<campaign name="testharvester">
<!-- Attack Arguments: none -->
<attacks>
<!-- Harvester Arguments:
   label: A unique label for the attack, alphanumeric without spaces
   deliverymethod: [sms | nfc | qrc]
   smstext: If sms, text to appear in message, alphanumeric, symbols, spaces
   directory: website directory for hosting destination page (set to same as label)
   page: website page destination, must be .html or .php
   clonepage: The URL of web login page to clone
      Experiment with this outside of dagah to get a good page.
      Avoid including cgi-arguments in the URL (?something=something) -->

       <harvester label="testharvesterlabel" deliverymethod="sms" smstext="This is only a test"
directory="testharvesterlabel" page="index.html" clonepage="template"
template="www.gmail.com"></harvester>
</attacks>
</campaign>
```

Agent example:

```
<!-- Sample Campaign for a Agent Phishing Attack
  Campaign Arguments: name: A unique label for the campaign,
    alphanumeric without spaces -->
<campaign name="testagent">
<!-- Attack Arguments: none -->
```

```
<attacks>
<!-- Agent Arguments:
  label: A unique label for the attack, alphanumeric without spaces
  deliverymethod: [sms | nfc | qrc]
  smstext: If sms, text to appear in message, alphanumeric, symbols, spaces
  directory: website directory for hosting destination page (set to same as label)
  backdoorapp: path and file name to apk to be backdoored with the agent
    or "none" for no backdoor.  Only for Android targets -->
     <agent label="testagentlabel" directory="testagentlabel" deliverymethod="external"
smstext="Install this app:" backdoorapp="none"></agent>
</attacks>
</campaign>
```

Profile example:

```
<campaign name="prof1"><attacks><profile page="index.html" label="prof1" directory="prof1"
deliverymethod="external" smstext="Install this profile"
profile_file="/home/dagah/dagah/profiles/dagahprofile.mobileprovision" /></attacks></campaign>
```

Bluetooth (and multiple attacks in a campaign) example:

```
<campaign name="testbluetooth">
<attacks>
<basic deliverymethod="qrc" directory="/testbasicqrcattack" label="testbasicqrcattack"
page="/index.html" webpagetext="test" />
<bluetooth/>
</attacks>
</campaign>
```

Email

```
<campaign name="testbasicemail"><attacks><basic deliverymethod="email" emailsubject="hello georgia"
directory="/testbasicemail" label="testbasicemail" page="/index.html" webpagetext="test"  emailtext="Hi
Everyone! Happy Australia Day! Cheers, Julian"/></attacks></campaign>
```

Twitter

```
<campaign name="testtwitter">
<attacks>
        <basic messagetext="This is only a test" label="testtwitterlabel" directory="testtwitterlabel"
deliverymethod="messagingapp" app ="twitter" webpagetext="This is a test"/>
</attacks>
</campaign>
```

Setup a campaign with the Campaign command.

```
[dagah@localhost dagah]$ python dagah2.pyc Campaign testagent.xml
Lic:
Version: 1
LicID: XXXXXX
Type: ENTERPRISE
Edition: COMMERCIAL
Created: Sun, 29 Jan 2017 01:24:13 JST
Expiration Date: Mon, 29 Jan 2018 01:24:13 JST
Seats: 10
Targets: 1000
Attacks: 100
AgentAttacks: True
ClientSideAttacks: True
MsgAppAttacks: True
licOrg: ACME

Dagah License Unchanged
Lic: free = 0
Dagah 2 --scripted configurations and XML input output
Creating Stored Campaign
```

To run the command you need target numbers, Twitter user names, etc.

```
[dagah@localhost dagah]$ cat numbers.txt
15555215556
```

```
[dagah@localhost dagah]$ cat twitters.txt
@georgiaweidman
@shevirahsec
@bulbsecurity
```

Run the campaign as shown below.

```
[dagah@localhost dagah]$ python dagah2.pyc Run numbers.txt testagent testagentsaved
Lic:
Version: 1
LicID: t8YC1CFzIED30fq4Xaj7CRLR3kX7HuqnqhdtLU718ME=
Type: ENTERPRISE
Edition: COMMERCIAL
Created: Sun, 29 Jan 2017 01:24:13 JST
Expiration Date: Mon, 29 Jan 2018 01:24:13 JST
Seats: 10
Targets: 1000
Attacks: 100
AgentAttacks: True
```

```
ClientSideAttacks: True
MsgAppAttacks: True
licOrg: ACME

Dagah License Unchanged
Lic: free = 0
Dagah 2 --scripted configurations and XML input output
Running Stored Campaign
Building Agent

BUILD SUCCESSFUL
```

When a user installs the agent, it checks in with the sever with basic information about itself. Results are stored in the savedruns folder.

```
[dagah@localhost 15555215556]$ pwd
/home/dagah/dagah/savedruns/testagentsaved/results/agents/15555215556
[dagah@localhost 15555215556]$ cat facts.txt
Phone Number: 15555215556
Type: Android
SDK Version: 19
IMEI: 000000000000000
Wifi IP: 10.0.2.15
Baseband Version:
Model: sdk
Kernel Version: 3.4.67-01422-gd3ffcc7-dirty
```

You can run post exploitation commands on a live agent.

```
[dagah@localhost dagah]$ python dagah2.pyc Agent command testagentsaved 15555215556 APKS
Lic:
Version: 1
LicID: XXXX
Type: ENTERPRISE
Edition: COMMERCIAL
Created: Sun, 29 Jan 2017 01:24:13 JST
Expiration Date: Mon, 29 Jan 2018 01:24:13 JST
Seats: 10
Targets: 1000
Attacks: 100
AgentAttacks: True
ClientSideAttacks: True
MsgAppAttacks: True
licOrg: ACME
```

```
Dagah License Unchanged
Lic: free = 0
Dagah 2 --scripted configurations and XML input output
```

Results of the post exploitation are again saved in the savedruns folder for the campaign.

```
[dagah@localhost dagah]$ cd savedruns/testagentsaved/results/agents/15555215556/
[dagah@localhost 15555215556]$ ls
APKS.txt  facts.txt
[dagah@localhost 15555215556]$ cat APKS.txt
com.android.soundrecorder
com.android.sdksetup
com.android.launcher
com.android.defcontainer
com.android.smoketest
com.android.quicksearchbox
com.android.contacts
com.android.inputmethod.latin
com.android.phone
com.android.calculator2
com.android.proxyhandler
com.android.htmlviewer
…
```

Twitter example:

```
[dagah@localhost dagah]$ python dagah2.pyc Run twitters.txt testtwitter testtwit
Lic:
Version: 1
LicID: t8YC1CFzIED30fq4Xaj7CRLR3kX7HuqnqhdtLU718ME=
Type: ENTERPRISE
Edition: COMMERCIAL
Created: Sun, 29 Jan 2017 01:24:13 JST
Expiration Date: Mon, 29 Jan 2018 01:24:13 JST
Seats: 10
Targets: 1000
Attacks: 100
AgentAttacks: True
ClientSideAttacks: True
MsgAppAttacks: True
licOrg: ACME

Dagah License Unchanged
Lic: free = 0
Dagah 2 --scripted configurations and XML input output
Running Stored Campaign
child.tag attacks
```

```
2 child.tag basic
Lic.n_Attacks() 100
[dagah@localhost dagah]$
```

Since I connected Dagah to my Twitter account (discussed previously) it sends the attack to the twitter accounts specified in twitters.txt.

If a user clicks they are recorded in the results.

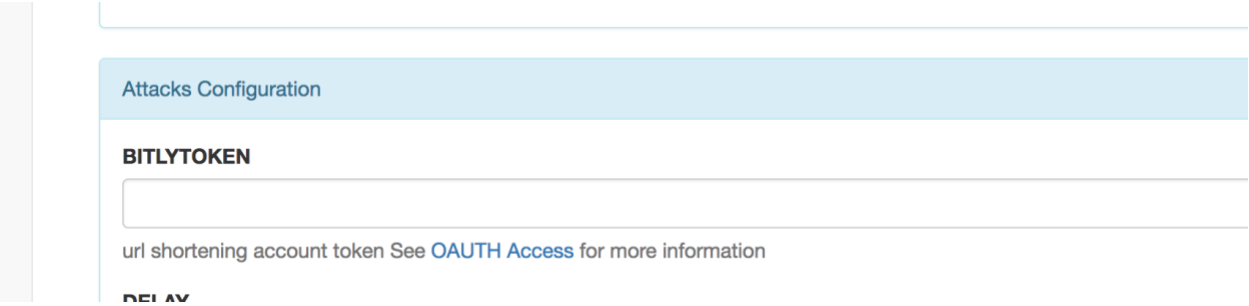
```
[dagah@localhost dagah]$ cd savedruns/testtwit/
[dagah@localhost testtwit]$ ls
campaignconfig.xml  numbers.txt  results
[dagah@localhost testtwit]$ cd results/
[dagah@localhost results]$ ls
testtwitterlabelbasic1-results
[dagah@localhost results]$ cat testtwitterlabelbasic1-results
[10/Nov/2017:15:03:49] @georgiaweidman Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/61.0.3163.100 Safari/537.36
[dagah@localhost results]$
```

(direct message support coming soon).

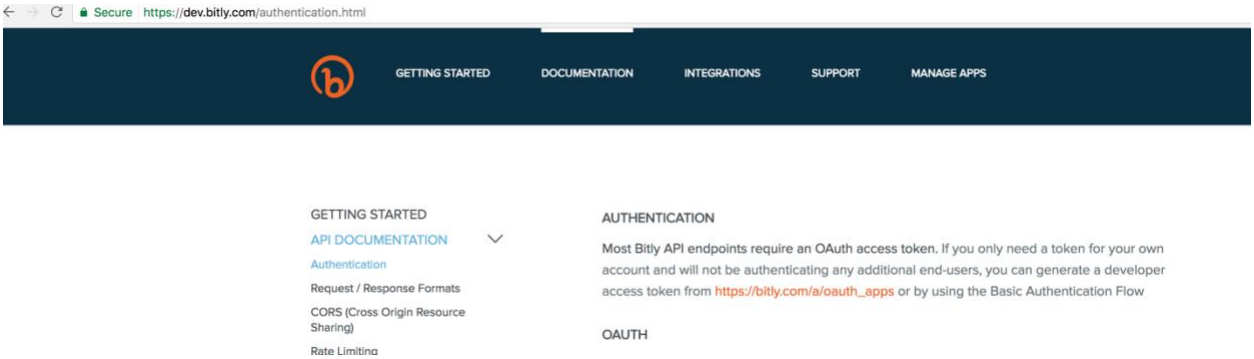
Using the GUI Example

Here's an example of using the GUI to run a harvester attack.

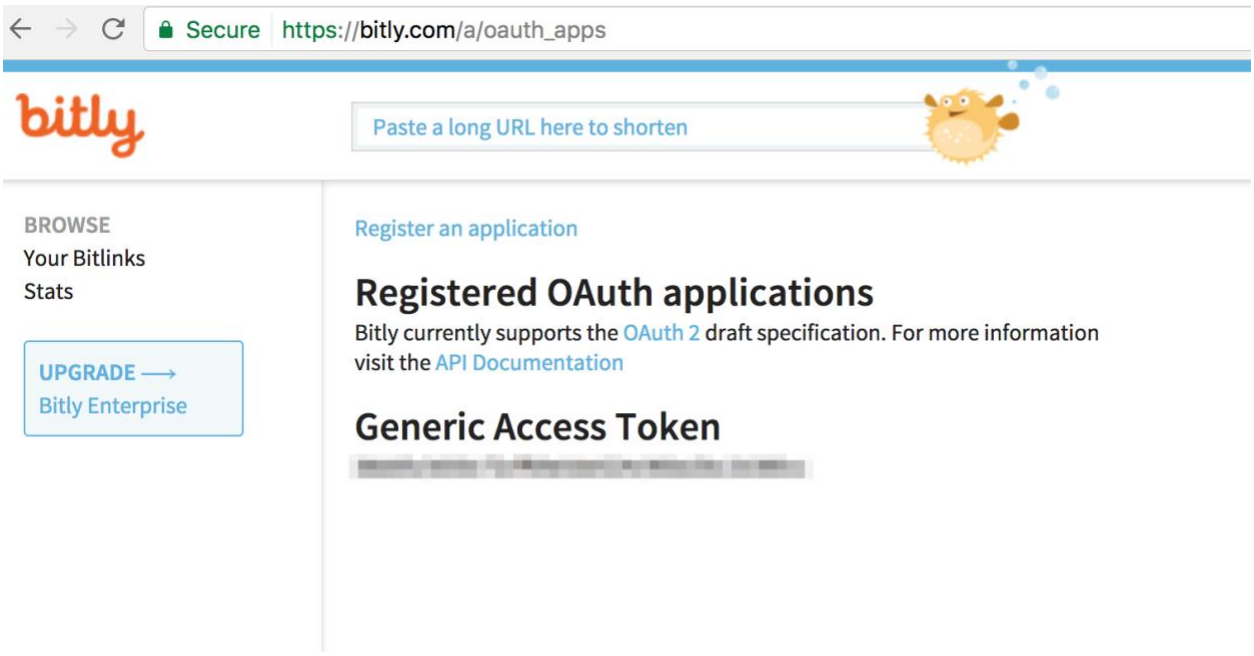
Let's start by setting a configuration option. Dagah can use Bit.ly to obfuscate URLs.



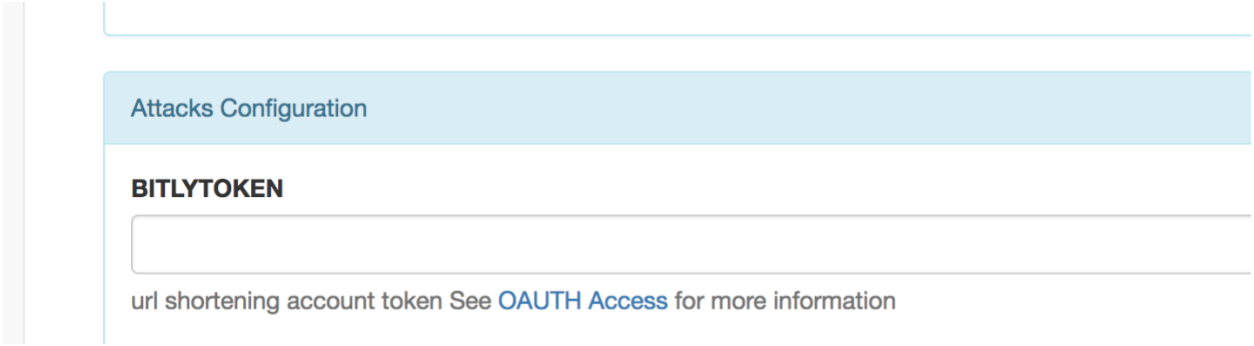
In the Site Configurator under BITLYTOKEN there's a link to information on how to get the token.



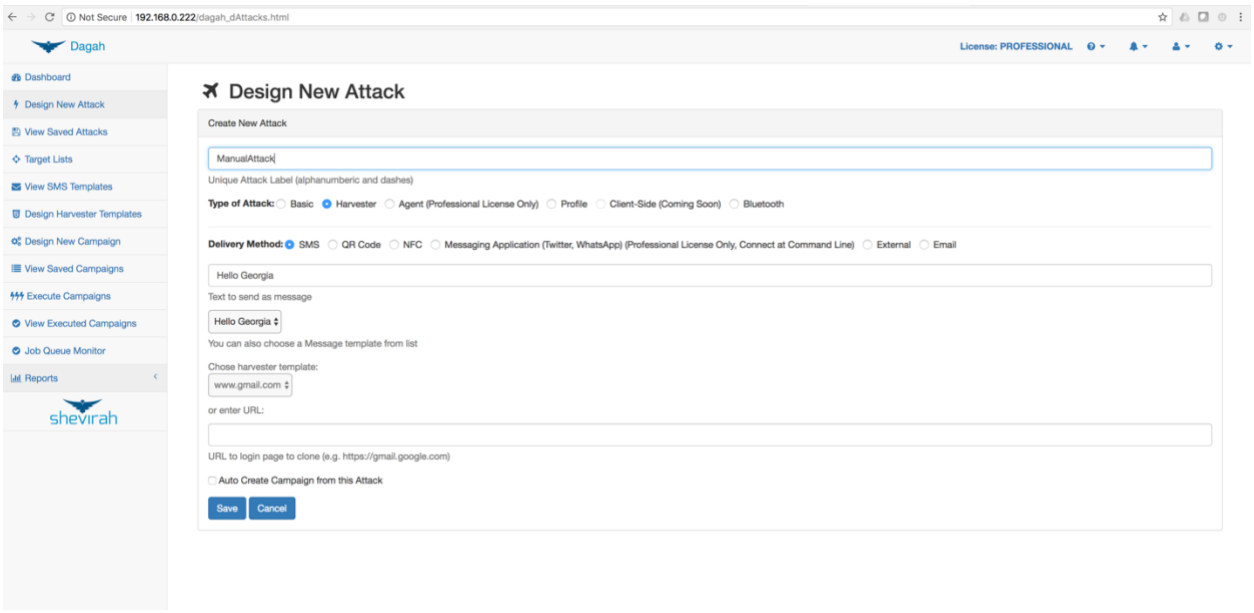
In your Bit.ly account generate an OAuth token as directed by the help.



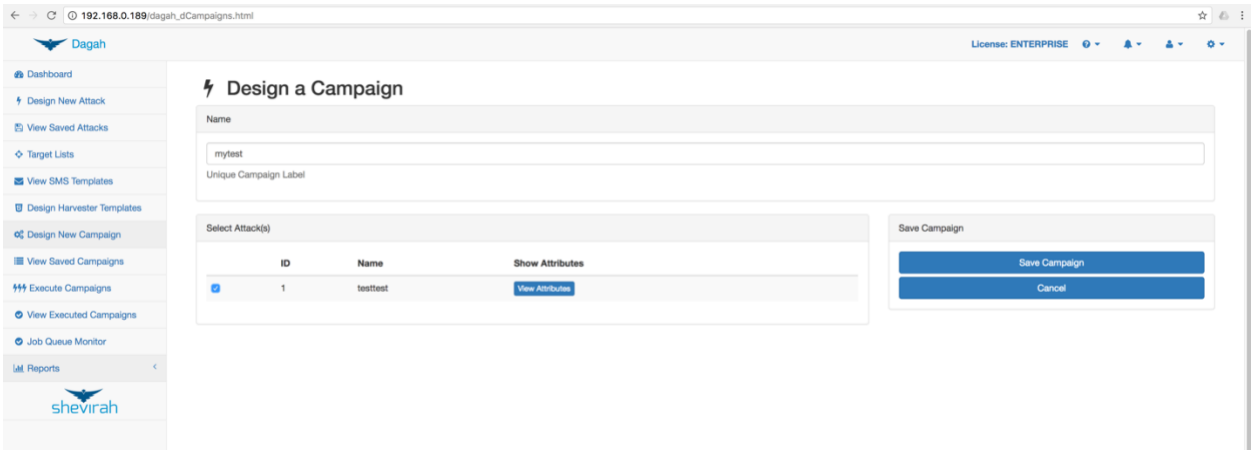
Save the token in the Site Configurator and relogin.



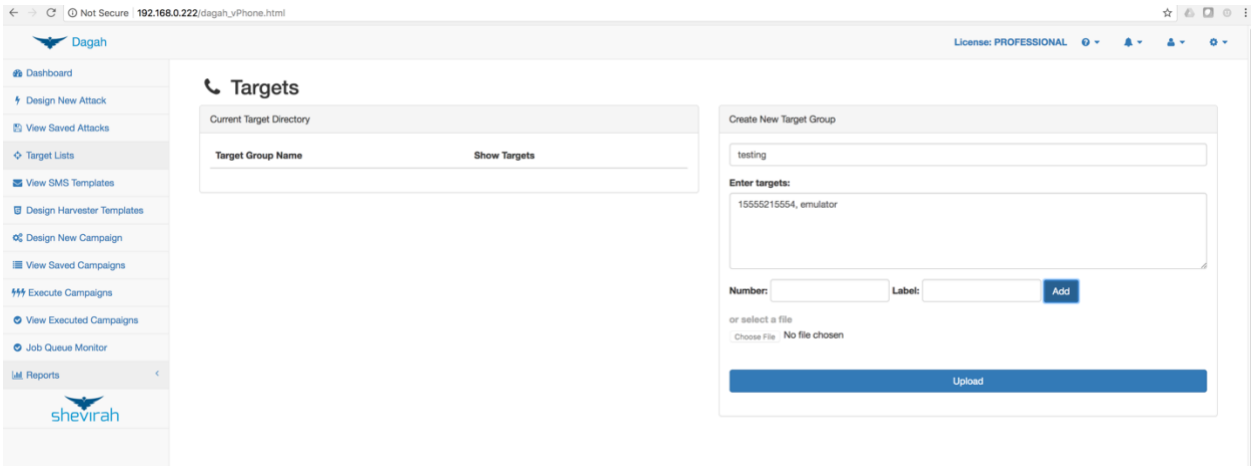
To create an attack go to Design New Attack on the left hand menu. Set the name of the attack, the type of attack (Harvester for this example), delivery method and other settings as necessary (in this case SMS template for the SMS delivery and harvester template for the harvester page). You can check the box to automatically create a campaign with this one attack to save a step, but let's not so we can look at creating a campaign.



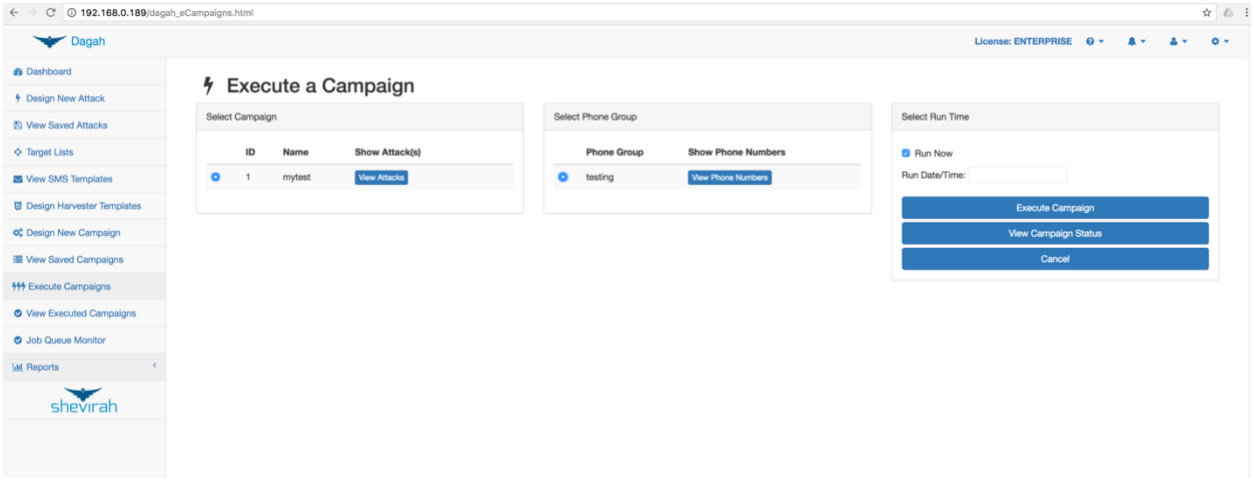
After saving the attack(s) to want to run, go to Design New Campaign on the left. Name the campaign and select the desired attacks(s) from the list of saved attacked. Click Save Campaign on the right.



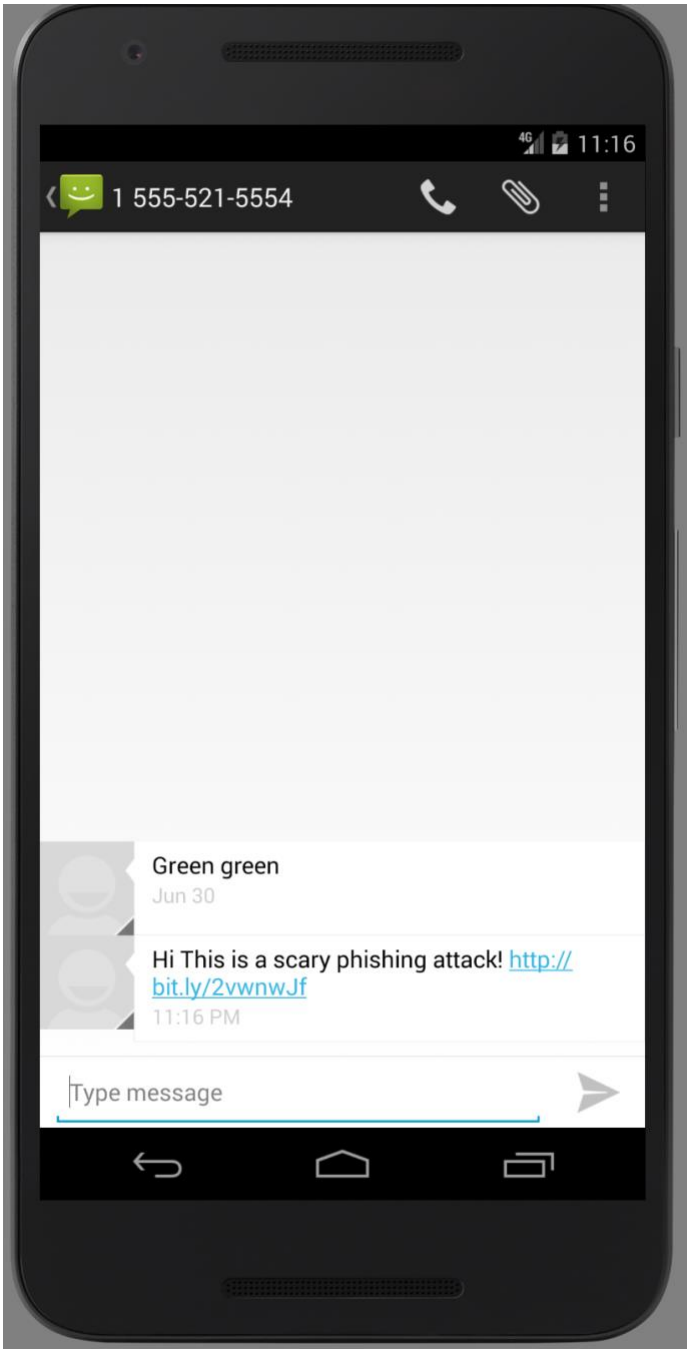
We need targets to attack. Click Target Lists on the left hand menu. You can upload a list or enter the list through the dialog. Targets can be phone numbers, email addresses, or Twitter handles. In this example we put in the number and a label of our target emulator. Save the target list with the upload button once you have entered all your targets.



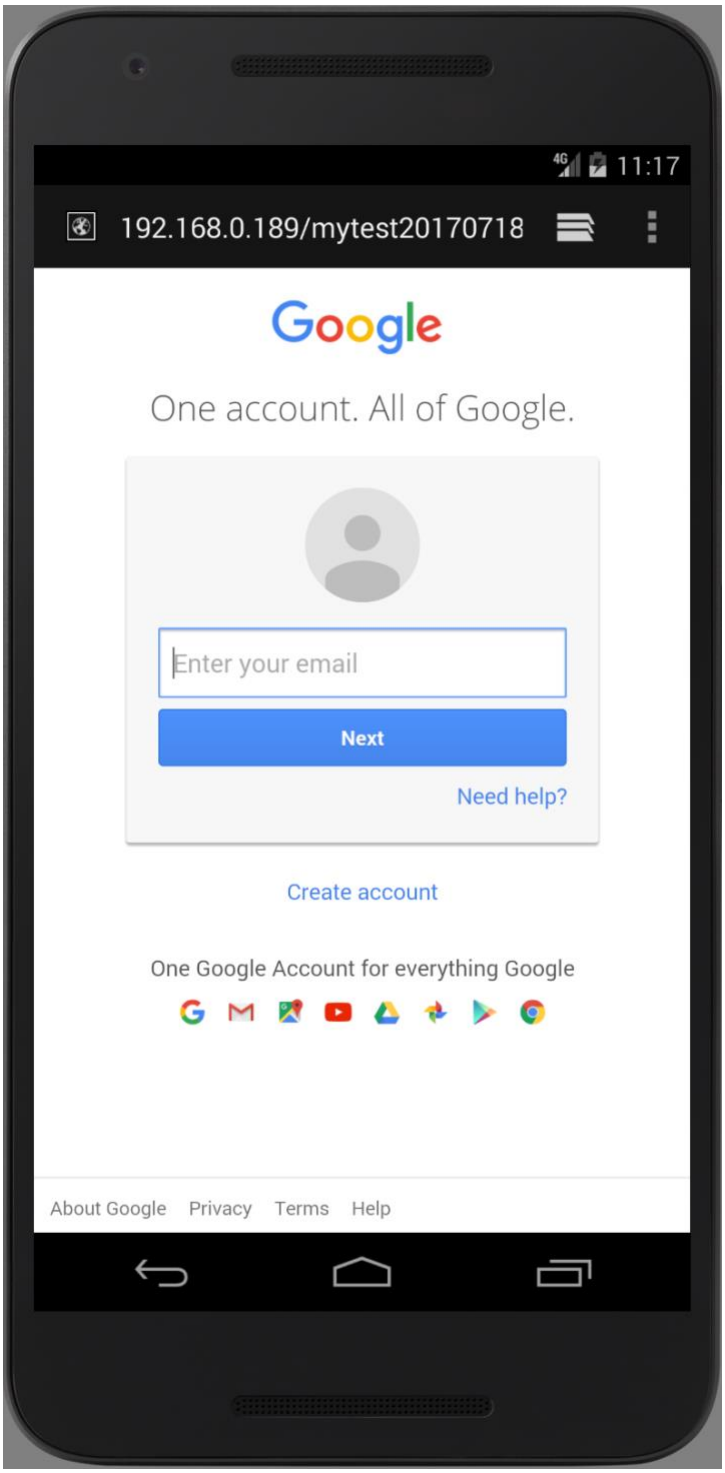
Now that we have a campaign and targets to run it against go to Execute Campaigns on the left. Choose a campaign to run from the saved campaigns on the left, a target list to turn it against, and click Execute Campaign.



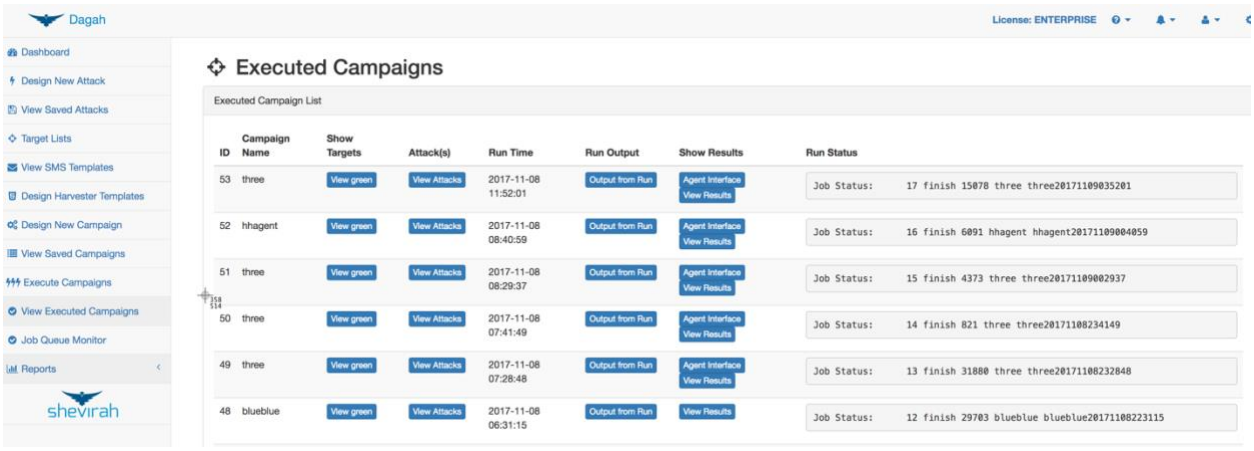
The targets will receive a text message with the SMS template text with a bit.ly obfuscated link.



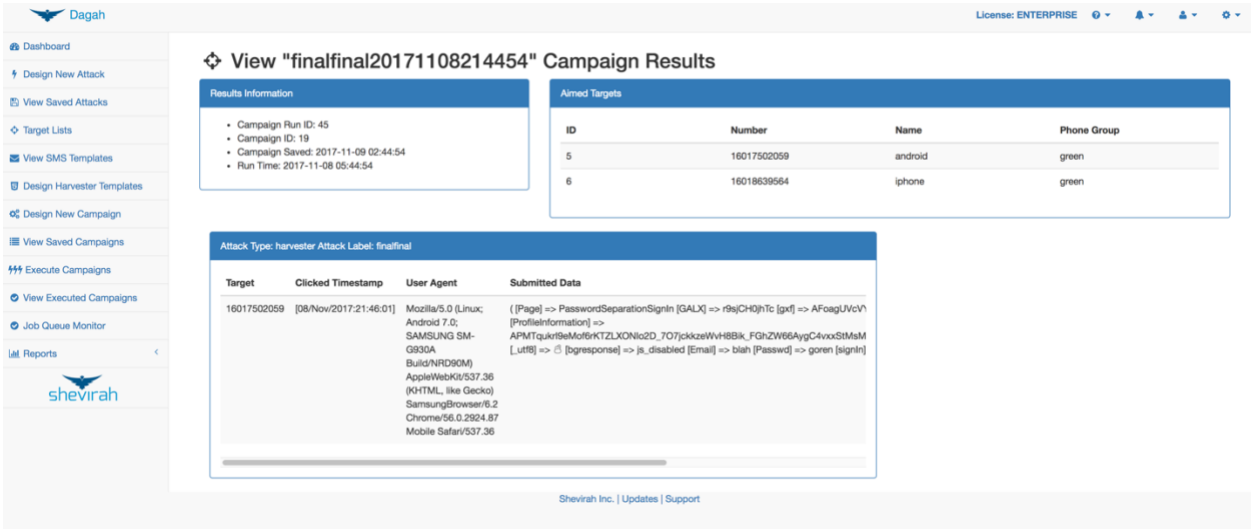
If the target clicks on the link it redirects to the harvester page. It looks and feels like the real gmail.com (and redirects to it after the user attempts to login) but captures the users requests.



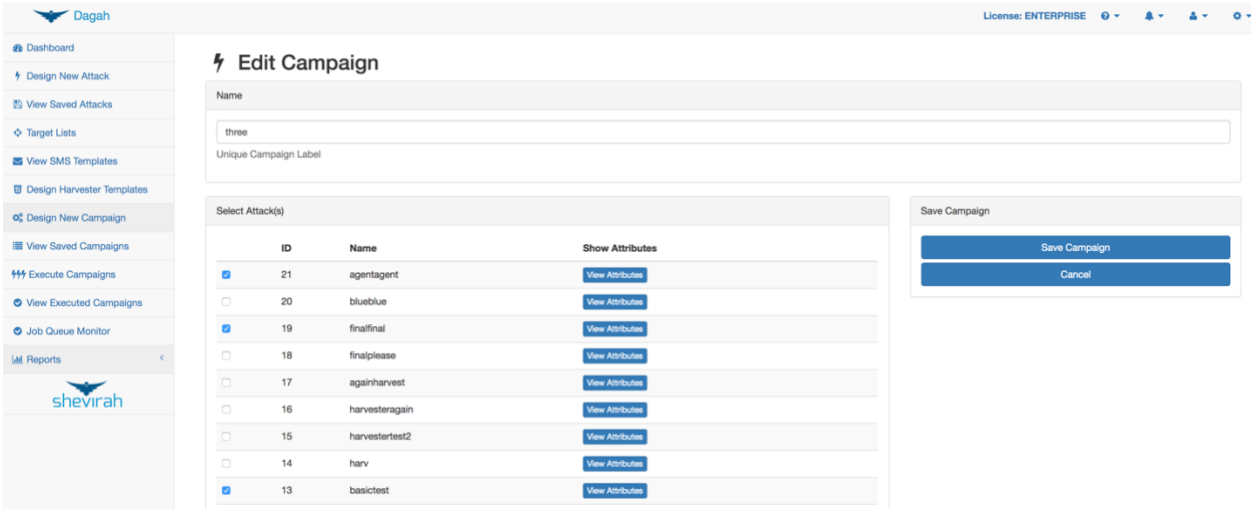
If the user clicks on the link on a basic or harvester attack, their user agent string will be recorded on the results page. If they submit data to a harvester page that will be recorded as well. To see campaign results click View Executed Campaigns on the left side menu.



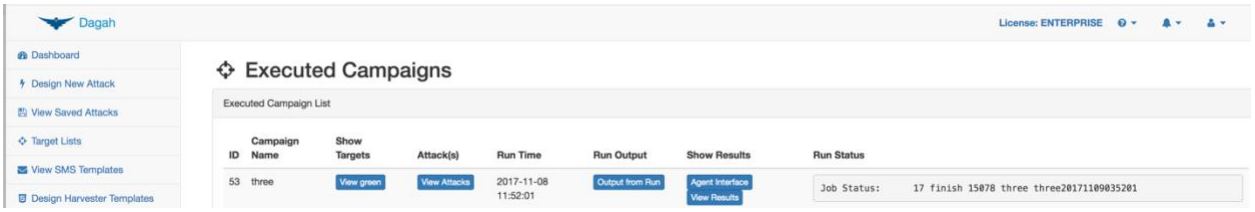
In the show results column click the View Results Button to see the results of your campaign.



With a professional license, you can make campaigns made up of multiple attacks by checking multiple boxes for attacks on the Create Campaigns page. Here is an example of a campaign with three attacks.



If a campaign includes an agent attack in addition to the View Results button on the View Executed Campaigns page you will see and Agent Interface button. If you click that it will take you directly to the Agent post exploitation interface discussed in the agents section earlier in this document.



You can view the status of your campaign runs by clicking the Job Queue Monitor on the left side menu.