Zero Degrees

CYBER-DEFENSE USING OPEN-SOURCE

Phish Defense with Gophish

Ashish Chalke
THEASHISHCHALKE

ZERO DEGREES

Table of Contents

Foreword

I’ve been associated with CyberSecurity [or infosec if you prefer] for nearly a decade now, mostly in a role that some people might hate (Sales Engineer). I really love discovering and working with open-source tools. In the field, organizations don’t implement solutions (read:tools) citing budget concerns. I broadly categorize tools into two categories – The Need to have – must have solutions that you need to spend money on such as Firewalls, endpoint security; and the – Good to have – solutions that you implement if you have the budget. Ideally, if you are going to buy a solution that you are only going to utilize 30% of it’s capabilities, that’s a Good to Have solution, unless those are supported by business goals. Secondly, tools play an important part in helping your team. Your team may have great analysts and some good analysts. Tools help to level the playing field.

I initially thought of writing a book that documents different open source tools that organizations can implement from their Good to Have list, but then thought of how could this make more impact? Rather than a 500 page book, how about I create quick books focusing on one aspect? Rather than charging for a book, how about I distribute it for free so that anyone can pick it up and implement!

I intend to publish at least 4 of these mini-books, looking at different aspects in 2020. The one you are reading aims to add to your Security Awareness program by harnessing active phishing simulations and converting the users into another layer of your Cyber Defense strategy.

If you find this useful, do share with others and help make the internet safer. After all Security = Safety.

The Basics

Social engineering is how majority of cyber-attacks start. Out of all social engineering tactics, Phishing is the most common. While we implement preventive measures such as Anti-spam/anti-phishing, link reputation, sandboxing etc., there is bound to be a chance where a malicious email bypasses these defenses and lands in a user's inbox. And then all the cyber criminal needs is Curious Karan to click on that link which will lead him to cute cat GIF heaven!

Most cyber security teams treat users as dumb zombies, performing ritual awareness trainings that are more of an HR exercise and do not account for the powerful role a user can play when properly educated and trained. Phishing Detection is a very vital training that should figure in your Security Education Training Awareness (SETA) program. However what end users are subject to are mindless hours of boring and repetitive video content that most of them skip forward [True story, I do that, so do you!].

Larger organizations add gamification or an active phishing training. Tools such as Proofpoint's PhishAlarm or KnowBe4's ESAT platform allow you to send a fake phishing mail to a user with which you can track user behavior – did the user click the phishing link, did they report it to Security Operations, did they ignore the mail [ha! Bet their manager would be interested in this]. Once you have the results, it's easy to know which users need more attention. You could even make this interesting by offering a small reward to users who successfully detect and forward the phishing campaigns. By rewarding user behavior, users will be actively on the lookout for suspicious emails, making your job of securing them much easier.

Unfortunately these tools fall under the Good to Have category. When deciding between a EPP upgrade (or the latest shiny NGFW) and a SETA tool, there's a higher chance that budget will be allocated to the former. This is where we leverage the power of opensource. Introducing **Gophish** [https://getgophish.com], an open source phishing framework developed in the Go programming language created by Jordan Wright [@jw_sec] and is available as a package for most popular OSes. From its website, Gophish is described as

> *Gophish is a powerful, open-source phishing framework that makes it easy to test your organization's exposure to phishing.*
>
> *For free.*

Gophish is deployed as either a VM or a package for an OS of your choice. You can then launch campaigns to your targets and review the results on a per campaign basis. This makes Gophish much more powerful than just a by the number's awareness exercise. We will be leveraging other tools that integrate with Gophish that cover some of the shortfalls compared to more commercial tools. Before we get down to the nitty gritty, let's cover some basics about Gophish.

The architecture is quite simple. Gophish has a listening service that listens for incoming connections from phished users, and an Admin service, accessed via the default TCP Port 3333. The capability to customize the listener allows practioners to leverage Gophish as a red team and a blue team tool. Gophish has the following elements:

- **Templates** – a template is just that. An email template baiting the end user to click on the link [or open the attachment if you're red]. The embedded link is uniquely tracked by Gophish using the rid parameter.
- **Landing pages** – are the web page a user sees when they click on the link from the phishing template. You can use your own HTML code or clone an existing website. Credentials can be

harvested from the landing page before redirecting the user to another page. Do note that credentials are stored in plaintext. I would recommend not selecting the “Capture Passwords” checkbox when capturing user submitted data to prevent from creating another attack surface.

- **Groups** – Are used to logically group targets. To create separate campaigns for sales and accounts, you would create two different target groups. You can then import users into the group by manually creating users by hand or using the bulk import option to import a csv.
- **Sending Profiles** – are SMTP relay settings to tell Gophish on how to send email for a particular campaign.
- **Campaign** – finally a campaign is where all of this comes together. A campaign is like a project where you select all of the above configured elements and track the success of this project by observing the user behavior. A campaign is brought together by the following:
  - **Group** – who is your target?
  - **Template** – how will they be baited?
  - **Landing Page** – what will they see?
  - **URL** – how will they see what they need to see?
  - **Sending Profile** – How will the campaign be launched?
  - **Send by** – When should it be launched?

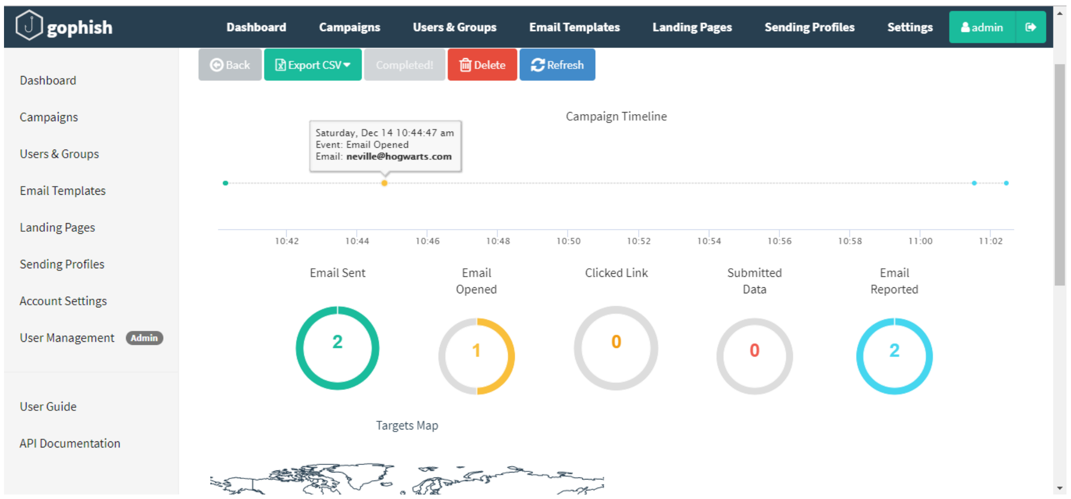
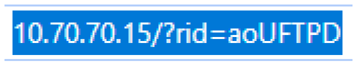


Fig. *Gophish campaign tracking.*

- **RID** – the RID or Result ID is the secret sauce that helps Gophish track the stages of user behavior as the campaign progresses. Whenever you launch a campaign, Gophish creates a unique id for each user being targeted. This ID is appended to the landing page embedded in the emails being sent out to the users, and whenever a user clicks on the Phishing link, the activity can be uniquely tracked. This ID is valid for that specific campaign. Meaning a user can be a part of multiple campaigns, and each campaign will have a unique rid for the user so that their activity can be tracked in the context of a specific campaign. **Example**: http://<landingpage-ip>/?rid=<random-id>

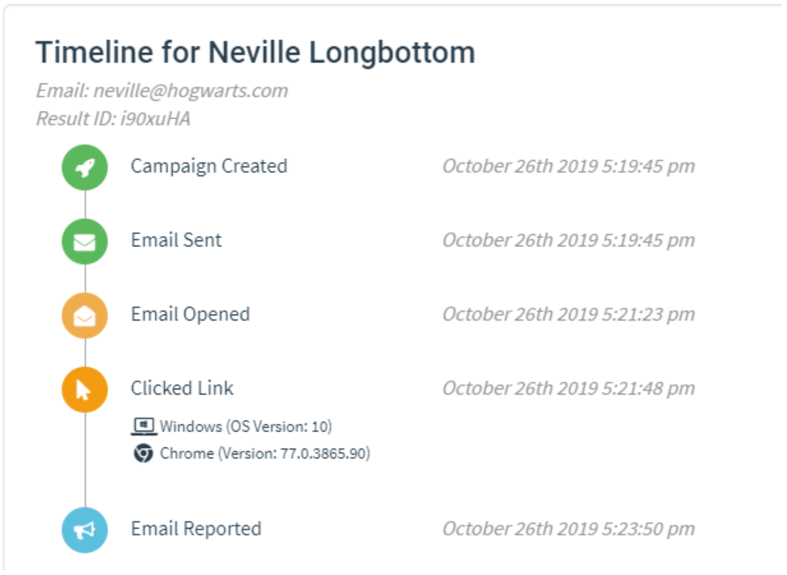


10.70.70.15/?rid=aoUFTPD

Fig. *a rid in the Landing page URL*.

Tracking actions that are available within Gophish are:

- **Email Sent** – tracked by the Gophish server on successful relay.
- **Email opened** – Each email includes an embedded remote image that invokes the landing page with the user rid. A word of caution, most email clients will block embedded images by default from an untrusted domain. Before launching a campaign, ensure that you whitelist your phishing domain to properly track this stage. (And if you're horrified at the idea of whitelisting a phishing domain, more on that later.)
- **Clicked on Link** – Each link in the email template is linked back to the landing page with the users rid. If the user clicks on a link, the event is captured.
- **Submitted Data** – If your landing page allows users to submit data such as credentials, this will track when a user successfully submits data via a POST to the landing page.
- **Email Reported** – tracks if your user successfully reports a phishing email. Currently as of Gophish version 0.8.1 this functionality is only enabled on the server side, and there is currently no client side interface (such as a plugin button for outlook) for the user to report an email. Fear not open source warrior! We shall explore a third party utility that helps you perform the client side of reporting that plugs into Gophish to give you a complete 360^{o} experience. [Written by yours truly, I might add]



Timeline for Neville Longbottom

Email: neville@hogwarts.com
Result ID: i90xuHA

Campaign Created	October 26th 2019 5:19:45 pm
Email Sent	October 26th 2019 5:19:45 pm
Email Opened	October 26th 2019 5:21:23 pm
Clicked Link Windows (OS Version: 10) Chrome (Version: 77.0.3865.90)	October 26th 2019 5:21:48 pm
Email Reported	October 26th 2019 5:23:50 pm

Fig: *Sample tracked actions for a User*.

Gophish also provides a rich REST and Pythonic API that can be used to orchestrate campaigns and perform reporting functions. Gophish however has some shortfalls that you would need to prepare for before launching the campaign. Like mentioned previously, the user side of the reporting functionality is yet to be developed. User Reporting is an important metric in your SETA program. This would tell you how effective your SETA program is, specific target areas and definitely 'try harder' (to quote a popular offsec ideology) when phishing your users. You would need to figure out a way to let users inform SecOps about a dodgy email. Most commercial tools include a plugin for outlook that can trigger an http request to the phishing server when clicked. However this does not work in a heterogenous environment consisting of mobile. The simplest method is coaching users to forward emails to a monitor mailbox such as report@xyz.com. SecOps could then validate and report the email to Gophish or use automation scripts for doing this. More on this in later sections. However if you're interested in a plugin approach, Societe Generale's Swordphish [https://github.com/certsocietegenerale/swordphish-awareness] has a NotifySecurity plugin [https://github.com/certsocietegenerale/NotifySecurity] that can be customized to your environment.

Gophish also doesn't respond to dns queries for your phishing domains. If you are hosting your Gophish instance offsite, you will need to make sure that DNS is properly setup for the domain, with SPF and DKIM records to make sure that your Anti-spam doesn't catch your mail. This functionality is currently being discussed for a future gophish release, so do keep an eye out on releases.

Deploying Gophish

Before we get to actually deploying the Gophish server, we need to think about how our users are going to interact with it and how realistic we want our simulations to be. Ideally, to simulate a real threat actor, we could look at deploying our Gophish server on the public cloud, AWS for instance. The server needs to be reachable by users who click on the links, hence it needs a public IP address as well as a domain name. We need an SMTP relay to send emails from our Gophish instance. Finally we need a legitimate domain name that can be queried for and also compel users to open. So summarizing:

- Hosting space – public cloud
- Public IP Address – for reachability
- Domain name – I would highly recommend using derivatives of your corporate domain. For example: if your corp domain is ashishchalke.com, try using derivatives such as ashishchalkecorp.com, ashishchalke.co, or even phonetic characters such as ashishchalkè.com [this one is still available at the time of writing]
- An SMTP relay service – If using AWS, just use Amazon SES.

Once we have the building blocks, we need to add the proper records for the domain so that it can properly respond when queried, so that it doesn't enter the junk folders or is outright rejected by your mail service provider if it doesn't find an origin. You would need to setup the below records:

- Mx – mail exchange records. These should point back to your SMTP Relay instance.
- SPF – txt records, specifies which servers are allowed to send email for your domain. An example:
  ```
  v=spf1 mx a ip4:1.1.1.1/32 ip4:2.2.21.2/32 a:mail.ashishchalke.com
  ~all
  ```

Or you could just whitelist the domain on your Anti-spam services. If your Anti-spam solution supports whitelisting via email headers, you can add a custom header in your email and whitelist that. Think about it – you craft the perfect phishing email, only to have it caught by your spam filter. This is actually good. You now know that your anti-spam services work against the "noise" attackers (aka script kiddies). However, what we really want to test are the inner layers as part of this exercise, specifically the User layer. Another note of caution, if you intend to use the public architecture described above, try to have a couple domains to prevent users from picking up on the test emails.

We could also setup Gophish as a private server on-premise, and have it relay off our mail server, making it easier to deploy. All we would need is an IP or FQDN that is reachable by end users. Finally, if you wish to setup Gophish in your testing lab, you can leverage the MorningCatch environment [https://blog.cobaltstrike.com/2014/08/06/introducing-morning-catch-a-phishing-paradise/] released by the creators of Cobalt Strike, that has a complete end-user environment setup for you to phish.

In the following section, we will look at setting up Gophish in a private datacenter on-premise, but you could follow the same steps to setup Gophish in AWS. The end result should be the same – we should have a working Gophish server with which to terrorize and phish our users!

Private Instance:

For setting up a private Gophish server, I will be utilizing my lab described below.

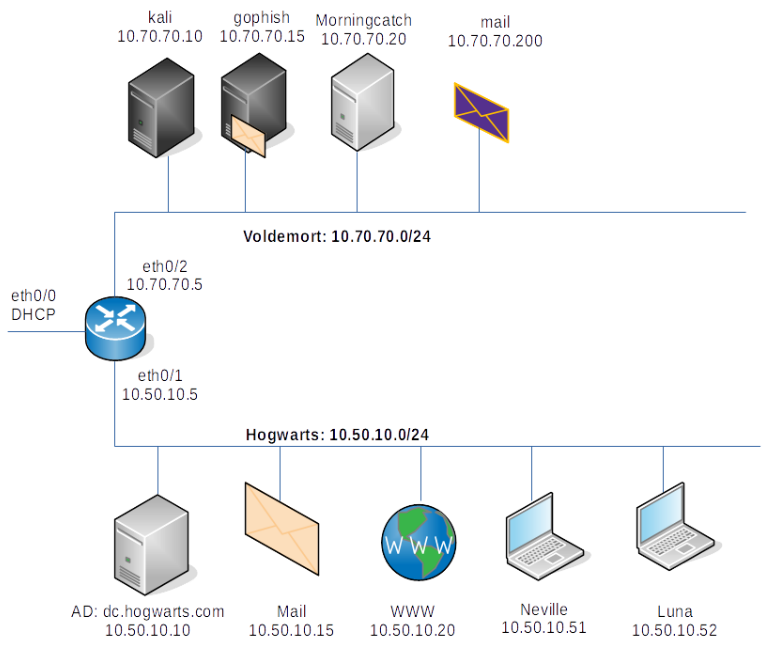


Fig: *Gophish Private lab*.

The lab consists of the Blue team – Hogwarts.com and their erstwhile nemesis Voldemort on the Red team. Both are separated by our trusty VyOS router, with the Red team network as 10.70.70.0/24 and Blue Team network as 10.50.10.0/24. The Blue Team aka Hogwarts elements have already been setup and are as follows:

- An Active Directory and an iRedmail server provide the infrastructure components.
- Two clients – Gryffindor (user: Neville) and Ravenclaw (user: Luna)
- An Administrator – Dumbledore: dumbledore@hogwarts.com

The Red Team aka Voldemort are composed as follows:

- Gophish server – we will set this up on 10.70.70.15
- A Postfix server – to relay emails
- Optional – MorningCatch – to test our phishing campaign before we unload it onto users.

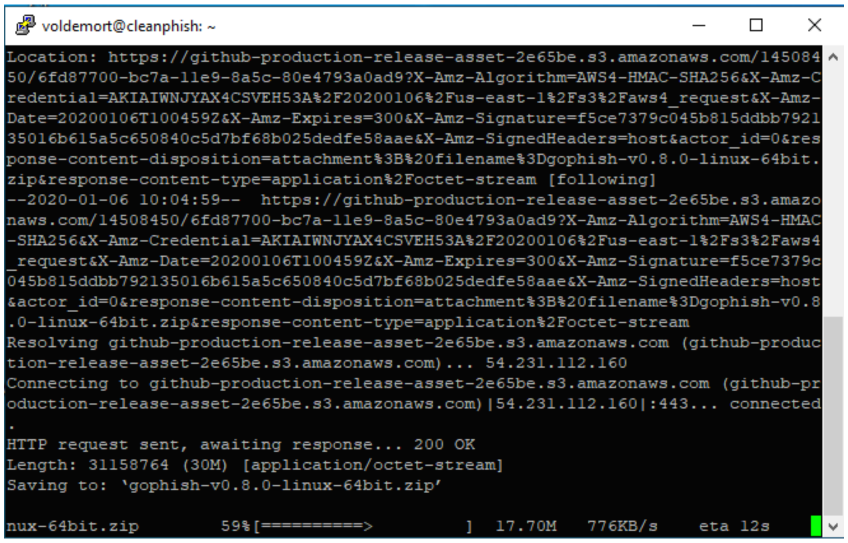
Setup – Private Instance

Installing Gophish is quite easy and accomplished in a few steps. I will be using an Ubuntu Server 18.04 to deploy, but setup is available for Windows, Linux and OSX. What I really like about using Ubuntu is that I can deploy it as a headless (non GUI) server, utilizing less resources. There are plenty of articles available on how to install Ubuntu, so we will skip that. Please note that the server should have a static IP Address.

Open a terminal. Install the `unzip` utility to extract zip files.

```
sudo apt-get install unzip
```

Download the latest Gophish release:
`wget` [`https://github.com/gophish/gophish/releases/download/v0.8.0/gophish-v0.8.0-linux-64bit.zip`](https://github.com/gophish/gophish/releases/download/v0.8.0/gophish-v0.8.0-linux-64bit.zip)



voldemort@cleanphish: ~

```
Location: https://github-production-release-asset-2e65be.s3.amazonaws.com/145084
50/6fd87700-bc7a-11e9-8a5c-80e4793a0ad9?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-C
redential=AKIAIWNJYAX4CSVEH53A%2F20200106%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-
Date=20200106T100459Z&X-Amz-Expires=300&X-Amz-Signature=f5ce7379c045b815ddbb7921
35016b615a5c650840c5d7bf68b025dedfe58aae&X-Amz-SignedHeaders=host&actor_id=0&res
ponse-content-disposition=attachment%3B%20filename%3Dgophish-v0.8.0-linux-64bit.
zip&response-content-type=application%2Foctet-stream [following]
--2020-01-06 10:04:59--  https://github-production-release-asset-2e65be.s3.amazo
naws.com/14508450/6fd87700-bc7a-11e9-8a5c-80e4793a0ad9?X-Amz-Algorithm=AWS4-HMAC
-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20200106%2Fus-east-1%2Fs3%2Faws4
_request&X-Amz-Date=20200106T100459Z&X-Amz-Expires=300&X-Amz-Signature=f5ce7379c
045b815ddbb792135016b615a5c650840c5d7bf68b025dedfe58aae&X-Amz-SignedHeaders=host
&actor_id=0&response-content-disposition=attachment%3B%20filename%3Dgophish-v0.8
.0-linux-64bit.zip&response-content-type=application%2Foctet-stream
Resolving github-production-release-asset-2e65be.s3.amazonaws.com (github-produc
tion-release-asset-2e65be.s3.amazonaws.com)... 54.231.112.160
Connecting to github-production-release-asset-2e65be.s3.amazonaws.com (github-pr
oduction-release-asset-2e65be.s3.amazonaws.com)|54.231.112.160|:443... connected
.
HTTP request sent, awaiting response... 200 OK
Length: 31158764 (30M) [application/octet-stream]
Saving to: 'gophish-v0.8.0-linux-64bit.zip'

nux-64bit.zip        59%[===========>          ]  17.70M   776KB/s    eta 12s
```

Fig: *Downloading the package*.

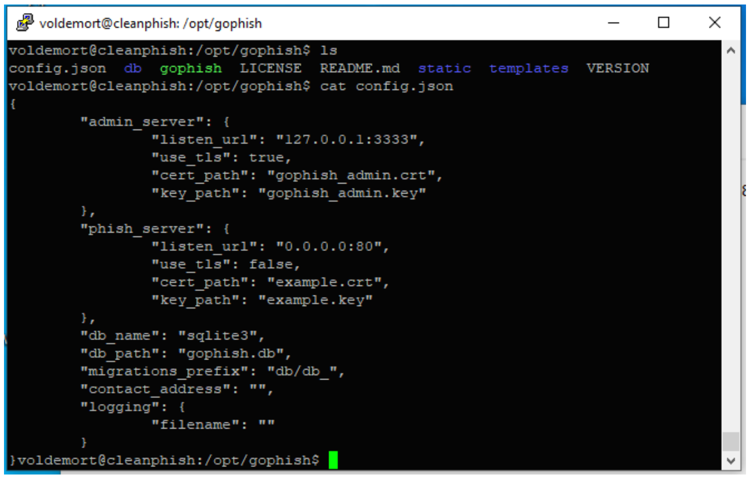
Since gophish is regularly updated, keep a lookout for new releases and update the url in the above command with the new download link. Releases page: https://github.com/gophish/gophish/releases

Unzip gophish to `/opt/gophish` folder using the unzip utility we installed earlier.
`sudo unzip gophish-v0.8.0-linux-64bit.zip -d /opt/gophish`

By default, the admin interface is only accessible on the local machine. Since we are on a headless server, we have no option to view it. Also, we will also need to access the admin interface from a remote workstation, and it would be more secure and easier to allow Web access to the server rather than the full console. We could provide direct web access to the admin interface by modifying the configuration file.

Change to the Gophish directory – `cd /opt/gophish`

To view the config the config.json file, use – `cat config.json`



```
voldemort@cleanphish:/opt/gophish$ ls
config.json  db  gophish  LICENSE  README.md  static  templates  VERSION
voldemort@cleanphish:/opt/gophish$ cat config.json
{
        "admin_server": {
                "listen_url": "127.0.0.1:3333",
                "use_tls": true,
                "cert_path": "gophish_admin.crt",
                "key_path": "gophish_admin.key"
        },
        "phish_server": {
                "listen_url": "0.0.0.0:80",
                "use_tls": false,
                "cert_path": "example.crt",
                "key_path": "example.key"
        },
        "db_name": "sqlite3",
        "db_path": "gophish.db",
        "migrations_prefix": "db/db_",
        "contact_address": "",
        "logging": {
                "filename": ""
        }
}voldemort@cleanphish:/opt/gophish$
```

Fig: *config.json file*

The **"listen_url"** under **admin_server** is currently listening on the loopback interface 127.0.0.1 which is not accessible over the network. We can change this to the IP Address of the server interface to enable access over the network. You can also specify 0.0.0.0 to listen on all available interfaces. This is not recommended for production or internet exposed servers.

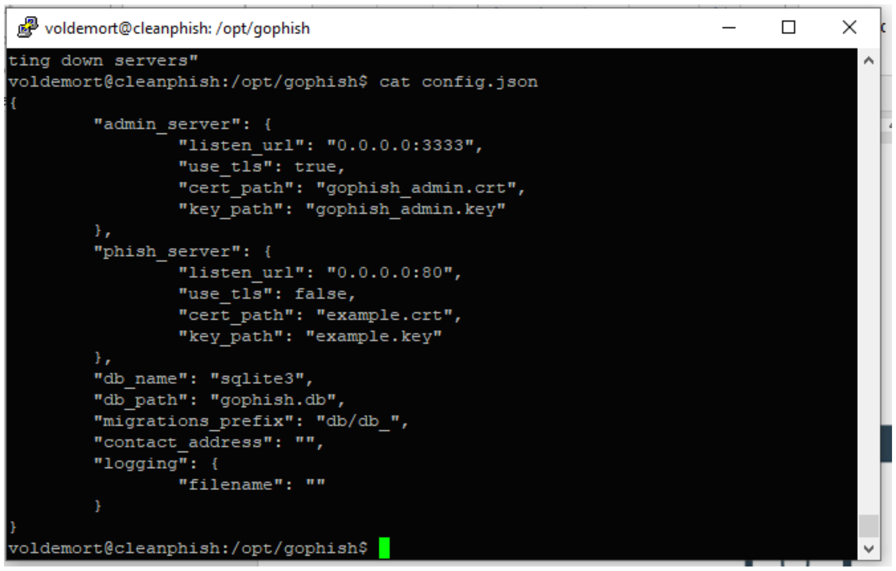
Original

```
{
    "admin_server": {
        "listen_url": "127.0.0.1:3333",
        "use_tls": true,
        "cert_path": "gophish_admin.crt",
        "key_path": "gophish_admin.key"
    },
```

Changed

```
{
    "admin_server": {
        "listen_url": "SERVER-IP:3333",
        "use_tls": true,
        "cert_path": "gophish_admin.crt",
        "key_path": "gophish_admin.key"
    },
```

We can use `nano` – a text editor – to change the configuration. Use `sudo nano config.json` to edit the file in the text editor. Once changed, press `CTRL+X` to exit. Type `Y` to save. Our configuration file should now reflect the updated configuration.



voldemort@cleanphish: /opt/gophish

```
ting down servers"
voldemort@cleanphish:/opt/gophish$ cat config.json
{
        "admin_server": {
                "listen_url": "0.0.0.0:3333",
                "use_tls": true,
                "cert_path": "gophish_admin.crt",
                "key_path": "gophish_admin.key"
        },
        "phish_server": {
                "listen_url": "0.0.0.0:80",
                "use_tls": false,
                "cert_path": "example.crt",
                "key_path": "example.key"
        },
        "db_name": "sqlite3",
        "db_path": "gophish.db",
        "migrations_prefix": "db/db_",
        "contact_address": "",
        "logging": {
                "filename": ""
        }
}
voldemort@cleanphish:/opt/gophish$
```

Fig: The updated *config.json file*

If you are using the Uncomplicated Firewall (`ufw`), you may need to allow external access to the 3333 port. You can do this with the command: `sudo ufw allow 3333`

We can now start Gophish by typing: `sudo ./gophish`

The management interface will now be accessible via your favorite browser on https://SERVER-IP:3333.

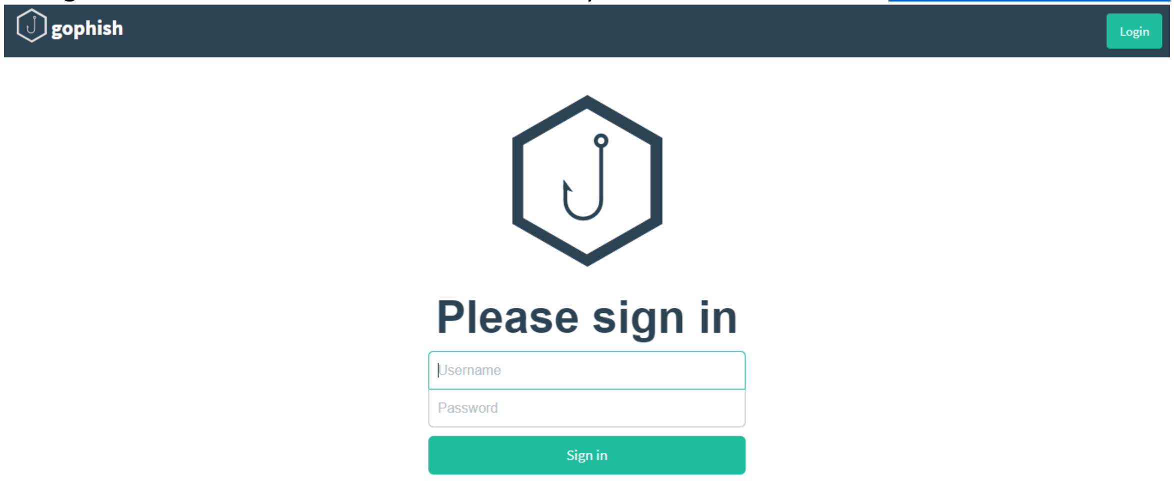


Fig: *Sign-in Page*

That's it! You've setup Gophish successfully. However, the workflow involves manually starting Gophish everytime the server reboots. This can be quite cumbersome operationally. We can create a service that starts Gophish at boot.

First we will create a user call `gophish` who will run the service.

```
sudo useradd -r gophish
```

Next we will use nano to create a Gophish Service file:

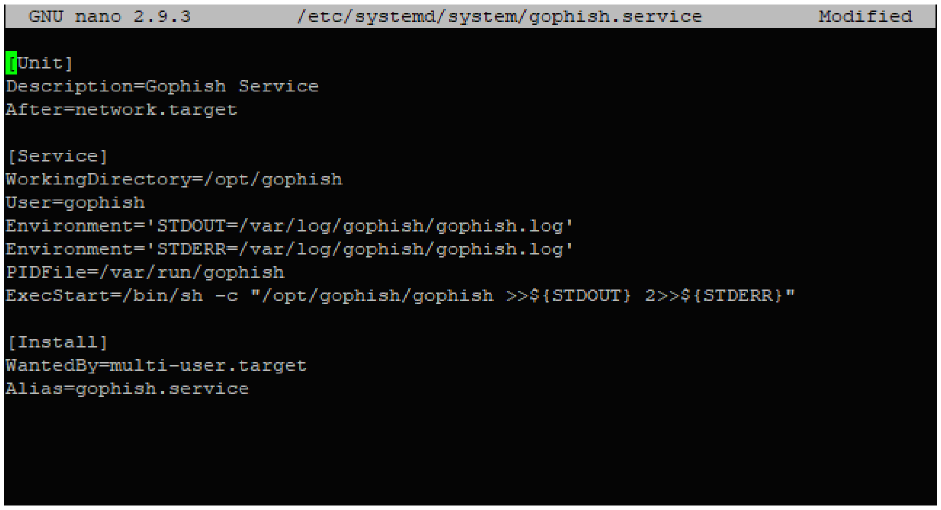
```
sudo nano /etc/systemd/system/gophish.service
```

Copy and paste the codeblock below:

Gophish Service Code

```
[Unit]
Description=Gophish Service
After=network.target

[Service]
WorkingDirectory=/opt/gophish
User=gophish
Environment='STDOUT=/var/log/gophish/gophish.log'
Environment='STDERR=/var/log/gophish/gophish.log'
PIDFile=/var/run/gophish
ExecStart=/bin/sh -c "/opt/gophish/gophish >>${STDOUT}
2>>${STDERR}"

[Install]
WantedBy=multi-user.target
Alias=gophish.service
```

Your service file should look like the screenshot below. Save and exit nano: CTRL+X -> Y



```
  GNU nano 2.9.3           /etc/systemd/system/gophish.service            Modified

[Unit]
Description=Gophish Service
After=network.target

[Service]
WorkingDirectory=/opt/gophish
User=gophish
Environment='STDOUT=/var/log/gophish/gophish.log'
Environment='STDERR=/var/log/gophish/gophish.log'
PIDFile=/var/run/gophish
ExecStart=/bin/sh -c "/opt/gophish/gophish >>${STDOUT} 2>>${STDERR}"

[Install]
WantedBy=multi-user.target
Alias=gophish.service
```

Fig: *Service file*

We will now create a directory to store the service logs and make sure that the gophish user is assigned ownership permissions to the gophish application and log directories.:

```
sudo mkdir /var/log/gophish
sudo chown -R gophish:gophish /opt/gophish/ /var/log/gophish/
```

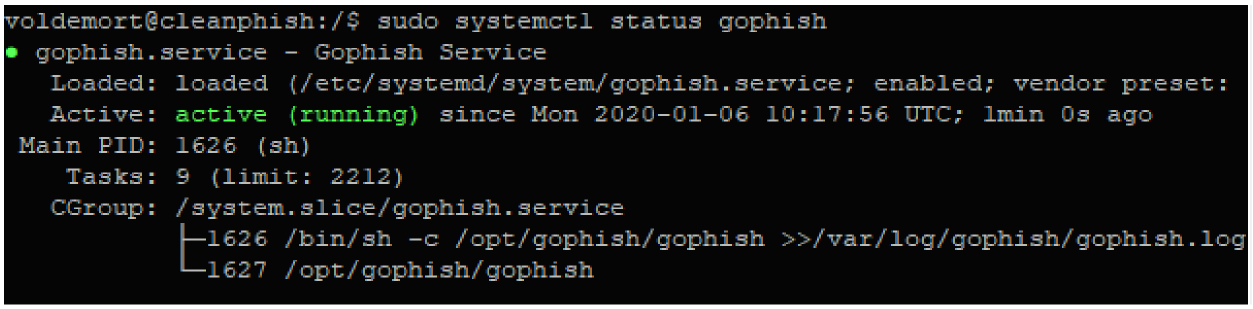
We will enable the Gophish application to bind itself to the ports specified in the configuration file:

```
sudo setcap cap_net_bind_service=+ep /opt/gophish/gophish
```

We will now reload and start the Gophish service:

```
sudo systemctl daemon-reload
sudo systemctl start gophish
sudo systemctl enable gophish
```

If we check the service status, we can see that the service is started and the admin interface is accessible. Try rebooting the server and checking if gophish is automatically started. To check service status use the command: `sudo systemctl status gophish`



```
voldemort@cleanphish:/$ sudo systemctl status gophish
● gophish.service - Gophish Service
   Loaded: loaded (/etc/systemd/system/gophish.service; enabled; vendor preset:
   Active: active (running) since Mon 2020-01-06 10:17:56 UTC; 1min 0s ago
 Main PID: 1626 (sh)
    Tasks: 9 (limit: 2212)
   CGroup: /system.slice/gophish.service
           ├─1626 /bin/sh -c /opt/gophish/gophish >>/var/log/gophish/gophish.log
           └─1627 /opt/gophish/gophish
```

Campaign

In this section we will deploy our first campaign. To launch a campaign, we will first create a Phishing template that will lure a user to click on our phishing link. Then we will create a landing page that asks the unsuspecting user to submit some data to us and then redirects to an educational site. We will create a sending profile to send our phishing campaign to our users. Finally we will create our campaign that combines all these elements together.

Creating an Email Template

Login to Gophish, go to **Email Templates** and click on **New Template**. A new template editor will popup.

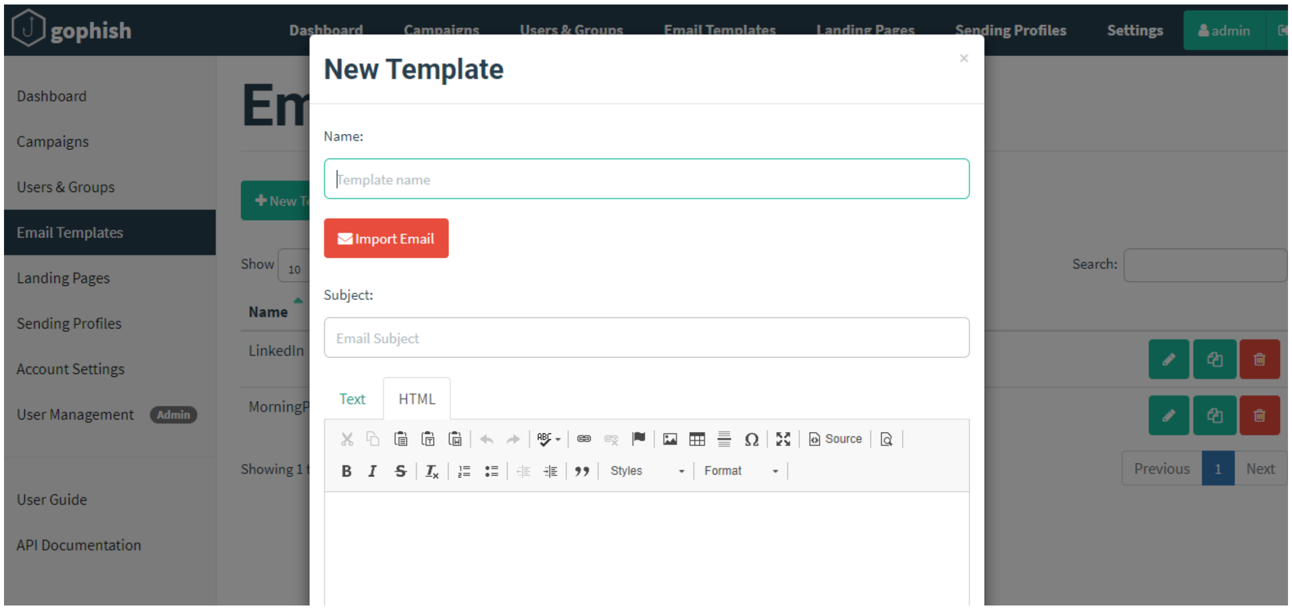
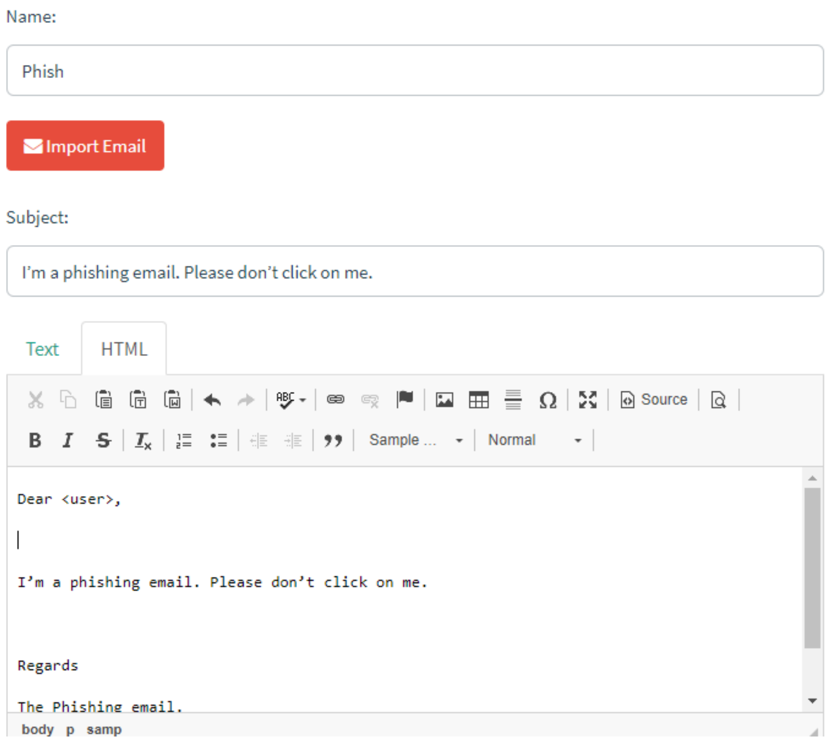


Fig: *New Template*

Select **HTML** and uncheck **Source** if selected. You now have a standard WYSIWYG editor. Give the template a name. Notice the **Import Email** button. This will let us import an actual email and change all links to our landing page, making a more legitimate bait. However we'll focus on manually creating our first Phishing email since our goal is to test the platform. How about we use the email below:

```
Subject: I'm a phishing email. Please don't click on me.

Dear <user>,

I'm a phishing email. Please don't click on me.

Regards
The Phishing email.
```

A simple to the point email. Lets paste it in our editor.



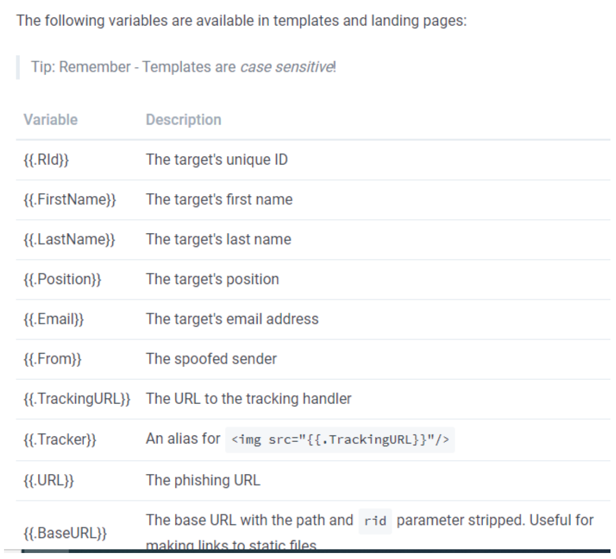
Phish

I'm a phishing email. Please don't click on me.

```
Dear <user>,

I'm a phishing email. Please don't click on me.


Regards

The Phishing email.
```

Doesn't look very genuine. Remember, the more personal an email, the more chances you have of the user clicking the email. However, creating a template per user is a tedious task. Template variables to our rescue! The Template Reference guide [https://docs.getgophish.com/user-guide/template-reference] provides us with a list of variables that we can use in our phishing email, and gophish will replace the variable with a value from the Database. Some cool variables to use:

- **{{.FirstName}}** - The target's first name, based on the name configured while creating the user.
- **{{.URL}}** - The phishing URL aka Landing page.
- **{{.RId}}** - The target's unique ID

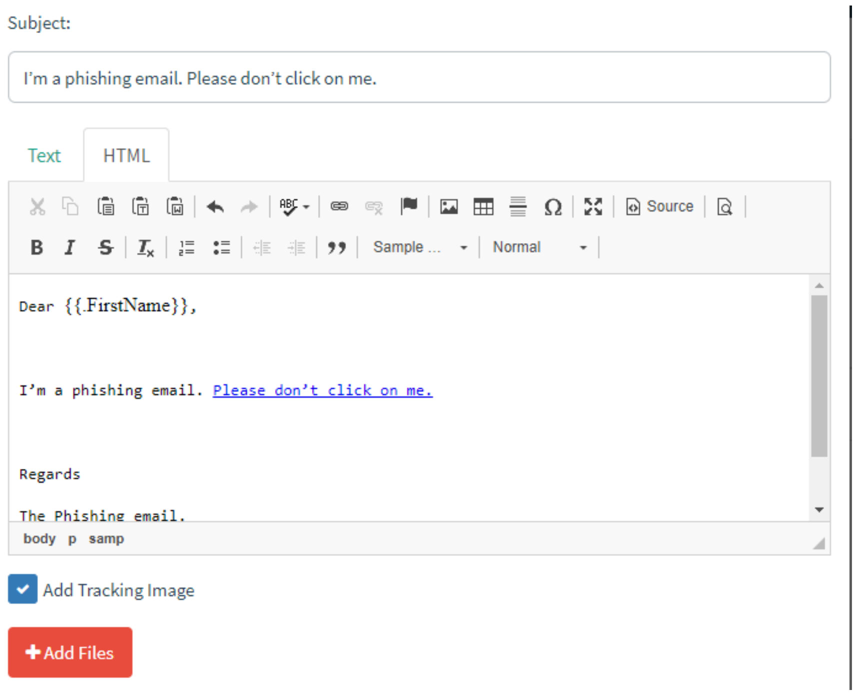


The following variables are available in templates and landing pages:

Tip: Remember - Templates are *case sensitive*!

Variable	Description
{{.RId}}	The target's unique ID
{{.FirstName}}	The target's first name
{{.LastName}}	The target's last name
{{.Position}}	The target's position
{{.Email}}	The target's email address
{{.From}}	The spoofed sender
{{.TrackingURL}}	The URL to the tracking handler
{{.Tracker}}	An alias for `<img src="{{.TrackingURL}}"/>`
{{.URL}}	The phishing URL
{{.BaseURL}}	The base URL with the path and `rid` parameter stripped. Useful for making links to static files

We will use the **FirstName** variable to make our email more personal. We will also use the URL variable to hyperlink the intended text. This makes our job easier so that we do not need to hardcode the url, and makes this template reusable against another landing page.



I'm a phishing email. Please don't click on me.

Dear {{.FirstName}},

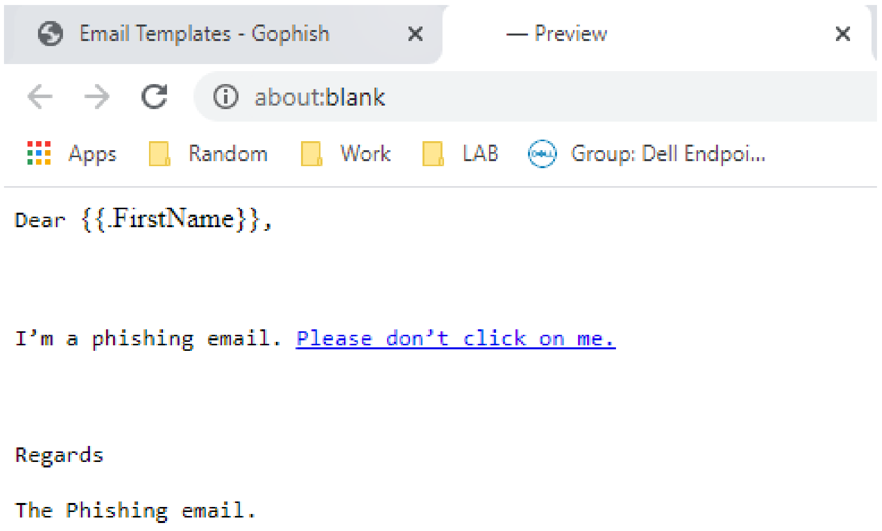
I'm a phishing email. Please don't click on me.

Regards

The Phishing email.

Fig: *Final email template*

Notice the **Add Tracking Image** checkbox. This will create a 1x1 pixel image that links back to our phishing URL and helps track if a user has opened our email. Click on the **Preview** button to view a preview of the final message. Once done, Click on **Save Template** to save our phishing mail.



Dear {{.FirstName}},

I'm a phishing email. Please don't click on me.

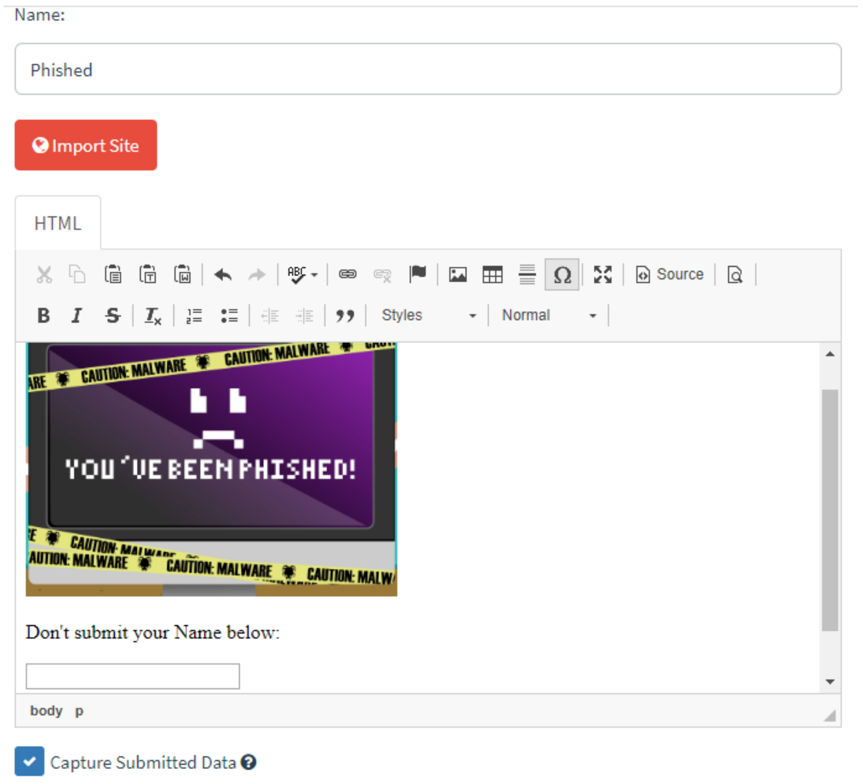
Regards

The Phishing email.

Fig: *Email Preview*

Create a Landing Page

Next we'll create our landing page. **Landing pages** are a great way to coach users or advance the attack one stage further, such as gathering user submitted data such as credentials. You can clone an existing website or create your own in the Editor. In the **Navigation Menu**, click on **Landing Pages** -> **New Landing Page**. I've used an image from the internet to coach the user. But to make this interesting, I've also create an input form to capture the name of the user. Notice we have setup the **Capture User submitted Data** checkbox.

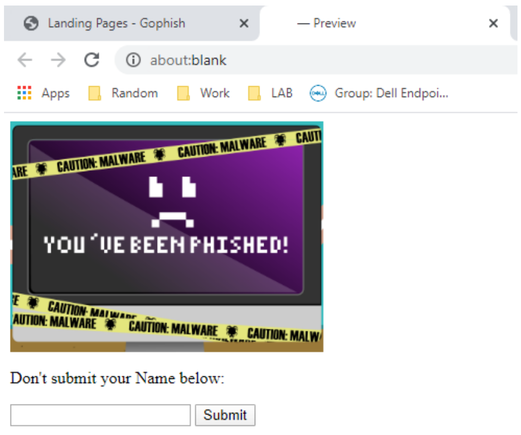

Name: Phished

Don't submit your Name below:

☑ Capture Submitted Data

Fig: *Landing Page editor*

Previewing it, our landing page seems legit.



Don't submit your Name below:

Fig: *Preview of the Landing page*

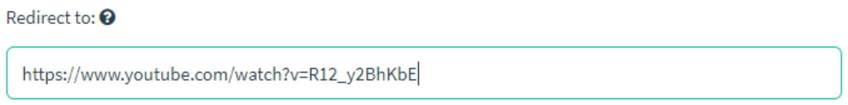
You can use simple HTML code in Source mode to create your own form. Here is what I used.

HTML Form

```
<p>Don&#39;t submit your Name below:</p>
<form action="" method="post"
name="form"><input name="username"
type="text"/> <input
type="submit"> </form>
```

__Tip__: Remember, the form method must be POST for gophish to capture any user submitted data.

The Landing page also gives us some advanced options such as the ability to capture passwords. Please note that Gophish at this time saves all data in cleartext, hence it's not such a good idea to capture passwords. You can also redirect users to another website once they have submitted the data. In practice, you would have a landing page that entices the user to submit credentials or other sensitive data and once submitted, redirect them to a page that coaches them. We will redirect users to a Youtube video on phishing scams [https://www.youtube.com/watch?v=R12_y2BhKbE]

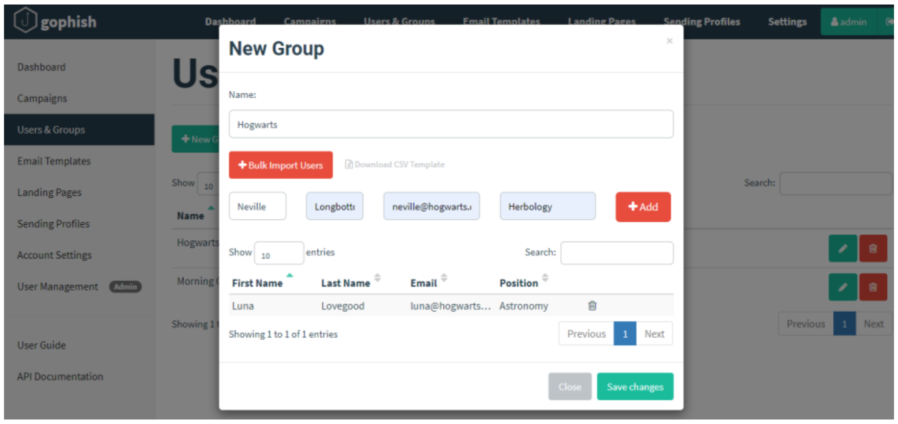


Redirect to: https://www.youtube.com/watch?v=R12_y2BhKbE

Go ahead and save the Landing page.

Adding Users & Groups

When you create a phishing campaign you target a group. A group is nothing but a collection of users such as Sales, or Marketing. Creating Users and Groups is quite straightforward in Gophish. In the navigation pane, click on **Users & Groups** -> Click on **New Group**. You can manually specify users with their Full Name, Email and Position or use the Bulk Import option to import a CSV containing this data. There's a handy CSV template you can use to format the data the way Gophish expects it. For this exercise, we're going to manually add users. Once added, click on **Save Changes**.



New Group

Name: Hogwarts

Neville | Longbott | neville@hogwarts. | Herbology

Show 10 entries Search:

First Name	Last Name	Email	Position
Luna	Lovegood	luna@hogwarts...	Astronomy

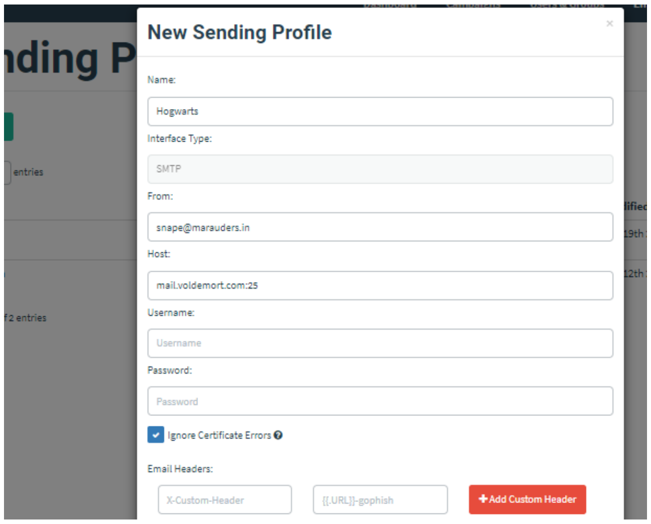
Showing 1 to 1 of 1 entries

Create a Sending Profile

A Sending Profile tells the gophish server how to send Phishing emails. As this element can be reused, you can have the same sending profile for multiple campaigns. You define:

- The SMTP Host – Who to Send? Use a `host/ip:port` to add an smtp host.
- The From User – email from whom?
- Username and Password (optional) to authenticate to the SMTP host

Gophish also supports adding custom headers in the phishing emails. You can use these headers to whitelist emails from gophish in your anti-spam appliance. You can set up something like `X-Gophish-Campaign <value = True>` which can be whitelisted in your anti-spam appliance. Headers are also help in differentiating your Gophish campaigns from actual phishing emails to an incident responder (provided you have disseminated which header you are going to use.) We will not be using headers for now, so go ahead and send a test email to verify if your configuration works.


New Sending Profile

Name: Hogwarts

Interface Type: SMTP

From: snape@marauders.in

Host: mail.voldemort.com:25

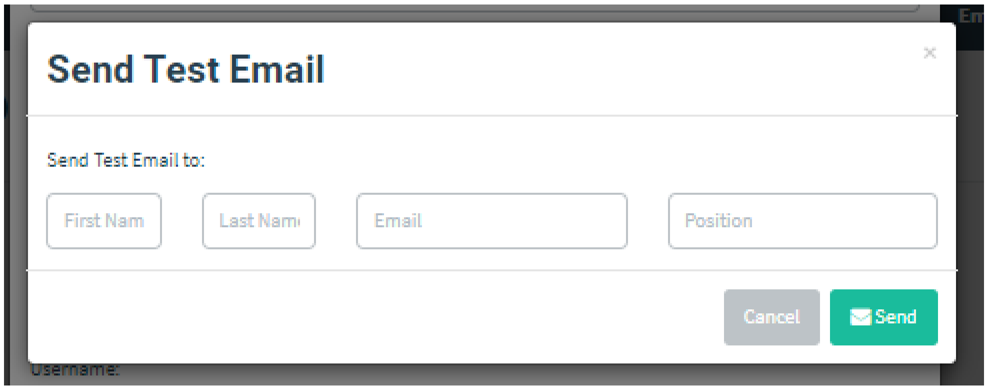
Username: Username

Password: Password

☑ Ignore Certificate Errors

Email Headers: X-Custom-Header {{.URL}}-gophish

Fig: *Sending Profile.*


Send Test Email

Send Test Email to:

First Nam | Last Nam | Email | Position

Fig: *Send a Test Email*

If your test is successful, go ahead and save. If your test fails, check the following:

- Is there a reachability problem? – Name resolution, IP connectivity, Port
- Is the information for the email server accurate? – hostname/ip, port, authentication credentials

- Is your test email being rejected by your mail server? – is the phishing domain whitelisted in your server, can it be resolved by your mail server.

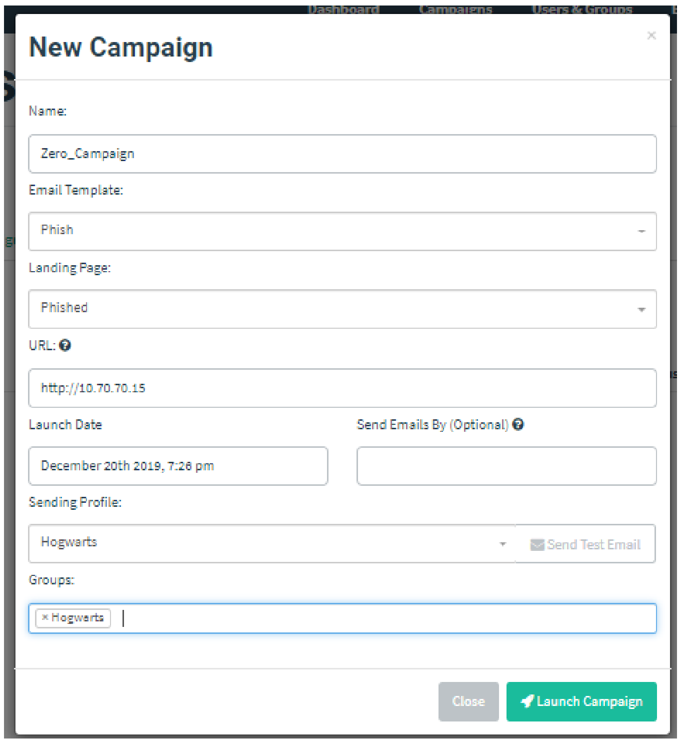
Bringing it together – the Campaign

Now that we have all our elements, it's time to bring them together and launch a campaign. As noted previously, a campaign is like a project where you select individual elements – Groups to target, Sending profiles, Email templates, Landing pages – and launch an active phishing campaign. The campaign actively tracks statuses such as User activity and provides reporting. Every campaign will generate a unique RID for each user targeted in the campaign, to enable tracking individual actions. If you need to learn more about RID, just flip over to the basics section where I go over all the elements of Gophish.

To launch a New Campaign, click on **Campaigns** -> New **Campaign**. The new campaign menu lets you select the individual elements that make a campaign:

- The Email Template to use
- A Landing Page
- A Sending Profile
- Target User Group(s)

Additionally you also need to specify the URL for the Landing page. If you remember when we created the Landing Page and the email template, we never specified an IP address or FQDN for the landing pages, but rather a variable. This allows our templates to be modular and reused while having the option to have a different FQDN serving the landing page. For using gophish to deliver the landing page, we will just specify the URL in this format – http://<ipaddress/fqdn> You also have options to schedule a launch by specifying a **launch date**, as well as a target end date by specifying a **Send all emails** by schedule. Here is our campaign build-out using all our elements:

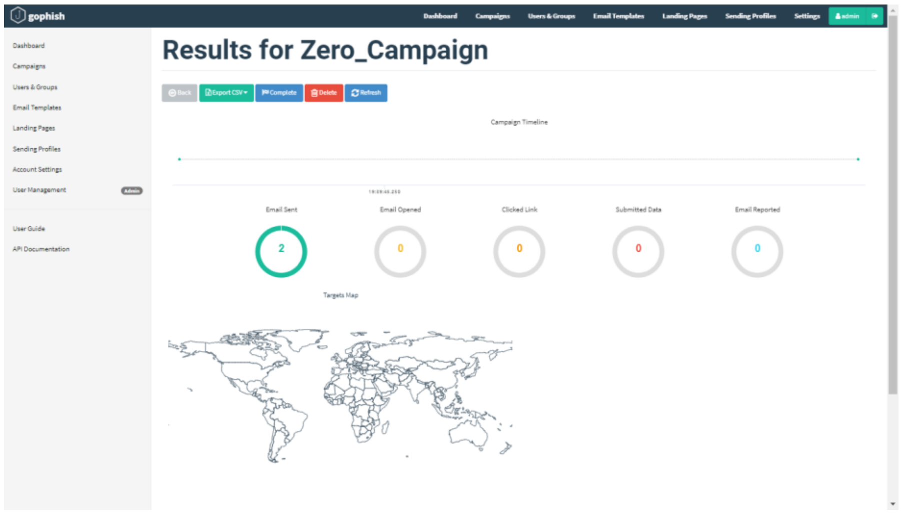


Name: Zero_Campaign
Email Template: Phish
Landing Page: Phished
URL: http://10.70.70.15
Launch Date: December 20th 2019, 7:26 pm
Send Emails By (Optional):
Sending Profile: Hogwarts
Groups: × Hogwarts

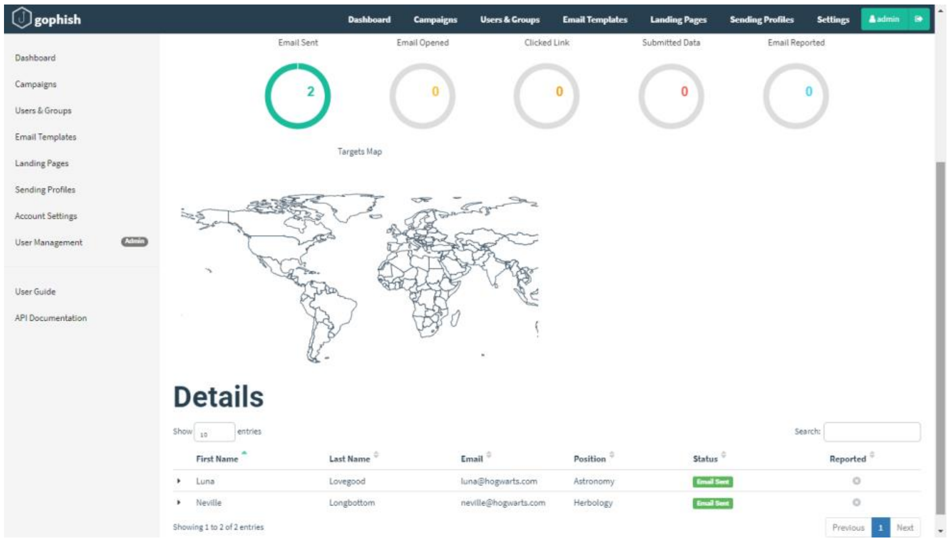
Fig: *New Campaign wizard.*

It’s time to launch our campaign and terrorize..err educate the masses! Click **Launch Campaign**!

You will be redirected to the campaign results page, showing you the current status of the campaign, which should ideally be at **Email Sent**. Since this is your first campaign, try to limit it to a couple of users for testing purposes.



Results for Zero_Campaign



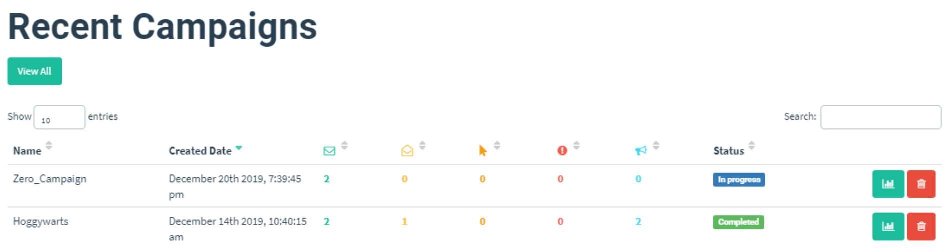
Details

Show 10 entries Search:

	First Name	Last Name	Email	Position	Status	Reported
▸	Luna	Lovegood	luna@hogwarts.com	Astronomy	Email Sent	○
▸	Neville	Longbottom	neville@hogwarts.com	Herbology	Email Sent	○

Showing 1 to 2 of 2 entries Previous 1 Next

You can always navigate to the campaign results via the **Dashboard** or **Campaigns** menu.



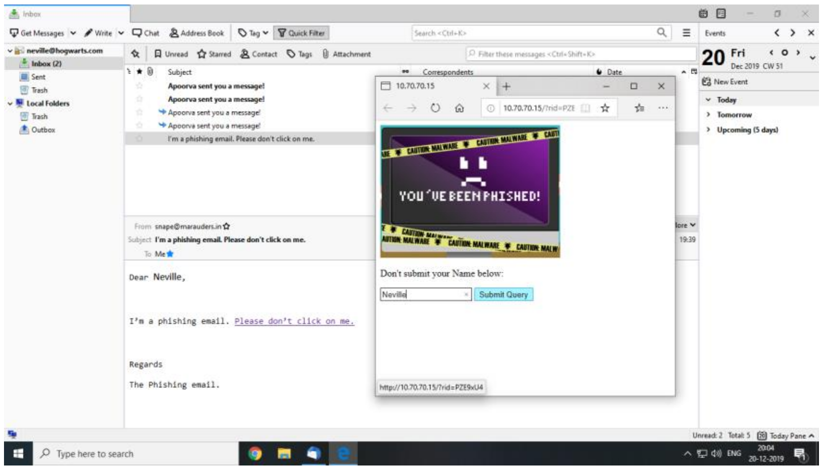
Recent Campaigns

View All

Show 10 entries Search:

Name	Created Date	✉	✉	➤	❗	📢	Status
Zero_Campaign	December 20th 2019, 7:39:45 pm	2	0	0	0	0	In progress
Hoggywarts	December 14th 2019, 10:40:15 am	2	1	0	0	2	Completed

Let’s get back to our campaign. We have emails sent out, and now we should be able to track activities. Mr.Longbottom after a hard day at the office [some mandrakes became unruly] has just opened his inbox and see’s our phishing mail. His curiosity is obviously piqued by a phishing email telling him it’s a phishing email. He opens it and clicks on the link, and just for fun submits his name to the landing page, thinking it to be a prank. (If he’d only read the books, he would know Snape was never in the marauders, but oh well..).


Apoorva sent you a message!
Apoorva sent you a message!
Apoorva sent you a message!
Apoorva sent you a message!
I'm a phishing email. Please don't click on me.

From snape@marauders.in
Subject I'm a phishing email. Please don't click on me.
To Me

Dear Neville,

I'm a phishing email. Please don't click on me.

Regards
The Phishing email.

YOU'VE BEEN PHISHED!

Don't submit your Name below:
Neville
Submit Query

http://10.70.70.15/?rid=PZE9xU4

Fig: *Opens email, clicks on the link, submit’s his name.*

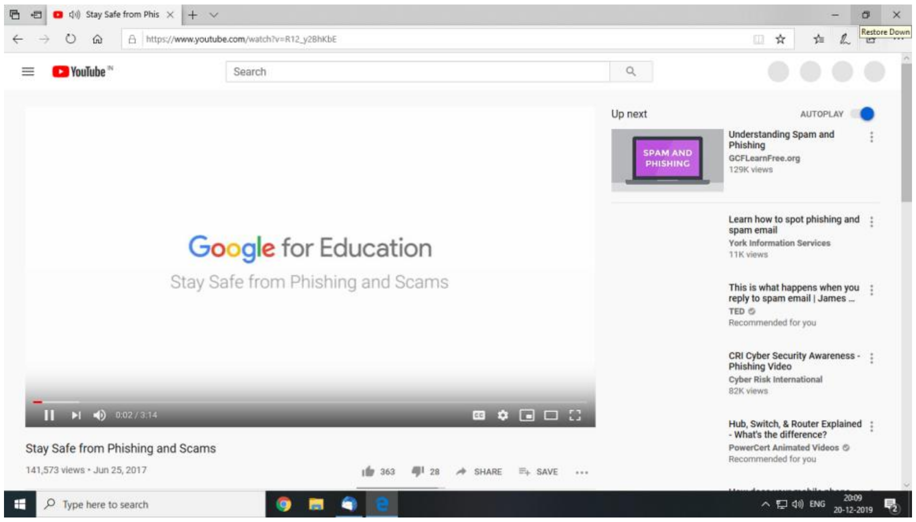

Stay Safe from Phis
https://www.youtube.com/watch?v=R12_y2BhKbE
Google for Education
Stay Safe from Phishing and Scams
Up next
Understanding Spam and Phishing
Learn how to spot phishing and spam email
This is what happens when you reply to spam email | James ...
CRI Cyber Security Awareness - Phishing Video
Hub, Switch, & Router Explained - What's the difference?
Stay Safe from Phishing and Scams
141,573 views • Jun 25, 2017

Fig: *Redirected to the Google Education video.*

The Campaign results allow us to accurately track the stages of a particular campaign. Not only does it track the user activity, but also timestamps it for verification. If a user clicks on a link multiple times, even that is captured. Additionally, the OS and browser version are captured.

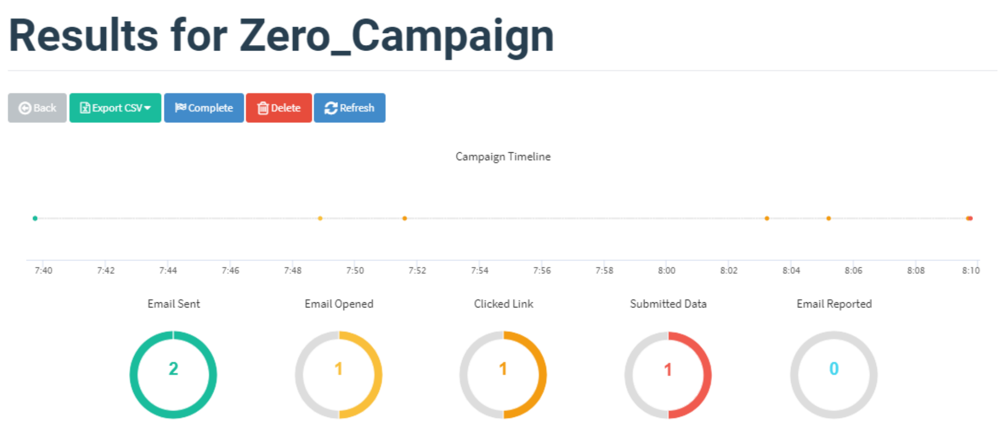
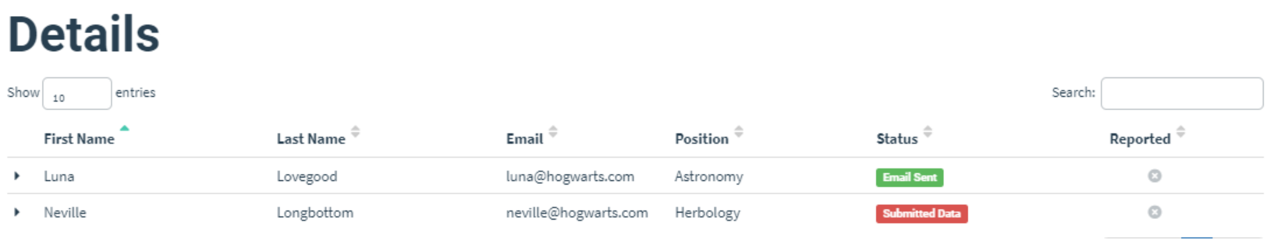
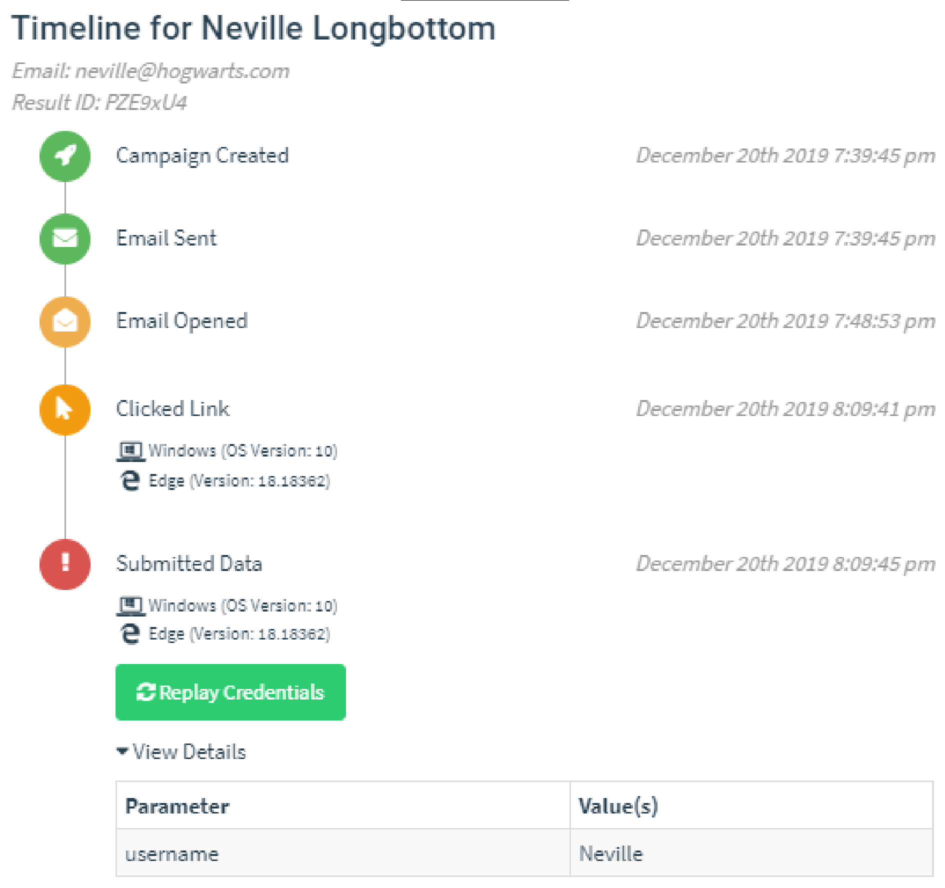

Results for Zero_Campaign

Fig: *Campaign Results – high level stats*


Details

	First Name	Last Name	Email	Position	Status	Reported
▸	Luna	Lovegood	luna@hogwarts.com	Astronomy	Email Sent	⊗
▸	Neville	Longbottom	neville@hogwarts.com	Herbology	Submitted Data	⊗

Fig: *User stats*


Timeline for Neville Longbottom

Email: neville@hogwarts.com

Result ID: PZE9xU4

Campaign Created — *December 20th 2019 7:39:45 pm*

Email Sent — *December 20th 2019 7:39:45 pm*

Email Opened — *December 20th 2019 7:48:53 pm*

Clicked Link — *December 20th 2019 8:09:41 pm*

Windows (OS Version: 10)

Edge (Version: 18.18362)

Submitted Data — *December 20th 2019 8:09:45 pm*

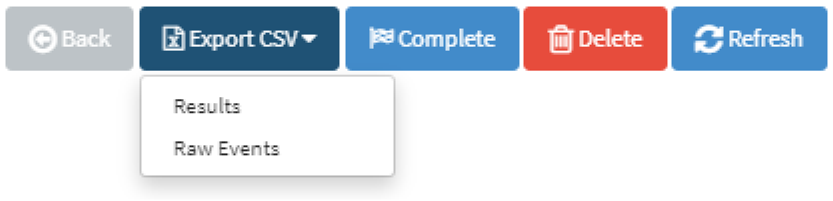
Windows (OS Version: 10)

Edge (Version: 18.18362)

Parameter	Value(s)
username	Neville

Fig: *Campaign tracking Neville's activities.*

Eureka! We have just completed our first phishing campaign. We can download results for our campaign to highlight the effectiveness of the campaign. Raw events can also be downloaded to track user activity timeline. Do remember to mark a Campaign as **Complete** to archive the data for the campaign. Archived Campaign data can be viewed under **Campaigns** -> **Archived Campaigns**.

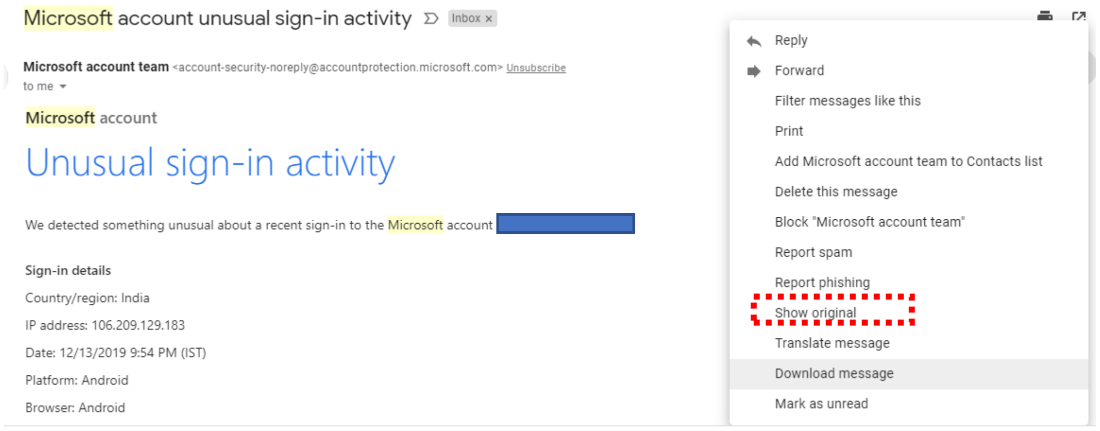


Templates

To make realistic campaigns, you will need to emulate actual real-life targets. For example, hosting a clone of your corporate website is a great idea to trick unsuspecting users. Adding on to it, a domain name playing on the actual domain name will make the illusion much more realistic. Gophish supports importing websites to emulate landing pages and importing emails to emulate real emails.

Importing Email Templates

To emulate an email, we need a template. Using real emails as the base really adds to the authenticity of the message. Gophish requires Raw Source code to import an email template. Gmail allows has a handy "Show Original" option in the mail menu to view the source code of a particular email.


Microsoft account unusual sign-in activity

Inbox ×

Microsoft account team <account-security-noreply@accountprotection.microsoft.com> Unsubscribe

to me

Microsoft account

Unusual sign-in activity

We detected something unusual about a recent sign-in to the Microsoft account [redacted]

Sign-in details

Country/region: India

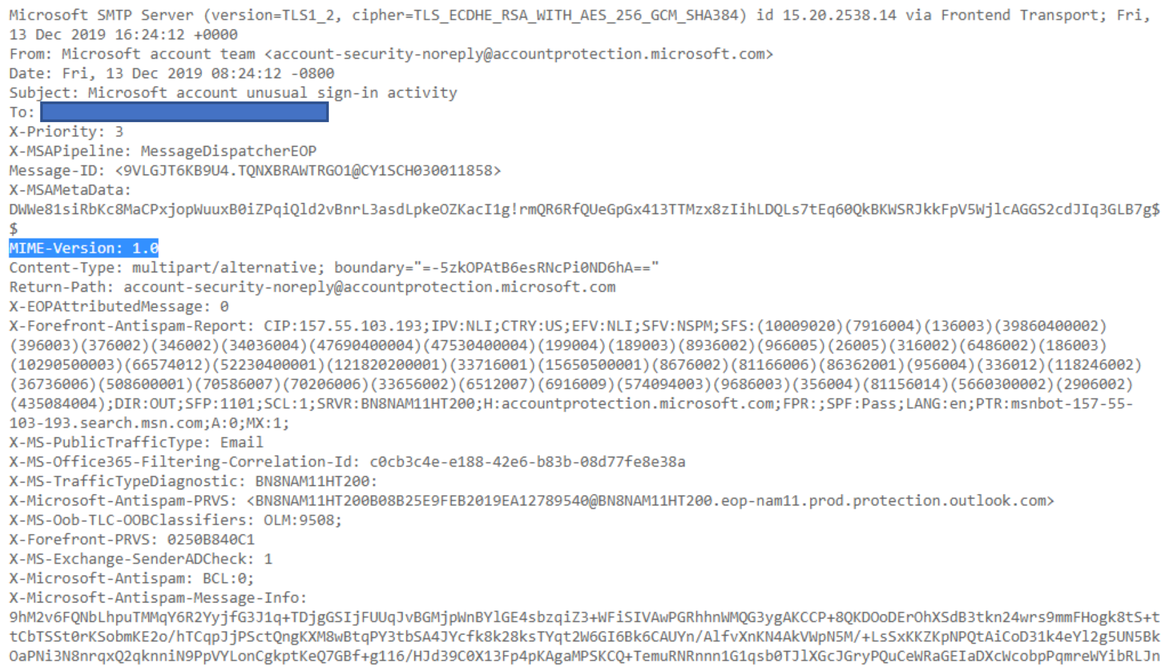
IP address: 106.209.129.183

Date: 12/13/2019 9:54 PM (IST)

Platform: Android

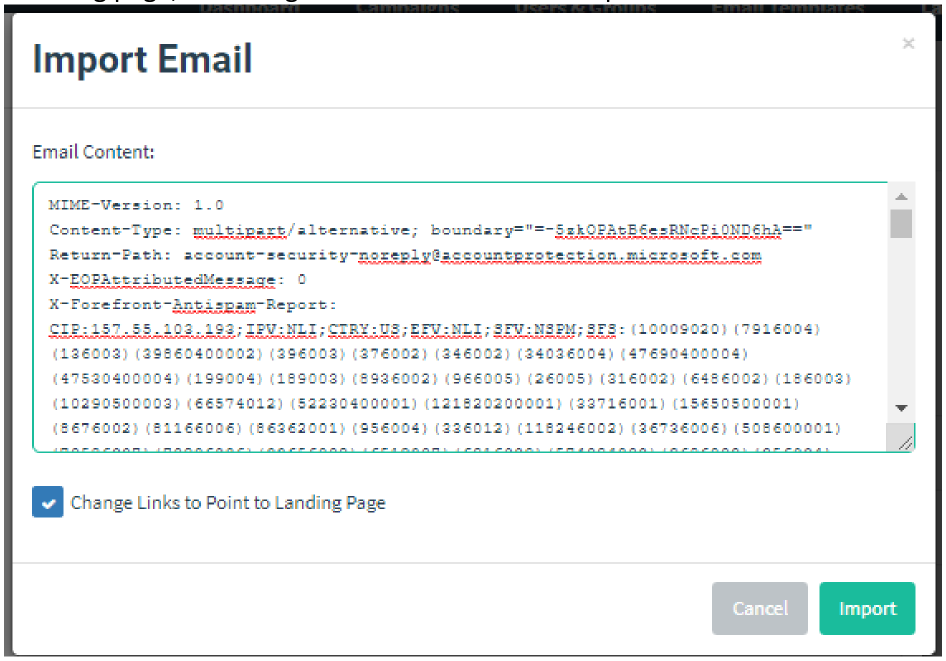
Browser: Android

A new tab with the raw email source will be opened. Gophish expects a complete email including headers. Since we will be setting up our own Subject & From addresses, we don't need to import them. A good place to start copying from is header: MIME-Version 1.0. Copy all the source code till the end of the message.



```
Microsoft SMTP Server (version=TLS1_2, cipher=TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384) id 15.20.2538.14 via Frontend Transport; Fri,
13 Dec 2019 16:24:12 +0000
From: Microsoft account team <account-security-noreply@accountprotection.microsoft.com>
Date: Fri, 13 Dec 2019 08:24:12 -0800
Subject: Microsoft account unusual sign-in activity
To: [redacted]
X-Priority: 3
X-MSAPipeline: MessageDispatcherEOP
Message-ID: <9VLGJT6KB9U4.TQNXBRAWTRGO1@CY1SCH030011858>
X-MSAMetaData:
DWWe81siRbKc8MaCPxjopWuuxB0iZPqiQld2vBnrL3asdLpkeOZKacI1g!rmQR6RfQUeGpGx413TTMzx8zIihLDQLs7tEq60QkBKWSRJkkFpV5WjlcAGGS2cdJIq3GLB7g$
$
MIME-Version: 1.0
Content-Type: multipart/alternative; boundary="=-5zkOPAtB6esRNcPi0ND6hA=="
Return-Path: account-security-noreply@accountprotection.microsoft.com
X-EOPAttributedMessage: 0
X-Forefront-Antispam-Report: CIP:157.55.103.193;IPV:NLI;CTRY:US;EFV:NLI;SFV:NSPM;SFS:(10009020)(7916004)(136003)(39860400002)
(396003)(376002)(346002)(34036004)(47690400004)(47530400004)(199004)(189003)(8936002)(966005)(26005)(316002)(6486002)(186003)
(10290500003)(66574012)(52230400001)(121820200001)(33716001)(15650500001)(8676002)(81166006)(86362001)(956004)(336012)(118246002)
(36736006)(508600001)(70586007)(70206006)(33656002)(6512007)(6916009)(574094003)(9686003)(356004)(81156014)(5660300002)(2906002)
(435084004);DIR:OUT;SFP:1101;SCL:1;SRVR:BN8NAM11HT200;H:accountprotection.microsoft.com;FPR:;SPF:Pass;LANG:en;PTR:msnbot-157-55-
103-193.search.msn.com;A:0;MX:1;
X-MS-PublicTrafficType: Email
X-MS-Office365-Filtering-Correlation-Id: c0cb3c4e-e188-42e6-b83b-08d77fe8e38a
X-MS-TrafficTypeDiagnostic: BN8NAM11HT200:
X-Microsoft-Antispam-PRVS: <BN8NAM11HT200B08B25E9FEB2019EA12789540@BN8NAM11HT200.eop-nam11.prod.protection.outlook.com>
X-MS-Oob-TLC-OOBClassifiers: OLM:9508;
X-Forefront-PRVS: 0250B840C1
X-MS-Exchange-SenderADCheck: 1
X-Microsoft-Antispam: BCL:0;
X-Microsoft-Antispam-Message-Info:
9hM2v6FQNbLhpuTMMqY6R2YyjfG3J1q+TDjgGSIjFUUqJvBGMjpWnBYlGE4sbzqiZ3+WFiSIVAwPGRhhnWMQG3ygAKCCP+8QKDOoDErOhXSdB3tkn24wrs9mmFHogk8tS+t
tCbTSSt0rKSobmKE2o/hTCqpJjPSctQngKXM8wBtqPY3tbSA4JYcfk8k28ksTYqt2W6GI6Bk6CAUYn/AlfvXnKN4AkVWpN5M/+LsSxKKZKpNPQtAiCoD31k4eYl2g5UN5Bk
OaPNi3N8nrqxQ2qknniN9PpVYLonCgkptKeQ7GBf+g116/HJd39C0X13Fp4pKAgaMPSKCQ+TemuRNRnnn1G1qsb0TJlXGcJGryPQuCeWRaGEIaDXcWcobpPqmreWYibRLJn
```

In Gophish, go to Email Templates -> New Template, add a Name and click on Import Email and paste the source code in the form. Notice that Gophish can automatically convert any links found in the email message to the landing page, removing manual work. Click on Import.



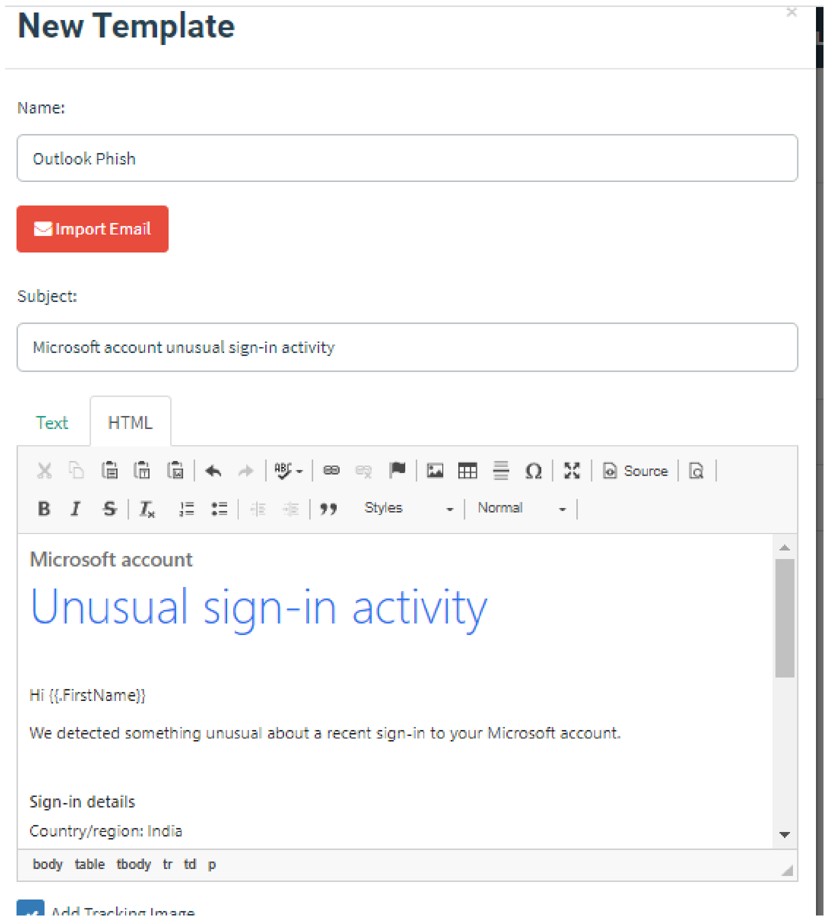
Import Email

Email Content:

```
MIME-Version: 1.0
Content-Type: multipart/alternative; boundary="=-5zkOPAtB6esRNcPi0ND6hA=="
Return-Path: account-security-noreply@accountprotection.microsoft.com
X-EOPAttributedMessage: 0
X-Forefront-Antispam-Report:
CIP:157.55.103.193;IPV:NLI;CTRY:US;EFV:NLI;SFV:NSPM;SFS:(10009020)(7916004)
(136003)(39860400002)(396003)(376002)(346002)(34036004)(47690400004)
(47530400004)(199004)(189003)(8936002)(966005)(26005)(316002)(6486002)(186003)
(10290500003)(66574012)(52230400001)(121820200001)(33716001)(15650500001)
(8676002)(81166006)(86362001)(956004)(336012)(118246002)(36736006)(508600001)
```

✓ Change Links to Point to Landing Page

We now have our spoofed email template to use. You can customize it, to make it more personal – such as saluting the user via their First name with variables. Once done, save the template and start using it in your campaigns!



New Template

Name:

Outlook Phish

Subject:

Microsoft account unusual sign-in activity

Microsoft account

Unusual sign-in activity

Hi {{.FirstName}}

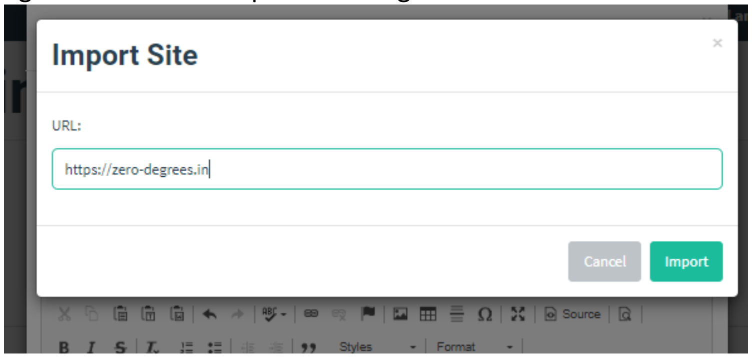
We detected something unusual about a recent sign-in to your Microsoft account.

Sign-in details

Country/region: India

Importing Landing Page Templates

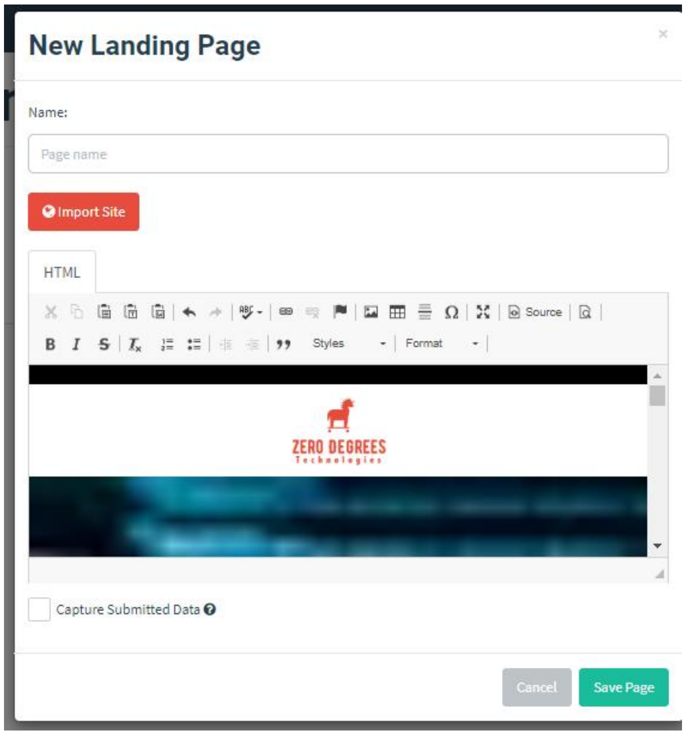
Similarly, you can import landing pages. Go to **Landing Pages -> New Page -> Import Site**. Enter the URL for Gophish to clone the page. Here is an example of cloning a website.



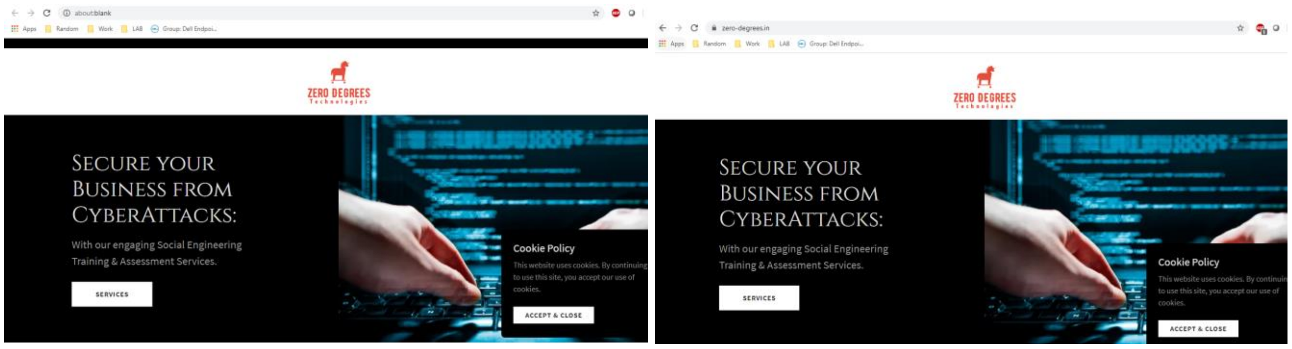
Import Site

URL:

https://zero-degrees.in



Comparing the cloned website with the original website, we can see the clone resembles the original website.


ZERO DEGREES
SECURE YOUR BUSINESS FROM CYBERATTACKS:
With our engaging Social Engineering Training & Assessment Services.
SERVICES
Cookie Policy
ACCEPT & CLOSE

Couple of caveats to remember before using this feature. Any links are not transformed to reflect Gophish addresses. This means that the page will behave as the original page and redirect to the original links when clicked. Again, as this is the landing page, we really want the user to think of this as the original website but also want to entice the user to submit credentials or data to YOU. So a good idea is to clone a form based web page – such as a login. Importing complex web pages can also result in broken page displays with images not rendering properly. For a corporate phishing campaign, a good idea is to run an HR campaign – asking the user to update details on the HRMS or an IT campaign asking a user to reset a password. Internal web pages can be cloned and presented. For external websites, a good idea would be to import assets such as the styling and logos, and replace the form with your own custom form similar to the one we used in our campaign setup. This way you have control over the elements and data requests presented to the user.

While the typical ideas such as Facebook password reset work, you will need to evolve your phishing campaigns to be more sophisticated. Based on personal experience, you could look at the following scenarios to run phishing campaigns:

- **Application Password Reset**: Campaign that targets normal users or operational users of an internal application such as SAP, and asks for password reset. Email styling should match IT mails but sender email should be an external domain as a clue for users.
- **HR Process**: Campaign that targets normal users or operations users to update details or review KRA in HR Tools, with form to capture submitted data. Email styling and landing page should match HR and HRMS Tool, but sender email should be a typosquatting domain or similar to corporate domain (think example.co for example.com) as a clue for users.
- **Potential Buyer**: Campaign that targets sales team. Very easy to do – dangle a potential requirement and they become click happy – I have a requirement of <xxxx>, please check the google spreadsheet and advice. The landing page has a google logo and a form for asking some data, but doesn't really resemble a google drive page. The email address could be from a yahoo or gmail.
- **Higher Up**: Campaign that targets secretaries or accounts personnel. Email emulates a higher authority such as a CEO or CFO asking for urgent funds to be transferred to a personal account (via our phish link) or ask to update some information on a website for an upcoming compliance requirement or registration. Do not match email styling and phish should be from a similar email address on your corporate domain such as achalke@example.com or ashishchalke@example.in for ashishchalke@example.com. The campaigns idea is to give enough visual cues that scream it is a phish, but invoke the users curiosity to click the link. Please do take the relevant stakeholder into confidence before launching such a campaign to prevent inadvertent fallout.
- **Transactional Email**: Campaign that targets Accounts teams emulating a regular 3rd party but asking for updating account details in their new automated payment system accessed via our phish link. Landing page hosts some sensitive form data to capture. Email styling should match the 3rd party but should be from a typosquatting domain as a clue for users.
- **Sensitive Update**: Campaign that targets IT personnel with a email informing about a critical update for a device or application with a link to the release notes. Added bonus for landing page to capture credentials.

Do remember to incorporate these clues in your Security Awareness trainings so that employees are educated to lookout for these in emails. Ideally your training should cover the following aspects for end-users to detect phishing emails:

- **Sender Email address** – verify the domain. Is the password reset email really from facebook?
- **Visual Cues** – does the style and language match the regular emails from this entity.
- **Verify links before clicking** – they could contain malware or worse still, steal your credentials. Highlight tools such as urlscan.io that the curious can use to verify suspicious links.
- **Typosquatting** – Educate on identifying typosquatting. Disseminate information on corporate domains used for emails. Any emails looking similar but other than the corporate domains should be treated with high suspicion.
- **URL Verification** – Does the URL in the browser match the domain? Does the TLS Certificate match?
- **Sensitive Data** – Educate on verifying before submitting any sensitive data to a link. Educate on what data should not be submitted on any website.
- **Out of band verification** – For sensitive transactions, verify out of band such as phone with recipient. I have seen multiple instances at small or medium sized businesses which work with

offshore suppliers, where the target has tried to verify via email before transferring money and the attacker having access to the source's email account has replied back confirming the same. Imagine if an attacker has access to the CEO's email account, asking a secretary to transfer x dollars to a private account and the secretary tries to verify via a new email only for the attacker having put in an email rule to send emails from the secretary to a folder such as Notes and replying in the affirmative… yes this was a real event.

- **Report** – More on this in the next section, but setup processes to enable users to report suspicious emails that have passed through your layers of defenses.
- **Use Technology for Sensitive Transactions** – work with 3rd parties to adapt email signing or encryption if sensitive transactions are being carried out over email.

Make quick reference posters that cover highlights of this information and paste where users can notice and action. These posters should ideally also have the Reporting procedure outlined for users.

Finally, to create better campaigns the web is your friend. Lots of phishing emails and landing pages are available ready to use with few modifications, If you need inspiration or something quick to use, check out these resources:

- Criggs626 Phishing Templates [https://github.com/criggs626/PhishingTemplates] has email and landing page templates for gophish.
- Phishing Campaign Ideas [https://github.com/gophish/gophish/issues/951]
- Cooper [https://github.com/chrismaddalena/Cooper] a Python script to clone emails and websites to templates, and convert links.

Phish Reporting

While phishing our users and understanding how many of them could be phished is great, telling your boss that you've verified that 40% users in your organization click on phishing emails and 20% ignore mails completely, doesn't really add value to the business. What would really add value would be understanding how many users report phishing emails, and then coach the ones that do not. Adding in gamification, such as small rewards program for users who successfully report your phishing tests would go a long way in securing your organization by motivating others to do the same and adding the user as another layer to your defenses.

Reporting is highly desirable but can be complicated to achieve. Most commercial tools include plugins for Outlook for users to report emails, but users today access email in a variety of ways – desktop clients [outlook, thunderbird], browsers, mobile clients – and there really is no solution that covers all these clients. As mentioned in this book previously, Gophish does have Reporting capabilities built-in but only the server-side components to support the reporting. The client-side functionality is yet being conceptualized.

Whenever gophish send's emails as part of a campaign, it uniquely tracks each email-user-campaign relationship via a unique RID that is embedded in the email – http://landingpage/?rid=**abc756**. To report a phishing email on behalf of the user, we modify the url to add the report keyword - http://landingpage/report/?rid=**abc756**. This method doesn't scale well when you have a 100 users wanting to report suspicious mails.

Thankfully, gophish comes with REST & Pythonic API's that allow us to develop our own extensions. There are a lot of projects by users that allow you to report on behalf of the user either via an external script or modifications of the webpages. However most of them require you as admin to fetch the RID manually – the RID that tracks a user's activity for a particular campaign and is unique. I felt the need to have something automated that could automatically fetch a user's RID for a particular campaign and report.

I developed GoEmailReporter (in hindsight it should have been GoPHISHReporter) as a project while learning Python to do exactly this – a simple script where I specify a campaign and a user and the script does the background reporting for me, and then I went further. As of the writing of this book, the GoEmailReporter script [https://github.com/theashishchalke/gophish-tools/wiki] enables you to report on behalf of the user in 3 ways:

- Manually enter a user email address to report.
- Import a list of email addresses
- Connect to an imap mailbox and report

In an enterprise scenario, the last one is the most convenient and impactful. We setup a mailbox that will be exclusively used by users to report phishing emails. We then coach our users [email blasts, training, landing pages] to forward suspected phishing emails to that mailbox and then use our script or manual methods to report. Remembering email addresses is hard, but making it easy to remember such as phish@domain.com and then repeated coaching & awareness (posters anyone?), with gamification should help. Ideally, we should have buttons and plugins for all clients but unless someone develops that, the above approach is the best. If you are interested in developing a plugin, have a look at NotifySecurity [https://github.com/certsocietegenerale/NotifySecurity] by the excellent folks at Societe Generale for their Swordphish solution as an excellent starting point.

Getting back to gophish, so we've setup a phish-report mailbox and coached our users to forward suspicious emails. You've launched a campaign and the emails starts pouring in. Now on to reporting those emails! GoEmailReporter has the following workflow available to admins to report phishing emails on behalf of the user:

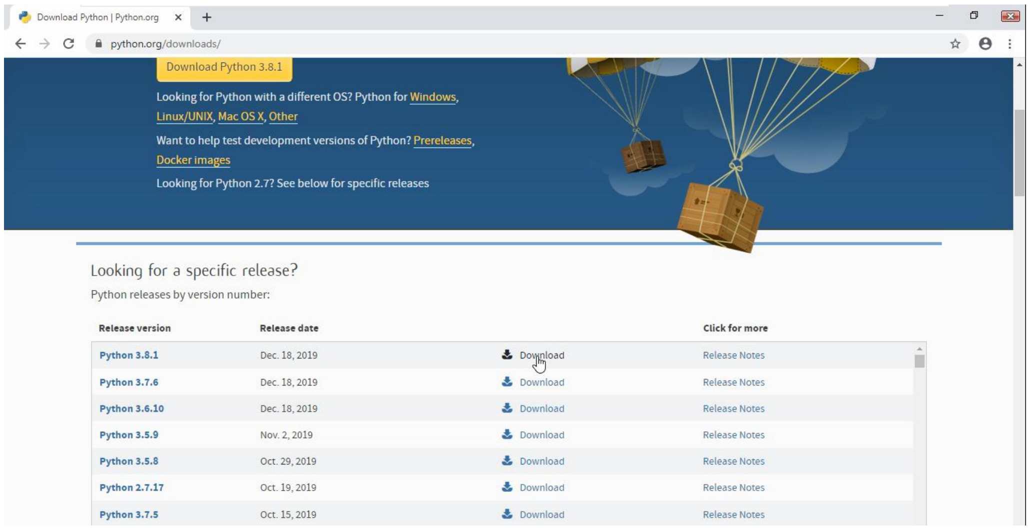
1. Users forward suspicious phishing emails to the phish-report mailbox.
2. Admin launches the GoEmailReporter python script.
3. The script pulls a list of all available campaigns and displays them with the name and campaign id.
4. **Asks** the admin for the campaign name to pull the rID. [Admin input]
5. Admin enters campaign name.
6. Script uses the entered Campaign name to compare campaign ID and pulls JSON data of all users, rid attached to that campaign via the RESTAPI
7. The Results section is parsed to a dictionary with UserEmail:rID format.
8. **Asks** the admin for a choice:
    a. **Manually Enter an Email Address**
    b. **Import a file containing a list of email addresses**
    c. **Read emails from a mailbox and report automatically**
9. If admin chose option **a - Manually enter the address :**
    1. **Asks** the admin for the Users email address for whom Email Reporting has to be done.
    2. Admin enters the email address.
    3. Script pulls the RID against the email address and stores in a variable.
    4. **Asks** the admin whether they want to Report - Y or N
    5. If Y, Constructs the report URL with the RID fetched earlier and makes request. If N, exits.
10. If admin chose option **b - Import address from file :**
    1. **Asks** the admin to enter a file name, can be either txt or csv. Should be in the same directory.
    2. Uses a function called fileimport [imported via `fileimport.py`] to extract email addresses to a list with csv delimiters.
    3. Main script takes extracted emails from list and imports only unique emails to a new python list.
    4. In a `for loop` for every email address in the list, constructs an http Report query by getting their RID from the dictionary and makes the request to the constructed url aka http://landingpage/report/?rid=**abc756**
11. If admin chose option **c - Connect to email account and pull :**
    1. **Asks** the admin to enter a subject to search. Should be as specific as possible, partial searches allowed. Double quotes [""] to enclose sentences.
    2. Imports mailbox user, password and hostname from `config.py` file and makes an imap connection.
    3. Searches for mails in the inbox which are UNREAD/UNSEEN and match the Subject specified by the admin.
    4. If mails are matched, extracts email addresses in the from field to a list, removes duplicates and constructs an http Report query by getting the RID from the dictionary and makes the request to the constructed url.
    5. Marks the message as READ/SEEN so as not to report the same ID again from the forwarded emails.

Installing GoEmailReporter

GoEmailReporter is available in two versions – a python script or a Jupyter notebook. We'll first take a look at the script variant. You can install GoEmailReporter in 3 steps:

- Download and Install Python. This script was coded on 3.7.4 but if you are on 3.6.x or above you should be good.
- Install Gophish library.
- Download the script and configure for your environment.

First download and install Python in PATH. Installing Python in the PATH environment enables us to use python as a command action to initiate python scripts or one liners, such as python script.py. Go to www.python.org/downloads and download the Python release for your environment. You should be fine with anything after 3.6.



Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Prereleases, Docker images

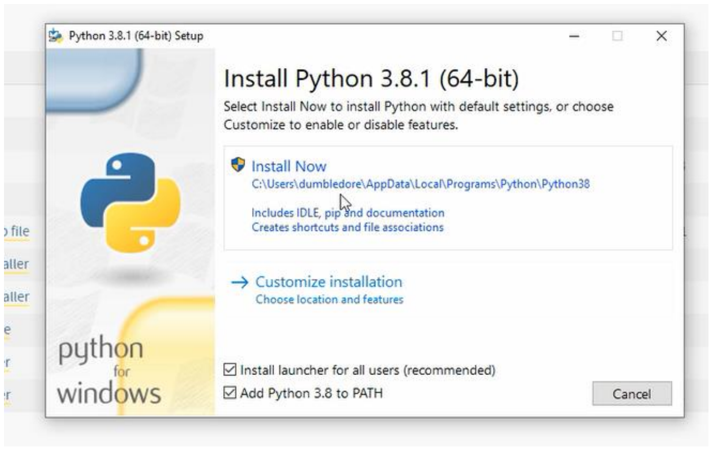
Looking for Python 2.7? See below for specific releases

Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.8.1	Dec. 18, 2019	Download	Release Notes
Python 3.7.6	Dec. 18, 2019	Download	Release Notes
Python 3.6.10	Dec. 18, 2019	Download	Release Notes
Python 3.5.9	Nov. 2, 2019	Download	Release Notes
Python 3.5.8	Oct. 29, 2019	Download	Release Notes
Python 2.7.17	Oct. 19, 2019	Download	Release Notes
Python 3.7.5	Oct. 15, 2019	Download	Release Notes

Download and execute the appropriate installer x86 for 32bit / x64 for 64bit, remember to select Add Python 3.8 to PATH. You will be asked for Administrator permissions to install. [no worries, you are the admin anyway 😉] You can optionally also disable the MAX_PATH limit when setup completes to use super long directory names or paths. :



Install Python 3.8.1 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

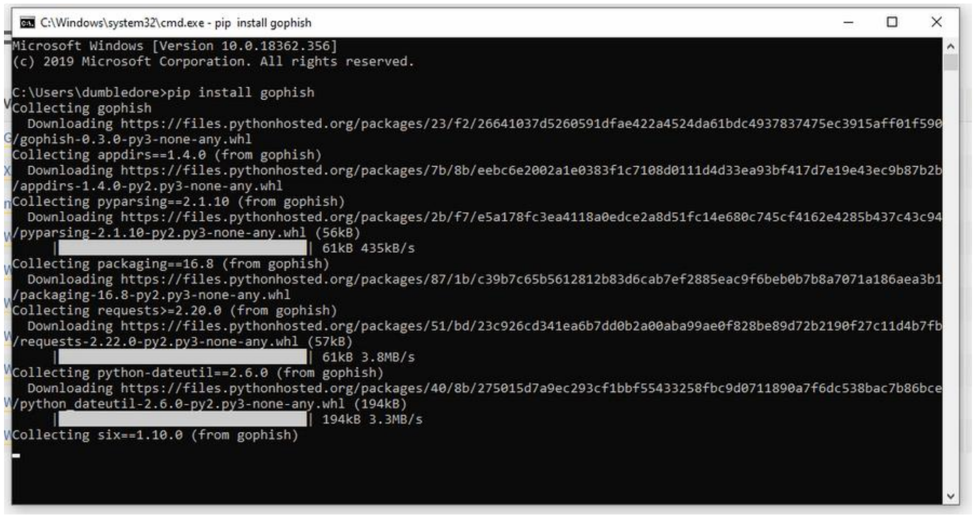
Install Now
C:\Users\dumbledore\AppData\Local\Programs\Python\Python38

Includes IDLE, pip and documentation
Creates shortcuts and file associations

Customize installation
Choose location and features

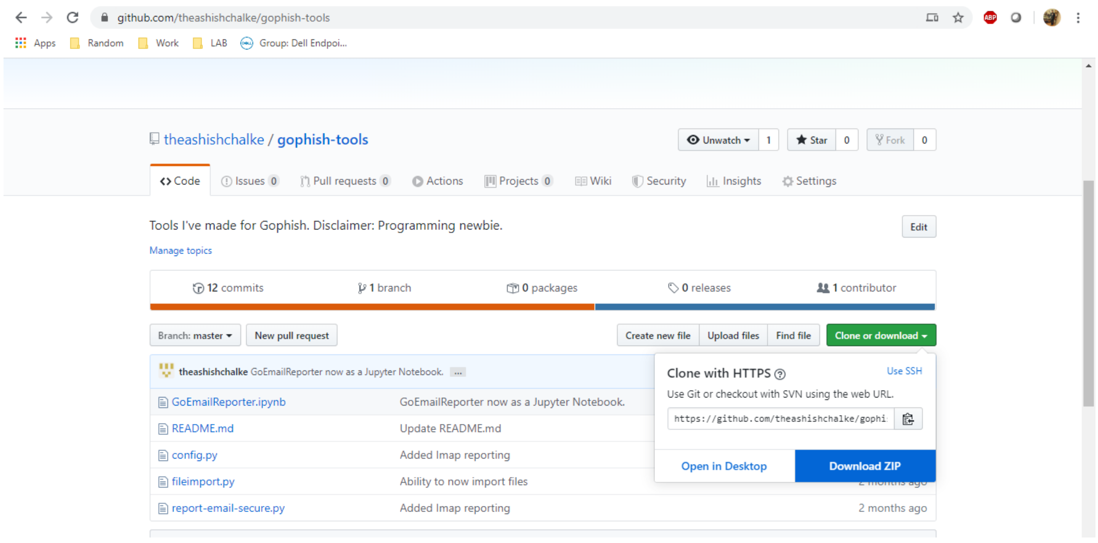
☑ Install launcher for all users (recommended)
☑ Add Python 3.8 to PATH

Next install the gophish library in python. Open a command prompt [as admin], and type `pip install gophish`. If you've successfully installed python, this should automatically download and install the gophish python library.



```
C:\Windows\system32\cmd.exe - pip  install gophish
Microsoft Windows [Version 10.0.18362.356]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\dumbledore>pip install gophish
Collecting gophish
  Downloading https://files.pythonhosted.org/packages/23/f2/26641037d5260591dfae422a4524da61bdc4937837475ec3915aff01f590
/gophish-0.3.0-py3-none-any.whl
Collecting appdirs==1.4.0 (from gophish)
  Downloading https://files.pythonhosted.org/packages/7b/8b/eebc6e2002a1e0383f1c7108d0111d4d33ea93bf417d7e19e43ec9b87b2b
/appdirs-1.4.0-py2.py3-none-any.whl
Collecting pyparsing==2.1.10 (from gophish)
  Downloading https://files.pythonhosted.org/packages/2b/f7/e5a178fc3ea4118a0edce2a8d51fc14e680c745cf4162e4285b437c43c94
/pyparsing-2.1.10-py2.py3-none-any.whl (56kB)
     |████████████████████████████████| 61kB 435kB/s
Collecting packaging==16.8 (from gophish)
  Downloading https://files.pythonhosted.org/packages/87/1b/c39b7c65b5612812b83d6cab7ef2885eac9f6beb0b7b8a7071a186aea3b1
/packaging-16.8-py2.py3-none-any.whl
Collecting requests>=2.20.0 (from gophish)
  Downloading https://files.pythonhosted.org/packages/51/bd/23c926cd341ea6b7dd0b2a00aba99ae0f828be89d72b2190f27c11d4b7fb
/requests-2.22.0-py2.py3-none-any.whl (57kB)
     |████████████████████████████████| 61kB 3.8MB/s
Collecting python-dateutil==2.6.0 (from gophish)
  Downloading https://files.pythonhosted.org/packages/40/8b/275015d7a9ec293cf1bbf55433258fbc9d0711890a7f6dc538bac7b86bce
/python_dateutil-2.6.0-py2.py3-none-any.whl (194kB)
     |████████████████████████████████| 194kB 3.3MB/s
Collecting six==1.10.0 (from gophish)
_
```

Finally download the GoEmailReporter scripts and place them in a handy directory to execute. Navigate to the Github repo [https://github.com/theashishchalke/gophish-tools] and click on Clone or Download -> Download as ZIP.



Tools I've made for Gophish. Disclaimer: Programming newbie.

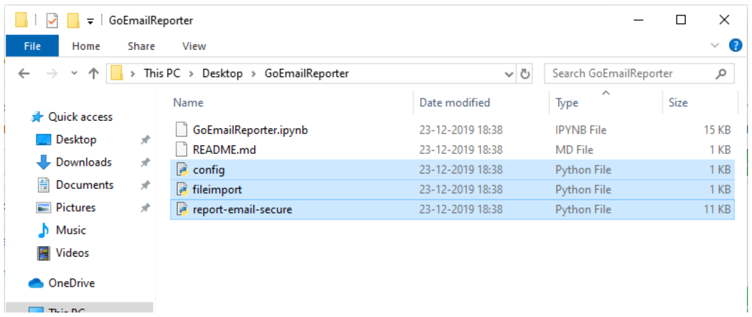
theashishchalke GoEmailReporter now as a Jupyter Notebook.

GoEmailReporter.ipynb	GoEmailReporter now as a Jupyter Notebook.	
README.md	Update README.md	
config.py	Added Imap reporting	
fileimport.py	Ability to now import files	2 months ago
report-email-secure.py	Added Imap reporting	2 months ago

Clone with HTTPS
Use Git or checkout with SVN using the web URL.
https://github.com/theashishchalke/gophi

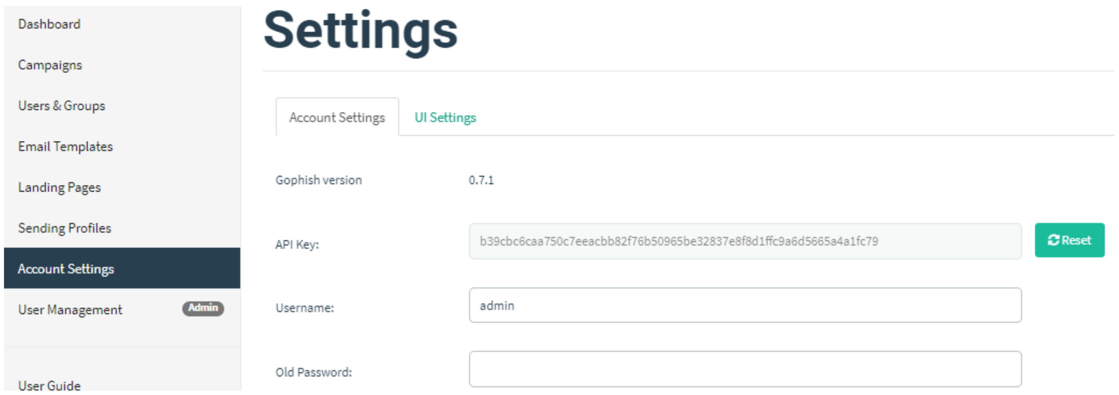
Download and extract this ZIP file to a convenient directory. This should give us the below 3 files. Ignore the .ipynb file for now.

- **config.py** – This file is used to set the API Key for Gophish, as well as the mailbox credentials if you wish to use option 3.
- **fileimport.py** – This is a function file used to hold the fileimport function. This code is not mine, but one I modified from the original authors – [http://csestack.org/python-extract-emails-read-file/]
- **report-email-secure.py** – is the actual script and what we will execute to run GoEmailReporter.



Name	Date modified	Type	Size
GoEmailReporter.ipynb	23-12-2019 18:38	IPYNB File	15 KB
README.md	23-12-2019 18:38	MD File	1 KB
config	23-12-2019 18:38	Python File	1 KB
fileimport	23-12-2019 18:38	Python File	1 KB
report-email-secure	23-12-2019 18:38	Python File	11 KB

But before we run the script, we need to populate the configuration file. We need the api key. To get the API key, navigate to the **Gophish Mgmt console -> Account Settings**. The API key should be visible under API Key.



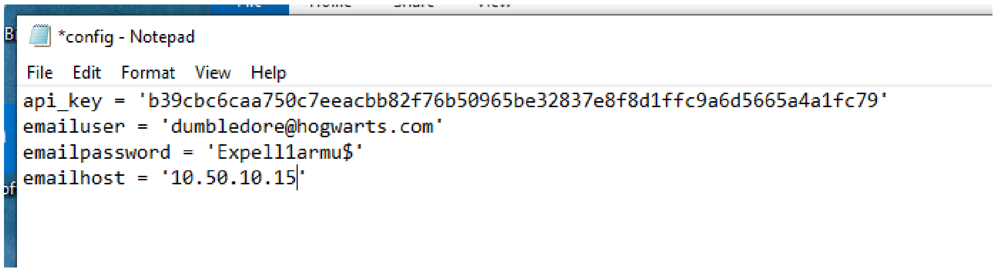
Gophish version 0.7.1

API Key: b39cbc6caa750c7eeacbb82f76b50965be32837e8f8d1ffc9a6d5665a4a1fc79

Username: admin

Old Password:

Copy and paste this in the **config.py** file we downloaded earlier [Use an editor such as Notepad], and enter the mailbox credentials and save the file. The final file should be similar to the below screenshot:

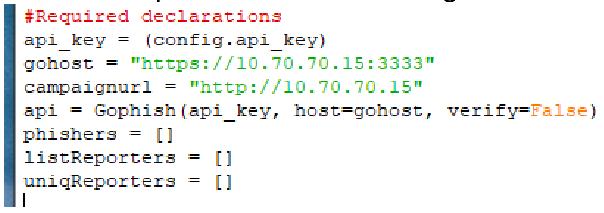


```
api_key = 'b39cbc6caa750c7eeacbb82f76b50965be32837e8f8d1ffc9a6d5665a4a1fc79'
emailuser = 'dumbledore@hogwarts.com'
emailpassword = 'Expell1armu$'
emailhost = '10.50.10.15'
```

Next, edit the report-secure-email.py script and modify the below variables under **#Required declarations** for your environment:

- **gohost** – Gophish Admin URL in https://ip-host:port format
- **campaignurl** – Gophish landing page url in http://ip-host format

The final script should look something like the below. Save it:



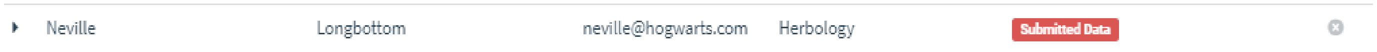
```
#Required declarations
api_key = (config.api_key)
gohost = "https://10.70.70.15:3333"
campaignurl = "http://10.70.70.15"
api = Gophish(api_key, host=gohost, verify=False)
phishers = []
listReporters = []
uniqReporters = []
```

Alright! We're now set to run GoEmailReporter script version.

Reporting Phish with GoEmailReporter

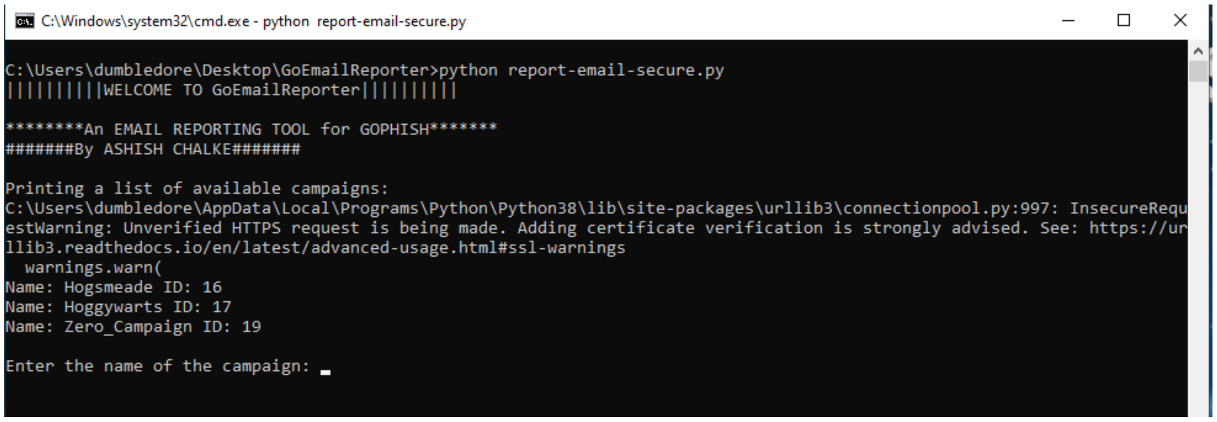
We'll use two ways of reporting with the script – manual entry and mailbox connect. Let's do a simple manual entry. The story so far – Neville, after a hard day at the office, got hooked – he opened our phishing email, clicked on the link and even went so far as to submit data to the phishing website. Upon going through the coaching video, he called IT and reported that he was phished – the right thing to do. That's +1 to IT as not only did they find out how bored Neville could be, but they've trained him to be more aware in future and always report suspicious emails to IT. IT needs to update the metrics to tell gophish that Neville has reported the email by using… you guessed it – GoEmailReporter! [Really rolls off the tongue doesn't it?]

Before we proceed, let's verify the campaign results once. Click on Campaigns -> and click on the Results icon next to the campaign name. Neville's status shows Submitted data and has not been reported [the gray x]. Let's Report.



Neville	Longbottom	neville@hogwarts.com	Herbology	Submitted Data	✗

Open a command prompt, and navigate to the folder where we extracted the zip file previously – `cd SOMEDRIVE:\SOMEPATH\FOLDER` and run `python report-secure-email.py.` The Welcome message should display.

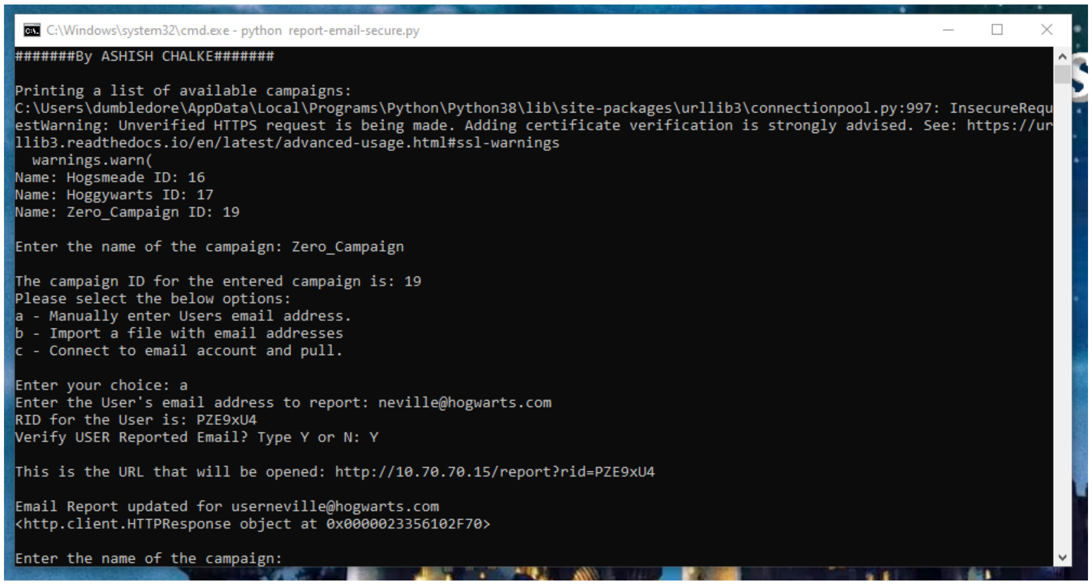


C:\Windows\system32\cmd.exe - python report-email-secure.py

```
C:\Users\dumbledore\Desktop\GoEmailReporter>python report-email-secure.py
|||||||||WELCOME TO GoEmailReporter|||||||||

********An EMAIL REPORTING TOOL for GOPHISH*******
#######By ASHISH CHALKE#######

Printing a list of available campaigns:
C:\Users\dumbledore\AppData\Local\Programs\Python\Python38\lib\site-packages\urllib3\connectionpool.py:997: InsecureRequ
estWarning: Unverified HTTPS request is being made. Adding certificate verification is strongly advised. See: https://ur
llib3.readthedocs.io/en/latest/advanced-usage.html#ssl-warnings
  warnings.warn(
Name: Hogsmeade ID: 16
Name: Hoggywarts ID: 17
Name: Zero_Campaign ID: 19

Enter the name of the campaign: _
```

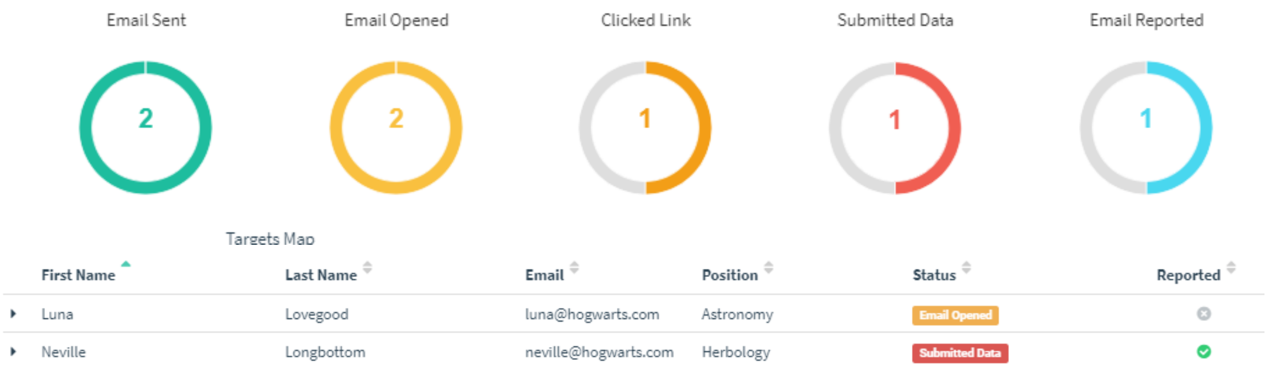
Follow the wizard to report.

- The script displays the list of campaigns available. Enter the **name** of the campaign to select it. Remember to enter the name EXACTLY as displayed.
- Enter **a** to select the Manual option to enter users email address.
- Enter the users **email address** to report.
- GoEmailReporter will **display** a RID if the user was part of the campaign and asks you once more to confirm. Enter **Y**.
- The final reporting url is confirmed and the Report is updated for the user.



```
#######By ASHISH CHALKE#######

Printing a list of available campaigns:
C:\Users\dumbledore\AppData\Local\Programs\Python\Python38\lib\site-packages\urllib3\connectionpool.py:997: InsecureRequ
estWarning: Unverified HTTPS request is being made. Adding certificate verification is strongly advised. See: https://ur
llib3.readthedocs.io/en/latest/advanced-usage.html#ssl-warnings
  warnings.warn(
Name: Hogsmeade ID: 16
Name: Hoggywarts ID: 17
Name: Zero_Campaign ID: 19

Enter the name of the campaign: Zero_Campaign

The campaign ID for the entered campaign is: 19
Please select the below options:
a - Manually enter Users email address.
b - Import a file with email addresses
c - Connect to email account and pull.

Enter your choice: a
Enter the User's email address to report: neville@hogwarts.com
RID for the User is: PZE9xU4
Verify USER Reported Email? Type Y or N: Y

This is the URL that will be opened: http://10.70.70.15/report?rid=PZE9xU4

Email Report updated for userneville@hogwarts.com
<http.client.HTTPResponse object at 0x0000023356102F70>

Enter the name of the campaign:
```

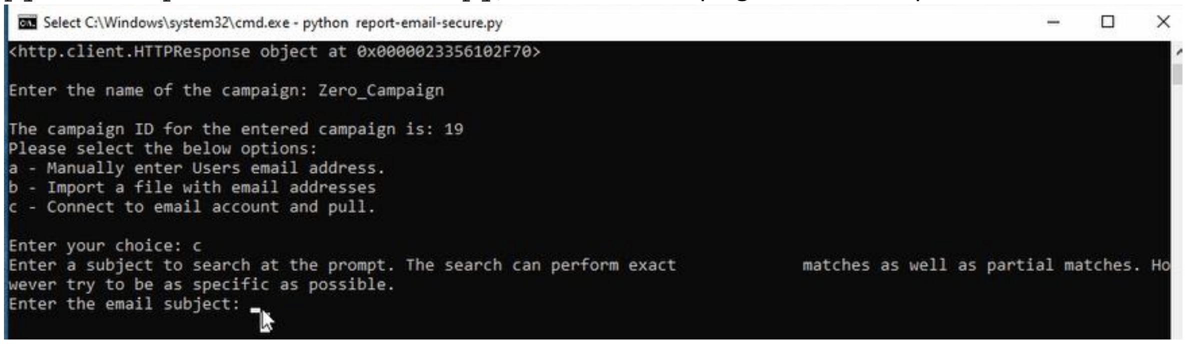
We can verify the same in our Gophish console:



First Name	Last Name	Email	Position	Status	Reported
Luna	Lovegood	luna@hogwarts.com	Astronomy	Email Opened	✗
Neville	Longbottom	neville@hogwarts.com	Herbology	Submitted Data	✓

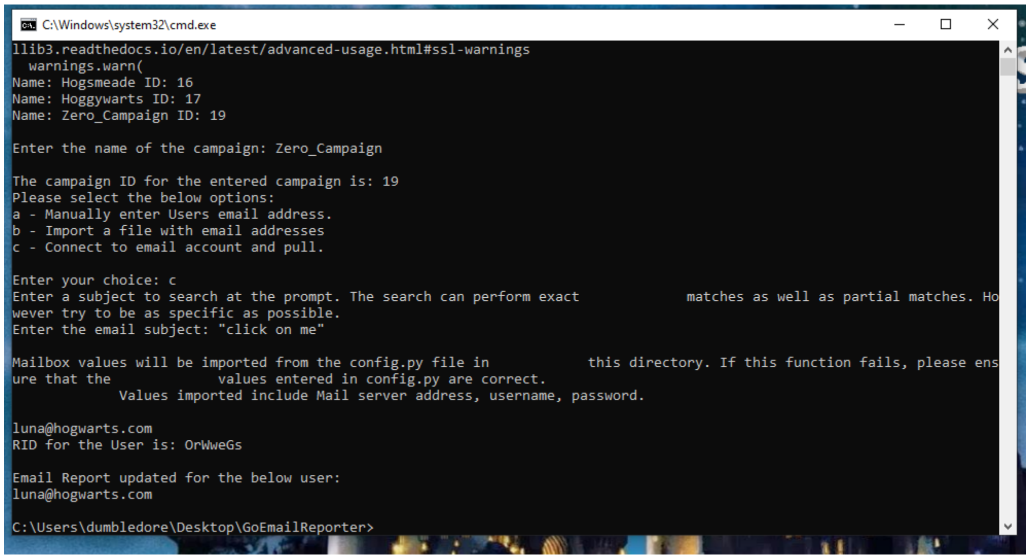
Great! We have successfully reported one phish on behalf of our users. Now for the second user – Luna, we'll use the mailbox connect method. Luna did open our phish, but because she's Luna, she used her divination powers to learn this was a phishing email and reported it to IT by sending it to the phish-report mailbox at Hogwarts – dumbledore@hogwarts.com. Why she couldn't divine that it was a phish before she opened it, or why Dumbledore is the IT admin at Hogwarts, we cannot answer at this time.

Anyway, the phish is in the reporting box and it's time to take action! Run the GoEmailReporter script – `python report-secure-mail.py`, choose the campaign and select option **c**.


Select C:\Windows\system32\cmd.exe - python report-email-secure.py

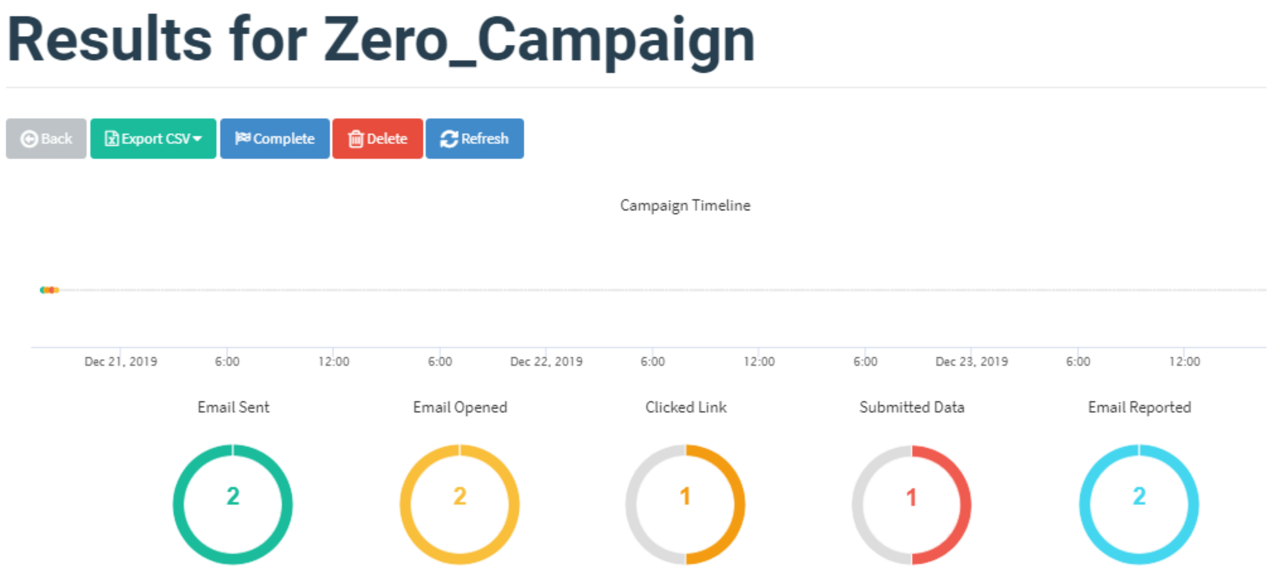
```
<http.client.HTTPResponse object at 0x0000023356102F70>

Enter the name of the campaign: Zero_Campaign

The campaign ID for the entered campaign is: 19
Please select the below options:
a - Manually enter Users email address.
b - Import a file with email addresses
c - Connect to email account and pull.

Enter your choice: c
Enter a subject to search at the prompt. The search can perform exact             matches as well as partial matches. Ho
wever try to be as specific as possible.
Enter the email subject:
```

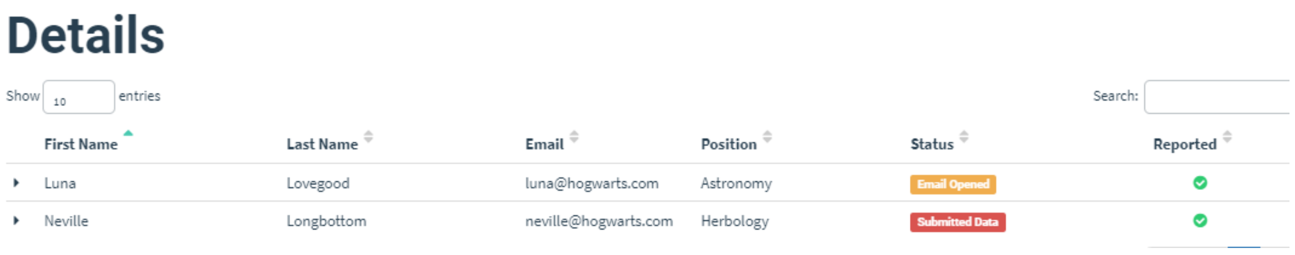
Enter the email subject, this can be a partial match or an exact match. The script doesn't play nice with single quotes ('), so if that is in your subject, use a portion that doesn't include that character. Also if you are entering phrases, use double quotes (") to enclose them. In our scenario, the subject line is "I'm a phishing email. Please don't click on me." So I've entered "click on me" for partial matching. The script mentions some troubleshooting tips incase it fails at this point (usually email credentials in config.py) and displays the email addresses and RIDs for each user whose forwarded emails are found in the mailbox and proceeds to report them. Only unique email addresses from UNREAD emails are reported when the script is run, so make sure you've marked emails as UNREAD if you open them. The script will parse unique email addresses, report them and then mark the email as READ so as not to parse them next time the script is run. Once all unique email addresses are processed, it will display the Email addresses reported.



```
llib3.readthedocs.io/en/latest/advanced-usage.html#ssl-warnings
  warnings.warn(
Name: Hogsmeade ID: 16
Name: Hoggywarts ID: 17
Name: Zero_Campaign ID: 19

Enter the name of the campaign: Zero_Campaign

The campaign ID for the entered campaign is: 19
Please select the below options:
a - Manually enter Users email address.
b - Import a file with email addresses
c - Connect to email account and pull.

Enter your choice: c
Enter a subject to search at the prompt. The search can perform exact             matches as well as partial matches. Ho
wever try to be as specific as possible.
Enter the email subject: "click on me"

Mailbox values will be imported from the config.py file in            this directory. If this function fails, please ens
ure that the            values entered in config.py are correct.
             Values imported include Mail server address, username, password.

luna@hogwarts.com
RID for the User is: OrWweGs

Email Report updated for the below user:
luna@hogwarts.com

C:\Users\dumbledore\Desktop\GoEmailReporter>
```

We can verify in the Gophish console for our campaign.





Details

Show 10 entries Search:

	First Name	Last Name	Email	Position	Status	Reported
▸	Luna	Lovegood	luna@hogwarts.com	Astronomy	Email Opened	✓
▸	Neville	Longbottom	neville@hogwarts.com	Herbology	Submitted Data	✓

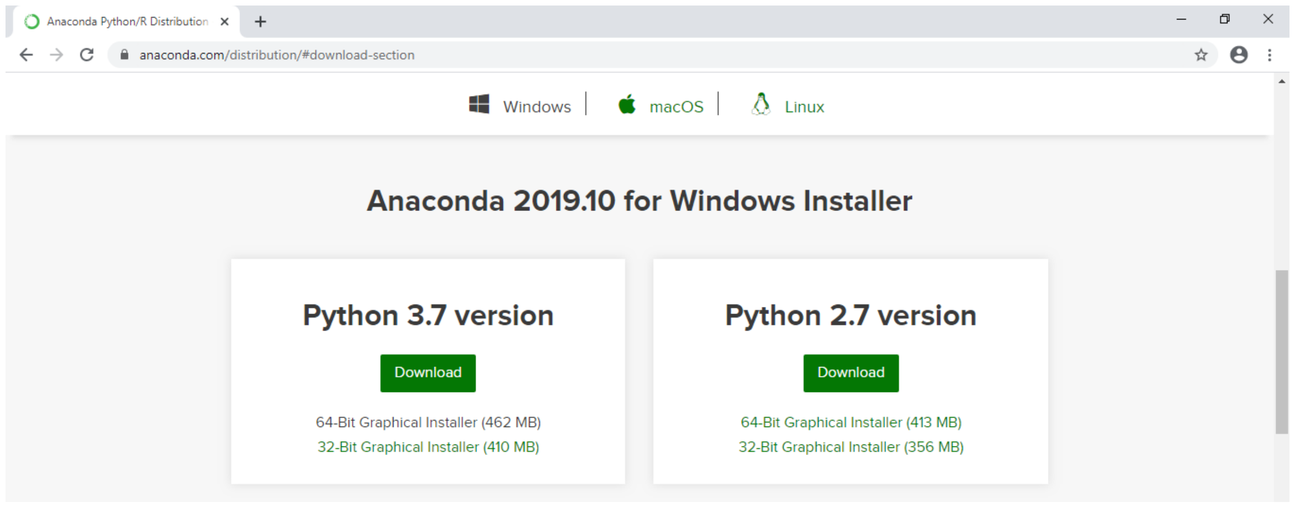
And that's it! We've just completed our objective – Actually educating users! Way to go CyberSecurity Warrior.

Phish Reporting with Jupyter Notebook

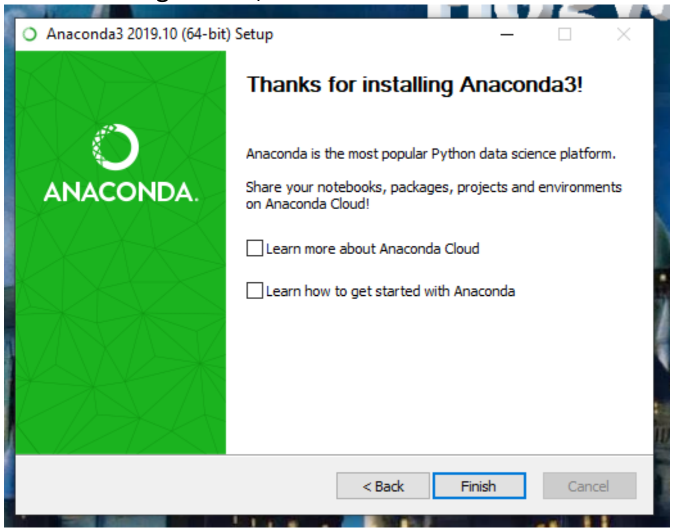
As another option, I've also created a **Jupyter Notebook** for GoEmailReporter that should be available in the zip file you downloaded from github – `GoEmailReporter.ipynb`

Jupyter Notebooks are web pages that contain live interactive python code that can be run by anyone who accesses the page. From the official website [https://jupyter.org]: The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. As our usecase, We can host our GoEmailReporter via a Jupyter Notebook and any IT admin on your team can run the reporting tool by accessing the page. Sounds awesome right? Let's get to it.

The easiest method to install Jupyter on Windows is via the Anaconda distribution [https://www.anaconda.com/distribution/]. It includes python, jupyter notebook and a lot of libraries for data science. Download the Python 3.7+ version appropriate for your OS.


Anaconda Python/R Distribution
anaconda.com/distribution/#download-section
Windows
macOS
Linux
Anaconda 2019.10 for Windows Installer
Python 3.7 version
Download
64-Bit Graphical Installer (462 MB)
32-Bit Graphical Installer (410 MB)
Python 2.7 version
Download
64-Bit Graphical Installer (413 MB)
32-Bit Graphical Installer (356 MB)

Install Anaconda with the default configuration, Next -> …Next -> Finish!


Anaconda3 2019.10 (64-bit) Setup

Thanks for installing Anaconda3!

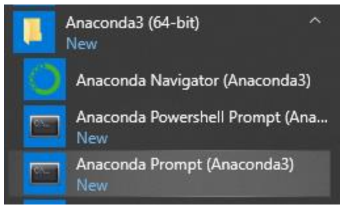
Anaconda is the most popular Python data science platform.

Share your notebooks, packages, projects and environments on Anaconda Cloud!

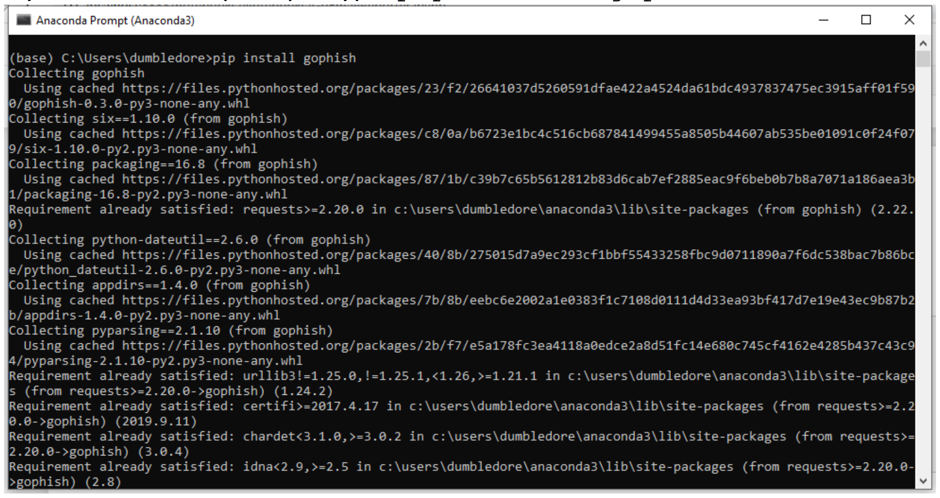
☐ Learn more about Anaconda Cloud

☐ Learn how to get started with Anaconda

We need to install the Gophish library for Anaconda. Open the Anaconda Prompt in Start Menu -> Anaconda3

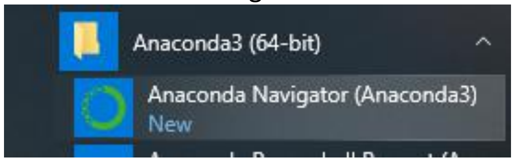

Anaconda3 (64-bit)
New
Anaconda Navigator (Anaconda3)
Anaconda Powershell Prompt (Ana...
New
Anaconda Prompt (Anaconda3)
New

This should give you a command prompt. Type `pip install gophish`

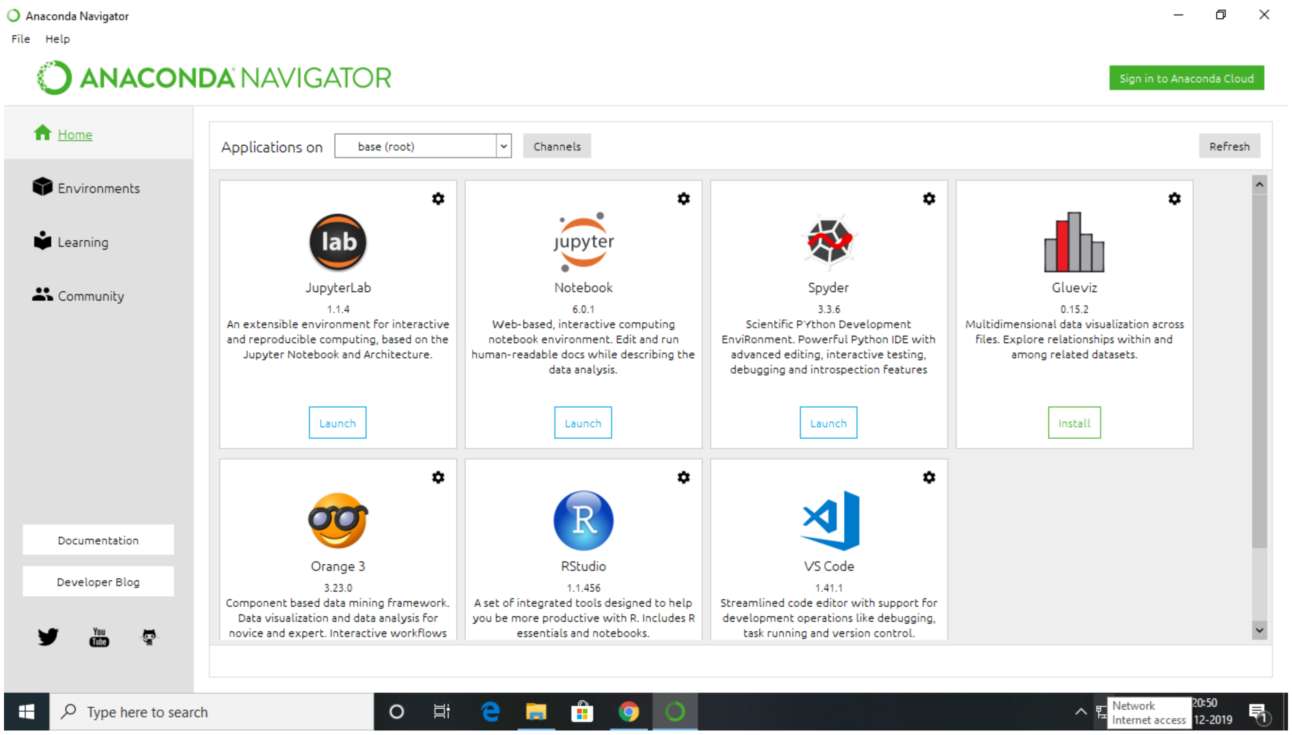

Anaconda Prompt (Anaconda3)

```
(base) C:\Users\dumbledore>pip install gophish
Collecting gophish
  Using cached https://files.pythonhosted.org/packages/23/f2/26641037d5260591dfae422a4524da61bdc4937837475ec3915aff01f59
0/gophish-0.3.0-py3-none-any.whl
Collecting six==1.10.0 (from gophish)
  Using cached https://files.pythonhosted.org/packages/c8/0a/b6723e1bc4c516cb687841499455a8505b44607ab535be01091c0f24f07
9/six-1.10.0-py2.py3-none-any.whl
Collecting packaging==16.8 (from gophish)
  Using cached https://files.pythonhosted.org/packages/87/1b/c39b7c65b5612812b83d6cab7ef2885eac9f6beb0b7b8a7071a186aea3b
1/packaging-16.8-py2.py3-none-any.whl
Requirement already satisfied: requests>=2.20.0 in c:\users\dumbledore\anaconda3\lib\site-packages (from gophish) (2.22.
0)
Collecting python-dateutil==2.6.0 (from gophish)
  Using cached https://files.pythonhosted.org/packages/40/8b/275015d7a9ec293cf1bbf55433258fbc9d0711890a7f6dc538bac7b86bc
e/python_dateutil-2.6.0-py2.py3-none-any.whl
Collecting appdirs==1.4.0 (from gophish)
  Using cached https://files.pythonhosted.org/packages/7b/8b/eebc6e2002a1e0383f1c7108d0111d4d33ea93bf417d7e19e43ec9b87b2
b/appdirs-1.4.0-py2.py3-none-any.whl
Collecting pyparsing==2.1.10 (from gophish)
  Using cached https://files.pythonhosted.org/packages/2b/f7/e5a178fc3ea4118a0edce2a8d51fc14e680c745cf4162e4285b437c43c9
4/pyparsing-2.1.10-py2.py3-none-any.whl
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in c:\users\dumbledore\anaconda3\lib\site-package
s (from requests>=2.20.0->gophish) (1.24.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\dumbledore\anaconda3\lib\site-packages (from requests>=2.2
0.0->gophish) (2019.9.11)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in c:\users\dumbledore\anaconda3\lib\site-packages (from requests>=
2.20.0->gophish) (3.0.4)
Requirement already satisfied: idna<2.9,>=2.5 in c:\users\dumbledore\anaconda3\lib\site-packages (from requests>=2.20.0-
>gophish) (2.8)
```

Now lets run our notebook. Open the Anaconda Navigator from the start menu:



As you can see, Anaconda includes a lot of tools for Python. But our interest is in the Jupyter Notebook. Click on Launch under Jupyter Notebook.



JupyterLab
1.1.4
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook
6.0.1
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Spyder
3.3.6
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

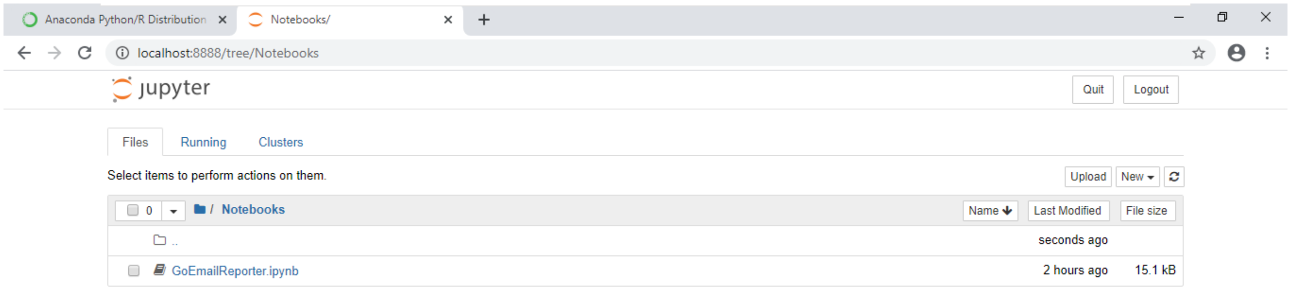
Glueviz
0.15.2
Multidimensional data visualization across files. Explore relationships within and among related datasets.

Orange 3
3.23.0
Component based data mining Framework. Data visualization and data analysis for novice and expert. Interactive workflows

RStudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

VS Code
1.41.1
Streamlined code editor with support for development operations like debugging, task running and version control.

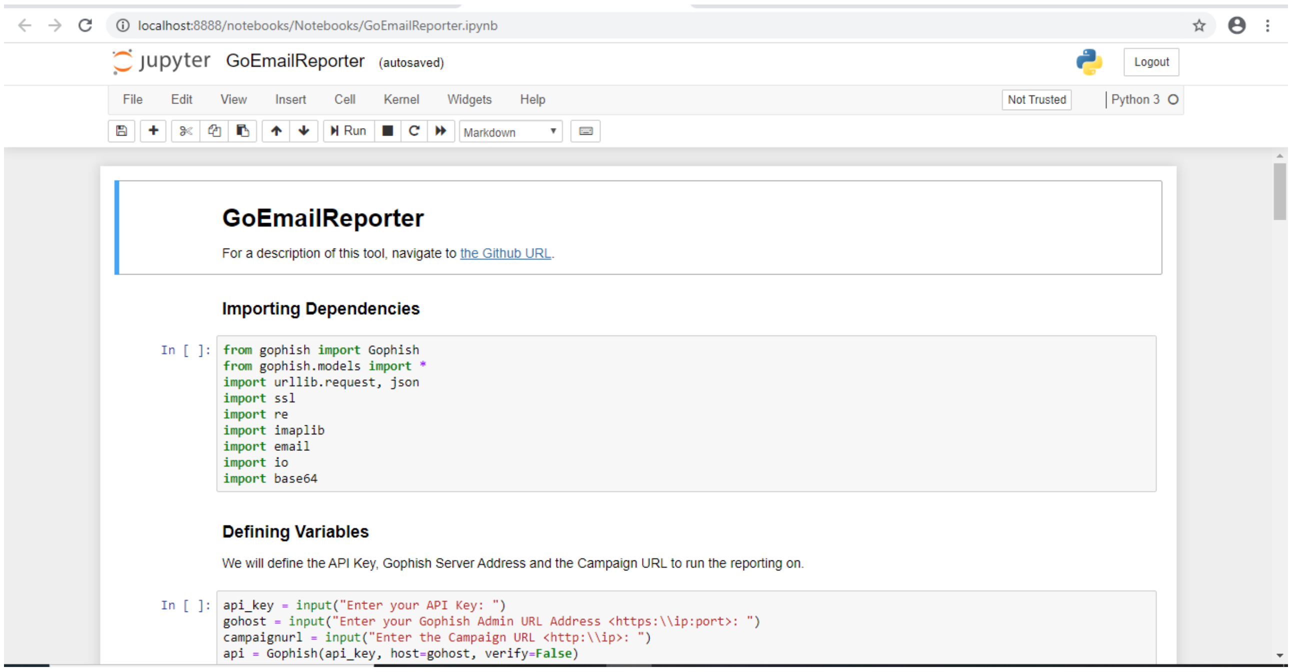
A Web browser will launch Jupyter with a directory browser. Either browse to the directory of our GoEmailReporter notebook or create a new folder in **C:\Users\<username>** and copy the .ipynb file in it to launch. Finally, click on the file **GoEmailReporter.ipynb** file to launch.



Select items to perform actions on them.

/ Notebooks		
..	seconds ago	
GoEmailReporter.ipynb	2 hours ago	15.1 kB

A Jupyter notebook can be comprised of Code or Text based [Markdown] blocks (among others) called cells. Each cell can be individually run to run that particular piece of code or run serially like a script. Cells marked with In [ ] are Code cells. The In [ ] tells the user that this is the input for python. When a cell is run, the output will be displayed below the Cell.


localhost:8888/notebooks/Notebooks/GoEmailReporter.ipynb
jupyter GoEmailReporter (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown

GoEmailReporter

For a description of this tool, navigate to the Github URL.

Importing Dependencies

```
In [ ]: from gophish import Gophish
        from gophish.models import *
        import urllib.request, json
        import ssl
        import re
        import imaplib
        import email
        import io
        import base64
```

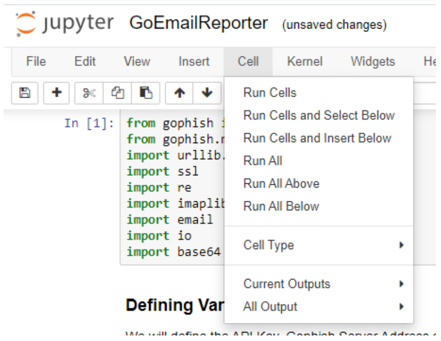
Defining Variables

We will define the API Key, Gophish Server Address and the Campaign URL to run the reporting on.

```
In [ ]: api_key = input("Enter your API Key: ")
        gohost = input("Enter your Gophish Admin URL Address <https:\\ip:port>: ")
        campaignurl = input("Enter the Campaign URL <http:\\ip>: ")
        api = Gophish(api_key, host=gohost, verify=False)
```

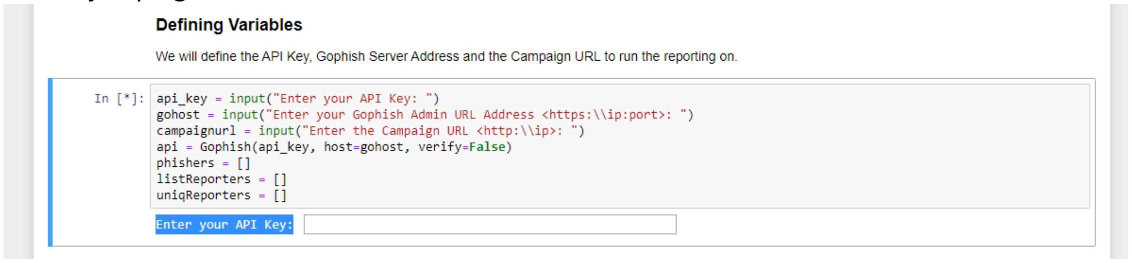
The default version of the GoEmailReporter notebook is setup as a single script for all functions eschewing a separate config.py file. Instead input fields allow admins to specify the variables each time the notebook is run. If you wish to hardcode the variable parameters, you replace the input commands with your data in quotes. Email credentials are part of the final cell block in option c.

I will be running the notebook in the default mode. Click on **Cells -> Run all**.


jupyter GoEmailReporter (unsaved changes)
File Edit View Insert Cell Kernel Widgets He
Run Cells
Run Cells and Select Below
Run Cells and Insert Below
Run All
Run All Above
Run All Below
Cell Type
Current Outputs
All Output
Defining Var

```
In [1]: from gophish i
        from gophish.r
        import urllib.
        import ssl
        import re
        import imaplib
        import email
        import io
        import base64
```

The first code cell should prompt for input to set our variables. Any prompts within that cell are run serially, before jumping to the next code cell.

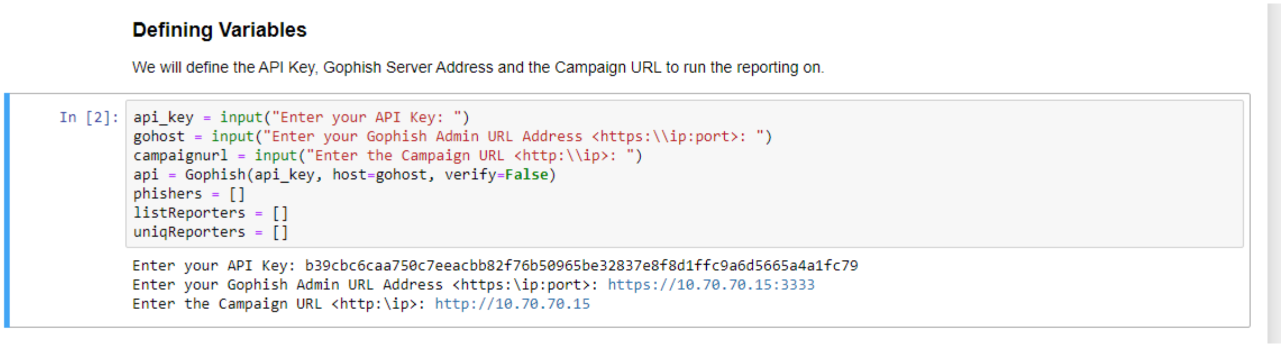


Defining Variables

We will define the API Key, Gophish Server Address and the Campaign URL to run the reporting on.

```
In [*]: api_key = input("Enter your API Key: ")
        gohost = input("Enter your Gophish Admin URL Address <https:\\ip:port>: ")
        campaignurl = input("Enter the Campaign URL <http:\\ip>: ")
        api = Gophish(api_key, host=gohost, verify=False)
        phishers = []
        listReporters = []
        uniqReporters = []

        Enter your API Key:
```

Once I finish entering all input required for that particular cell, the notebook runs the next cell and so on. This is our finished cell block:

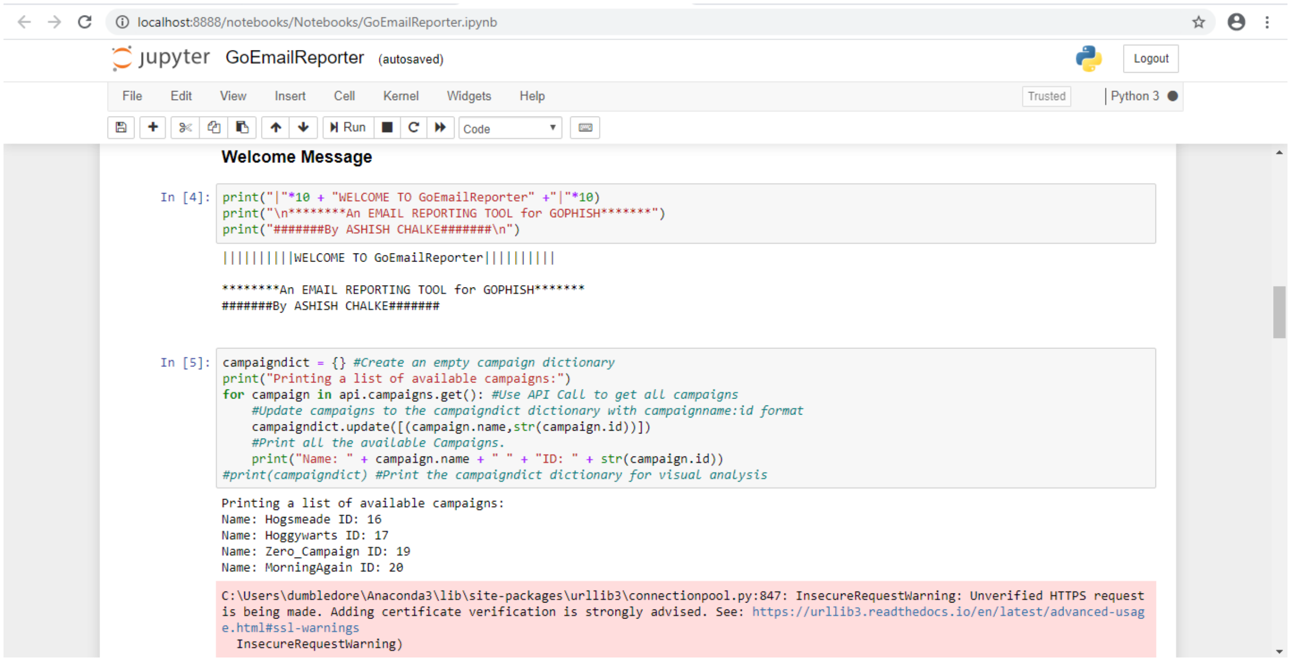


Defining Variables

We will define the API Key, Gophish Server Address and the Campaign URL to run the reporting on.

```
In [2]: api_key = input("Enter your API Key: ")
        gohost = input("Enter your Gophish Admin URL Address <https:\\ip:port>: ")
        campaignurl = input("Enter the Campaign URL <http:\\ip>: ")
        api = Gophish(api_key, host=gohost, verify=False)
        phishers = []
        listReporters = []
        uniqReporters = []

        Enter your API Key: b39cbc6caa750c7eeacbb82f76b50965be32837e8f8d1ffc9a6d5665a4a1fc79
        Enter your Gophish Admin URL Address <https:\ip:port>: https://10.70.70.15:3333
        Enter the Campaign URL <http:\ip>: http://10.70.70.15
```

Unfortunately, with the current setup of GoEmailReporter, it jumps directly to the last cell for input so you may need to scroll up to view output for the campaign cell block so that you can correctly input data in the last cell. The campaigns can be viewed under the Welcome Message heading.

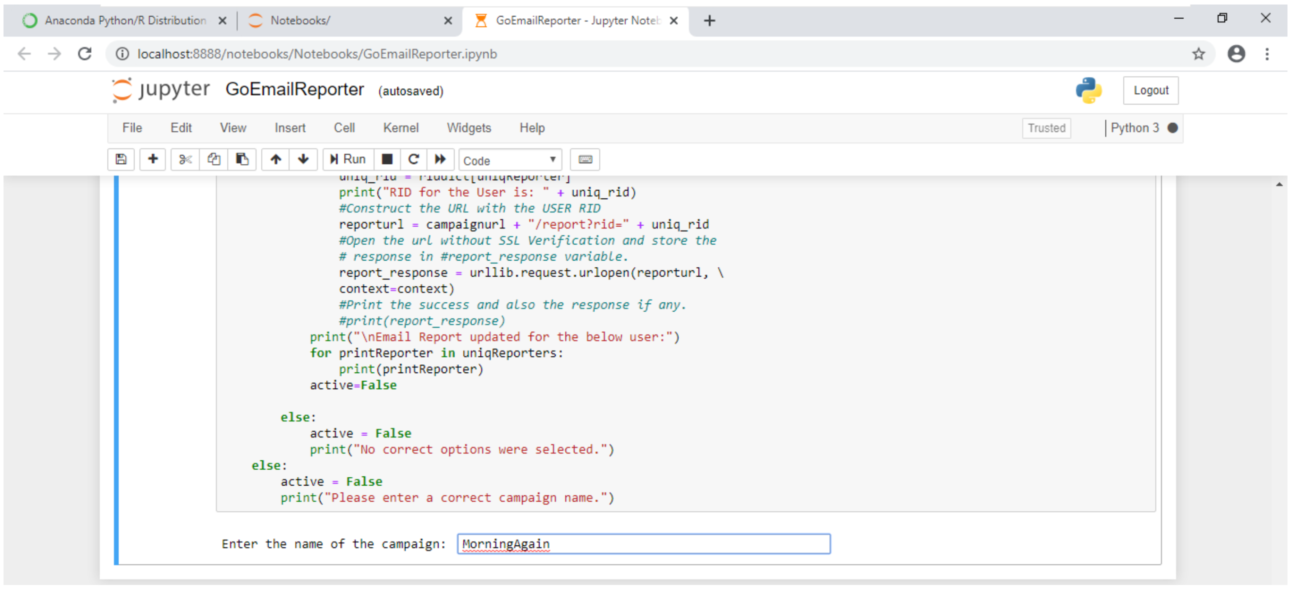


localhost:8888/notebooks/Notebooks/GoEmailReporter.ipynb
jupyter GoEmailReporter (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

Welcome Message

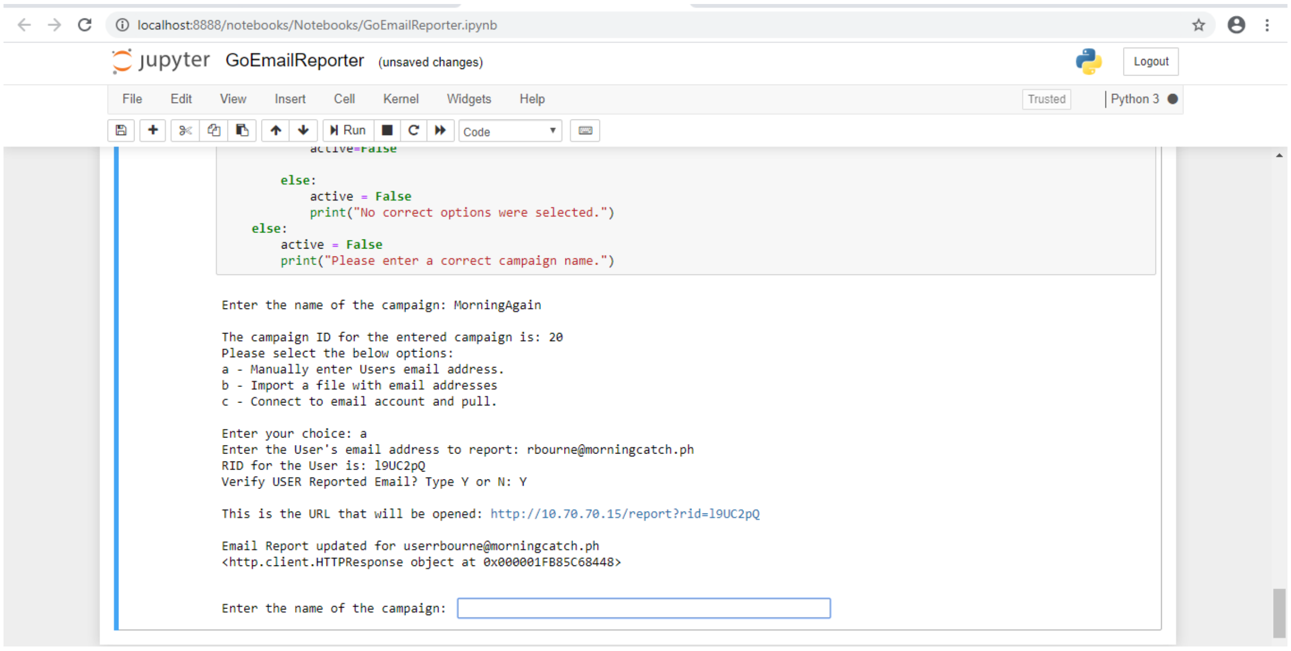
```
In [4]: print("|"*10 + "WELCOME TO GoEmailReporter" +"|"*10)
        print("\n*******An EMAIL REPORTING TOOL for GOPHISH*******")
        print("#######By ASHISH CHALKE#######\n")

        ||||||||||WELCOME TO GoEmailReporter||||||||||

        *******An EMAIL REPORTING TOOL for GOPHISH*******
        #######By ASHISH CHALKE#######


In [5]: campaigndict = {} #Create an empty campaign dictionary
        print("Printing a list of available campaigns:")
        for campaign in api.campaigns.get(): #Use API Call to get all campaigns
            #Update campaigns to the campaigndict dictionary with campaignname:id format
            campaigndict.update([(campaign.name,str(campaign.id))])
            #Print all the available Campaigns.
            print("Name: " + campaign.name + " " + "ID: " + str(campaign.id))
        #print(campaigndict) #Print the campaigndict dictionary for visual analysis

        Printing a list of available campaigns:
        Name: Hogsmeade ID: 16
        Name: Hoggywarts ID: 17
        Name: Zero_Campaign ID: 19
        Name: MorningAgain ID: 20

        C:\Users\dumbledore\Anaconda3\lib\site-packages\urllib3\connectionpool.py:847: InsecureRequestWarning: Unverified HTTPS request
        is being made. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usag
        e.html#ssl-warnings
          InsecureRequestWarning)
```

We copy the name of the campaign and scroll to the output of the last code cell to paste it as input.



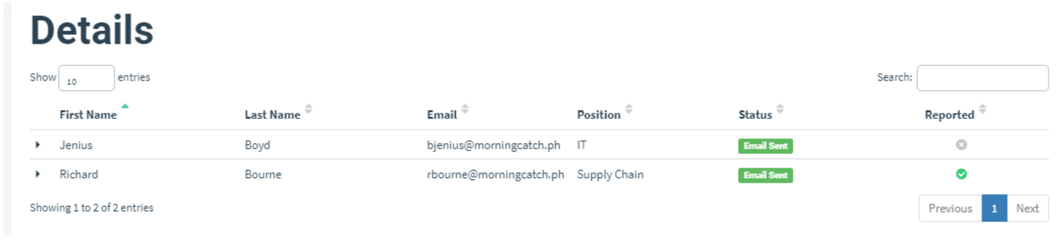
```
                uniq_rid = ridDict[uniqReporter]
                print("RID for the User is: " + uniq_rid)
                #Construct the URL with the USER RID
                reporturl = campaignurl + "/report?rid=" + uniq_rid
                #Open the url without SSL Verification and store the
                # response in #report_response variable.
                report_response = urllib.request.urlopen(reporturl, \
                context=context)
                #Print the success and also the response if any.
                #print(report_response)
            print("\nEmail Report updated for the below user:")
            for printReporter in uniqReporters:
                print(printReporter)
            active=False

        else:
            active = False
            print("No correct options were selected.")
    else:
        active = False
        print("Please enter a correct campaign name.")

Enter the name of the campaign: MorningAgain
```

Follow the prompts to report this user in the fashion you want! I went ahead and reported via the manual method.



```
            active=False

        else:
            active = False
            print("No correct options were selected.")
    else:
        active = False
        print("Please enter a correct campaign name.")

Enter the name of the campaign: MorningAgain

The campaign ID for the entered campaign is: 20
Please select the below options:
a - Manually enter Users email address.
b - Import a file with email addresses
c - Connect to email account and pull.

Enter your choice: a
Enter the User's email address to report: rbourne@morningcatch.ph
RID for the User is: l9UC2pQ
Verify USER Reported Email? Type Y or N: Y

This is the URL that will be opened: http://10.70.70.15/report?rid=l9UC2pQ

Email Report updated for the userrbourne@morningcatch.ph
<http.client.HTTPResponse object at 0x000001FB85C68448>

Enter the name of the campaign:
```

And we verified it on the Gophish Admin page that the user has successfully been reported!



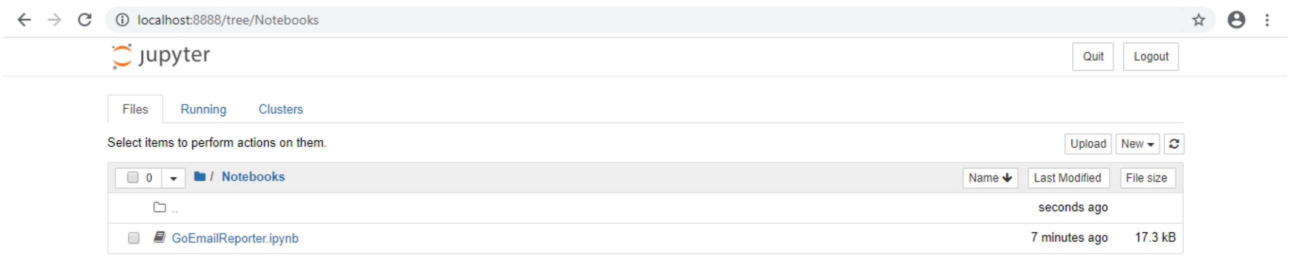
Details

Show 10 entries Search:

First Name	Last Name	Email	Position	Status	Reported
Jenius	Boyd	bjenius@morningcatch.ph	IT	Email Sent	⊗
Richard	Bourne	rbourne@morningcatch.ph	Supply Chain	Email Sent	✓

Showing 1 to 2 of 2 entries

Pretty easy huh? You can rerun a notebook by going to **Kernel -> Restart & Run all**. This will clear all variables entered. I will recommend this method if you are running in default mode. Once finished, Click on **File -> Close and Halt** to stop the python kernel and close the notebook. The notebook icon will go back to gray. Click on **Quit** on the top right to close jupyter.



Select items to perform actions on them.

/ Notebooks		
..	seconds ago	
GoEmailReporter.ipynb	7 minutes ago	17.3 kB

But what's the use of setting up a web based interactive notebook if other admins can't access it on the network and use it? We will need to modify Jupyter's configuration to enable it to listen on the network and allow remote connections. And while we are at it, how about deploying it as a web app that enables IT admins to access it on their web browsers and perform reporting? There are exhaustive tutorials available on the web to set this up. The below can act as a quick setup guide on setting up the app on Ubuntu.

1. Install Ubuntu Server 18.04 LTS. Note that this is a headless server only having shell access as the need is setting up a remote-access tool. If you need GUI, install the Desktop version.
2. Upgrade Python3 and Install pip
   a. `sudo apt-get upgrade python3`
   b. `sudo apt-get install python3-pip python3-dev`
3. Install our dependencies:
   a. `sudo -H pip3 install gophish`
4. Install jupyter:
   a. `sudo -H pip3 install jupyter`
5. Create a Jupyter configuration and edit the configuration using nano or your favorite text editor:
   a. `sudo jupyter notebook --generate-config`
   b. `sudo nano /home/<user>/.jupyter/jupyter-notebook-config.py`
6. Add these two values. You can add them anywhere in the file, or find the default ones, uncomment and modify them.
   a. `c.NotebookApp.allow_origin = '*'` - **allows connections from all IPs**
   b. `c.NotebookApp.ip = '0.0.0.0'` - **listens on all ip addresses. Change to specific if needed.**
   c. SAVE the file
7. Optional: Set a password to access the Jupyter notebook.
   a. jupyter notebook password
8. Allow inbound access to the Jupyter default port
   a. sudo ufw allow 8888

9. Start the Jupyter notebook
    a. sudo jupyter notebook --allow-root
10. Setup GoEmailReporter for the Jupyter Notebook:
    a. **Make a directory to hold the notebook**: `sudo mkdir /home/<user>/.jupyter/GoEmailReporter`
    b. **Move into the directory**: `cd /home/<user>/.jupyter/GoEmailReporter`
    c. **Download the GoEmailReporter Notebook**: sudo wget https://raw.githubusercontent.com/theashishchalke/gophish-tools/master/GoEmailReporter.ipynb
11. Connect to the Jupyter server on a browser on the network, enter the password and use!
    a. http://ip:8888/login

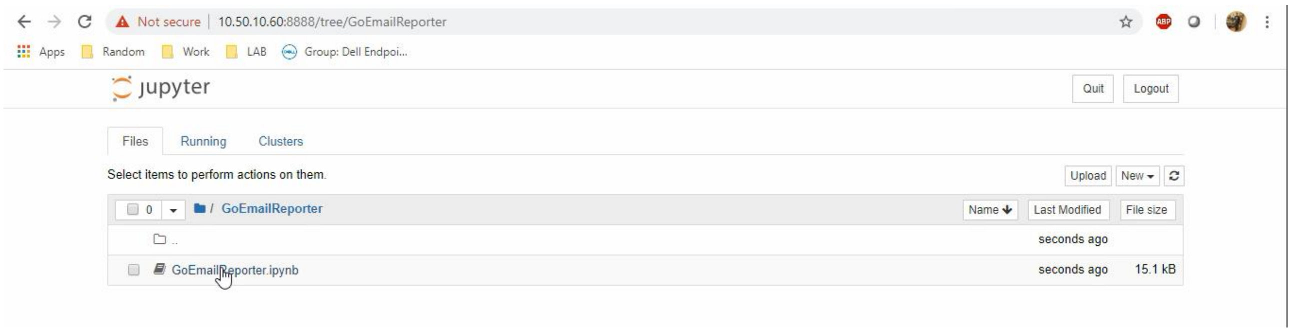
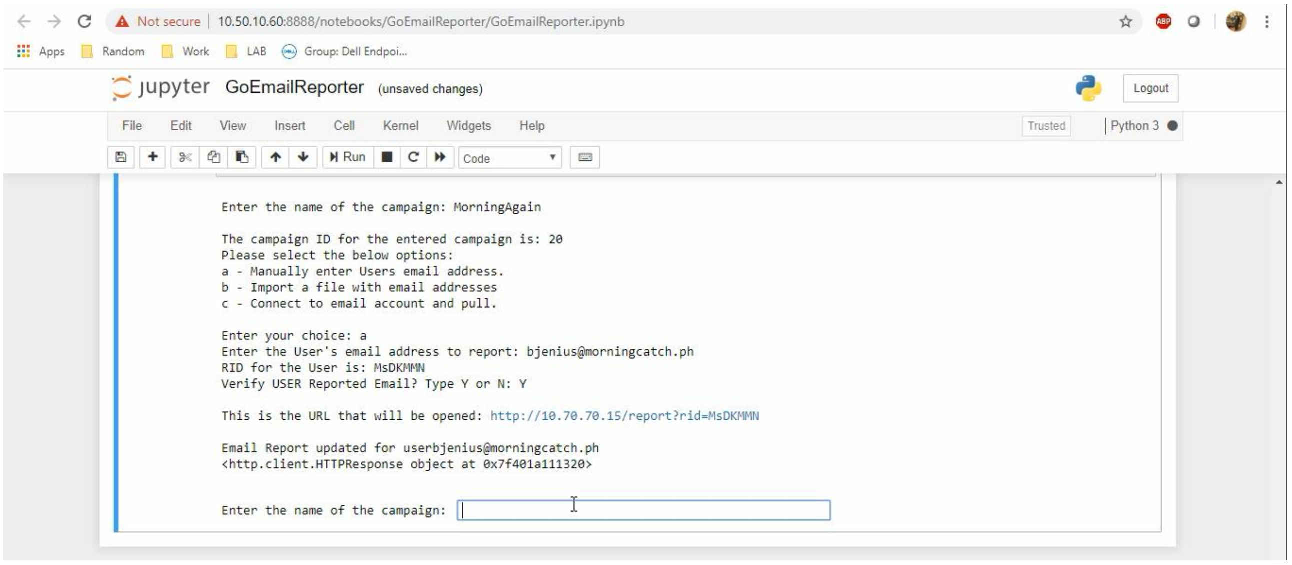


Fig: Accessing GoEmailReporter hosted on a remote Jupyter server.



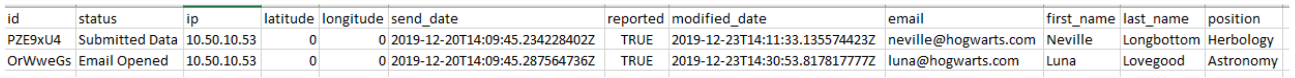
```
Enter the name of the campaign: MorningAgain

The campaign ID for the entered campaign is: 20
Please select the below options:
a - Manually enter Users email address.
b - Import a file with email addresses
c - Connect to email account and pull.

Enter your choice: a
Enter the User's email address to report: bjenius@morningcatch.ph
RID for the User is: MsDKMMN
Verify USER Reported Email? Type Y or N: Y

This is the URL that will be opened: http://10.70.70.15/report?rid=MsDKMMN

Email Report updated for userbjenius@morningcatch.ph
<http.client.HTTPResponse object at 0x7f401a111320>


Enter the name of the campaign:
```

Fig: Running GoEmailReporter.

A few points before we exit this section on (one final time) GoEmailReporter – This is an in progress project and some functionality may change or add. Be sure to keep up with the wiki [https://github.com/theashishchalke/gophish-tools/wiki] for up to date instructions as well as some gif demos if that is your thing. Secondly, I'm a script kiddie level programmer – hence it's mostly been writing what I know and googling [read:pillaging] what I do not [with proper credits of course]. But what this means is there could be ways where things could have been done better, and if you have a suggestion do contact me via the project page or create your fork! More the merrier. Finally, this is an admin tool to be used for a specific functionality. I decided to keep the code light and not do a lot of failure checks. If you follow the wizard prompts you should be good. Again, if you wish to add this functionality for added value, go ahead.

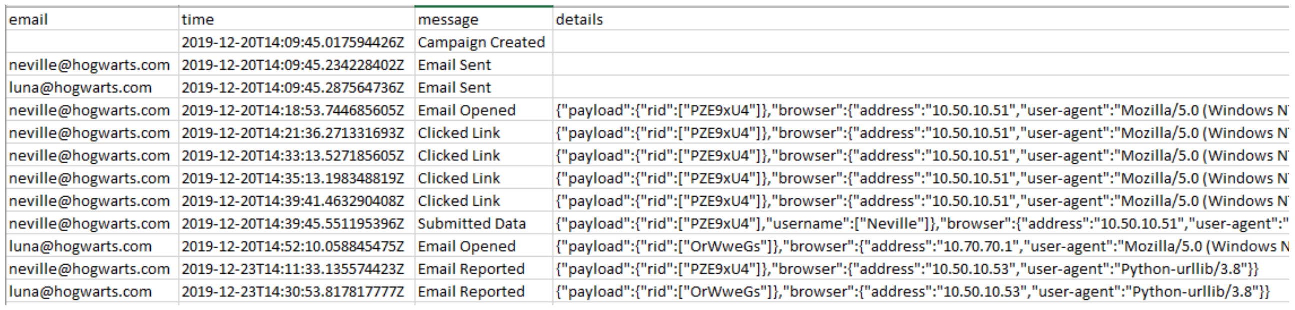
Reporting

Gophish has built-in reporting that can be accessed from the Campaign page or the Dashboard. Results for a campaign can be exported on a per campaign level. We have two available formats – Results is the final stage for a user – did they just open the email, click on the link, submit data. Exporting the Results view gives us the below report format. Notice that Reporting is a separate column.



id	status	ip	latitude	longitude	send_date	reported	modified_date	email	first_name	last_name	position
PZE9xU4	Submitted Data	10.50.10.53	0	0	2019-12-20T14:09:45.234228402Z	TRUE	2019-12-23T14:11:33.135574423Z	neville@hogwarts.com	Neville	Longbottom	Herbology
OrWweGs	Email Opened	10.50.10.53	0	0	2019-12-20T14:09:45.287564736Z	TRUE	2019-12-23T14:30:53.817817777Z	luna@hogwarts.com	Luna	Lovegood	Astronomy

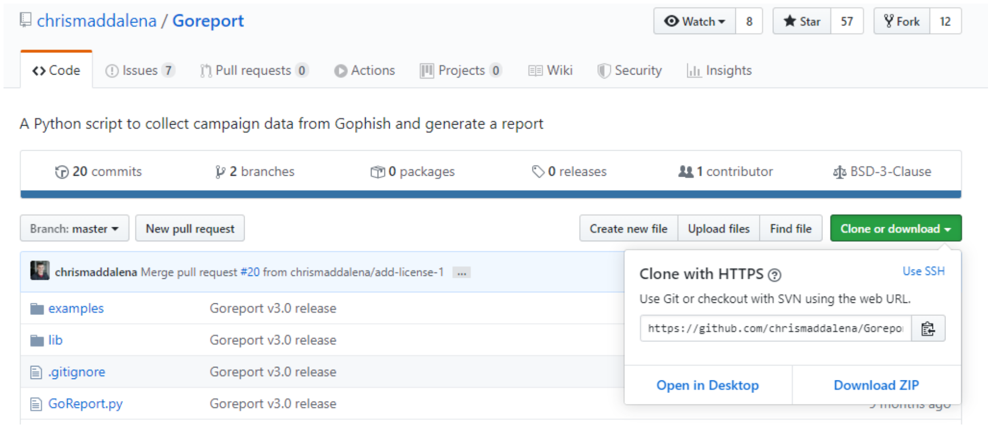
The Raw Events view gives us insights into individual events as the phish progressed. The meta data of the event is presented in json in the details column.



email	time	message	details
	2019-12-20T14:09:45.017594426Z	Campaign Created	
neville@hogwarts.com	2019-12-20T14:09:45.234228402Z	Email Sent	
luna@hogwarts.com	2019-12-20T14:09:45.287564736Z	Email Sent	
neville@hogwarts.com	2019-12-20T14:18:53.744685605Z	Email Opened	{"payload":{"rid":["PZE9xU4"]},"browser":{"address":"10.50.10.51","user-agent":"Mozilla/5.0 (Windows N
neville@hogwarts.com	2019-12-20T14:21:36.271331693Z	Clicked Link	{"payload":{"rid":["PZE9xU4"]},"browser":{"address":"10.50.10.51","user-agent":"Mozilla/5.0 (Windows N
neville@hogwarts.com	2019-12-20T14:33:13.527185605Z	Clicked Link	{"payload":{"rid":["PZE9xU4"]},"browser":{"address":"10.50.10.51","user-agent":"Mozilla/5.0 (Windows N
neville@hogwarts.com	2019-12-20T14:35:13.198348819Z	Clicked Link	{"payload":{"rid":["PZE9xU4"]},"browser":{"address":"10.50.10.51","user-agent":"Mozilla/5.0 (Windows N
neville@hogwarts.com	2019-12-20T14:39:41.463290408Z	Clicked Link	{"payload":{"rid":["PZE9xU4"]},"browser":{"address":"10.50.10.51","user-agent":"Mozilla/5.0 (Windows N
neville@hogwarts.com	2019-12-20T14:39:45.551195396Z	Submitted Data	{"payload":{"rid":["PZE9xU4"],"username":["Neville"]},"browser":{"address":"10.50.10.51","user-agent":"
luna@hogwarts.com	2019-12-20T14:52:10.058845475Z	Email Opened	{"payload":{"rid":["OrWweGs"]},"browser":{"address":"10.70.70.1","user-agent":"Mozilla/5.0 (Windows N
neville@hogwarts.com	2019-12-23T14:11:33.135574423Z	Email Reported	{"payload":{"rid":["PZE9xU4"]},"browser":{"address":"10.50.10.53","user-agent":"Python-urllib/3.8"}}
luna@hogwarts.com	2019-12-23T14:30:53.817817777Z	Email Reported	{"payload":{"rid":["OrWweGs"]},"browser":{"address":"10.50.10.53","user-agent":"Python-urllib/3.8"}}

While great, sometimes we may need a more customized version. For example, if you're running an elaborate phishing exercise where every campaign is targeted towards a particular set of users, you'll still want a consolidated report that lists who all were successfully targeted and how many reported your phishing emails. Making consolidated reports within Gophish requires a lot of manual work. This is where **GoReport** by **Chris Maddelena** [https://github.com/chrismaddalena/Goreport] comes in. GoReport is a python script that allows you to create detailed reports exported in excel and word, combines multiple campaigns into one report and also has capabilities to integrate with the google maps API for enhanced geolocation capabilities than what's built-in with gophish.

Since we've already installed Python, we will move directly to the GoReport setup. However if you need a refresher, see the previous section. To download GoReport, go to the github page [https://github.com/chrismaddalena/Goreport] -> Clone or Download -> Download ZIP. Download and extract this to a folder of your choosing.



chrismaddalena / Goreport

A Python script to collect campaign data from Gophish and generate a report

20 commits · 2 branches · 0 packages · 0 releases · 1 contributor · BSD-3-Clause

chrismaddalena Merge pull request #20 from chrismaddalena/add-license-1

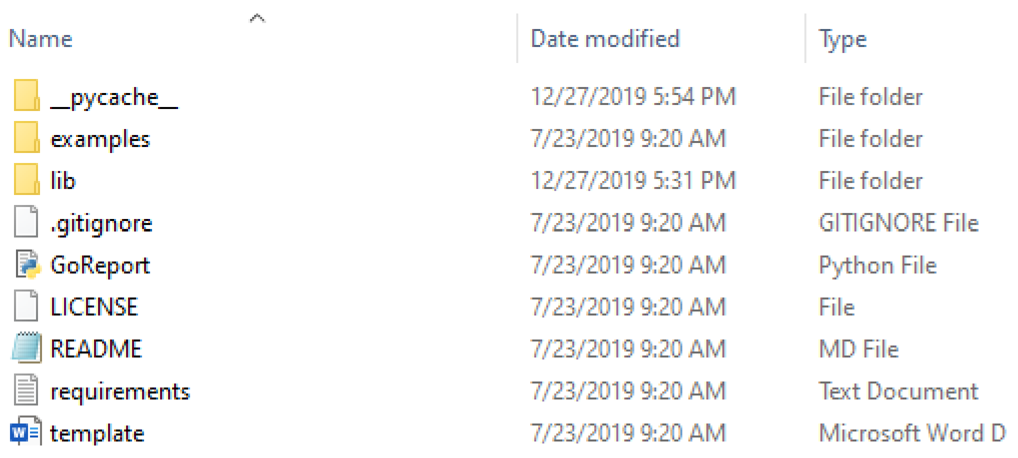
examples	Goreport v3.0 release
lib	Goreport v3.0 release
.gitignore	Goreport v3.0 release
GoReport.py	Goreport v3.0 release

Clone with HTTPS

Use Git or checkout with SVN using the web URL.

https://github.com/chrismaddalena/Gorepo

You should have the below directory structure.

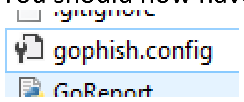


Name	Date modified	Type
__pycache__	12/27/2019 5:54 PM	File folder
examples	7/23/2019 9:20 AM	File folder
lib	12/27/2019 5:31 PM	File folder
.gitignore	7/23/2019 9:20 AM	GITIGNORE File
GoReport	7/23/2019 9:20 AM	Python File
LICENSE	7/23/2019 9:20 AM	File
README	7/23/2019 9:20 AM	MD File
requirements	7/23/2019 9:20 AM	Text Document
template	7/23/2019 9:20 AM	Microsoft Word D

Before running GoReport, we need to first add a config file with our api key and API tokens for ipinfo / Google used during geolocation. The tokens for IPINFO and Google are optional. Create a new notepad file with a custom extension [Save As Type -> All] called **gophish.config** and enter the below information in the file:

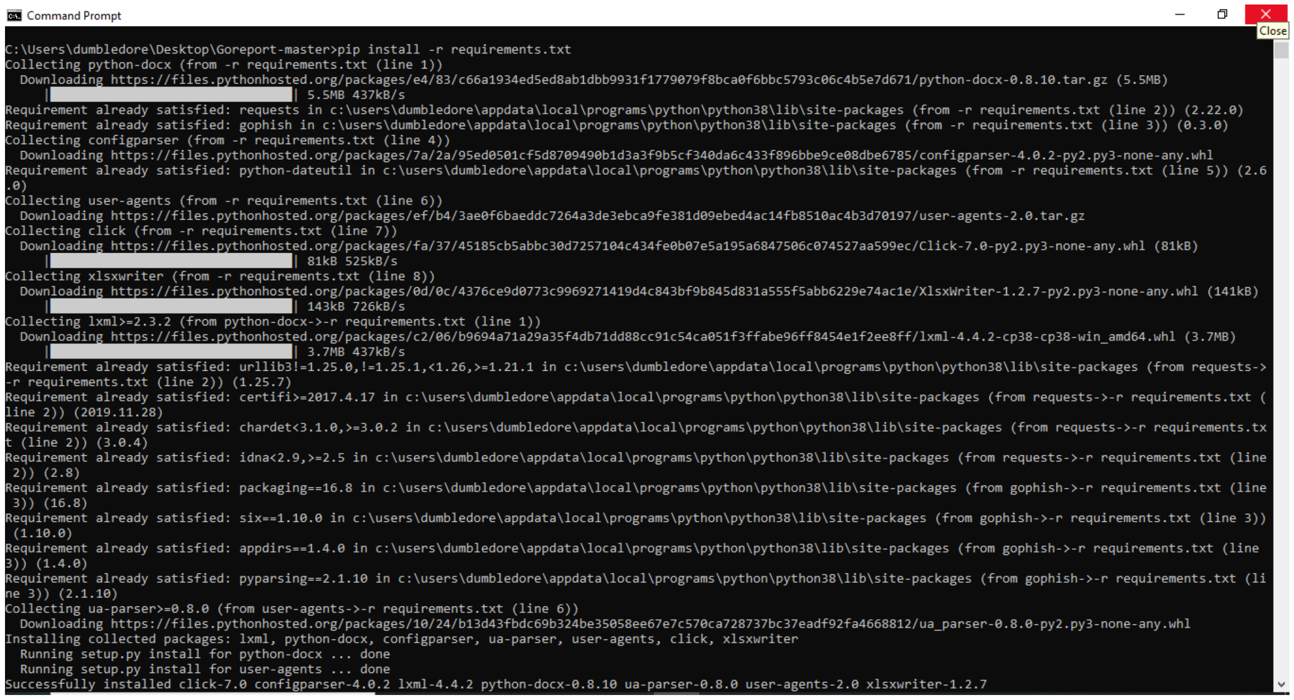
```
[Gophish]
gp_host: <GOPHISH ADMIN URL – https://iphost:port>
api_key: <GOPHISH API KEY>

[ipinfo.io]
ipinfo_token: <IPINFOTOKEN>

[Google]
geolocate_key: <GEOLOCATE_API_KEY>
```

You should now have a config file in the same directory.



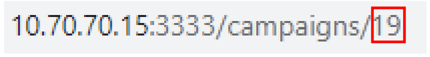
.gitignore
gophish.config
GoReport

Now we will begin installation of GoReport. Open a command prompt and navigate to the GoReport folder [`cd PATH`] . Type the command to install dependencies: `pip install -r requirements.txt`



```
C:\Users\dumbledore\Desktop\Goreport-master>pip install -r requirements.txt
Collecting python-docx (from -r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/e4/83/c66a1934ed5ed8ab1dbb9931f1779079f8bca0f6bbc5793c06c4b5e7d671/python-docx-0.8.10.tar.gz (5.5MB)
     |████████████████████████████████| 5.5MB 437kB/s
Requirement already satisfied: requests in c:\users\dumbledore\appdata\local\programs\python\python38\lib\site-packages (from -r requirements.txt (line 2)) (2.22.0)
Requirement already satisfied: gophish in c:\users\dumbledore\appdata\local\programs\python\python38\lib\site-packages (from -r requirements.txt (line 3)) (0.3.0)
Collecting configparser (from -r requirements.txt (line 4))
  Downloading https://files.pythonhosted.org/packages/7a/2a/95ed0501cf5d8709490b1d3a3f9b5cf340da6c433f896bbe9ce08dbe6785/configparser-4.0.2-py2.py3-none-any.whl
Requirement already satisfied: python-dateutil in c:\users\dumbledore\appdata\local\programs\python\python38\lib\site-packages (from -r requirements.txt (line 5)) (2.6
.0)
Collecting user-agents (from -r requirements.txt (line 6))
  Downloading https://files.pythonhosted.org/packages/ef/b4/3ae0f6baeddc7264a3de3ebca9fe381d09ebed4ac14fb8510ac4b3d70197/user-agents-2.0.tar.gz
Collecting click (from -r requirements.txt (line 7))
  Downloading https://files.pythonhosted.org/packages/fa/37/45185cb5abbc30d7257104c434fe0b07e5a195a6847506c074527aa599ec/Click-7.0-py2.py3-none-any.whl (81kB)
     |████████████████████████████████| 81kB 525kB/s
Collecting xlsxwriter (from -r requirements.txt (line 8))
  Downloading https://files.pythonhosted.org/packages/0d/0c/4376ce9d0773c9969271419d4c843bf9b845d831a555f5abb6229e74ac1e/XlsxWriter-1.2.7-py2.py3-none-any.whl (141kB)
     |████████████████████████████████| 143kB 726kB/s
Collecting lxml>=2.3.2 (from python-docx->-r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/c2/06/b9694a71a29a35f4db71dd88cc91c54ca051f3ffabe96ff8454e1f2ee8ff/lxml-4.4.2-cp38-cp38-win_amd64.whl (3.7MB)
     |████████████████████████████████| 3.7MB 437kB/s
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in c:\users\dumbledore\appdata\local\programs\python\python38\lib\site-packages (from requests->
-r requirements.txt (line 2)) (1.25.7)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\dumbledore\appdata\local\programs\python\python38\lib\site-packages (from requests->-r requirements.txt (
line 2)) (2019.11.28)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in c:\users\dumbledore\appdata\local\programs\python\python38\lib\site-packages (from requests->-r requirements.tx
t (line 2)) (3.0.4)
Requirement already satisfied: idna<2.9,>=2.5 in c:\users\dumbledore\appdata\local\programs\python\python38\lib\site-packages (from requests->-r requirements.txt (line
 2)) (2.8)
Requirement already satisfied: packaging==16.8 in c:\users\dumbledore\appdata\local\programs\python\python38\lib\site-packages (from gophish->-r requirements.txt (line
 3)) (16.8)
Requirement already satisfied: six==1.10.0 in c:\users\dumbledore\appdata\local\programs\python\python38\lib\site-packages (from gophish->-r requirements.txt (line 3))
 (1.10.0)
Requirement already satisfied: appdirs==1.4.0 in c:\users\dumbledore\appdata\local\programs\python\python38\lib\site-packages (from gophish->-r requirements.txt (line
3)) (1.4.0)
Requirement already satisfied: pyparsing==2.1.10 in c:\users\dumbledore\appdata\local\programs\python\python38\lib\site-packages (from gophish->-r requirements.txt (li
ne 3)) (2.1.10)
Collecting ua-parser>=0.8.0 (from user-agents->-r requirements.txt (line 6))
  Downloading https://files.pythonhosted.org/packages/10/24/b13d43fbdc69b324be35058ee67e7c570ca728737bc37eadf92fa4668812/ua_parser-0.8.0-py2.py3-none-any.whl
Installing collected packages: lxml, python-docx, configparser, ua-parser, user-agents, click, xlsxwriter
  Running setup.py install for python-docx ... done
  Running setup.py install for user-agents ... done
Successfully installed click-7.0 configparser-4.0.2 lxml-4.4.2 python-docx-0.8.10 ua-parser-0.8.0 user-agents-2.0 xlsxwriter-1.2.7
```

Now we can run reports. For a basic report, all we need are 2 things – The Campaign ID and the format to output the report – word or excel. You can get the campaign ID in the URL when you navigate to that particular campaign in Gophish Admin. The 19 in the below screenshot is the campaign ID.

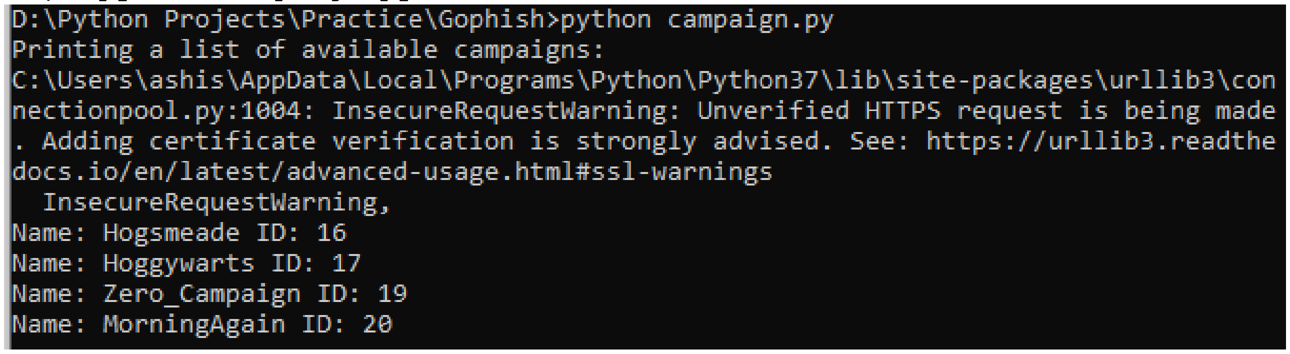


```
10.70.70.15:3333/campaigns/19
```

If you need something a bit more automated, you can use a script that I created to poll Gophish campaigns – campaign.py [https://raw.githubusercontent.com/theashishchalke/pythonscripts/master/campaign.py] Edit and modify the below variables to reflect your environment.

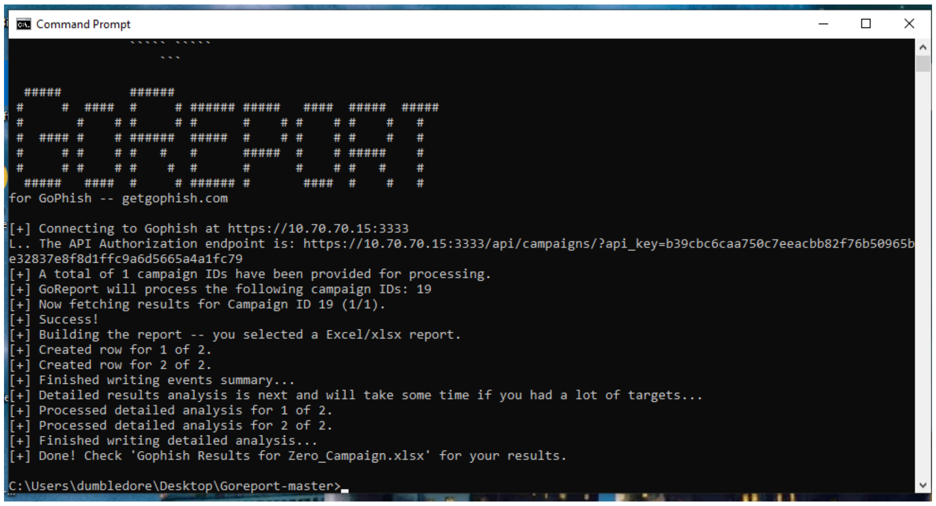
```
api_key = "APIKEYHERE"
gohost = "GOPHISH_ADMIN_URL"
```

You should now be able to run the script to poll all the campaigns on your Gophish server by executing the script – `python campaign.py`



```
D:\Python Projects\Practice\Gophish>python campaign.py
Printing a list of available campaigns:
C:\Users\ashis\AppData\Local\Programs\Python\Python37\lib\site-packages\urllib3\con
nectionpool.py:1004: InsecureRequestWarning: Unverified HTTPS request is being made
. Adding certificate verification is strongly advised. See: https://urllib3.readthe
docs.io/en/latest/advanced-usage.html#ssl-warnings
  InsecureRequestWarning,
Name: Hogsmeade ID: 16
Name: Hoggywarts ID: 17
Name: Zero_Campaign ID: 19
Name: MorningAgain ID: 20
```

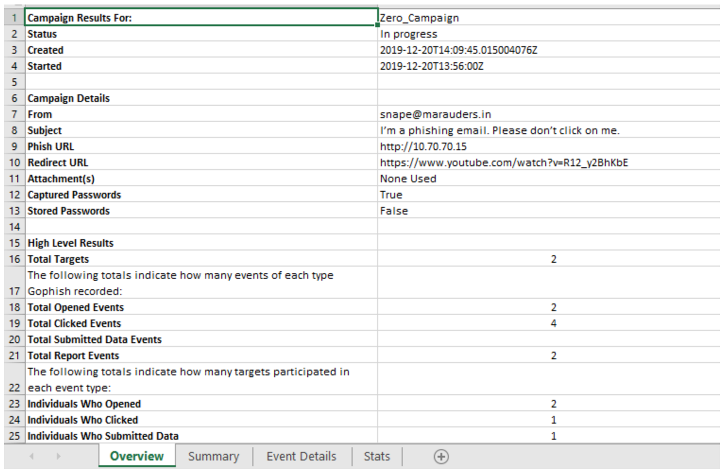
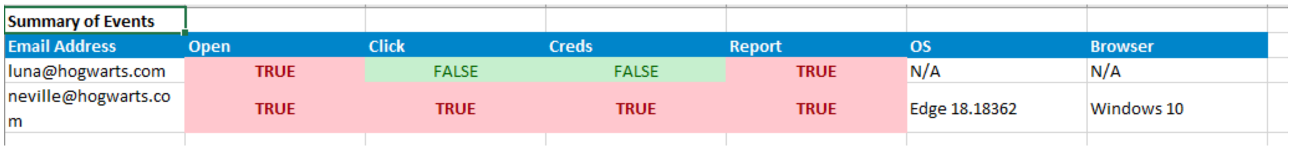
However you decide to get the campaign ID – manual or script method, you can now run a simple excel report for a campaign with GoReport using the command:
`python GoReport.py --id=<campaignID> --format=<word/excel>`
Example: `python GoReport.py --id=19 --format=excel`

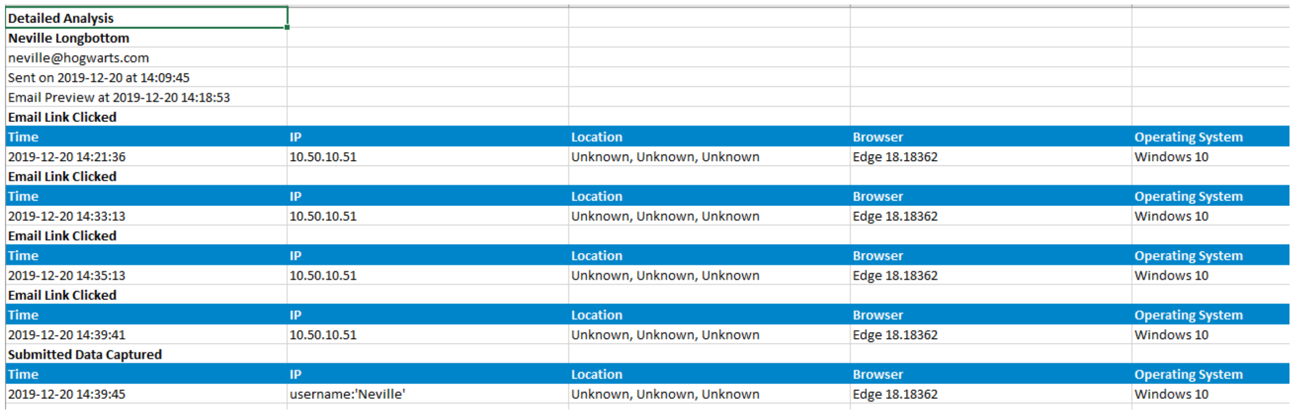


~~~
                ````` `````
                    ~~~

  #####   ######
 #   # #### #   # ###### #####   #### ##### #####
 #   #   # #   # #   #   # #   # #   #   #
 # #### #   # ###### ##### #   # #   # #   #   #
 #   # #   # #   #   #   ##### #   # #####   #
 #   # #   # #   # #   #   #   # #   #   #
  #####   #### #   # ###### #   #### #   #   #
for GoPhish -- getgophish.com

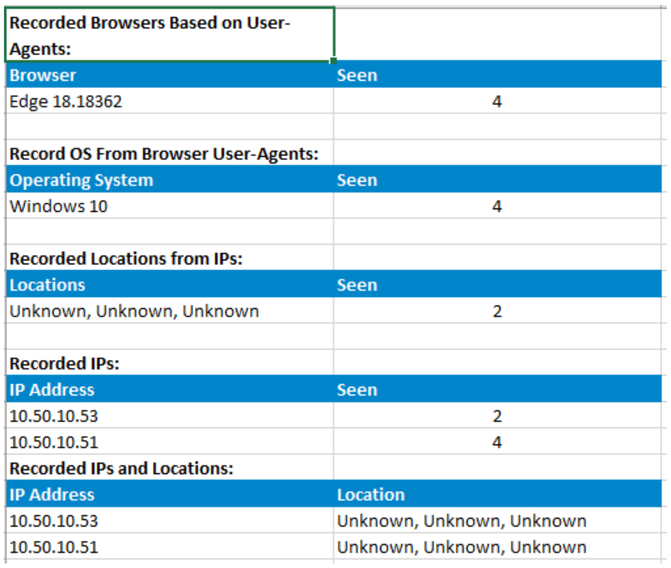
[+] Connecting to Gophish at https://10.70.70.15:3333
L.. The API Authorization endpoint is: https://10.70.70.15:3333/api/campaigns/?api_key=b39cbc6caa750c7eeacbb82f76b50965b
e32837e8f8d1ffc9a6d5665a4a1fc79
[+] A total of 1 campaign IDs have been provided for processing.
[+] GoReport will process the following campaign IDs: 19
[+] Now fetching results for Campaign ID 19 (1/1).
[+] Success!
[+] Building the report -- you selected a Excel/xlsx report.
[+] Created row for 1 of 2.
[+] Created row for 2 of 2.
[+] Finished writing events summary...
[+] Detailed results analysis is next and will take some time if you had a lot of targets...
[+] Processed detailed analysis for 1 of 2.
[+] Processed detailed analysis for 2 of 2.
[+] Finished writing detailed analysis...
[+] Done! Check 'Gophish Results for Zero_Campaign.xlsx' for your results.

C:\Users\dumbledore\Desktop\Goreport-master>
```

You should now have a report file in the GoReport folder. As you explore this file, you'll see that you have more details than what is exported out of the box.



1	**Campaign Results For:**	Zero_Campaign
2	**Status**	In progress
3	**Created**	2019-12-20T14:09:45.015004076Z
4	**Started**	2019-12-20T13:56:00Z
5		
6	**Campaign Details**	
7	**From**	snape@marauders.in
8	**Subject**	I'm a phishing email. Please don't click on me.
9	**Phish URL**	http://10.70.70.15
10	**Redirect URL**	https://www.youtube.com/watch?v=R12_y2BhKbE
11	**Attachment(s)**	None Used
12	**Captured Passwords**	True
13	**Stored Passwords**	False
14		
15	**High Level Results**	
16	**Total Targets**	2
17	The following totals indicate how many events of each type Gophish recorded:	
18	**Total Opened Events**	2
19	**Total Clicked Events**	4
20	**Total Submitted Data Events**	
21	**Total Report Events**	2
22	The following totals indicate how many targets participated in each event type:	
23	**Individuals Who Opened**	2
24	**Individuals Who Clicked**	1
25	**Individuals Who Submitted Data**	1

Fig: *Overview Tab*



Summary of Events						
Email Address	**Open**	**Click**	**Creds**	**Report**	**OS**	**Browser**
luna@hogwarts.com	TRUE	FALSE	FALSE	TRUE	N/A	N/A
neville@hogwarts.com	TRUE	TRUE	TRUE	TRUE	Edge 18.18362	Windows 10

Fig: *Summary Tab*
```



| | | | | |
|---|---|---|---|---|
| **Detailed Analysis** | | | | |
| **Neville Longbottom** | | | | |
| neville@hogwarts.com | | | | |
| Sent on 2019-12-20 at 14:09:45 | | | | |
| Email Preview at 2019-12-20 14:18:53 | | | | |
| **Email Link Clicked** | | | | |
| **Time** | **IP** | **Location** | **Browser** | **Operating System** |
| 2019-12-20 14:21:36 | 10.50.10.51 | Unknown, Unknown, Unknown | Edge 18.18362 | Windows 10 |
| **Email Link Clicked** | | | | |
| **Time** | **IP** | **Location** | **Browser** | **Operating System** |
| 2019-12-20 14:33:13 | 10.50.10.51 | Unknown, Unknown, Unknown | Edge 18.18362 | Windows 10 |
| **Email Link Clicked** | | | | |
| **Time** | **IP** | **Location** | **Browser** | **Operating System** |
| 2019-12-20 14:35:13 | 10.50.10.51 | Unknown, Unknown, Unknown | Edge 18.18362 | Windows 10 |
| **Email Link Clicked** | | | | |
| **Time** | **IP** | **Location** | **Browser** | **Operating System** |
| 2019-12-20 14:39:41 | 10.50.10.51 | Unknown, Unknown, Unknown | Edge 18.18362 | Windows 10 |
| **Submitted Data Captured** | | | | |
| **Time** | **IP** | **Location** | **Browser** | **Operating System** |
| 2019-12-20 14:39:45 | username:'Neville' | Unknown, Unknown, Unknown | Edge 18.18362 | Windows 10 |

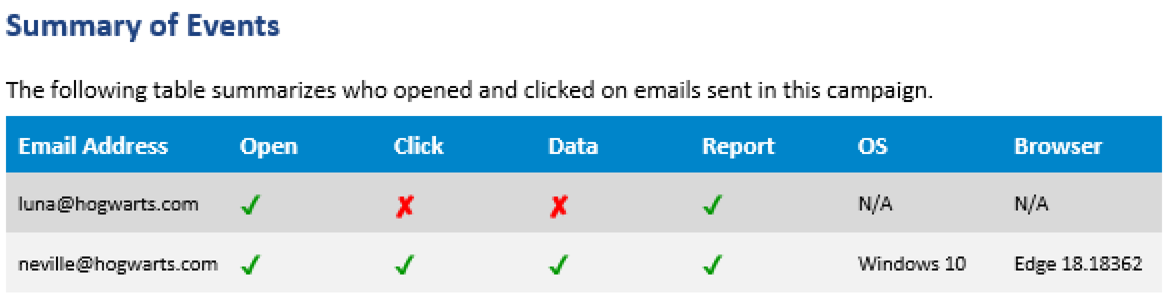
**<u>Fig</u>**<u>: *Event Details Tab*</u>



| | |
|---|---|
| **Recorded Browsers Based on User-Agents:** | |
| **Browser** | **Seen** |
| Edge 18.18362 | 4 |
| | |
| **Record OS From Browser User-Agents:** | |
| **Operating System** | **Seen** |
| Windows 10 | 4 |
| | |
| **Recorded Locations from IPs:** | |
| **Locations** | **Seen** |
| Unknown, Unknown, Unknown | 2 |
| | |
| **Recorded IPs:** | |
| **IP Address** | **Seen** |
| 10.50.10.53 | 2 |
| 10.50.10.51 | 4 |
| **Recorded IPs and Locations:** | |
| **IP Address** | **Location** |
| 10.50.10.53 | Unknown, Unknown, Unknown |
| 10.50.10.51 | Unknown, Unknown, Unknown |

**<u>Fig</u>**<u>: *Stats tab.*</u>

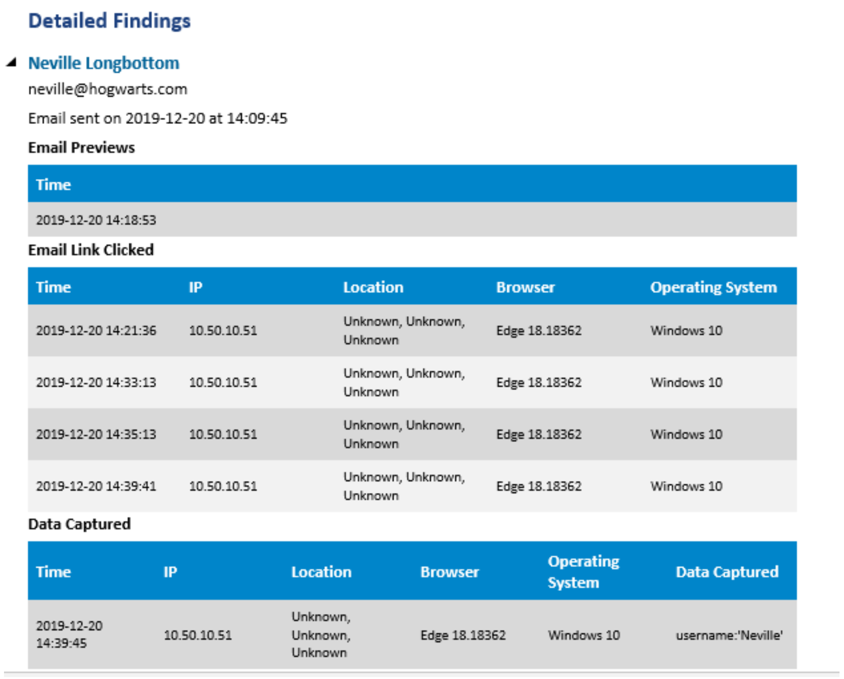
The Word version gives us a report in an Executive summary format.
Command: `python Goreport.py --id=<CampaignID> --format=word`



## Summary of Events

The following table summarizes who opened and clicked on emails sent in this campaign.

| Email Address | Open | Click | Data | Report | OS | Browser |
|---|---|---|---|---|---|---|
| luna@hogwarts.com | ✓ | ✗ | ✗ | ✓ | N/A | N/A |
| neville@hogwarts.com | ✓ | ✓ | ✓ | ✓ | Windows 10 | Edge 18.18362 |


**Detailed Findings**

**Neville Longbottom**

neville@hogwarts.com

Email sent on 2019-12-20 at 14:09:45

**Email Previews**

| Time |
|---|
| 2019-12-20 14:18:53 |

**Email Link Clicked**

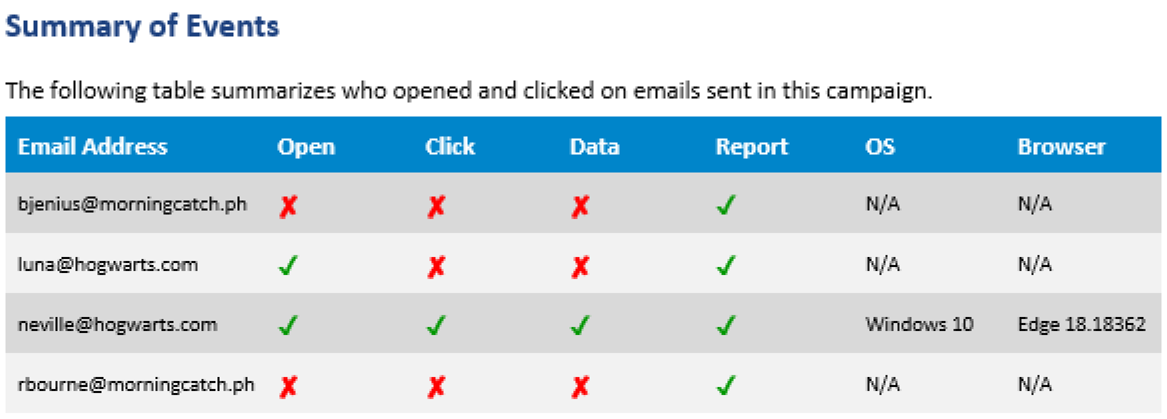
| Time | IP | Location | Browser | Operating System |
|---|---|---|---|---|
| 2019-12-20 14:21:36 | 10.50.10.51 | Unknown, Unknown, Unknown | Edge 18.18362 | Windows 10 |
| 2019-12-20 14:33:13 | 10.50.10.51 | Unknown, Unknown, Unknown | Edge 18.18362 | Windows 10 |
| 2019-12-20 14:35:13 | 10.50.10.51 | Unknown, Unknown, Unknown | Edge 18.18362 | Windows 10 |
| 2019-12-20 14:39:41 | 10.50.10.51 | Unknown, Unknown, Unknown | Edge 18.18362 | Windows 10 |

**Data Captured**

| Time | IP | Location | Browser | Operating System | Data Captured |
|---|---|---|---|---|---|
| 2019-12-20 14:39:45 | 10.50.10.51 | Unknown, Unknown, Unknown | Edge 18.18362 | Windows 10 | username:'Neville' |

To combine multiple campaigns in a single report, specify multiple campaign IDs:
**Command Example**: `python GoReport.py --id=19,20 --format=word --combine`

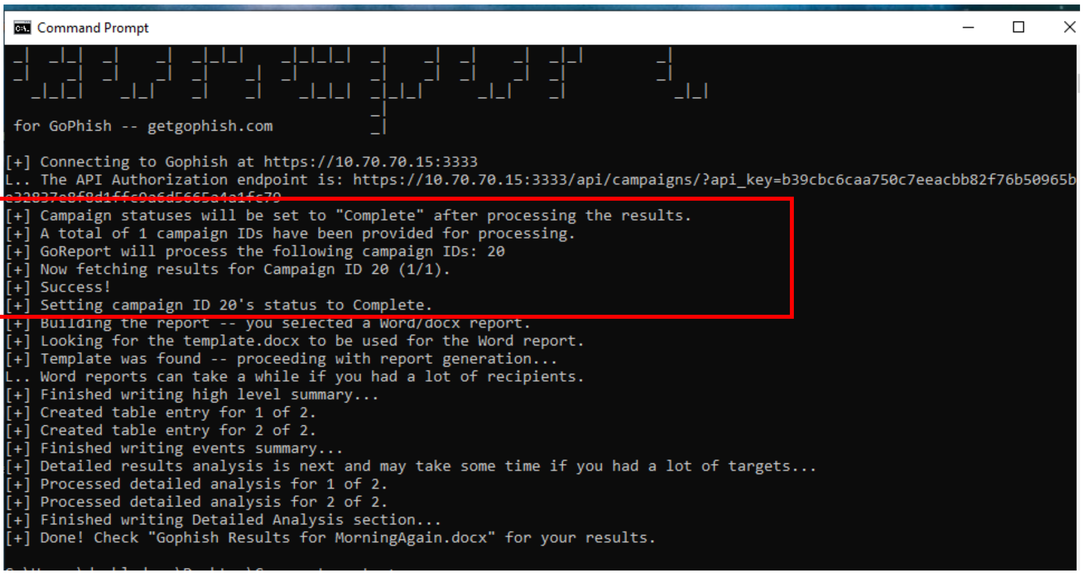
Do note that the name of the report will be the first campaign ID entered in the command.


**Summary of Events**

The following table summarizes who opened and clicked on emails sent in this campaign.

| Email Address | Open | Click | Data | Report | OS | Browser |
|---|---|---|---|---|---|---|
| bjenius@morningcatch.ph | ✗ | ✗ | ✗ | ✓ | N/A | N/A |
| luna@hogwarts.com | ✓ | ✗ | ✗ | ✓ | N/A | N/A |
| neville@hogwarts.com | ✓ | ✓ | ✓ | ✓ | Windows 10 | Edge 18.18362 |
| rbourne@morningcatch.ph | ✗ | ✗ | ✗ | ✓ | N/A | N/A |

You can also mark Campaigns as complete while retrieving reports with the `--complete` flag.
Command Example: `python GoReport.py --id=20 --format=word --complete`



```
_|  _|_|  _|    _|  _|_|_|    _|_|_|_|  _|    _|  _|    _|  _|_|        _|
_|    _|  _|    _|  _|    _|  _|        _|    _|  _|    _|  _|          _|
  _|_|_|    _|_|    _|    _|    _|_|_|  _|_|_|    _|_|    _|            _|_|
                                        _|
 for GoPhish -- getgophish.com          _|

[+] Connecting to Gophish at https://10.70.70.15:3333
L.. The API Authorization endpoint is: https://10.70.70.15:3333/api/campaigns/?api_key=b39cbc6caa750c7eeacbb82f76b50965b
e32837e8f8d1ffc9a6d5665a4a1fc79
[+] Campaign statuses will be set to "Complete" after processing the results.
[+] A total of 1 campaign IDs have been provided for processing.
[+] GoReport will process the following campaign IDs: 20
[+] Now fetching results for Campaign ID 20 (1/1).
[+] Success!
[+] Setting campaign ID 20's status to Complete.
[+] Building the report -- you selected a Word/docx report.
[+] Looking for the template.docx to be used for the Word report.
[+] Template was found -- proceeding with report generation...
L.. Word reports can take a while if you had a lot of recipients.
[+] Finished writing high level summary...
[+] Created table entry for 1 of 2.
[+] Created table entry for 2 of 2.
[+] Finished writing events summary...
[+] Detailed results analysis is next and may take some time if you had a lot of targets...
[+] Processed detailed analysis for 1 of 2.
[+] Processed detailed analysis for 2 of 2.
[+] Finished writing Detailed Analysis section...
[+] Done! Check "Gophish Results for MorningAgain.docx" for your results.
```

Our Campaign which was **In Progress** earlier is now marked as **Complete**d and archived.

Name	Created Date	Status
MorningAgain	December 23rd 2019, 9:18:45 pm	Completed

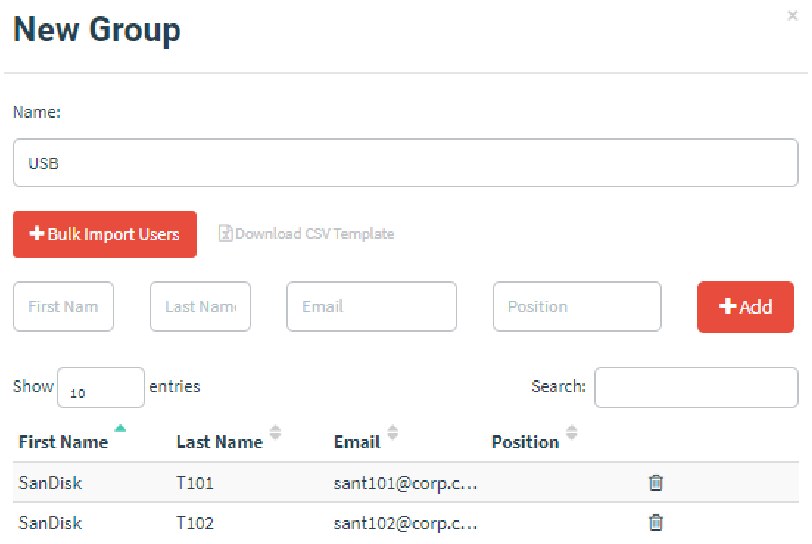
You can explore more GoReport functionality by getting yourself familiar with the command switches. Do read the ReadMe at https://github.com/chrismaddalena/Goreport to deliver powerful reports for your organization's needs.

USB Phishing with Gophish

Security around removable devices has been around for years now. Most Endpoint security tools allow you to block them, even Windows has a group policy that can provide limited or no access to mass storage media. However there are business cases where removable media access may be required or has been provisioned and forgotten. How do you audit that? One way to do that is to get the awesome USB Rubber Ducky [https://shop.hak5.org/products/usb-rubber-ducky-deluxe], combine with Metasploit and do some drops. But could we achieve something similar with Gophish and without expense? **Chris Merkel** wrote an article on Medium [https://medium.com/@chrismerkel/conducting-usb-drop-tests-with-gophish-44cc7e1a88b9] around conducting drop tests with Gophish that is quite interesting. I decided to take my own approach to building this. The idea however remains the same:

- We use a couple of USB drives and make users in Gophish for each of them.
- Create a campaign to for these USB Users.
- Build a Macro based excel with an enticing file and pack with other files.
- Drop the USB and observe the results.

First we create a USB group, and create users. Since these are not real users, you can just name them anything. I would suggest either using the model number and/or location that can be usable during the reporting phase.

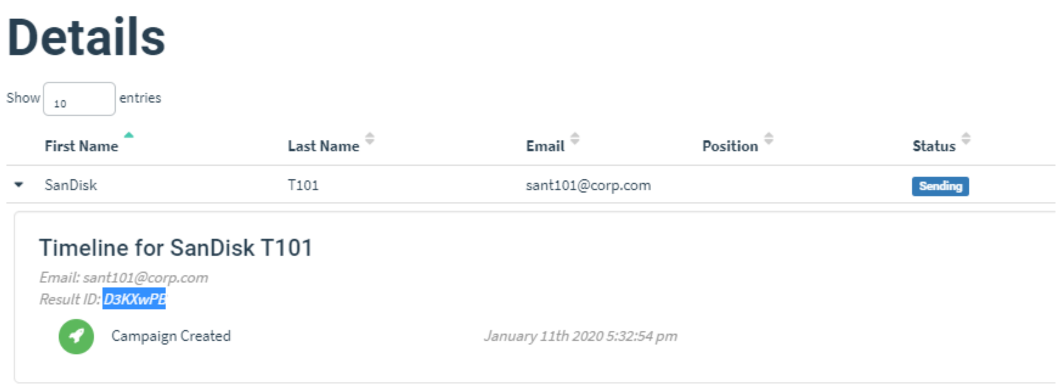


Name: USB

Show 10 entries

First Name	Last Name	Email	Position
SanDisk	T101	sant101@corp.c...	
SanDisk	T102	sant102@corp.c...	

We create a campaign and associate a Landing page (I use the one we created earlier) and attach a sending profile. To be honest, the Sending profile doesn’t really matter as these email addresses are undeliverable. What we need is the RID that will get created once the campaign is launched. Copy the RID for each USB drive once the campaign launches.



Details

Show 10 entries

First Name	Last Name	Email	Position	Status
SanDisk	T101	sant101@corp.com		Sending

Timeline for SanDisk T101

Email: sant101@corp.com

Result ID: D3KXwPB

Campaign Created January 11th 2020 5:32:54 pm

Now, in a normal campaign, whenever you click on a link, Gophish redirects to a unique url for that particular user. This is basically a construct of the Landing page and the RID of the user - http://<LANDINGPAGE>/?rid=<RID>

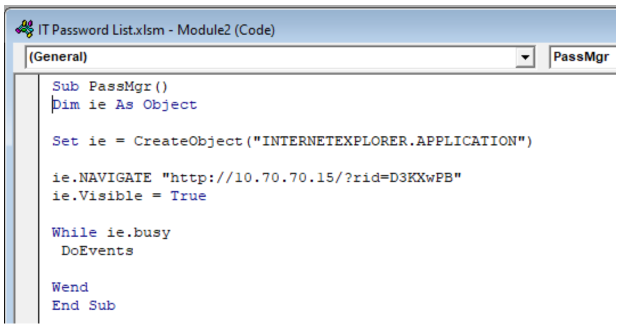
In my campaign, the URL for the first drive will be http://10.70.70.15/?rid=D3KXwPB . You should construct similar URLs for your other drives by changing the RID section.

Next we create the enticing document that triggers this URL. A document needs to be enticing enough yet not obvious to trigger a reaction from the user. I really liked the Password manager idea that Chris mentioned in his article, but you could also go for appraisal list etc. Just ensure you pad the drive with other dummy files as well.

Let's build our fake password manager! The secret sauce here is a simple macro that opens a web browser and hits our landing page. You can use the below macro code and just change the url per usb drive.

Macro Code

```
Sub PassMgr()
Dim ie As Object

Set ie = CreateObject("INTERNETEXPLORER.APPLICATION")

ie.NAVIGATE "http://10.70.70.15/?rid=D3KXwPB "
ie.Visible = True

While ie.busy
 DoEvents

Wend
End Sub
```

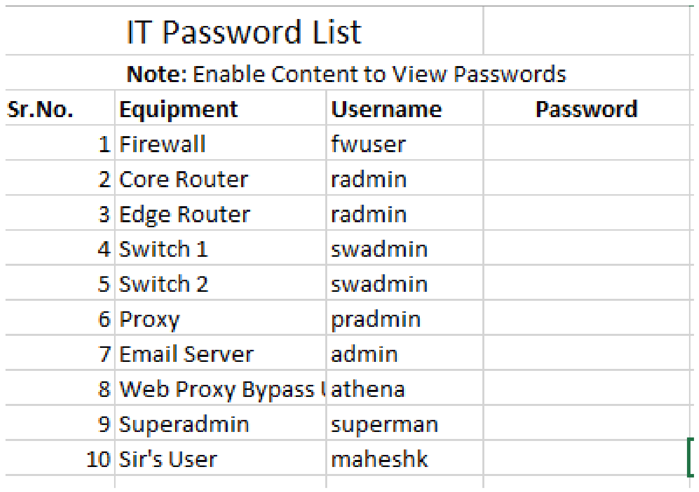


IT Password List.xlsm - Module2 (Code)

(General) PassMgr

```
Sub PassMgr()
Dim ie As Object

Set ie = CreateObject("INTERNETEXPLORER.APPLICATION")

ie.NAVIGATE "http://10.70.70.15/?rid=D3KXwPB"
ie.Visible = True

While ie.busy
 DoEvents

Wend
End Sub
```

We will create an excel document that holds some dummy rows with fake information. Note, leave the password column blank for now.

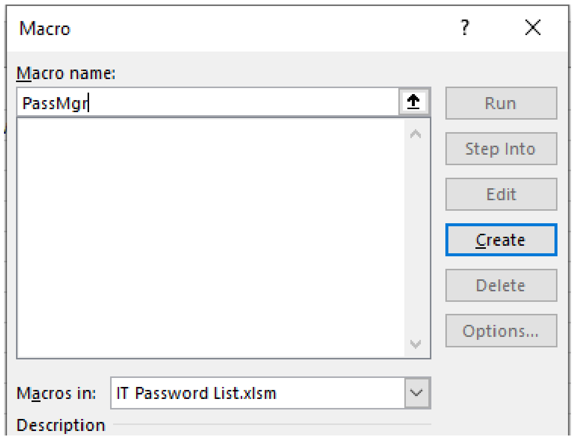


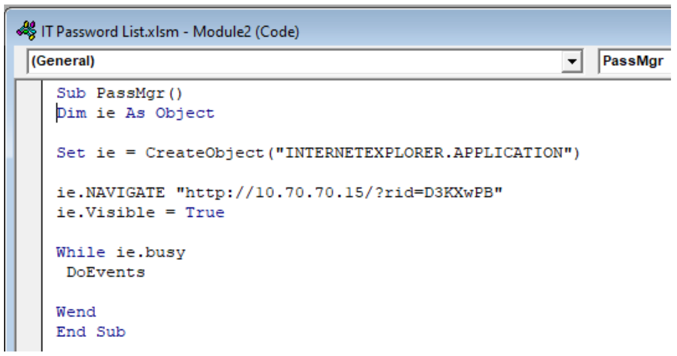
IT Password List

Note: Enable Content to View Passwords

Sr.No.	Equipment	Username	Password
1	Firewall	fwuser	
2	Core Router	radmin	
3	Edge Router	radmin	
4	Switch 1	swadmin	
5	Switch 2	swadmin	
6	Proxy	pradmin	
7	Email Server	admin	
8	Web Proxy Bypass U	athena	
9	Superadmin	superman	
10	Sir's User	maheshk	

To add a macro, you will need to enable the Developer tab in the ribbon. [https://www.excelcampus.com/vba/enable-developer-tab/] Once added, Click on **Developer Tab -> Macros -> Enter** a Macro name and click **Create**. Paste the macro code in the editor and Save the document as a **Macro Enabled Excel document**.

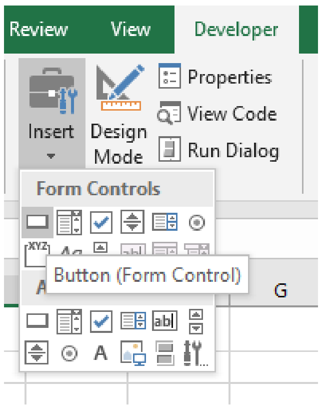


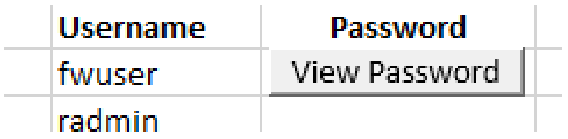


```
Sub PassMgr()
Dim ie As Object

Set ie = CreateObject("INTERNETEXPLORER.APPLICATION")

ie.NAVIGATE "http://10.70.70.15/?rid=D3KXwPB"
ie.Visible = True

While ie.busy
 DoEvents

Wend
End Sub
```

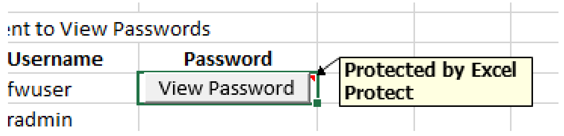
Once created, you can manually run the macro and verify if it opens IE and hits the landing page. Now to trigger user action, you could add dummy characters in the password field like Chris describes in his article. Tell the user they need to enable content to view them, then create an Automatic macro that triggers when the user clicks enable content. However, this isn't really that enticing. Rather than just one auto-trigger, how about making this more realistic and having multiple triggers? Rather than dummy characters, how about we create a button that when clicked triggers the macro and opens the landing page? Simple and effective!

To make this button, Go to the **Developer tab -> Insert -> Form Control -> Button**. You will get a + draw tool. Draw a button in the first empty password cell. It will prompt you to Assign a macro. Assign the macro we created previously and rename the button to something like – View Password. To be more realistic, you could add a comment saying Protected by Excel Protect or some DLP.



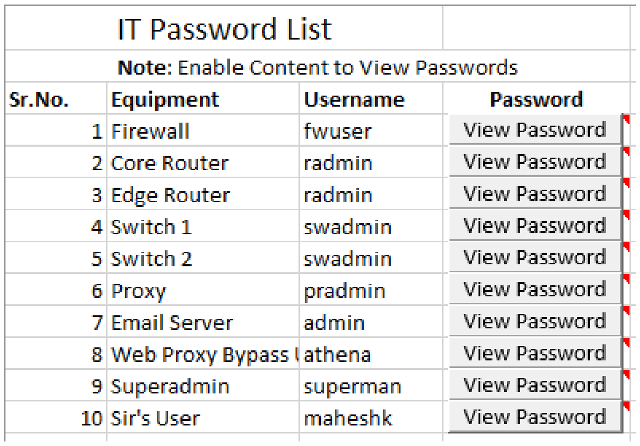


Username	Password
fwuser	View Password
radmin	



nt to View Passwords		
Username	Password	
fwuser	View Password	Protected by Excel Protect
radmin		

Now copy and paste this cell to the other empty cells and voila! You have just completed your fake password manager.

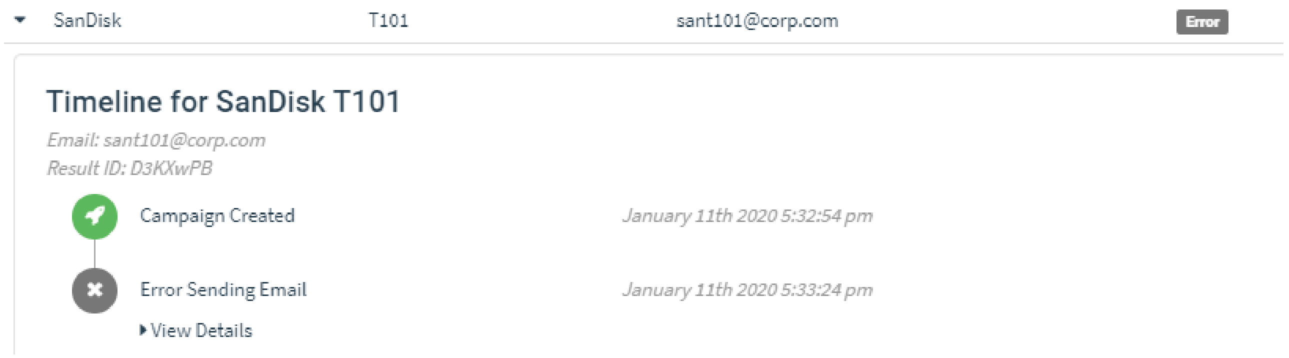


IT Password List

Note: Enable Content to View Passwords

Sr.No.	Equipment	Username	Password
1	Firewall	fwuser	View Password
2	Core Router	radmin	View Password
3	Edge Router	radmin	View Password
4	Switch 1	swadmin	View Password
5	Switch 2	swadmin	View Password
6	Proxy	pradmin	View Password
7	Email Server	admin	View Password
8	Web Proxy Bypass U	athena	View Password
9	Superadmin	superman	View Password
10	Sir's User	maheshk	View Password

To verify that the button really works, click on it. It should open IE and pop up the landing page. Do remember that if you use Macro filtering in your endpoint security, that could block the payload. In that case you would need to either allow this particular file to execute or monitor block events for our macro.



SanDisk T101 sant101@corp.com Error

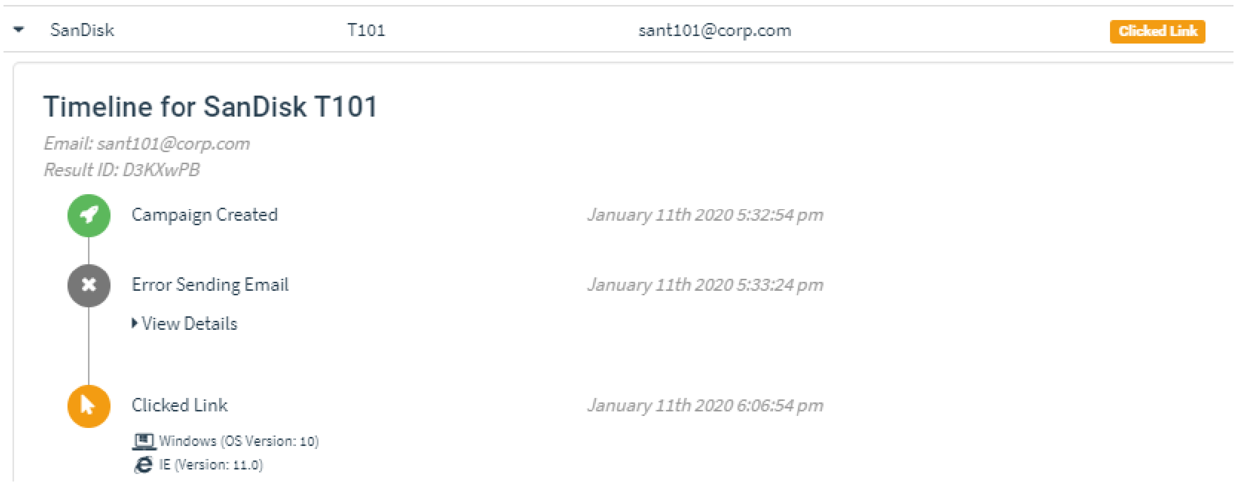
Timeline for SanDisk T101

Email: sant101@corp.com
Result ID: D3KXwPB

Campaign Created	*January 11th 2020 5:32:54 pm*
Error Sending Email	*January 11th 2020 5:33:24 pm*

View Details

Fig: *Gophish status before clicking on View Password*



SanDisk T101 sant101@corp.com Clicked Link

Timeline for SanDisk T101

Email: sant101@corp.com
Result ID: D3KXwPB

Campaign Created *January 11th 2020 5:32:54 pm*

Error Sending Email *January 11th 2020 5:33:24 pm*

View Details

Clicked Link *January 11th 2020 6:06:54 pm*

Windows (OS Version: 10)
IE (Version: 11.0)

<u>Fig</u>: *<u>Gophish status after clicking on View Password</u>*

And that's it! Your drop is ready. Copy it onto the usb, add the dummy files and drop it. Remember to change the URL in the macro to the RID of the usb stick you are dropping into. You may get users that take it home and try it on a personal device. That is OK. You will know exactly which USB was used. The main goal is to identify if users are trying to plug these devices into corporate devices.

Tip*: You can use my Fake Password Manager file and modify it to use it in your environment.*
https://github.com/theashishchalke/booktools/raw/master/IT%20Password%20List.xlsm

In Closing

Your users have started reporting phishing mails, now what? There are numerous resources on the internet that can help you investigate these emails. Do checkout http://gotphish.com by @SwiftonSecurity [https://twitter.com/swiftonsecurity] that has various resources on evaluating and reporting phishing emails. If you're on exchange as your mailing solution, also check out the SwiftFilter [https://github.com/SwiftOnSecurity/SwiftFilter] that has Transport rules to detect and enable response to basic phishing.

Ideally Gophish is just one cog in your awareness program which, as I've said before, should include Training and awareness assets. Just running the simulations will not bring value except some pretty reports on how poor your users are at security. Focus on using this data to deliver better and more focused awareness. This is what I believe should be a cycle for actually making the user a layer in your cyber-defense.

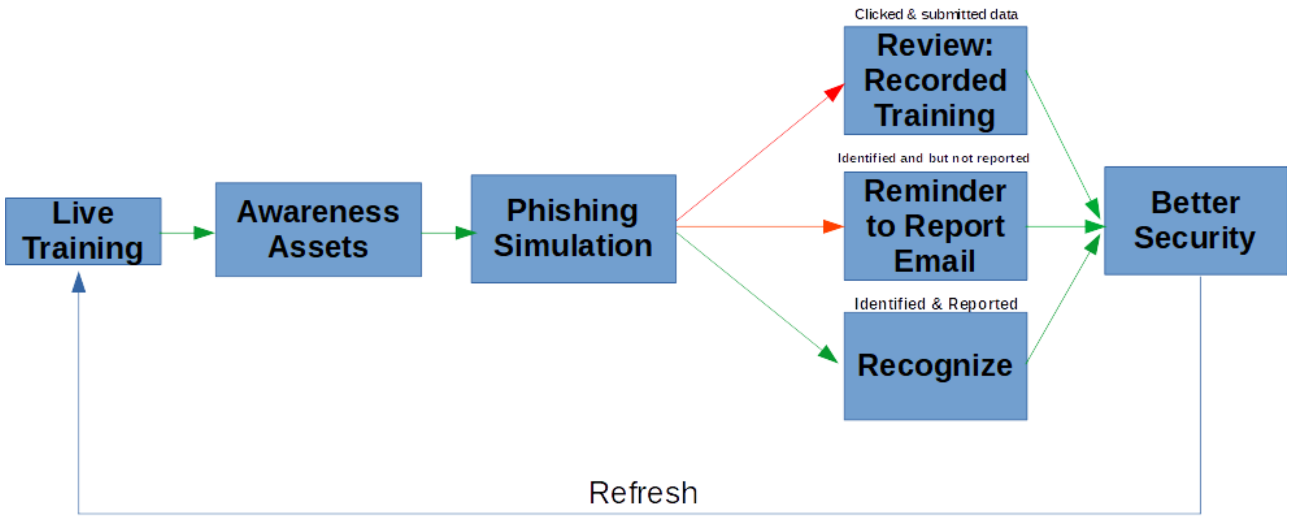


Fig: *The Training Loop*

Training is easier said than done. Engaging a live outside speaker really makes a buzz, but everyone has different mindsets. Some like DIY, some engage outside consultants. Whatever the category you fit in, if you need prebuilt assets, that you can redirect users to, try the below links:

- Google Education – Stay Safe from Phishing and Scams https://www.youtube.com/watch?v=R12_y2BhKbE
- SANS CyberSecurity Awareness - https://www.youtube.com/watch?v=5RHeJAEdiEc
- My github repository [this is in progress, will add assets] - https://github.com/theashishchalke/phishawassets

If you have issues in Gophish, go through Issues [https://github.com/gophish/gophish/issues] to see if anyone else has had the same issue or create a new one. You can also reference the documentation here - https://getgophish.com/documentation/

Finally, congrats on helping make your organization and the world safer. By educating users, you have ensured that they will have this security mindset not only within your organization but also in their personal lives! Congrats **PhishMan**!

Credits

If you liked this book/article/chapter or have some errata / suggestions, do connect with me on LinkedIn - https://www.linkedin.com/in/ashishchalke/ or Twitter - https://twitter.com/theashishchalke or my (future) website – https://ashishchalke.com

This wouldn't have been possible without –

- Jordan Wright and his amazing Gophish - https://twitter.com/jw_sec.
- Koromicha Kifarunix and their service setup guide - https://kifarunix.com/install-gophish-on-ubuntu-18-04-debian-9-8/
- GoReport by Chris Maddelena - https://github.com/chrismaddalena/Goreport
- Chris Merkel article on USB Drop - https://medium.com/@chrismerkel/conducting-usb-drop-tests-with-gophish-44cc7e1a88b9
- Python knowledge gained from –
    - Python Crash Course – Eric Matthes
    - Python Bible Udemy – Zayed Yehia - https://www.udemy.com/course-dashboard-redirect/?course_id=903378

Other Publications:

You can also find links to my works here:

- My CISSP Notes (tbh exam study only) - https://github.com/theashishchalke/cisspstudy-guide/raw/master/The%20OPEN%20CISSP%20STUDY%20GUIDE-FINAL.pptx
- Zero Degrees book series - https://github.com/theashishchalke/zerodegrees

ZERO DEGREES