LINUX VOICE
TUTORIAL

PENETRATION TESTING: SOCIAL ENGINEER TOOLKIT

BEN EVERARD

Don a stylish black hat and open up the software of choice for the discerning technical con man.

WHY DO THIS?

- Learn the tools of online scammers so you can protect yourself.
- Begin a lucrative career as a penetration tester.
- Get a better understanding of the technologies that underpin the web.

It doesn't matter how good your computer security systems are if your users just let attackers know how to log in. Social engineering is the black art of persuading victims to tell you everything you need to know to break into their computers. Sometimes this can mean persuading them to hand over usernames or passwords, sometimes it can mean granting you physical access to their computer, and sometimes it can mean deleting any incriminating evidence. In truth, the skilled social engineer can persuade victims to bypass all sorts of computer security that would be hard to compromise using just technical means.

The *Social-Engineer Toolkit* (*Set*) is a piece of software that helps you set up some social engineering attacks.

Stealing credentials

One of the most popular uses of social engineering is tricking people into revealing their login details. This could be through a simple confidence scheme, or through some technical trickery. The *Social-Engineer Toolkit* provides some ways to make this easier.

The easiest way to try *Set* is using a security-focused distro that comes with it already installed. Kali Linux is an excellent option for this (see boxout).

Alternatively, you can grab *Set* from the Git repository. First you'll need to make sure you've installed the **git** command through your package manager. On Debian and Ubuntu-based systems, this is done with:

```
sudo apt-get install git
```

Then you can clone the repository with:

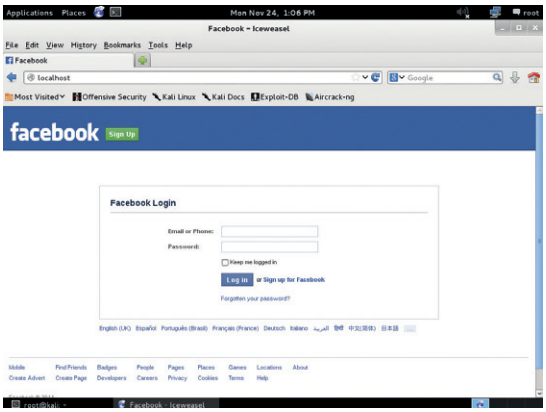
```
git clone https://github.com/trustedsec/social-engineer-toolkit/set/
cd set
sudo python setup.py install
```

You can now use the command **setoolkit** to access the various features of *Set*. The attack we're going to run is called credential harvesting. It will create a clone of a website with login details, then we'll have to try and get people to log in. When people do, it'll save their username and password, then forward them on to the real site where they'll be prompted to log in again. Most people will simply assume that they entered their password incorrectly the first time, or that there's been some form of network glitch, and log in again.

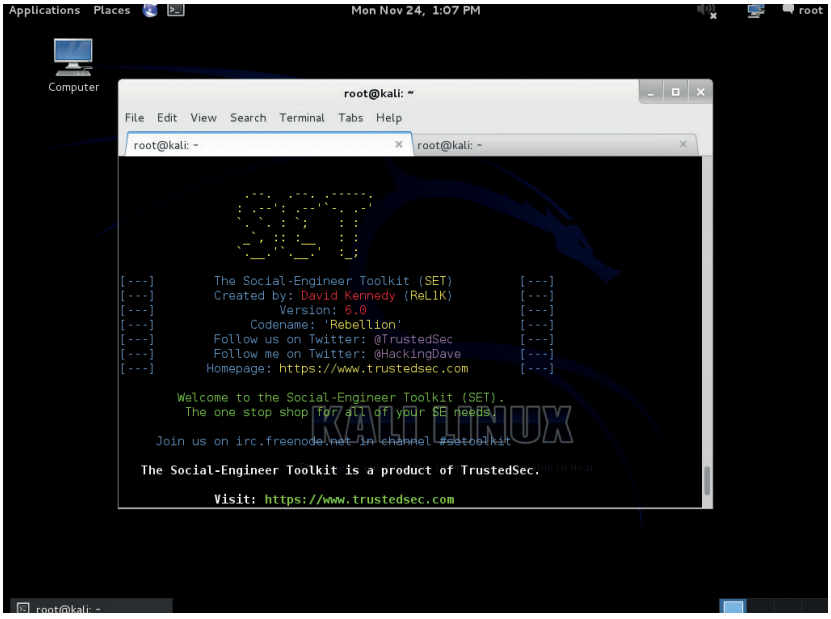
In order to run this attack, you'll need a webserver with PHP running on your local machine. You can get this in Debian- and Ubuntu-based systems with:

```
sudo apt-get install apache2 php5 libapache2-mod-php5
```

If you're trying this from Kali Linux, you'll already have all this installed. If you're using another distro and can't find *Apache* in your package manager, it's sometimes in a package called **httpd**.



Our login-stealing Facebook clone. Would you be fooled?



The *Social-Engineer Toolkit* can do far more than just clone websites. The best documentation is at **www.social-engineer.org** (under Framework).

Legalities

It should go without saying that the techniques we've discussed in this tutorial can be illegal if used against unsuspecting victims. While it's fine to try them out by yourself on your own local network, using them to try to break into computer networks can have very serious legal consequences. Just don't do it.

Now with everything set up, let's set up the cloned website. First, start *Set* running with root permissions:

```
sudo setoolkit
```

You may get a warning message about *Metasploit* not being installed. This isn't a problem for us since we won't be using any of the attacks that depend on it. However, if you want to fully investigate more of the capabilities of *Metasploit*, it's worth installing this as well (see boxout).

Note that these instructions will overwrite **/var/www/index.html** (or **/var/www/html/index.html**), so if you're already hosting any website on the machine, it's probably best to try this out in a live environment or a virtual machine.

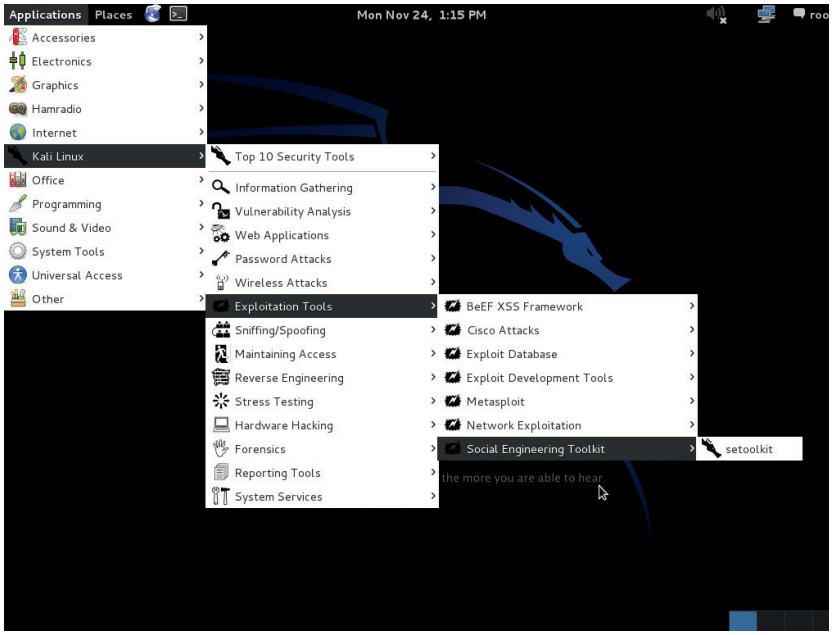
Setoolkit is controlled via a text-based menu system (which means it can easily be controlled remotely should you need to). The cloned website credential harvester is under: 1) Social-Engineering Attacks > 2) Website Attack Vectors > 3) Credential Harvester Attack Method > 2) Site Cloner.

Once you've selected this, you just need to enter the IP address of the machine you're running the attack from. If you're running the attack on a LAN, then this should be the local IP address of the machine on which you're running *Set*. You can find this out by running **sudo ifconfig**, and looking for something like:

```
wlan0     Link encap:Ethernet  HWaddr 00:13:e8:3d:92:7b
          inet addr:192.168.0.4  Bcast:192.168.0.255
Mask:255.255.255.0
```

In this case, the IP address for wlan0 (wireless lan – if you're using wired Ethernet, this will probably be eth0) is 192.168.0.4.

If you've got a publicly routable IP address, then you could also enter this here, but you should be careful: running a test attack on a local network is usually fine (see boxout on legalities), but if you're doing anything on the public internet, you're far more likely to run into problems with the law.



Set is just one of many penetration testing tools included in Kali (see boxout).

After you've entered this, you should enter the URL that you want to clone. For a test, we used Facebook (**https://facebook.com**), but you could put any website with a login here.

Once you've done that, it'll automatically make a copy of the website, host it locally, and set it to harvest the credentials.

You should now be able to point your browser to localhost and see the cloned site. If you don't see the cloned site, then it could be that *Set* has put the files in the wrong place. By default, it will put them in **/var/www**, but many modern Linux systems use **/var/www/html** as the web root. The easiest way to tell if this is the problem is by opening a new terminal and **cd**'ing to **/var/www** and seeing if the **html** folder exists. If it does, just move all three files created by *Set* to **html**:

Metasploit

The *Social-Engineer Toolkit* is designed to work hand-in-hand with *Metasploit*. *Metasploit* isn't so much a piece of software, as a complete framework for penetration testing. It includes everything from exploits to software for recording your attacks. *Set* uses some of *Metasploit*'s features and exploits, so unless you have *Metasploit* installed, you won't get the full functionality of *Set*.

The easiest way try out *Metasploit* is in a live environment that has it already installed, like Kali. However, you can install it on a regular Linux distro. To make this a bit easier on Debian-and Ubuntu-based systems, there's a script to automate installation. You can grab it from GitHub with:

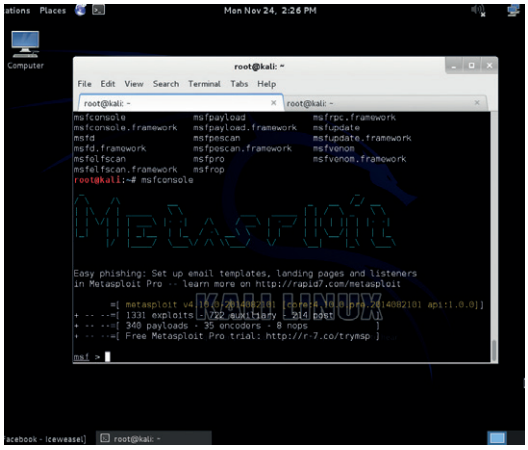
```
git clone https://github.com/darkoperator/MSF-Installer.git msf
```

Then, to install *Metasploit*, you just need to run:

```
cd msf
./msf_install.sh
```

You can find details on how to install it in Fedora at **http://fedoraproject.org/wiki/Metasploit**.

There's also a version of *Metasploit* with a HTML front-end. You can grab this from **www.rapid7.com/products/metasploit/download.jsp**.



msfconsole (shown here) is the usual interface to *Metasploit*, but there are others including *Armitage* (graphical) and **msfcli** (for using in scripts).

Kali Linux

If you do quite a bit of penetration testing, it's worthwhile setting up your own working environment. This is quite a worthwhile exercise; there's plenty of software that can be useful, and you'll be able to pick the ones that are useful to you. However, if you're just getting started, or if you just want to dabble, it's useful to use a ready-made penetration testing environment. In last month's Distrohopper, we took a look at Backbox. This is one good option, but by far the most popular is Kali (formerly BackTrack Linux). It comes in flavours for small ARM machines as well as x86 desktops.

You can install it, but you can also run it live, and it has almost every open source (and some closed source) penetration tool set up and ready to run. You can grab an ISO from **www.kali.org**, and then use it like any other distro. It also runs well in a virtual machine if you want to separate your penetration testing from your main desktop.

```
mv /var/www/index.html html
mv /var/www/post.php html
mv harvester* html
```

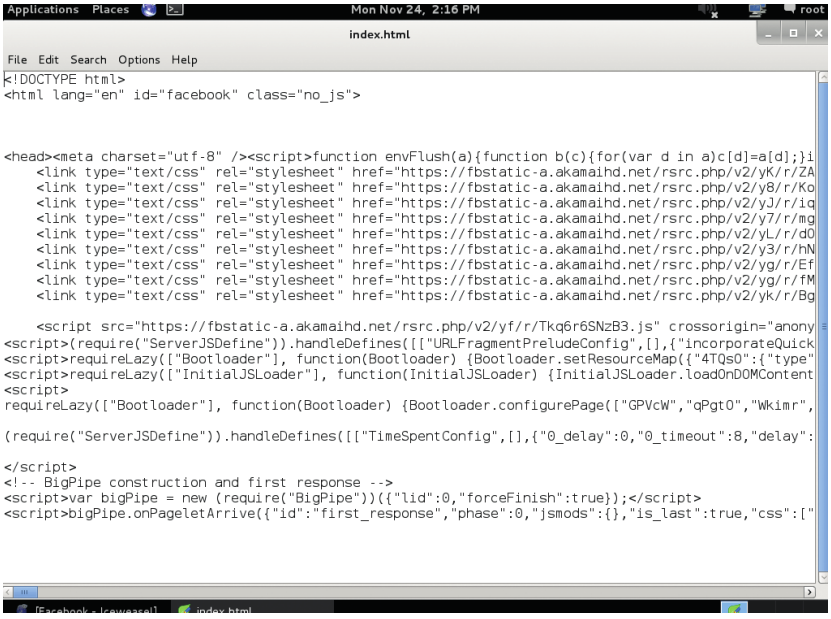
Now you should be able to point your browser to localhost and see the cloned site. If you enter dummy details in there, you should find that you get forwarded to the real Facebook page, and that whatever you enter is copied to the harvester file in the webroot.

After a visitor goes to your site, you should see an entry like the following in the harvester file:

```
Array
(
    [lsd] => AVrTM83X
    [display] =>
    [enable_profile_selector] =>
    [legacy_return] => 1
    [profile_selector_ids] =>
    [trynum] => 1
    [timezone] => 0
    [lgnrnd] => 125137_iouC
    [lgnjs] => 1416689527
    [email] => test@test.com
    [pass] => test
    [default_persistent] => 0
)
```

This contains all the POST data that the user sent back to the website. In this case, the important fields are **email** and **pass**.

Set just sets up the HTML and PHP file. The actual attack could be run on any computer with a webserver installed.



Getting visitors

The problem now is to get victims to go to your fake site. There are a few options here. Perhaps the simplest is simply tricking them to click on a link. The classic approach here is to send them an email with a link to Facebook that actually points to your clone. This, however, will look a little strange to anyone who clicks on the link because the URL in the address bar will be an IP number not the proper domain.

One way around this is to register a domain that looks similar to the one in you're cloning. For example, **www.facebock.com**. A casual glance won't show that there's anything wrong with that. What's more, you could even get a real SSL certificate for it so that you could encrypt the connection and make it look even less suspicious. For Facebook, you're unlikely to find a domain that looks similar that's not already taken. This approach does have the downside that it costs money, and leaves a paper trail.

Another approach is to attack the Domain Name System (DNS) in such a way that when the user enters a domain name, they get pointed to your site instead of the real site. The easiest way of doing this requires modifying a file on the victim's machine. This could be done in a few ways. You could get the victim to use your machine (with the attack already set up). If you can get physical access to their machine, you could do it using a live distro to bypass any passwords they may have (unless their hard drive is encrypted).

Whenever you type in a domain name, like **www.facebook.com** or **google.co.uk**, your computer first checks a file called **hosts**. If this domain isn't in that file, it then sends a message to its DNS server asking which machine corresponds to the domain name. To get the victim's computer to send the request to our malicious server, all we have to do is

Protect yourself

If the attacker does this well, it can be hard to spot a cloned site. One obvious giveaway is a lack of SSL on a login page. However, even this can be faked if the attacker is either using a real domain, or has managed to get access to your computer (where they could install a fake certificate).

You can avoid getting caught out by fake domains by always typing in the domain of any important sites rather than just clicking on links, but this won't protect you if they've managed to hijack your DNS connection. The best protection against an attacker installing fake certificates on your computer is full disk encryption.

Important sites should use two-factor authentication. This combines usual login details with a second form of security (such as a code that is sent via SMS). Using something like this means that the credentials the attacker steals won't be sufficient to log them in.

add an entry to their **hosts** file. On Linux systems, this file is in **/etc/hosts**. In most versions of Windows, it's in **Windows\System32\drivers\etc**. Entries are simply a domain name, then a space (or tab), then the IP address that domain should resolve to.

If we hijack **www.facebook.com**, then whenever the user goes to **www.facebook.com**, they will be directed to our site. However, this causes a problem because they won't be able to get to the actual Facebook site, and so will quickly see that there is a problem. However, if we redirect **login.facebook.com** to our site, we can then forward them on to **www.facebook.com**, and it should be fairly trivial to persuade a victim to click on a link to **login.facebook.com** (*Set* is, after all, an aid to social engineering, not a complete hacking solution).

The line you need to add to the **hosts** file is:

```
192.168.0.4 login.facebook.com
```

If you do this on the same machine that you're running the server on, you won't be able to clone the site once you've entered this because it will interfere with the way *Set* clones the URL. However, you can disable this line in the hosts file by adding a **#** character to the start of it.

You can get your cloned web page to point the user onto whatever other page you want to by editing the **post.php** that *Set* puts in **/var/www**. The file should contain a single line that's something like:

```
<?php $file = 'harvester_2014-11-22 20:51:37.547239.txt';file_put_contents($file, print_r($_POST, true), FILE_APPEND);?><meta http-equiv="refresh" content="0; url=https://www.facebook.com/login.php" />
```

The first part of this (in the **php** tag) harvests the credentials, while the second (in the **meta** tag) forwards the user onto the correct site. In this case, we've set it to forward to **https://www.facebook.com/login.php**, but this could be anything as long as it doesn't cause a problem with your **hosts** file.

Google dorks

If you want to use this approach on victims on your local network, you'll need to run this on a public server. This is legally dubious so it's probably best that you don't do it.

Another problem with running this sort of attack using a public server is that it uses a standard set of filenames. While you could quite easily change the **index.html**, **post.php** and harvester files, many people don't. The first two files there are generic enough that there are files with these names in lots of web apps. However, the harvester filename is quite unusual. If you can find a public server with a file whose name starts with **harvester_2014**, then there's a good chance that it's currently running *Set*'s credential harvester.

Finding files on the internet is easy, you just use Google. In this case, if you search for "inurl:harvester_2014" you'll find (among some other things) servers that are currently running *Set*.

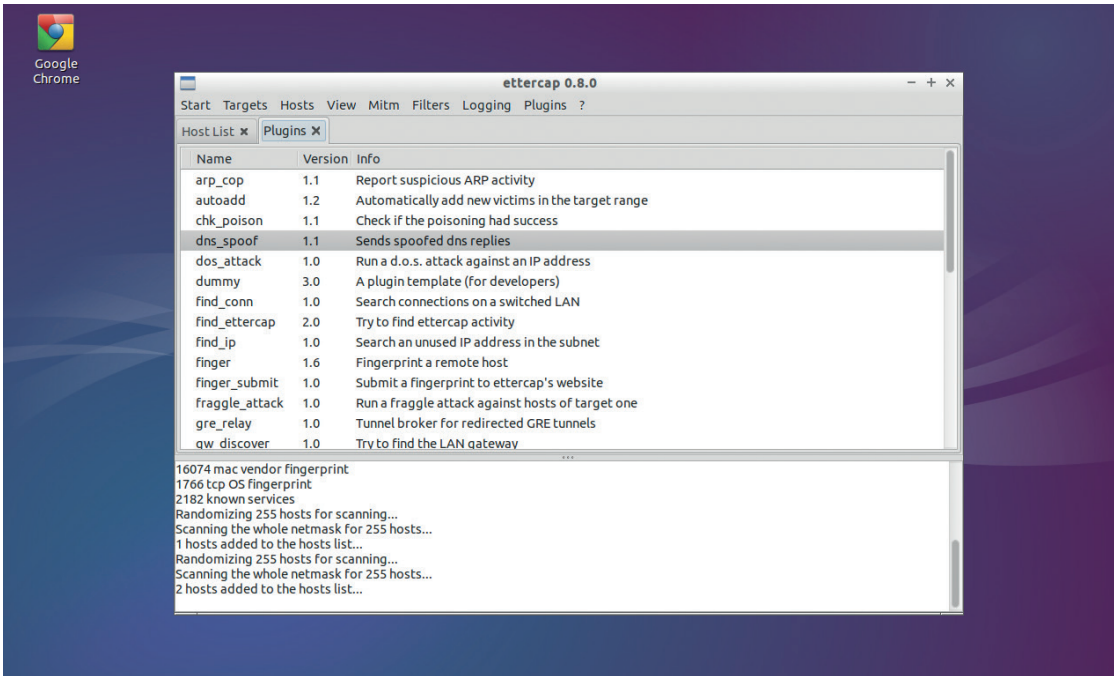
Obviously, these sites tend to go down and come back up quite regularly. When we did it, there were a couple of live harvesters, only one of which had any data in. Assuming they haven't changed the default, the cloned site will be **index.html** in the same directory. Sure enough, we found that it was a clone of Facebook. We didn't check to see if any of the credentials in the harvester file were real (and obviously, doing this would be illegal).

This style of using Google to find things that are useful to hackers is known as Google Dorks. If you know the more advanced syntax of Google searches, you can find all sorts of things that were never meant to be made public. There's a database full of useful examples at **www.exploit-db.com/google-dorks**.

Alternatively, you could change this to an HTML page that just contains an error saying that the website is down temporarily. As long as it has the first **php** tag, it will still harvest the credentials.

This approach is fairly simple to set up, but it does require you to have access to the machine that the victim's on. There are ways of getting around this requirement. To do this, you need to perform a man-in-the-middle attack which can either be physical (that is, you actually set up the network so that their connection has to flow through your computer), or by using an ARP spoofing attack, which will fool other computers on the local area network into routing their traffic through your machine.

Ben Everard is the best-selling author of *Learning Python With Raspberry Pi*. He really wants you to be careful on the web.



Ettercap can be used to manipulate a network to intercept DNS requests. Run it from the command line with **sudo ettercap -G** to get the graphical version shown here.