APT28 leverages multiple phishing techniques to target Ukrainian civil society

blog.sekoia.io/apt28-leverages-multiple-phishing-techniques-to-target-ukrainian-civil-society/

17 May 2023

Research & Threat Intelligence

The APT28 intrusion set (aka. Sofacy, PawnStorm, Fancy Bear), associated to the Russian GRU was observed using multiple phishing techniques to target the Ukrainian civil society.

APT28
phishing
Sofacy
ukraine

Felix Aimé and Threat & Detection Research Team - TDR May 17 2023
384 0

8 minutes reading

The APT28 intrusion set (aka. Sofacy, PawnStorm, Fancy Bear), associated to the Russian GRU and famous for its cyber espionage and sabotage campaigns, was observed using **multiple phishing techniques** to target the Ukrainian civil society. These techniques include using HTTP webhook services such as as Pipedream and Webhook, as well as

compromised Ubiquiti routers to steal victims' credentials. On one occasion, APT28 was seen using the "Browser in the Browser" technique to display a fake login page to the victim, purporting to decrypt a document.

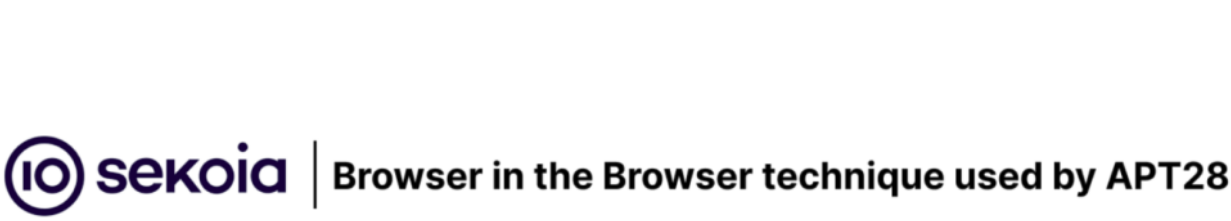
The majority of retrieved phishing webpages target the UKR.NET webmail service, which is popular among Ukrainian society. However, APT28 could use the same techniques against other webmail services used by western civil society which supports Ukraine.

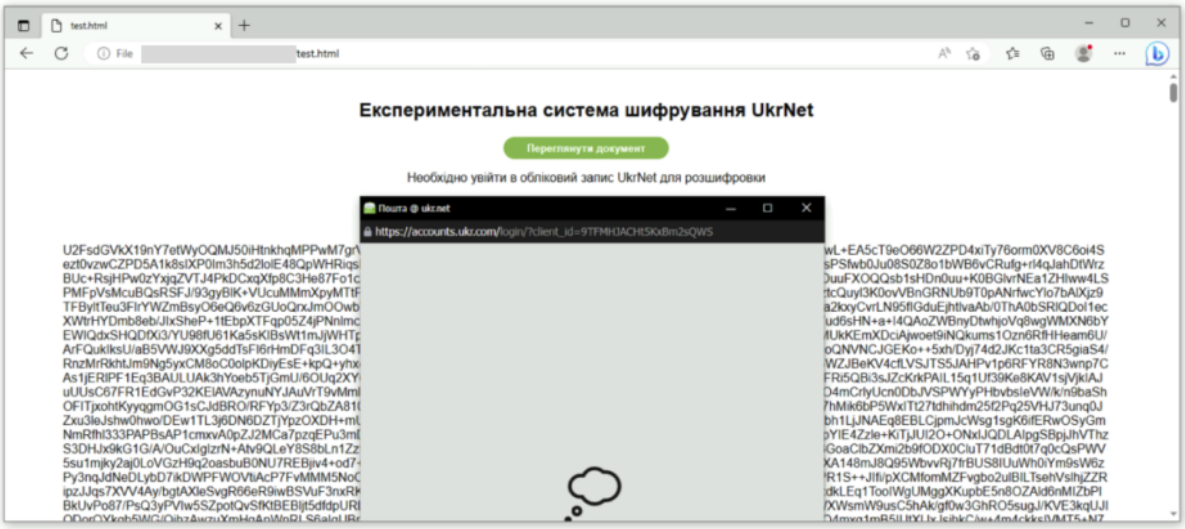
This small blogpost aims at presenting the different techniques used by APT28 to create their spear phishing webpages, as well as ways to detect them, including IOCs and YARA rules.

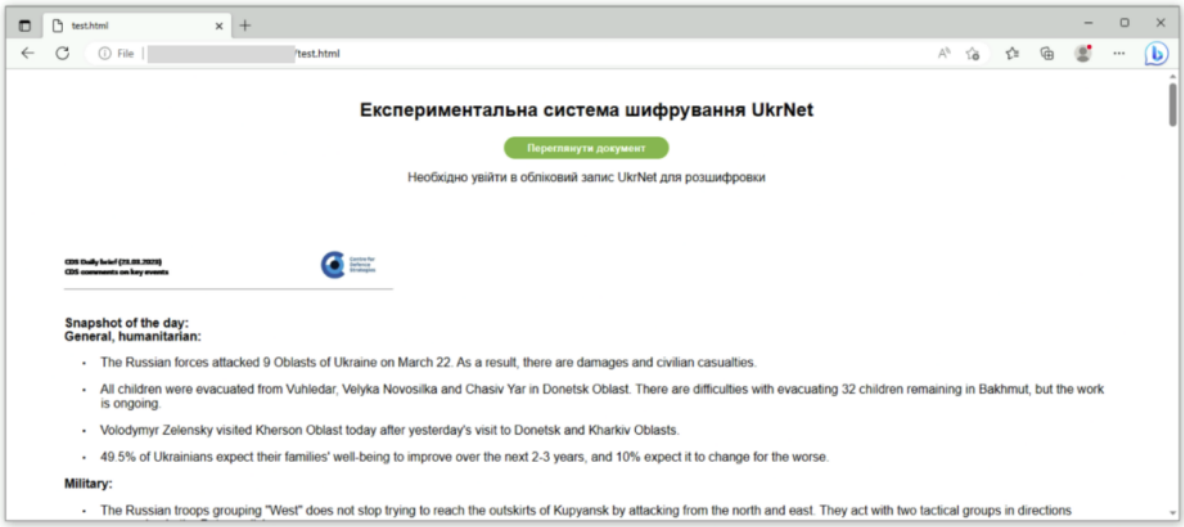
Technique 1: Man in the browser

During their investigations, Sekoia.io TDR analysts identified a file named CDS Daily Brief 23.03.2023.html, impersonating the Ukrainian security think tank Centre for Defence Strategies. The centre provides online daily briefings on the Russo-Ukrainian conflict. The retrieved decoy document trick the victim into click on a button to decrypt the page content, which is presumably secured by a UKR.NET technology.

Clicking on the button displays fake login window. This window contains an iframe that embeds a fake UKR.NET login webpage hosted on robot-876.frge[.]io – which was deactivated during the investigation and was already published by Google TAG. The intention here is to trick the user into entering their credentials.



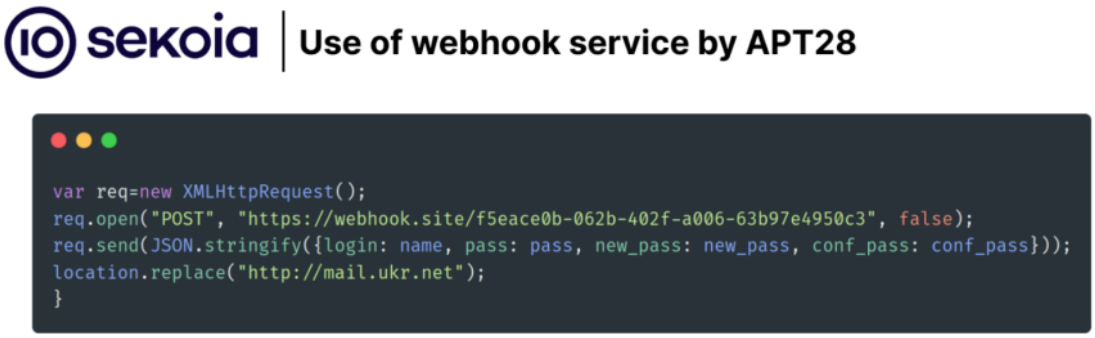




This is the only known phishing case from APT28 that we know of where the intrusion set uses an HTML attachment with the Man in the Browser technique to trick the user into entering its credentials. Although the code and the technique was copied from the mrd0x original blogpost dating back to 2022, the analysed document is currently only detected by one antivirus engine on VirusTotal (eScan) at the time of writing.

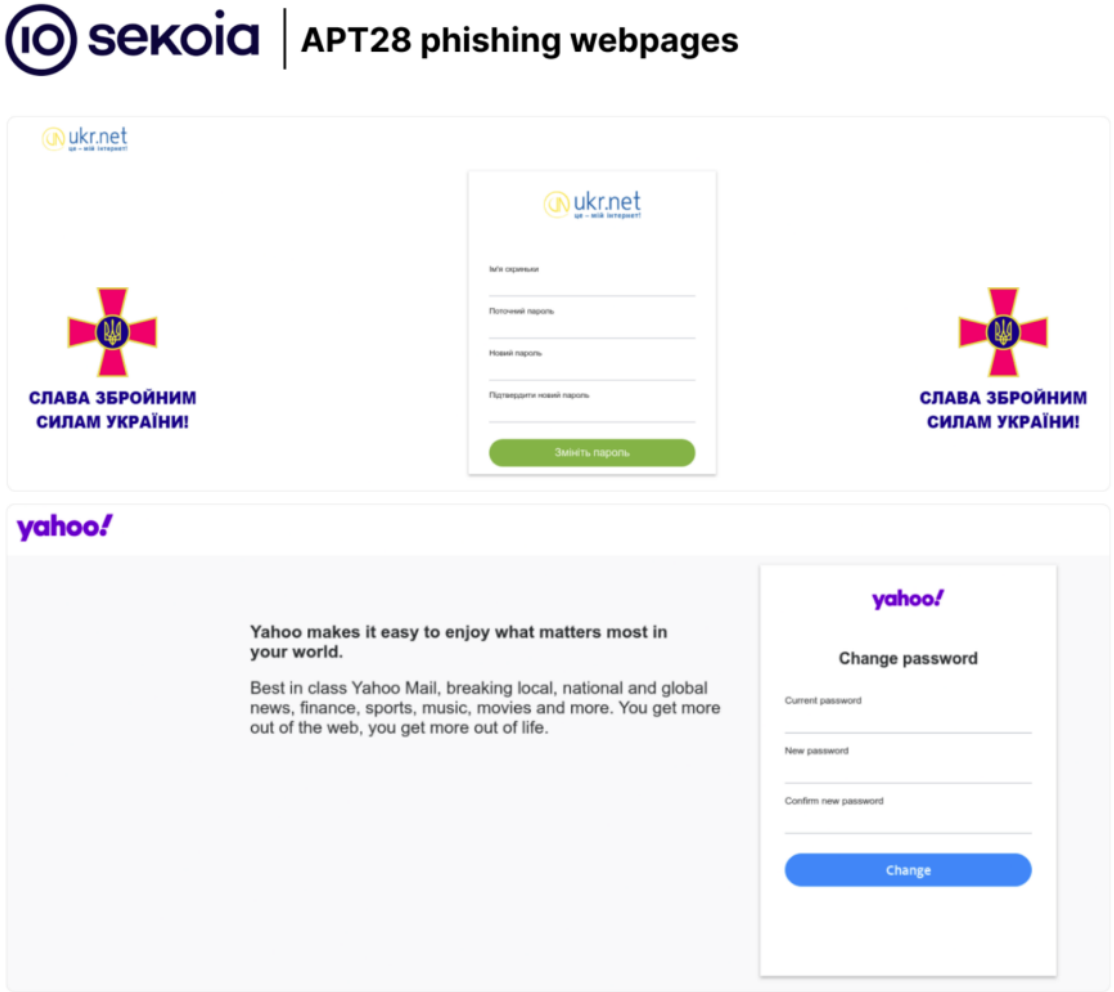
Technique 2: Use of HTTP debugging / webhook services by APT28

A second technique that we observed used by APT28 since more than a year is the use of public HTTP debugging / webhook services in their phishing webpages to retrieve stolen credentials, as shown below:



Therefore, APT28 operators don't need to setup any extra script or infrastructure to collect the credentials, but only to setup a webhook page on the service and wait to receive credentials from the victim. During the investigation, we identified two services abused by APT28, PipeDream.com and Webhook.site, both receiving HTTP requests without any user inscription.

This technique was at least on two phishing webpages operated by APT28, the first one targeting UKR.NET, and the second one targeting Yahoo.com, as shown below:

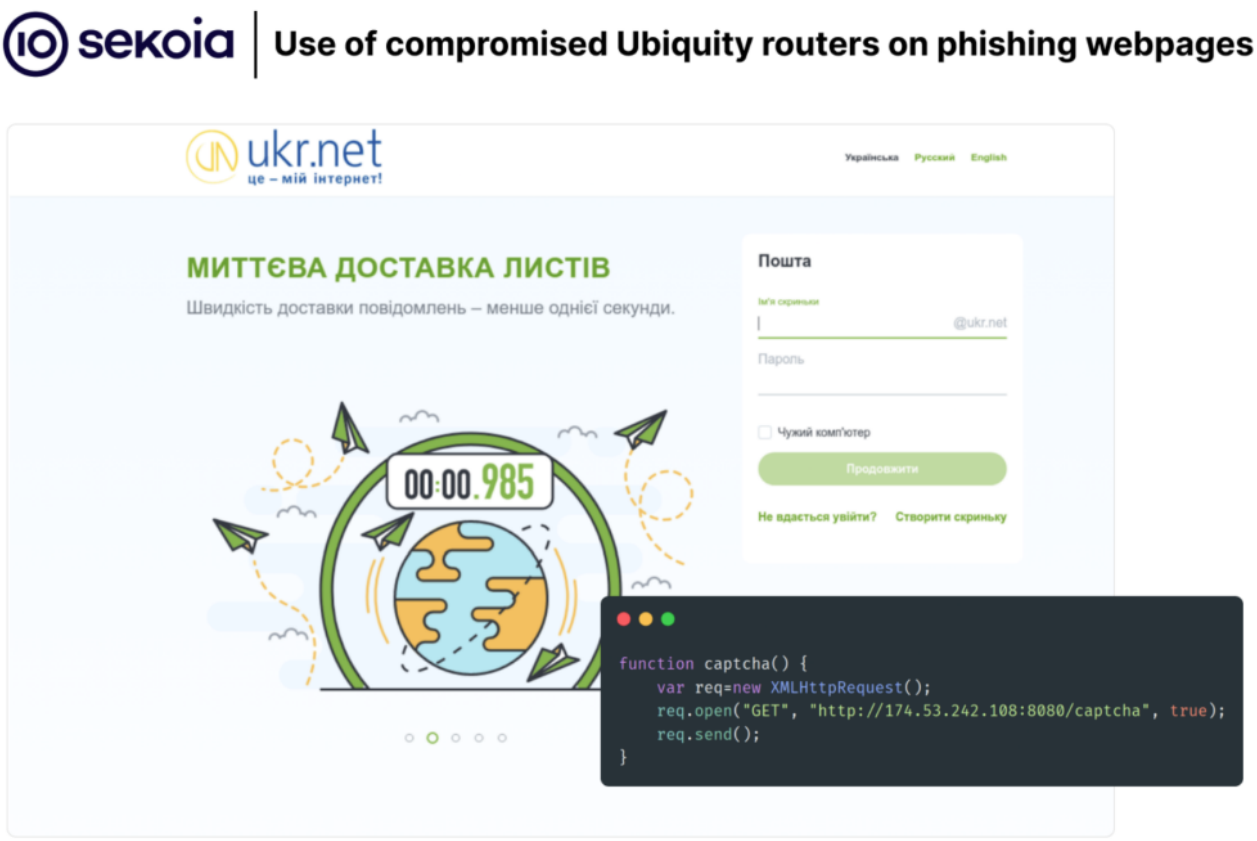


This is the first time we see a state-sponsored threat actor use these services for phishing operations. While this technique remains effective, from a security perspective, there is a concern that **some services might allow access to hooks created by simply by specifying the hook identifier**. This implies that anyone accessing the phishing page can view stolen credentials by looking at the specific hook.

Technique 3: Use of compromised Ubiquiti Routers by APT28.

The aforementioned technique is only applicable to accounts with 1FA authentication, as the webhook service lacks the capability to interact with the UKR.NET servers for validating the second factor and retrieving authentication cookies or tokens. For 2FA accounts, APT28 created dedicated webpages hosted on *.frge[.]io domains that interact with a **Python script running on compromised Ubiquiti routers**.

Here is an extract from the HTML code of a phishing webpage that points to a compromised Ubiquiti router (174.53.242.108):



The Python script, which was leaked on Twitter, interacts with the UKR.NET API to authenticate the user and bypass 2FA. Interestingly, as the UKR.NET service uses **RecaptchaV2** during the authentication process, the Python script contains code to bypass the captcha by using the anti-captcha.com service API with the customer key equal to acbb64c3de5ea5e5936df4a1eecf1235.

The script enables two other interesting features. The first one is that it activates **IMAP access via the UKR.NET API** for the compromised mail account and saves the generated password from UKR.NET, which will be used for this access, to a file as shown below. Embedding IMAP access to the mailbox is a known trick that allows APT28 operators to **automate mail exfiltration**.

sekoia | IMAP activation via the UKR.NET API

```
priv_url='https://accounts.ukr.net/api/v1/security_session/get_privileges'
priv_data='{"password":"'+unquote(password)+'","change_type":"apwds_add"}'
s.post(priv_url, priv_data,headers=headers3)
pass_url='https://accounts.ukr.net/api/v1/security_session/app_pwds/add'
pass_data='{"name":"Imap"}'
js_res=json.loads(s.post(pass_url, pass_data,headers=headers3).text)
imap_pass = js_res['password']['raw_password']
with open(login+'_imap.txt', 'a') as f:
    f.write(login+' '+imap_pass+'\n')
debug('imap success')
```

Another interesting feature embedded in this script is that it deletes the last emails received from noreply@ukr.net. Sekoia.io analysts assess this is almost certainly done to **cover up changes to the mailbox security settings**. It is also highly likely a technique to prevent the user from being alerted to a new access to the mailbox.

sekoia | "noreply@urk.net" emails removing from the mailbox

```
url_mail_all = 'https://mail.ukr.net/api/mail/batch?1675139356809'
data_mail_all ='[TRUNCATED]'
stop=False
while True:
    if stop==True:
        break
    mailbox = s.post(url_mail_all,data_mail_all,headers=headers1)
    js_mails = json.loads(mailbox.text)
    js_array = js_mails[2]['messages']
    for js_mail in js_array:
        if js_mail['from'][0][1]== 'noreply@ukr.net':
            mail_id = js_array[0]['id']
            delete_data = '[{"method":"messages/remove","args":{"messages":["'+mail_id+'"], [TRUNCATED]}}]'
            del_resp=s.post(url_mail_all, delete_data,headers=headers1)
            if del_resp.text=='[{}]':
                stop=True
```

Through the use of heuristics, Sekoia.io discovered five different Ubiquiti routers hosting the Python script. **All of them were already compromised by the same SSH rootkit**, as indicated by their unusual SSH banners. This rootkit is based on the openssh-backdoor-kit open-source project. **This rootkit has already compromised hundreds of Ubiquiti routers, as evidenced by the returned SSH banners**. It is unclear whether this rootkit is

used by APT28 or another threat actor who is also looking for shells on embedded devices. So far, no other compromised SOHO brands or VPS instances were matching our heuristics on the APT28's 2FA phishing script.

Update: An attentive reader said that it is likely that APT28 took over the routers by exploiting the SSH backdoor, as the backdoor has been already analyzed and its passwords published. Moreover, one of the backdoored sshd sample was known on VT since 2018.

As demonstrated in our FLINT report on CVE-2023-23397, **APT-28 recently used compromised Ubiquiti Edge devices, based on EdgeOS, as operational boxes to serve Responder instances on 445.** Attackers seem to find EdgeOs, which is based on the Debian fork Vyatta, particularly interesting due to its default password, WAN access, and Aptitude package management capabilities. These features make it easy for them to deploy their scripts.

Links to APT28

Sekoia.io analysts link their campaigns to APT28 with high confidence based on two artefacts.

APT28 uses the getforge service since 2022, which produces the frge[.]io domain names. In 2022, APT28 used this service to exploit the Follina vulnerability and drop its CredoMap stealer. This service allows for free hosting of static web pages and is no longer maintained by the company that created it. Sekoia.io analysts think that APT28 may see this as a good opportunity to host web pages without the risk of being taken down.

The final point pertains to the use of HTTP webhooks and mocking-free services in APT28's operations. They appear to use these services to host payloads and retrieve victim data because getforge doesn't allow to host dynamic webpages. Moreover, these services are free, legit and do not require specific infrastructure setup. Although we currently only see APT28 using these services, it is possible that other intrusion sets may follow, as was the case with the use of pastebin-like services to host payloads.

—

Featured image: Drone Shot of Motherland Monument and the City of Kyiv – CC Petkevich Evgeniy

IOCs of APT28 listed as part of this campaign targeting Ukrainian civil society

Malicious domains names

```
robot-876.frge[.]io
setnewcred.ukr.net.frge[.]io
panelunregistertle-348[.]frge.io
settings-panel.frge[.]io
ukrprivacysite.frge[.]io
config-panel.frge[.]io (medium confidence)
smtp-relay.frge[.]io (medium confidence)
```

Compromised Ubiquiti routers

```
68.76.150[.]97
174.53.242[.]108
24.11.70[.]85
202.175.177[.]238
85.240.182[.]23
```

SSH rootkit IOCs

```
69.28.64[.]137 - Attacker’s IP
packinstall.kozow[.]com - Installation script staging server
```

Yara rules to detect APT28

```
rule apt_APT28_Phishing_webpage_Ubiquiti {
    meta:
        id = "324e9b6f-45eb-4cb4-bca6-9012edc7170c"
        version = "1.0"
        intrusion_set = "APT28"
        description = "Detects APT28 phishing page sending to Ubiquiti devices"
        source = "SEKOIA"
        creation_date = "2023-04-28"
        classification = "TLP:WHITE"
    strings:
        $ = "req.open(\"GET\", \"http://"
        $ = "/captcha\", true);"
        $ = "[3].style.backgroundImage="
        $ = ".html?usr="
        $ = "full.forEach(element=>setInp"
    condition:
        filesize < 100KB and
        3 of them
}

rule apt_APT28_UKRNET_2FA_Bypass_Python_Script {
    meta:
        id = "b8e4418c-1b39-4f78-b2ec-a0b85e4e7ca6"
        version = "1.0"
        intrusion_set = "APT28"
        description = "Detects APT28 Python script used to bypass 2FA on UKR.NET"
        source = "SEKOIA"
        creation_date = "2023-04-28"
        classification = "TLP:WHITE"
    strings:
        $ = "debug('f_res ok')"
        $ = "make_response('BAD')"
        $ = "make_response('NOOP')"
        $ = "debug('task='+str(task))"
        $ = "f.write(str_cookie"
        $ = "if stop==True:"
        $ = "with open(login+"
    condition:
        filesize < 5KB and
        3 of them
}

rule apt_APT28_Phishing_webpage_webhook {
    meta:
        id = "46adc67b-8bcc-4cba-a480-502c7eb433a3"
        version = "1.0"
        intrusion_set = "APT28"
        description = "Detects webhook APT28 phishing page"
        source = "SEKOIA"
        creation_date = "2023-04-28"
        classification = "TLP:WHITE"
    strings:
```

```
        $ = "location.replace(\"http"
        $ = "name=location.search.split('=')[1];"
        $ = "req.send(JSON.stringify({login: name, pass: pass,"
    condition:
        filesize < 20KB and
        all of them
}

rule apt_APT28_Phishing_Browser_in_Browser {
    meta:
        id = "e997b893-3120-4121-a813-281abfaa59c8"
        version = "1.0"
        intrusion_set = "APT28"
        description = "Detects APT28 Browser in Browser phishing page"
        source = "SEKOIA"
        creation_date = "2023-04-28"
        classification = "TLP:WHITE"
    strings:
        $ = "U2FsdGVkX19nY7"
        $ = "').innerHTML=CryptoJS.AES.decrypt(document.getElementById("
        $ = ").toString(CryptoJS.enc.Utf8)"
        $ = "function sh()"
        $ = "$(\"#clickme\")[0].style.display='none'"
        $ = "<span id=\"domain-name\">"
    condition:
        4 of them and
        filesize < 500KB
}
```

Chat with our team!

Would you like to know more about our solutions?
Do you want to discover our XDR and CTI products?
Do you have a cybersecurity project in your organization?
Make an appointment and meet us!

Contact us

Thank you for reading this blog post. Feel free to share your feedback, and read other TDR reports here:

Comments are closed.