Advanced Use Case:

www[.]flowerexplosion[.]com Are We Compromised?

Scenario:

Several financial institutions have stated that fraudulent payment card activity has occurred after users made purchases from your website. The financial institutions believe that payment card skimming software was used.

Fraudulent activity started around back in June 2019.

Your website was placed on the Google Safe Browsing blacklist. Users were prevented from accessing your website and your organization started to lose a lot of money. Management gave the order to the Server Admins to get the website back up and running as quickly as possible. Server Admins blow away the website and load a clean known good instance in 20 minutes.

Server Admins did not have or make any backups or keep an instance of the website for you.

You are still tasked with investigating if payment cards were being stolen from your website.

You need to determine the following:

1. Were payment cards being stolen from www[.]flowerexplosion.com?
2. How were the payment cards being stolen?
3. How do you suspect the website was compromised?
4. How can you prevent similar attacks in the future from occurring on www[.]flowerexplosion[.]com?

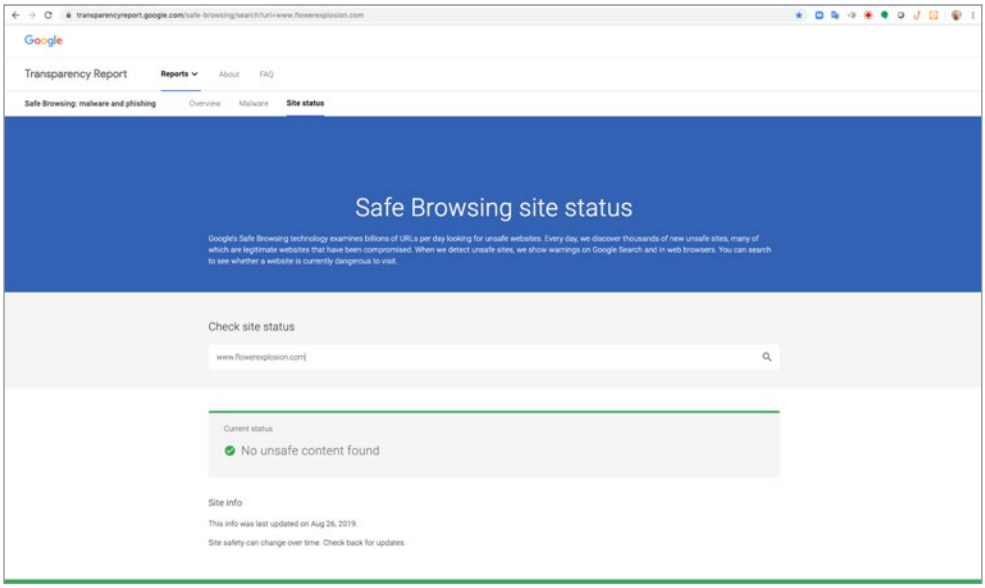
Step 1: Check to see if your organization's website www[.]flowerexplosion[.]com is still on the google safe browsing list.

Open your web browser and search for "google safe browsing"

In the results click on the link for google transparency report:
https://transparencyreport.google.com/safe-browsing/search?hl=en

Enter the website www[.]flowerexplosion[.]com

The URL should now be:
https://transparencyreport.google.com/safe-browsing/search?url=www.flowerexplosion.com

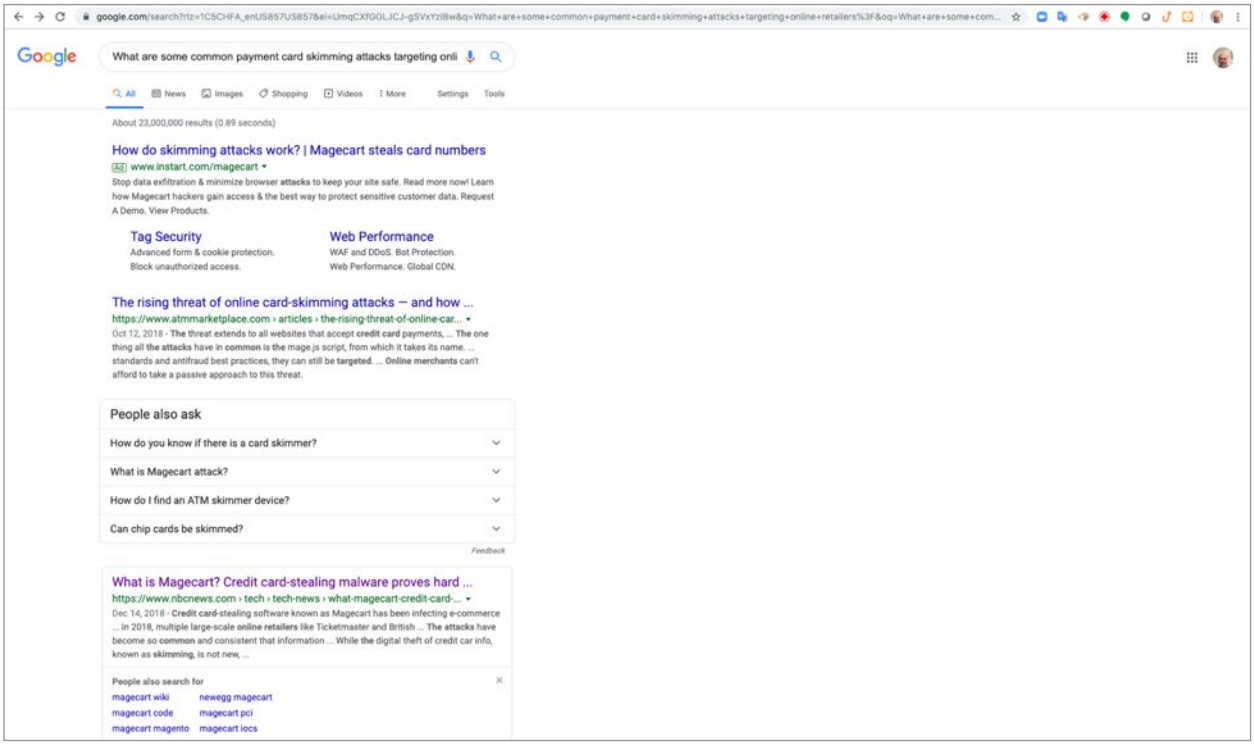


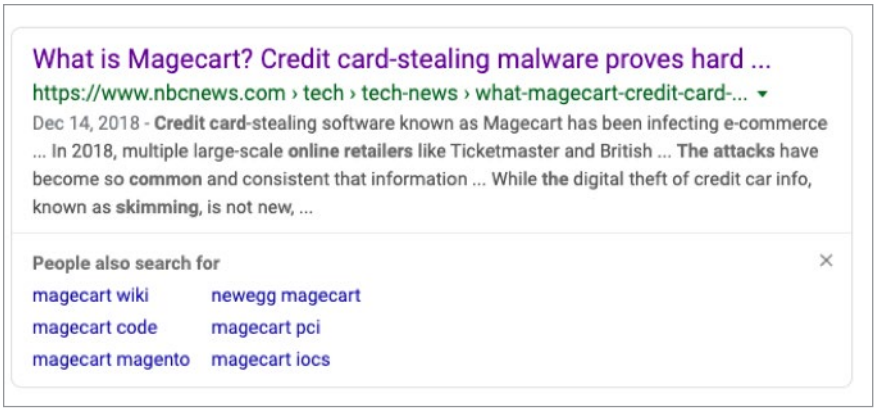
Google is not currently blocking your organization's website. This is a good indicator, but this does not mean that your organization is safe from future attacks.

Since your organization does not have a system or any backups to investigate, this does not mean that your investigation is over. It just means that you must rely upon tools that have historic information about your domain.

Step 2: Open a new tab and go to https://www.google.com

Search for the following question:
what are some common payment card skimming attacks targeting online retailers?





Just from reading the results you can see that one of the popular payment card skimmers is Magecart, a JavaScript attack that targets online merchants. If you click on the links, you will get more information about Magecart.

What is Magecart?

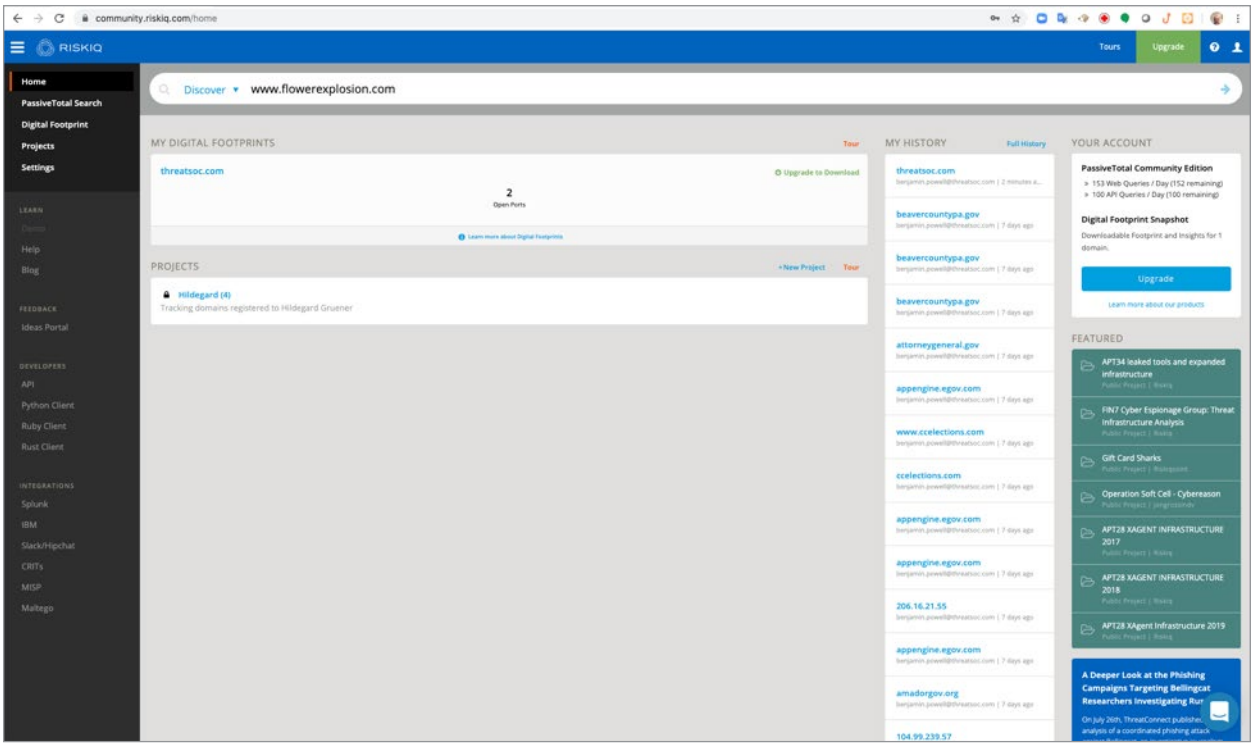
Magecart injects a script designed to steal sensitive data that consumers enter into online payment forms on e-commerce websites directly or through compromised third-party suppliers that websites might depend upon to make their sights function.

Now we are going to utilize RiskIQ's PassiveTotal™ threat hunting tool to further your investigation. PassiveTotal has over 10 years of rich internet from gathering information on the Open Internet (IPv4). This information allows threat hunters and researchers to understand information about a domain and the relationships the domain has had to other domains on the internet.

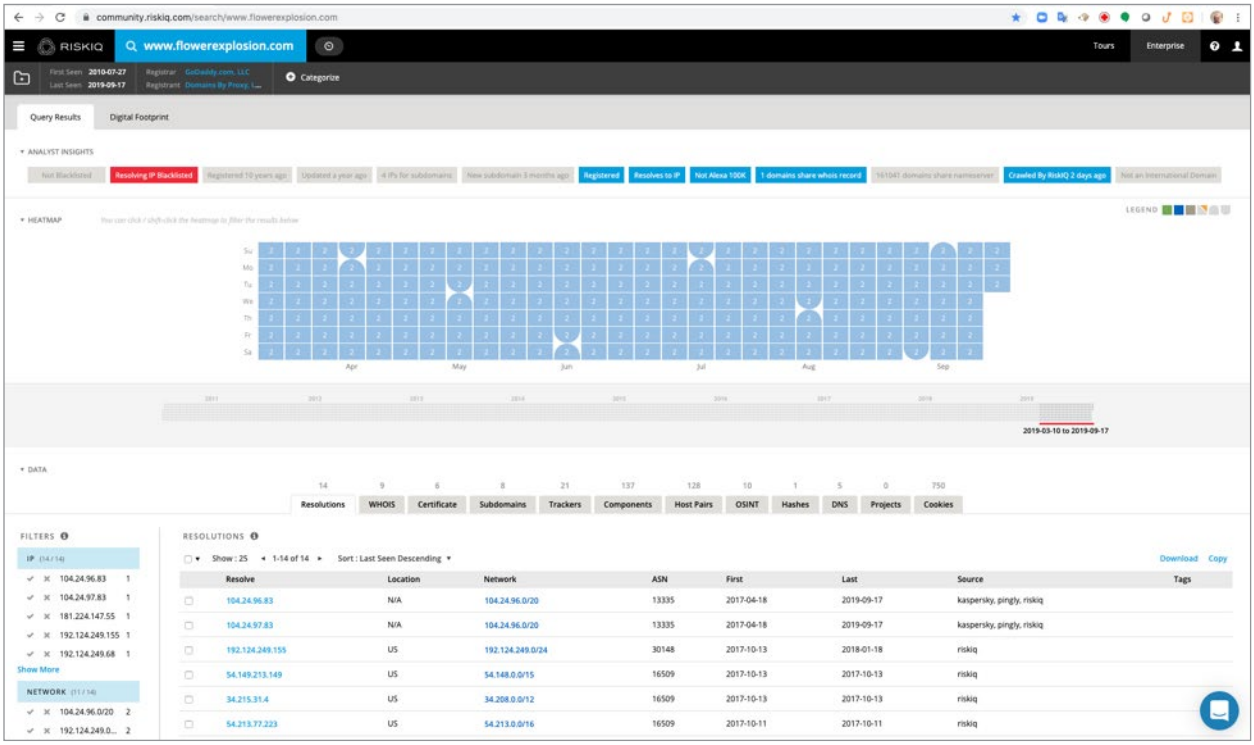
Step 3: Search for the domain www[.]flowerexplosion[.]com

Open a new tab in your web browser go to https://community.riskiq.com

Login using your credentials and begin by searching for www[.]fowerexplosion[.]com.

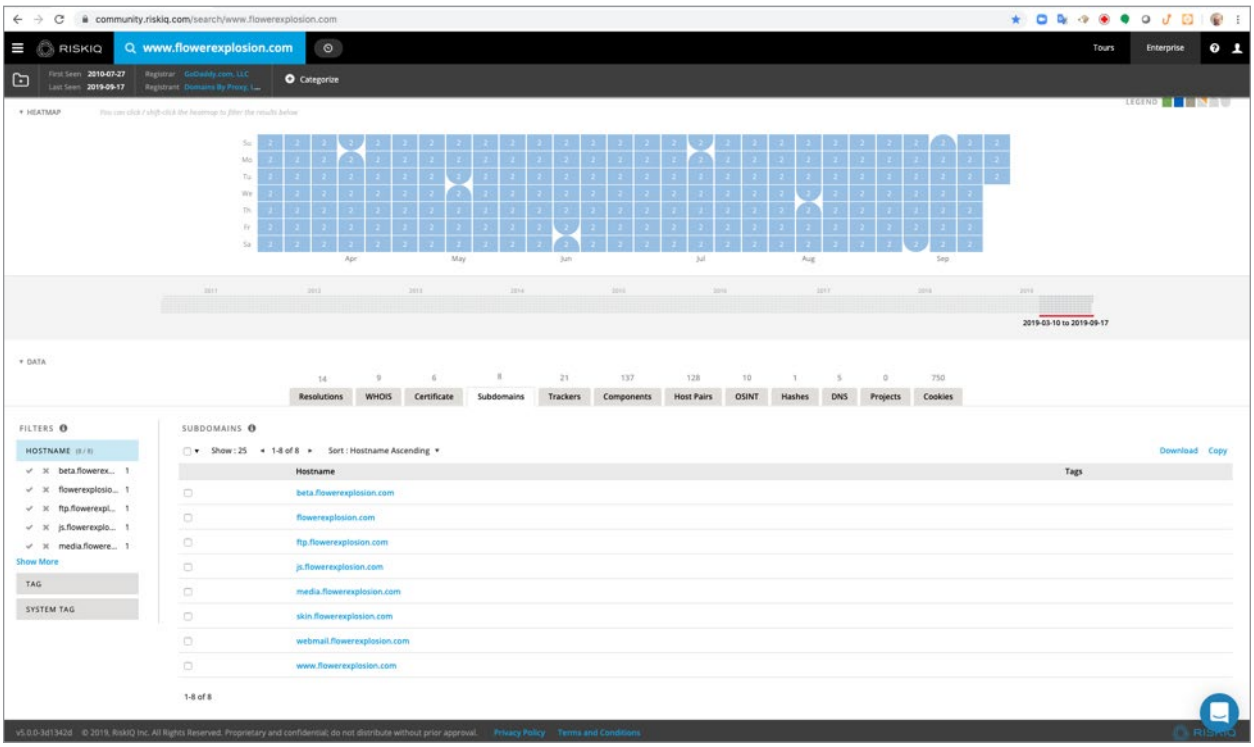


After your search the URL should be: https://community.riskiq.com/search/www.flowerexplosion.com



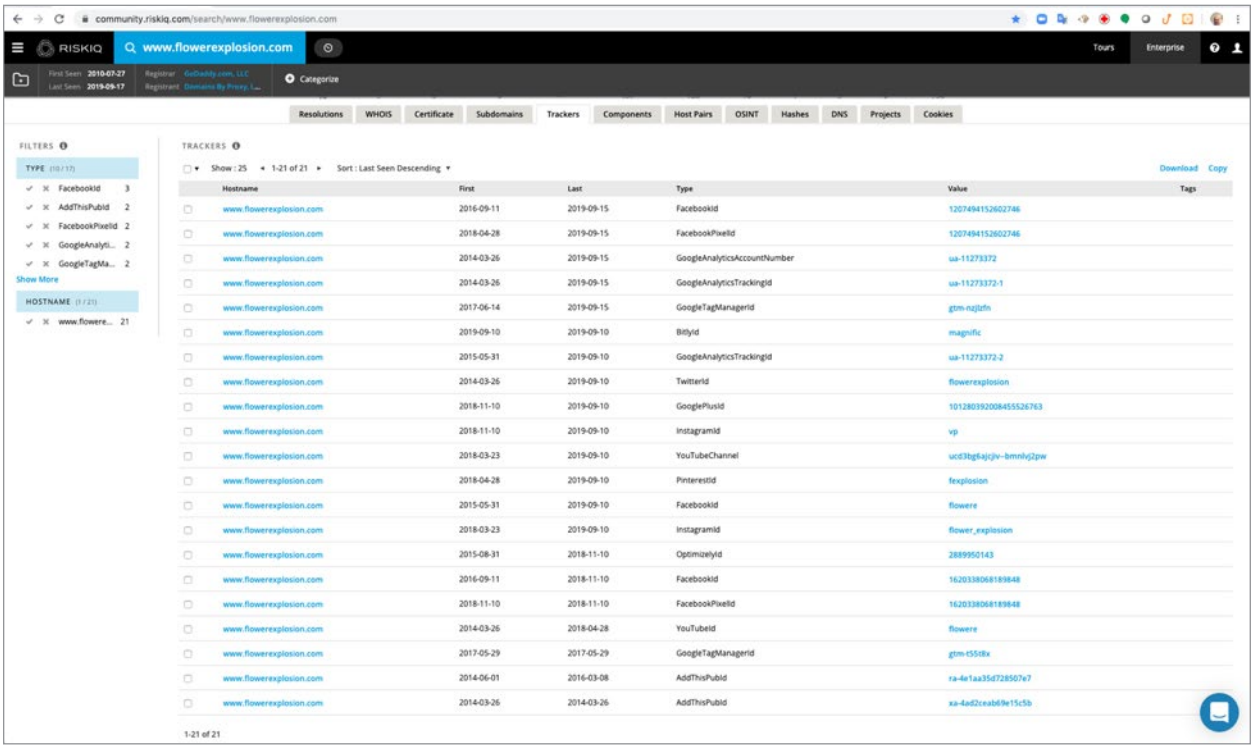
Examining the IP addresses, we see that they have not changed since 2017. Therefore, there is nothing unusual that sticks out.

Step 4: Click on the subdomains tab.



All of the domains are part of flowerexplosion[.]com and do not seem to be unusual and stick out to you.

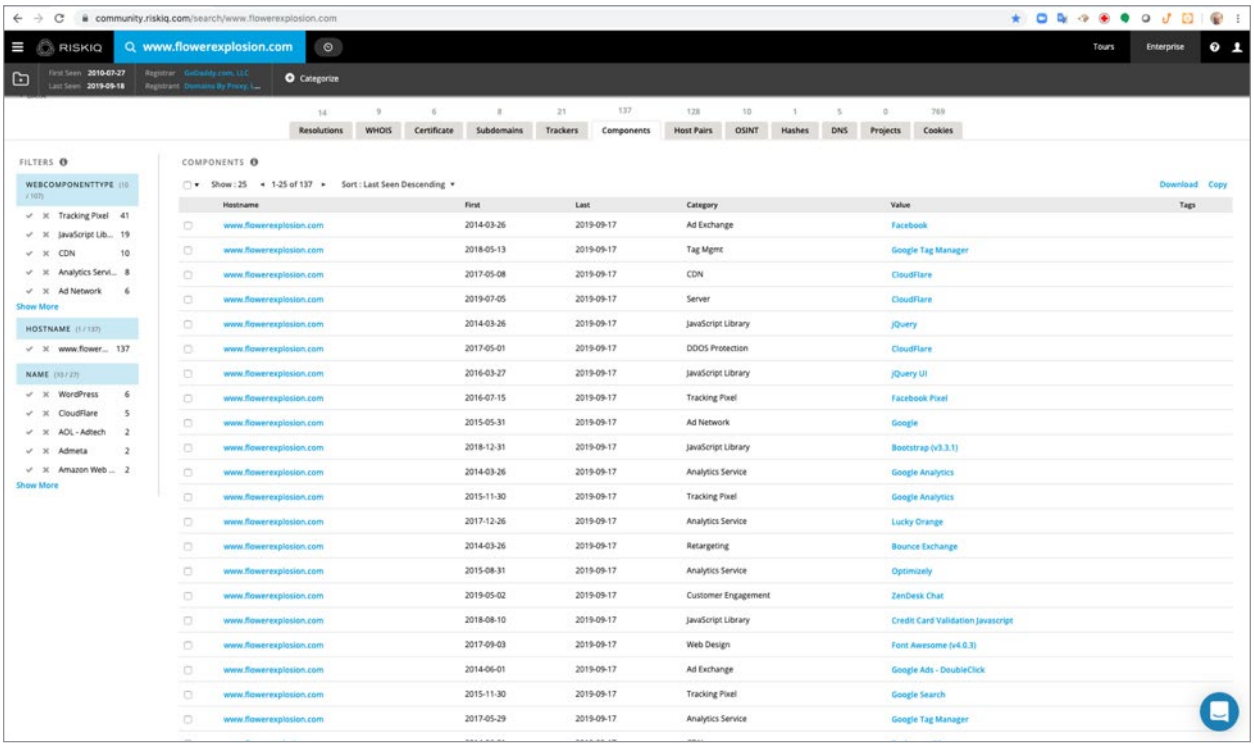
Step 5: Click on the Trackers tab.



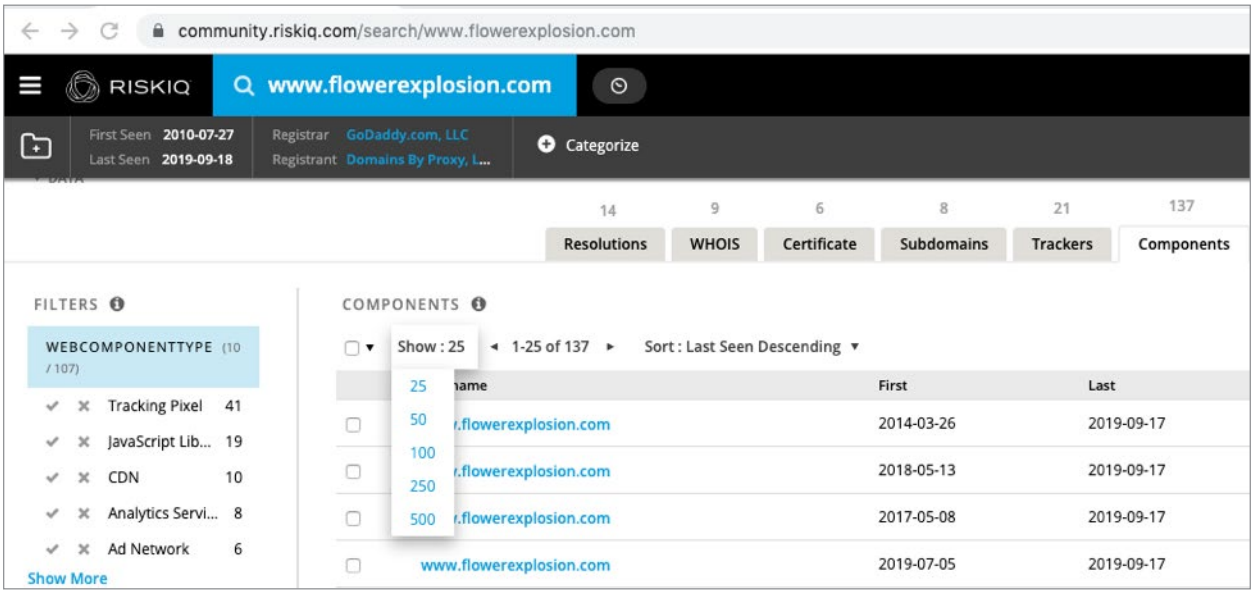
Look for trackers like MarkOfTheWeb or TorHiddenServiceAddress that are usually associated with threat actor activity. MarkOfTheWeb is created when someone duplicates your website using Internet Explorer. This is usually associated with phishing attacks. TorHiddenServiceAddress is associated with Tor exit nodes that bridge the open internet and the dark web.

The trackers results listed do not show anything unusual.

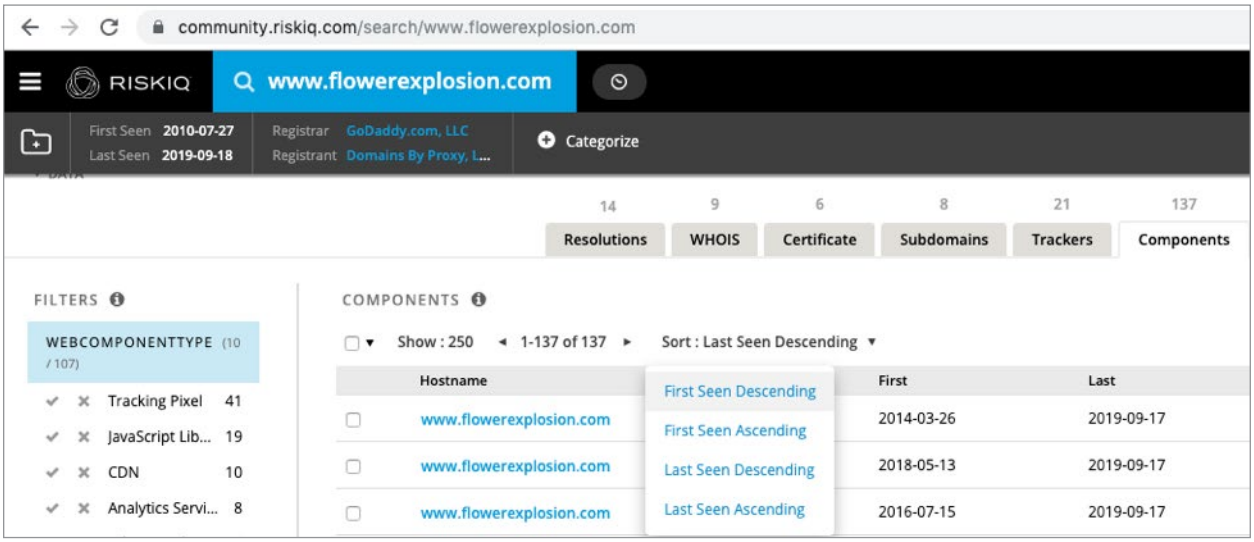
Step 6: Click on the Components tab

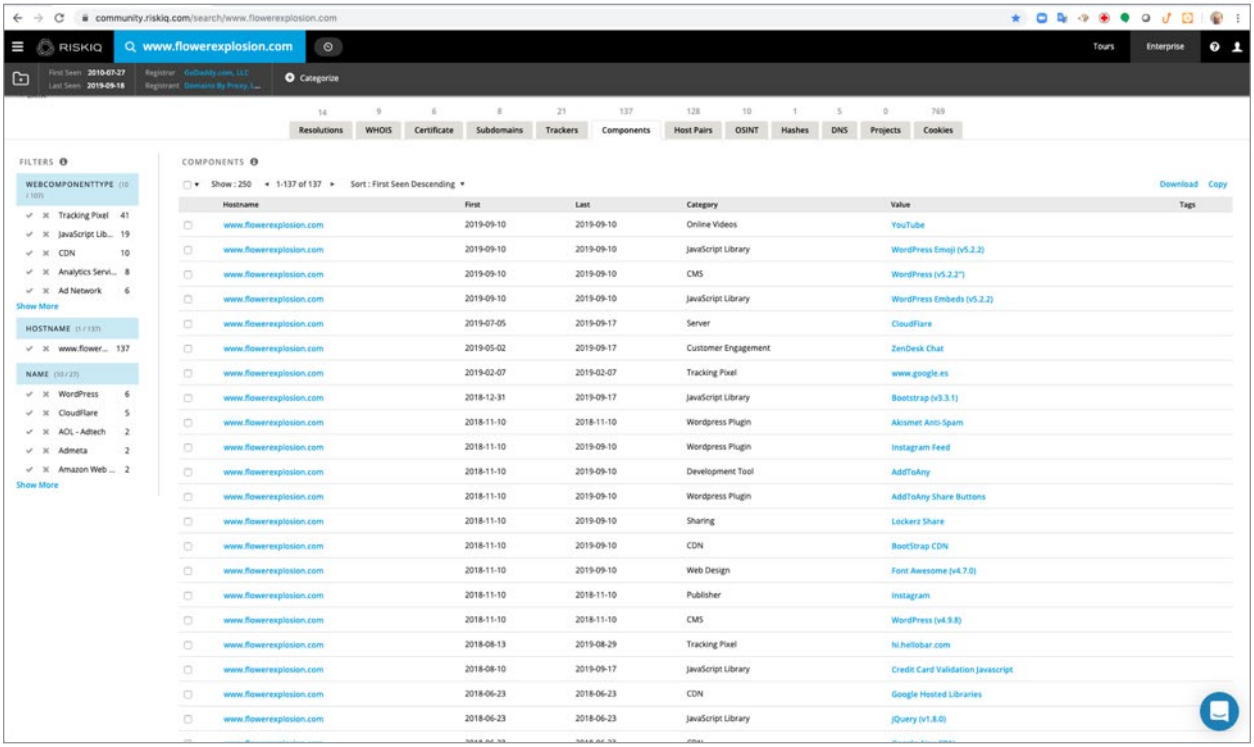


There are over 100 results, expand the number of results to 250 by clicking on Show: 25 and then clicking on 250.



Sort the results to First seen Descending.





In the results, if you look at entries from 2018-12-31 to present in the First seen column. We can see that Bootstrap 3.3.1, ZenDesk Chat, and WordPress v5.2.2 were added. Nothing unusual listed. But some of the entries might contain vulnerabilities. We will come back to investigate if vulnerabilities exist a little later.

Step 7: Investigate the JavaScripts used on www[.]flowerexplosion[.]com.

Now some information about payment card skimmers and identified one of the most popular slimmers as Magecart. We now have a potential avenue of attack via a malicious JavaScripts. We will now need to examine the JavaScripts used on your website to see if you can identify a potential JavaScript that needs to be further investigated. RiskIQ PassiveTotal will not display the contents of the JavaScripts it has detected it will only identify the sources where the JavaScripts came from. RiskIQ has other modules and solutions that can monitor and alert organizations to changes in JavaScripts or Malicious code contained in JavaScripts.

To start our JavaScript investigation in PassiveTotal, you need to click on the Host Pairs tab.

Note:

Host Pairs are the relationship between two websites that were observed during RiskIQ's crawl the website. For example, a website that you visit might be pulling in the logo from Amazon (Parent relationship) or the website might send analytic data to google to track user experience (child relationship).

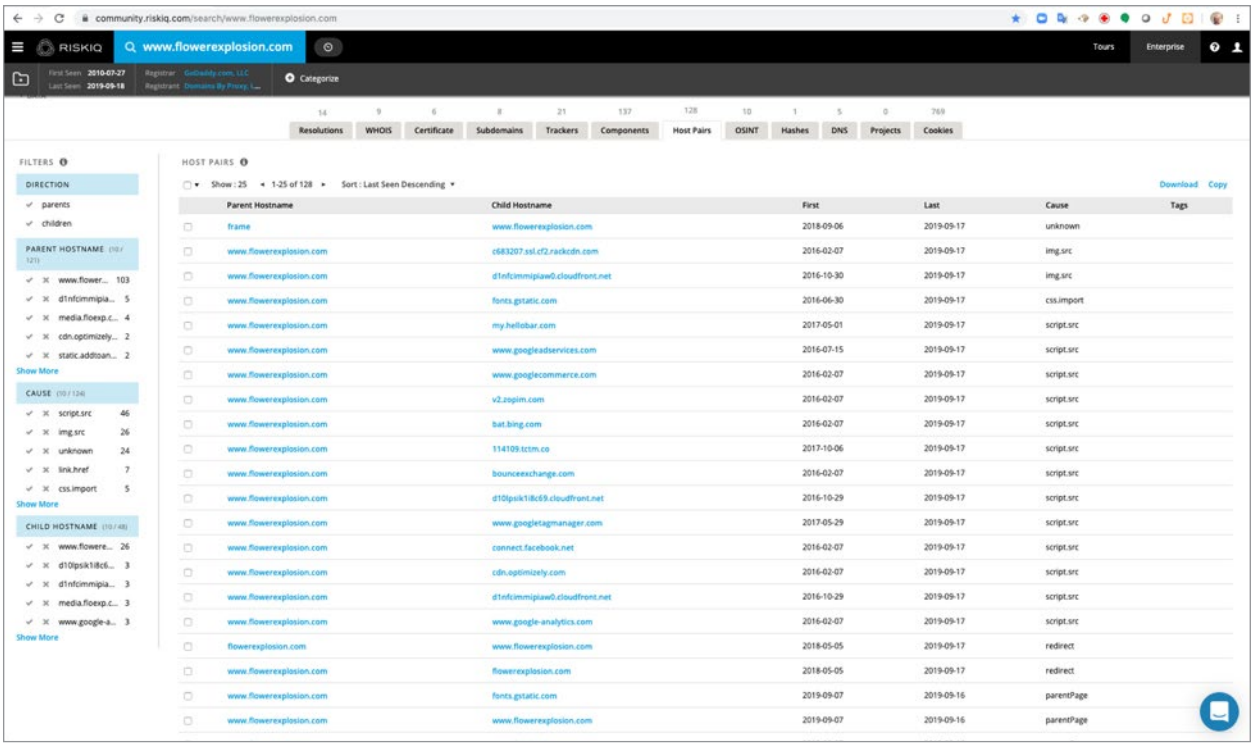
The connection could range from a top-level redirect (HTTP 302) to something more complex like an iframe or script source reference.

Think of the relationship with regard to what you have searched. In our case, we searched for www[.] flowerexplosion[.]com.

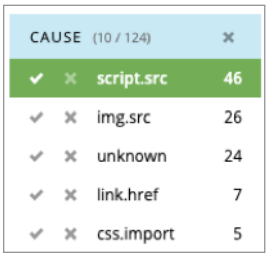
Host Relationships

Parent	→	RiskIQ observed relationships in www[.]flowerexplosion[.]com	→	Child
Parent	→	What was queried in PassiveTotal	→	Child
payments.amazon.com Cause: img.src	→	www[.]flowerexplosion[.]com	→	google.com Cause: script.src

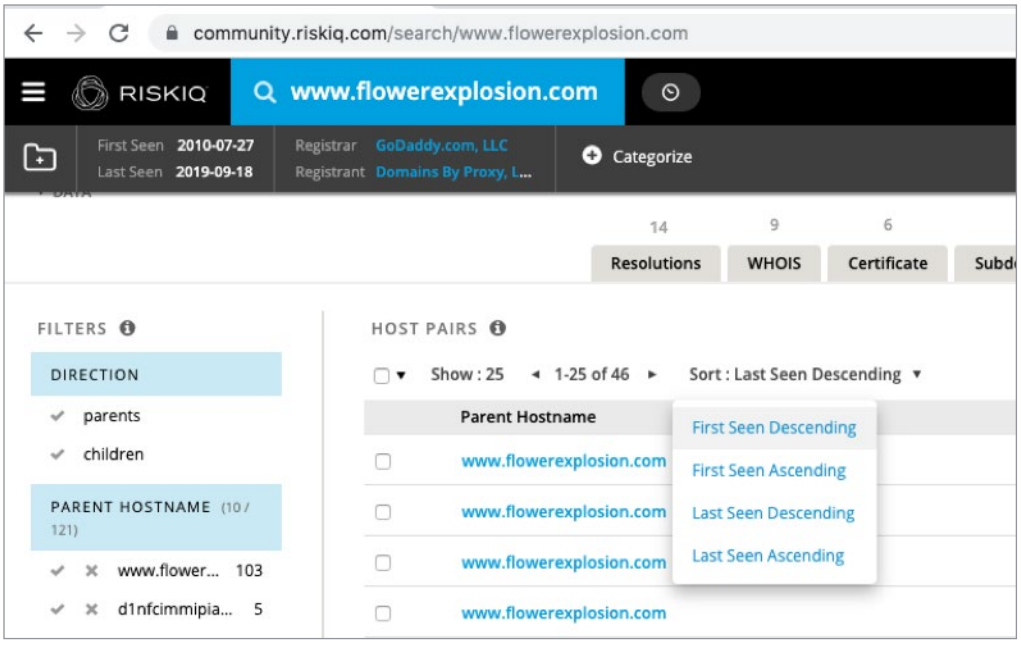
In our example: www[.]flowerexplosion[.]com is going to google.com and loading a JavaScript. It does not state which JavaScript was loaded, just where it was loaded from. www[.]flowerexplosion[.]com is also sending payment information to payments.amazon.com. This might be for processing online payments from orders.

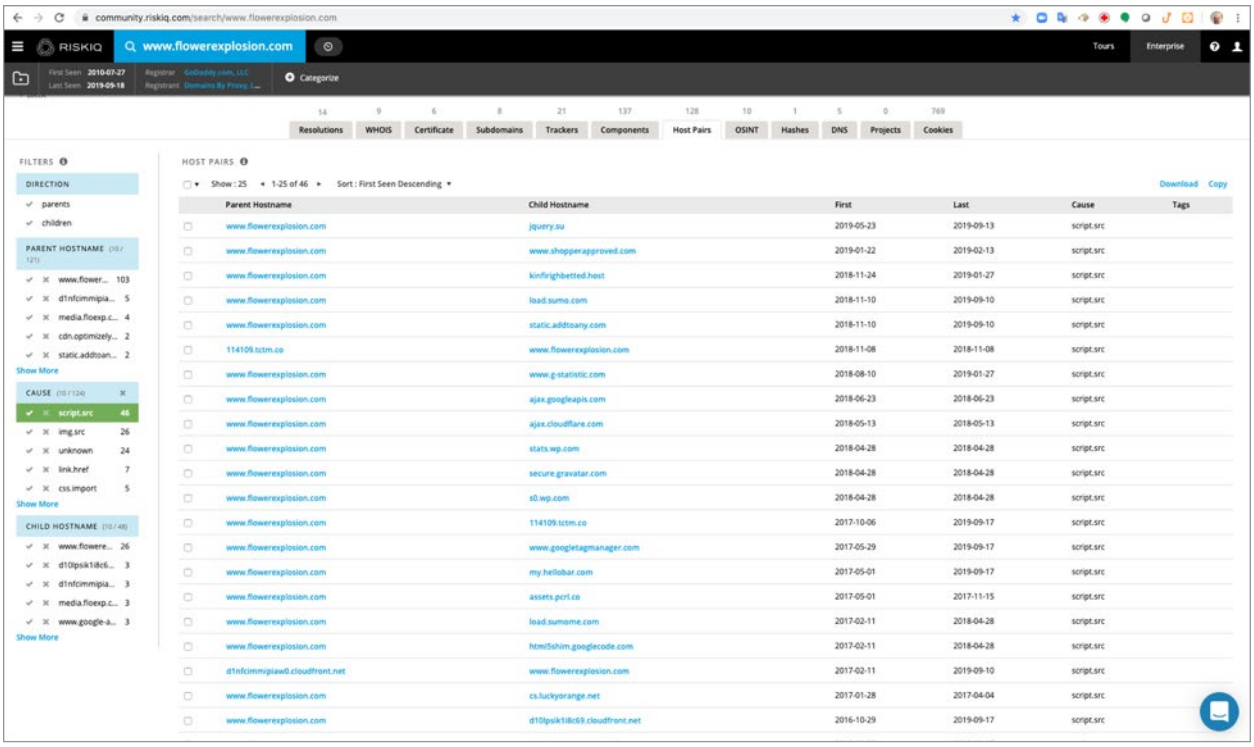


Since we are looking just for scripts in the CAUSE filter section on the left click on the check next to script.src. This will filter the results to only show causes in host pairs that were scripts.src.



Now sort the results on the screen to First Seen Descending:





The first entry is jquery.su first seen 2019-05-23 and last seen 2019-09-13. The server was rebuilt on 2019-09-13. The fraudulent payment card activity was seen in June 2019. This could be a potential candidate for further investigation. www[.]shopperapproved[.]com was first seen in January 2019 does not fit the time period.

Viewing the results, most we have seen in 2018 or before. Only one script item is was first observed in 2019, jquery.su. Jquery.su was first observed on 05-23-2019, this is right before the first fraud that was reported by the financial institutions.

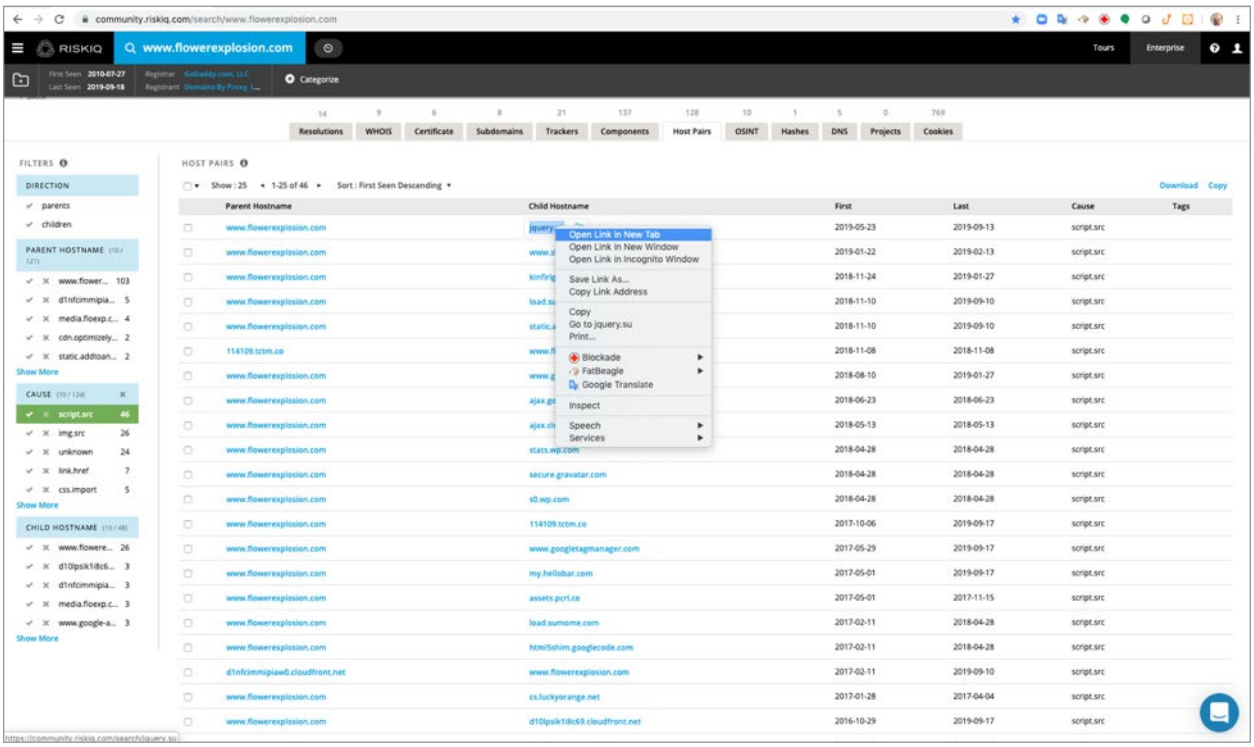
Now that we have the suspect script what can we do next?

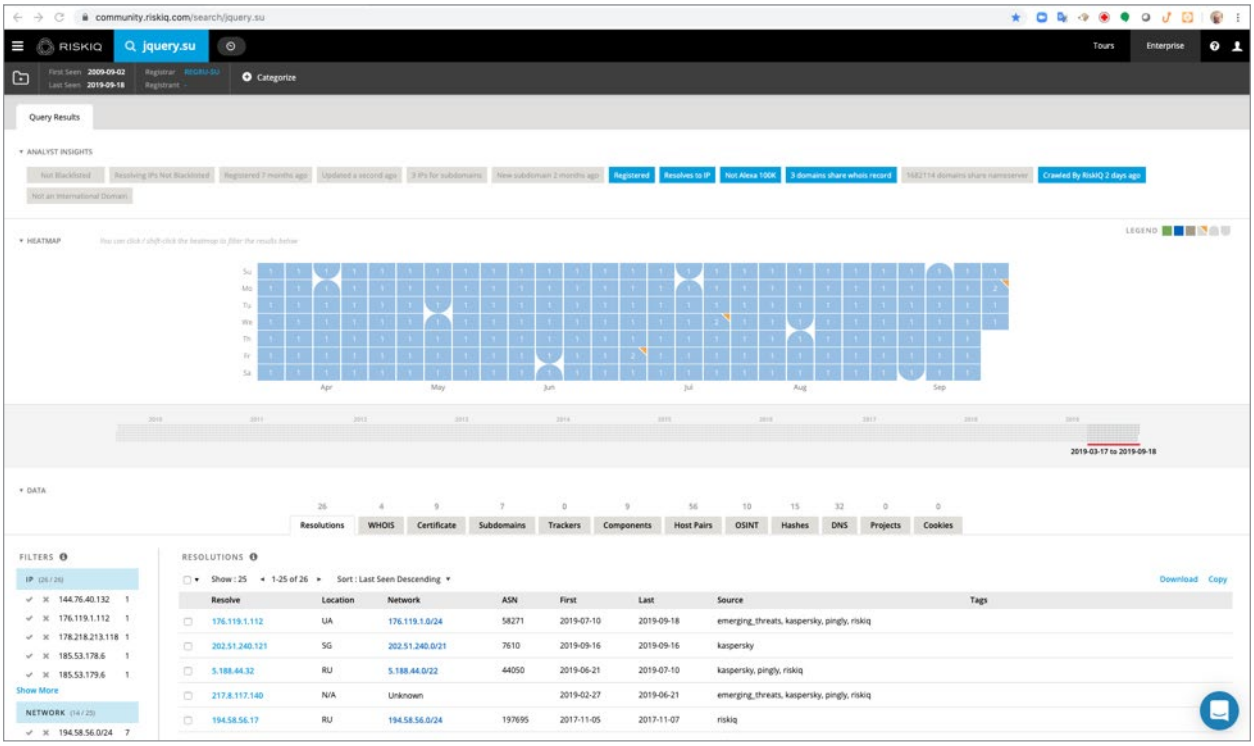
RiskIQ has enterprise products and features that would automatically monitor your websites and alert you to changes in the website's JavaScripts that you directly control or a third-party website you rely upon. But since we are using the RiskIQ Threat Investigation tool PassiveTotal we will have to manually investigate the domain to further understand what it is and if it is associated with malicious activity.

The next steps need to be done cautiously. Since we might be dealing with an active attack could infect your computer by visiting the website directly. It is important to have a safe way to visit the website and to not get compromised during your investigation.

You could just visit the website and view the source and see what is happening but that is really not safe. I will show you a safer way to do the investigation. You can investigate the websites and scripts and to not tip them off.

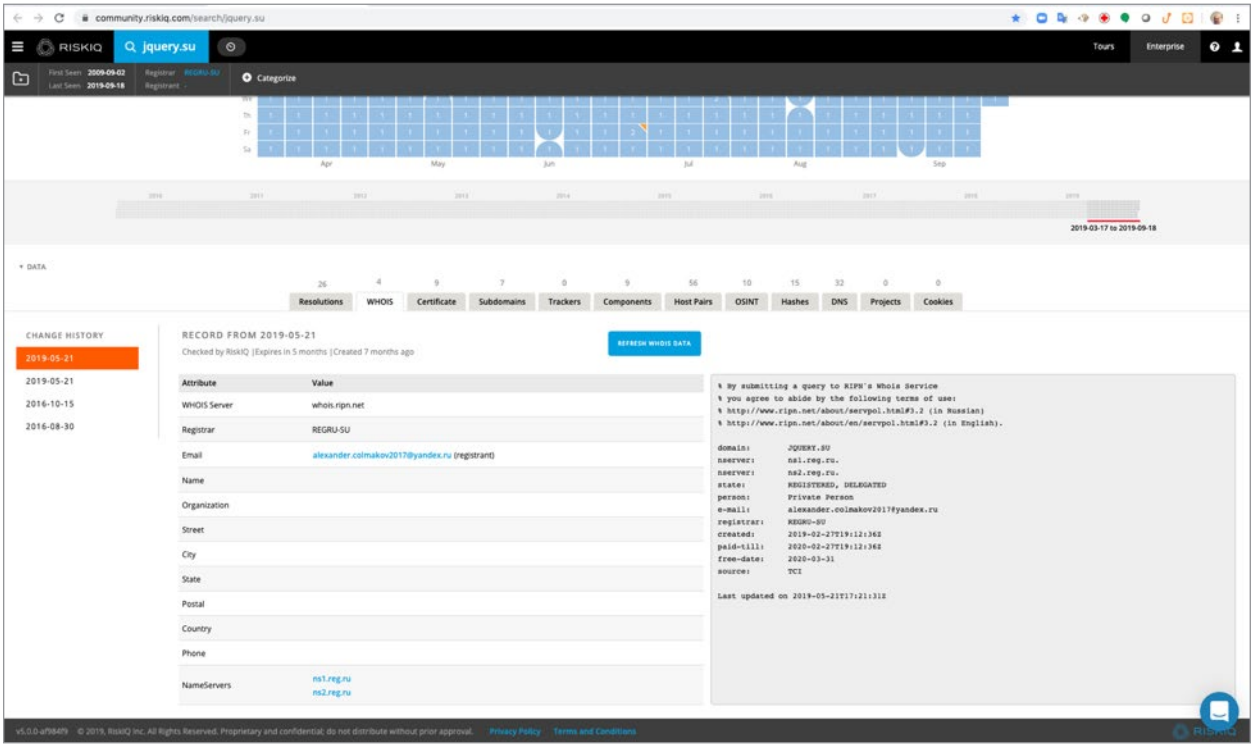
Step 8: Pivot search on jquery[.]su, right-click on jquery[.]su and open it in a new tab.





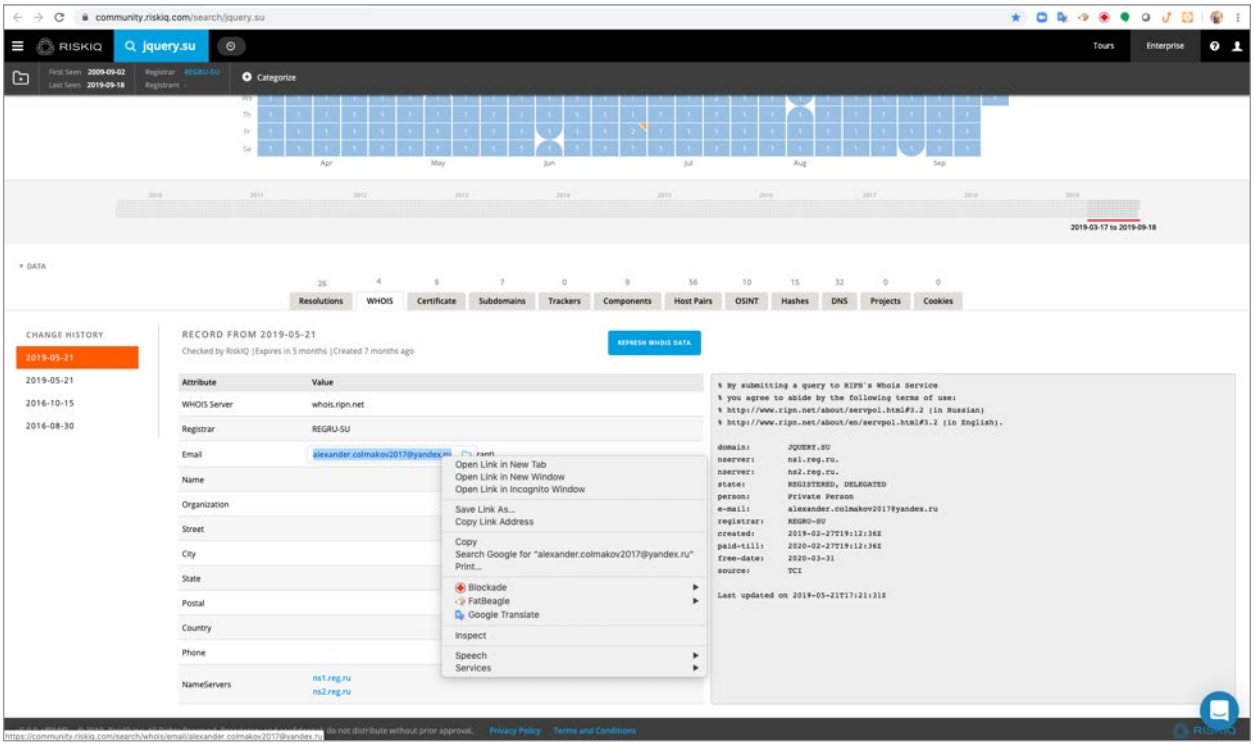
The location information on the Resolutions tab shows IP addresses from Ukraine, Singapore, and Russia. Flower Explosion is an American business and does not do business overseas. This could potentially be unusual.

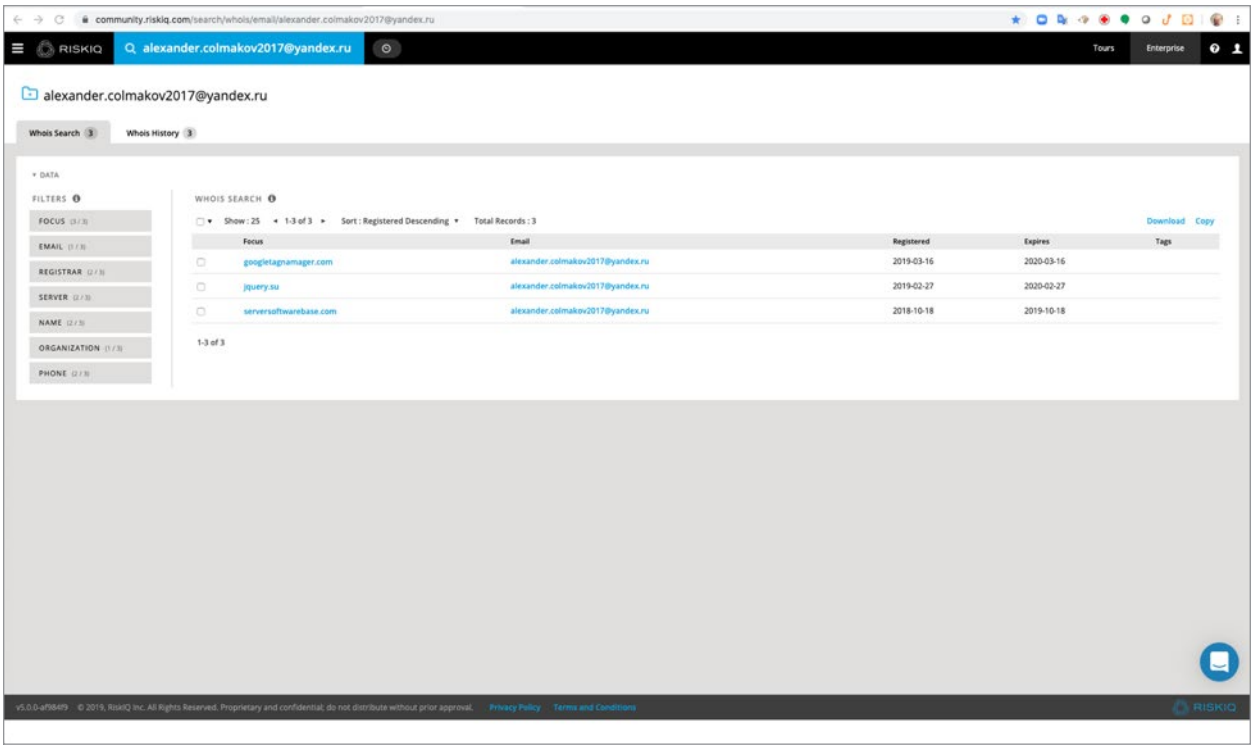
Step 9: Click on the WHOIS tab.



We see that the domain is registered to Alexander[.]colmakov2017[@]Yandex[.]ru.

Right-click on the email address and open it in a new tab.



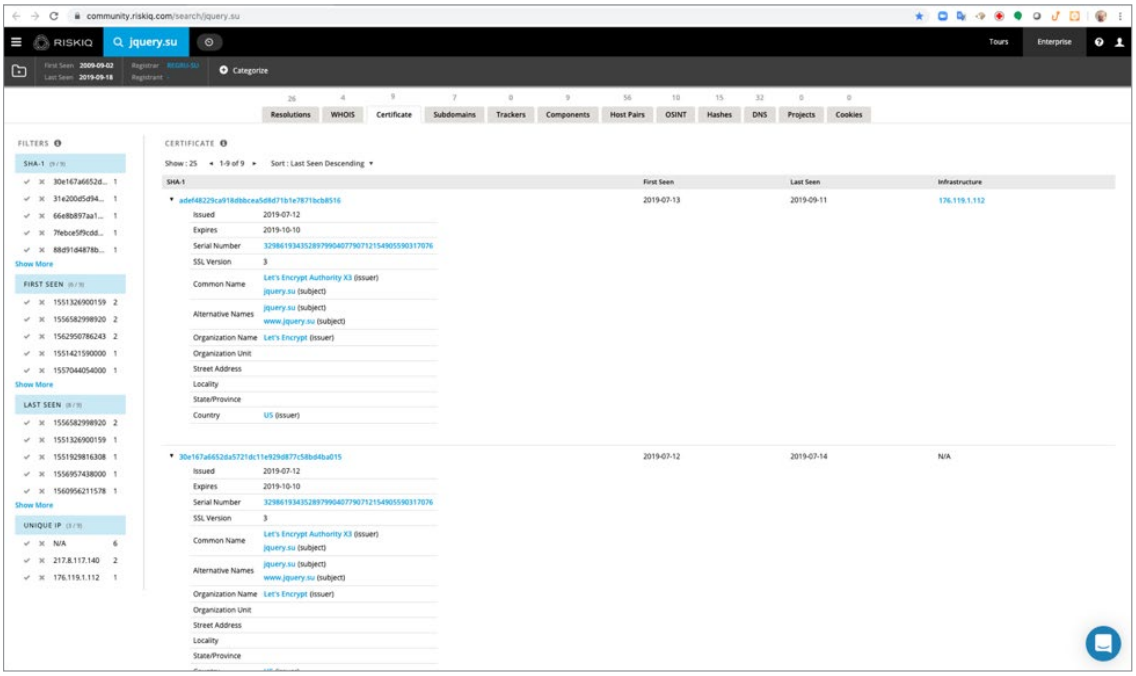


The URL should now show:
https://community.riskiq.com/search/whois/email/alexander.colmakov2017@yandex.ru

We now see Alexander has registered 3 different domains. A typo squatted googletagmanager domain named googletagnamager[.]com, jquery.[su], and serersoftwarebase[.]com.

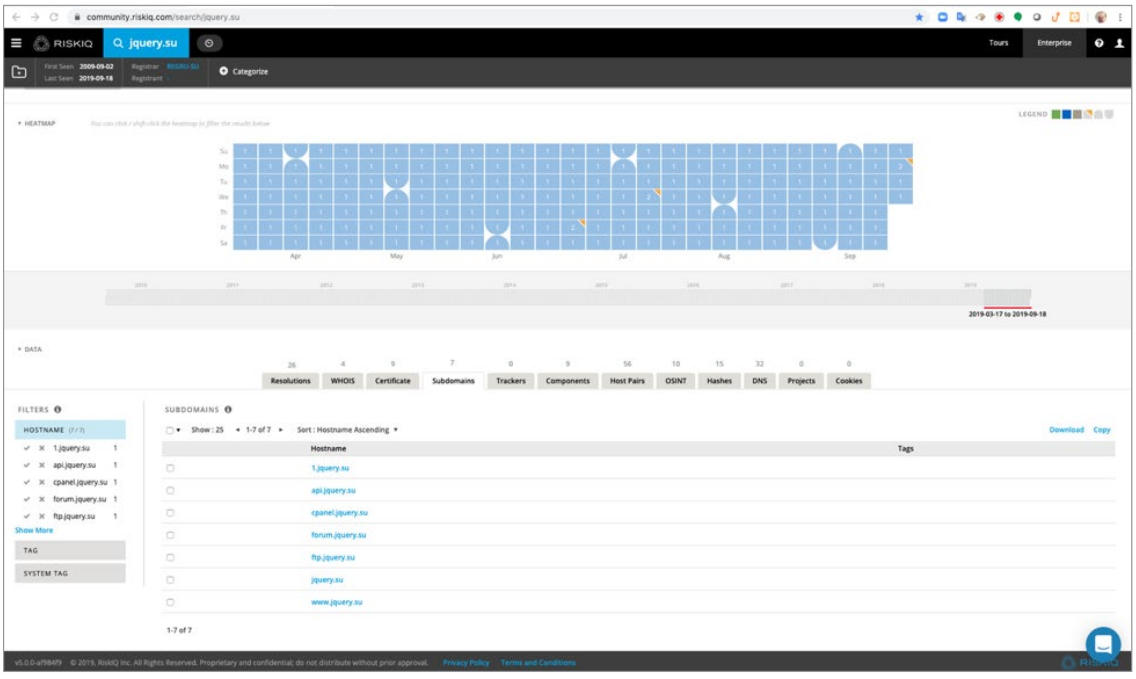
Step 10: Go back to the previous tab for the results for jquery[.]su.

Click on the Certificate tab and expand the SHA-1 results to identify where the certificate came from.



The results show that the domain jquery[.]su is utilizing free let's Encrypt SSL certificates. This is a common item we see with threat actors.

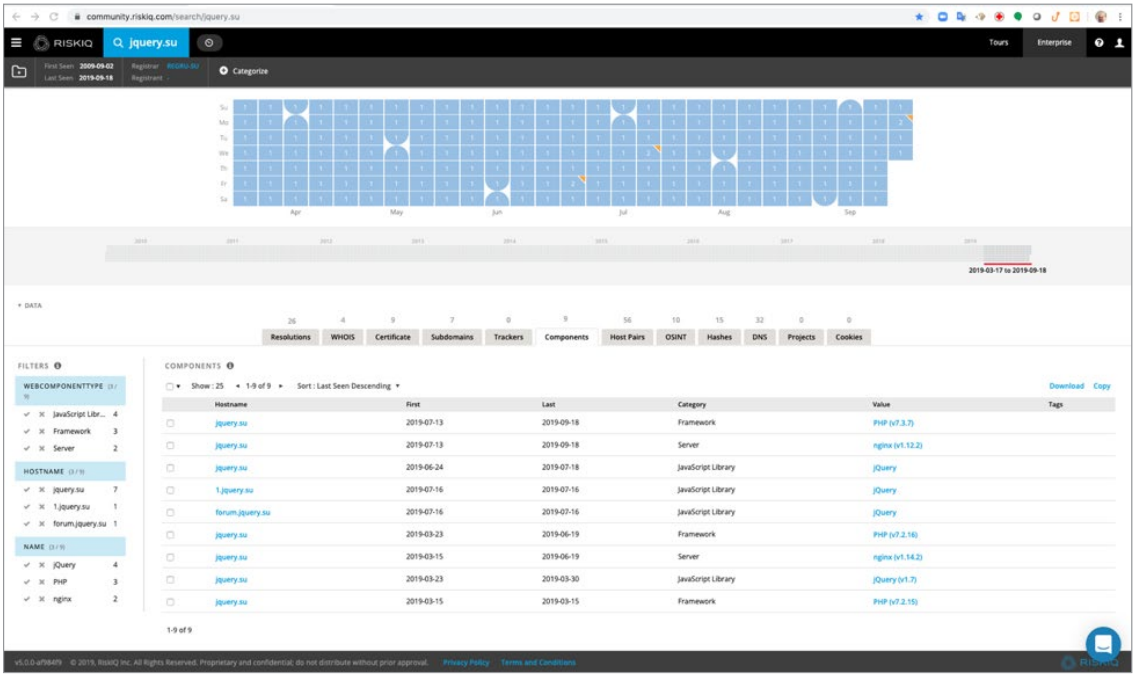
Step 11: Click on the Subdomains tab.



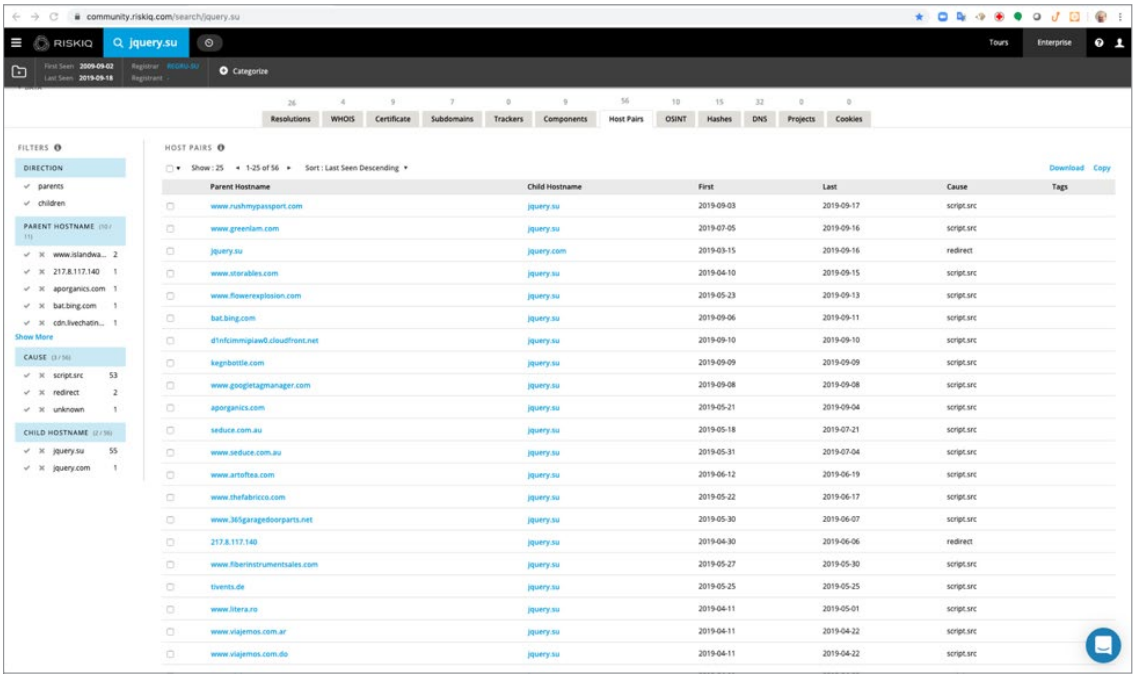
Each of the subdomains could have completely different infrastructure associated with it. We will investigate this later in a different use case.

Step 12: Click on the Components tab

We see a very lean components list. Threat actors do not usually stand up components they do not utilize. We can see they this domain utilizes PHP, nginx, and JQuery.



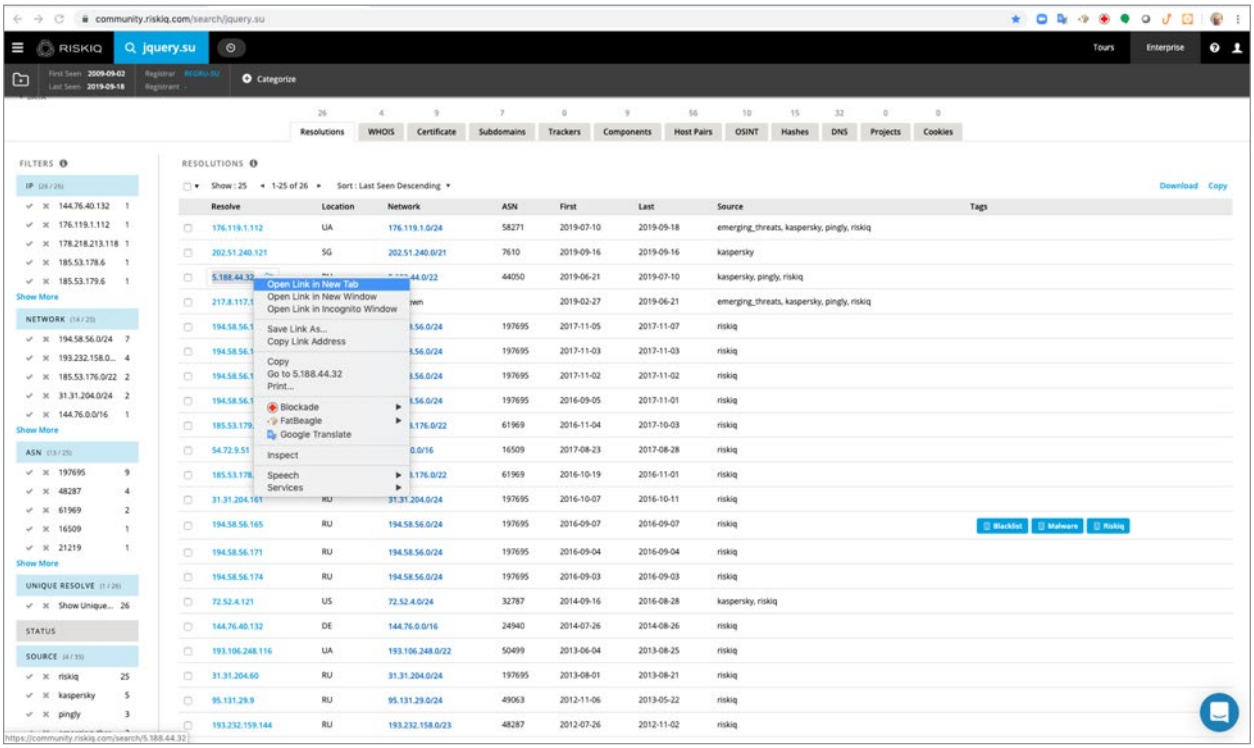
Step 13: Click on the Host Pairs tab

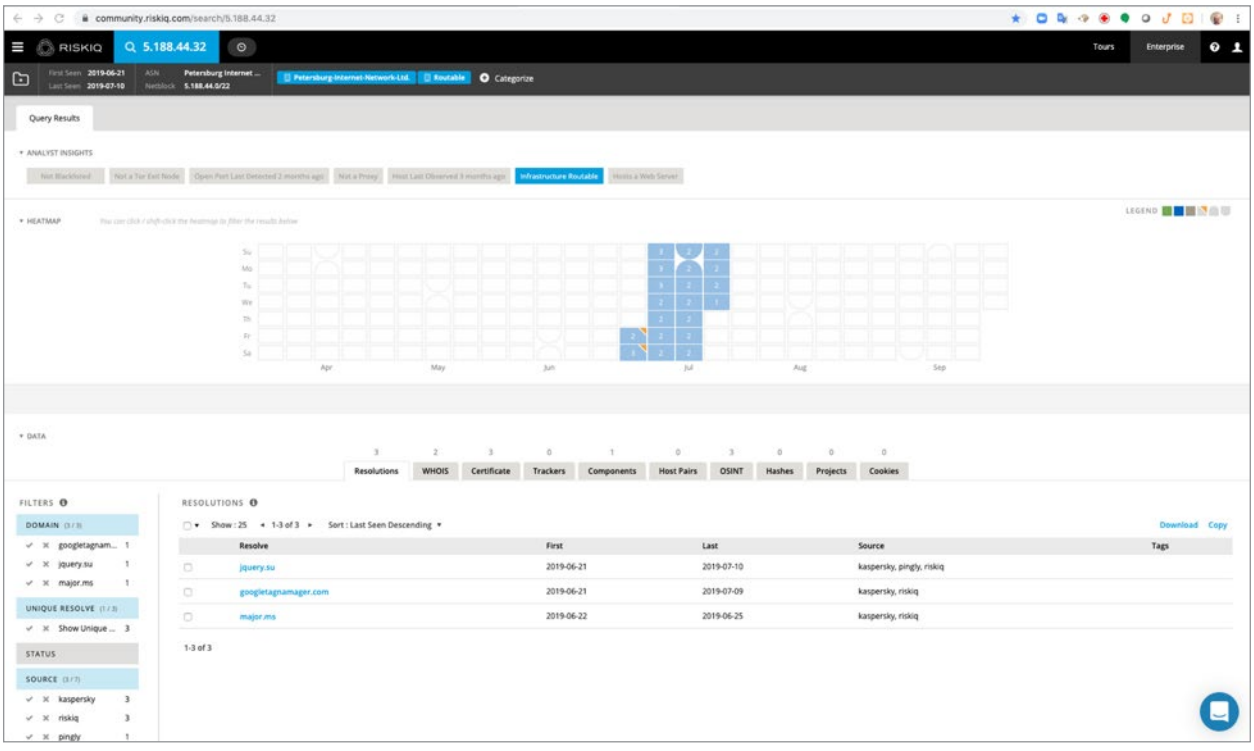


The results show us all of the domains that are going to jquery[.]su and running JavaScripts. From the names listed most appear to be online retailers. If jquery[.]su is determined to be malicious all of these domains might also be compromised.

Step 14: Click on the Resolutions tab

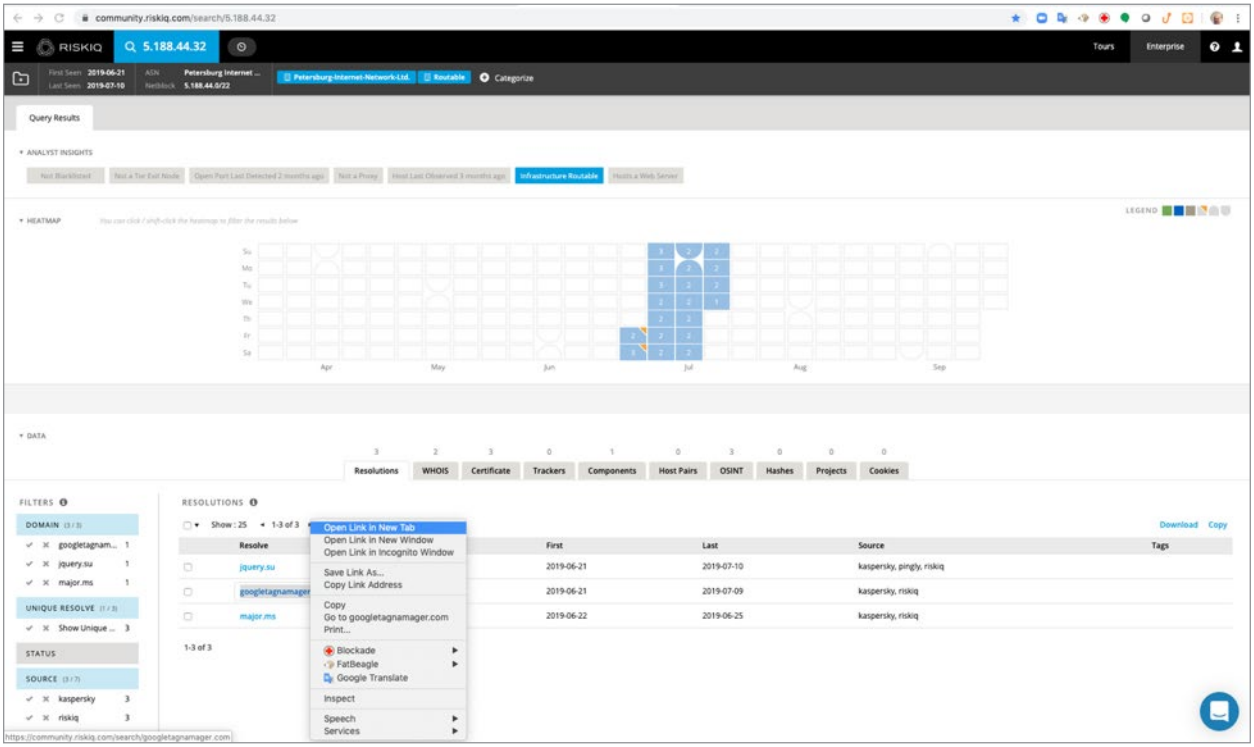
Right-click on the IP address 5[.]188[.]44[.]32 and open it in a new tab.

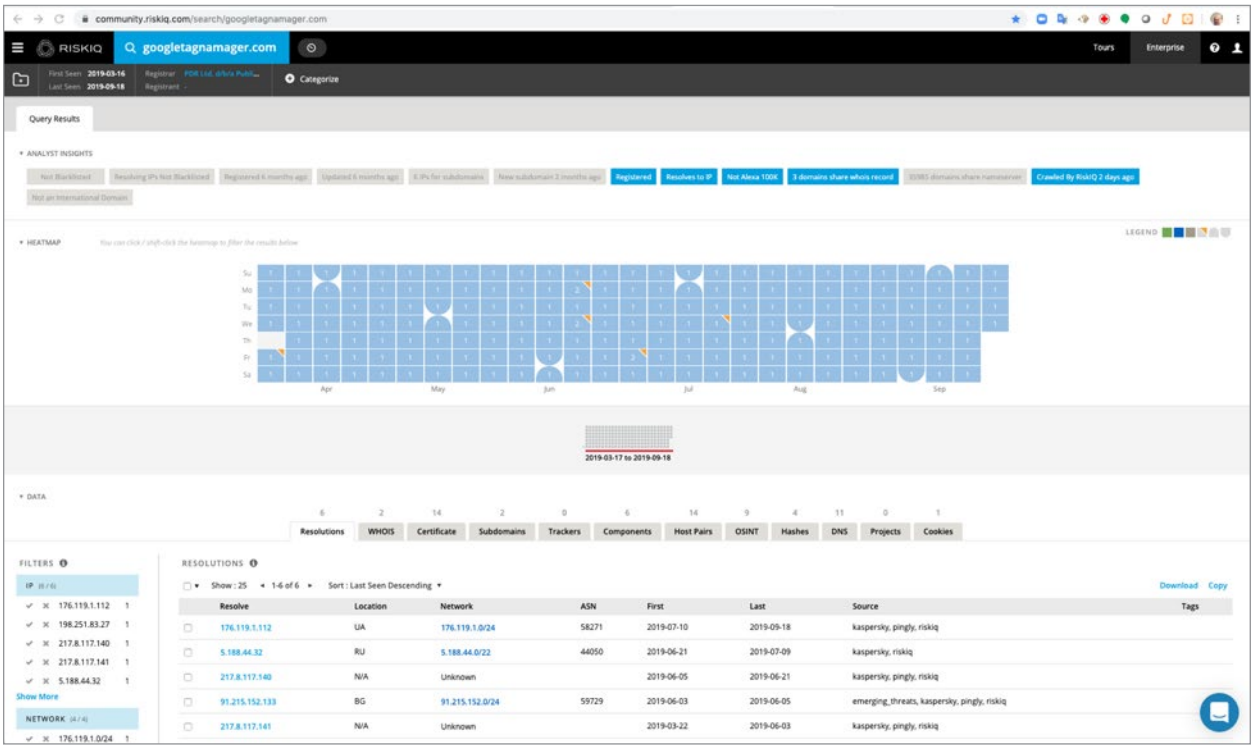




The results show 1 new domain. Now we have identified jquery[.]su, googletagnamager[.]com, serversoftwarebase[.]com, and now major[.]ms.

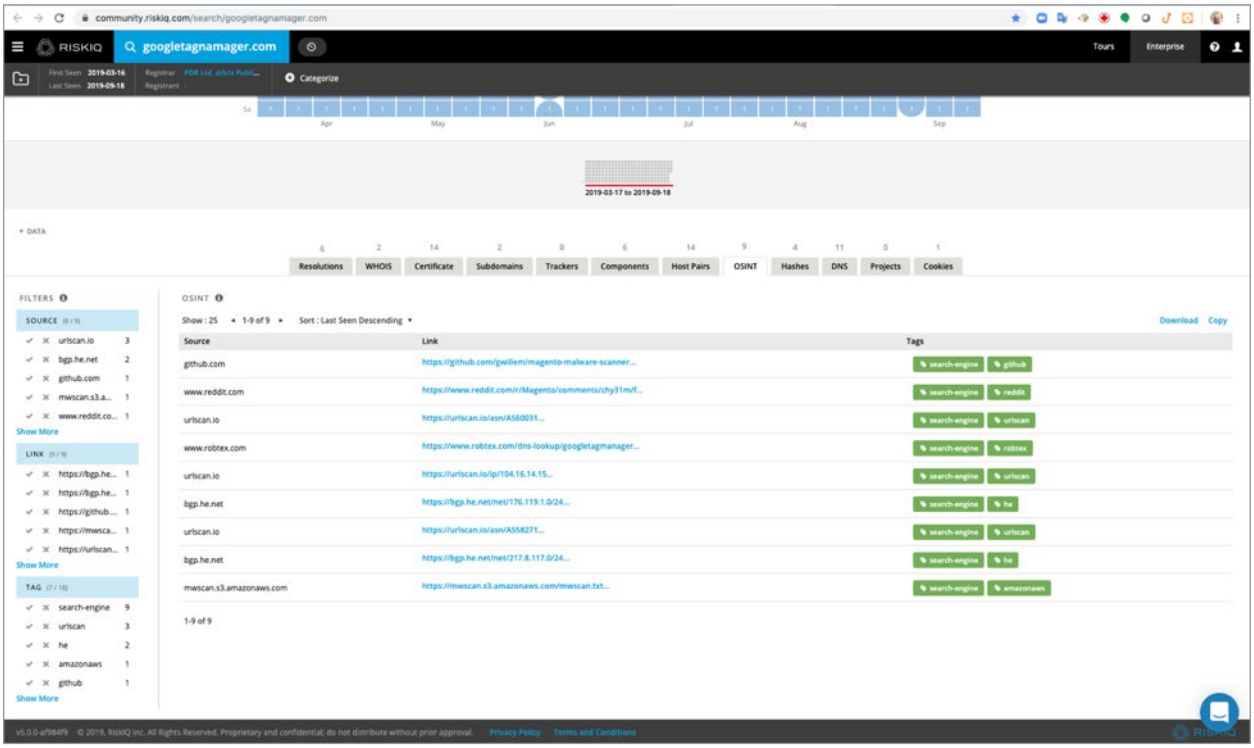
Step 15: Right-click on googletagnamager[.]com and open it in a new tab.



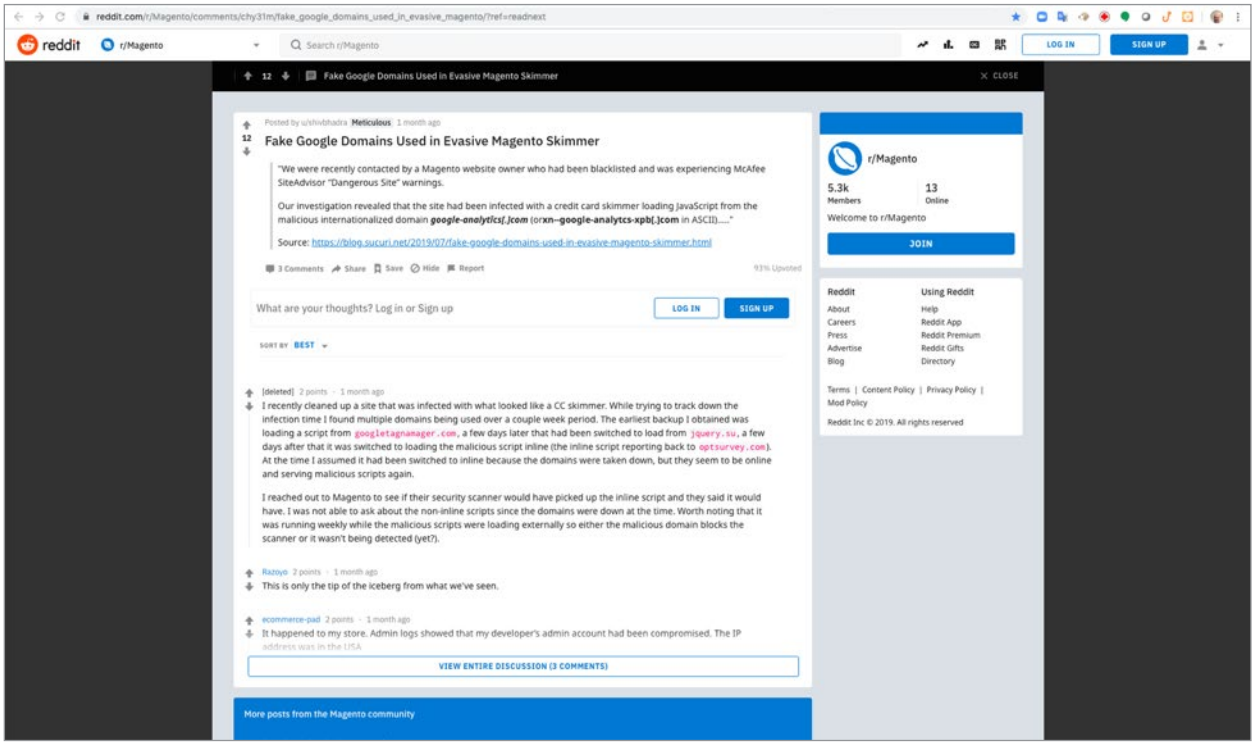


We can see that this domain has been active since 2019-03-16 to present. It is been located to IP addresses in Ukraine and Russia.

Step 16: Click on the OSINT tab.



Click on the link for Reddit.
https://www.reddit.com/r/Magento/comments/chy31m/fake_google_domains_used_in_evasive_magento/?ref=readnext



The results mention jquery[.]su, googletagnamager[.]com and how it is involved in a skimmer. Now we have confirmation that jquery[.]com is associated with malicious activity. It also says that the attack is against Magento and that is what our website uses for its online store.

But only if we had the server to see the jquery[.]su script. Is our investigation over?

No, we can utilize other tools that might have captured the website and preserved it.

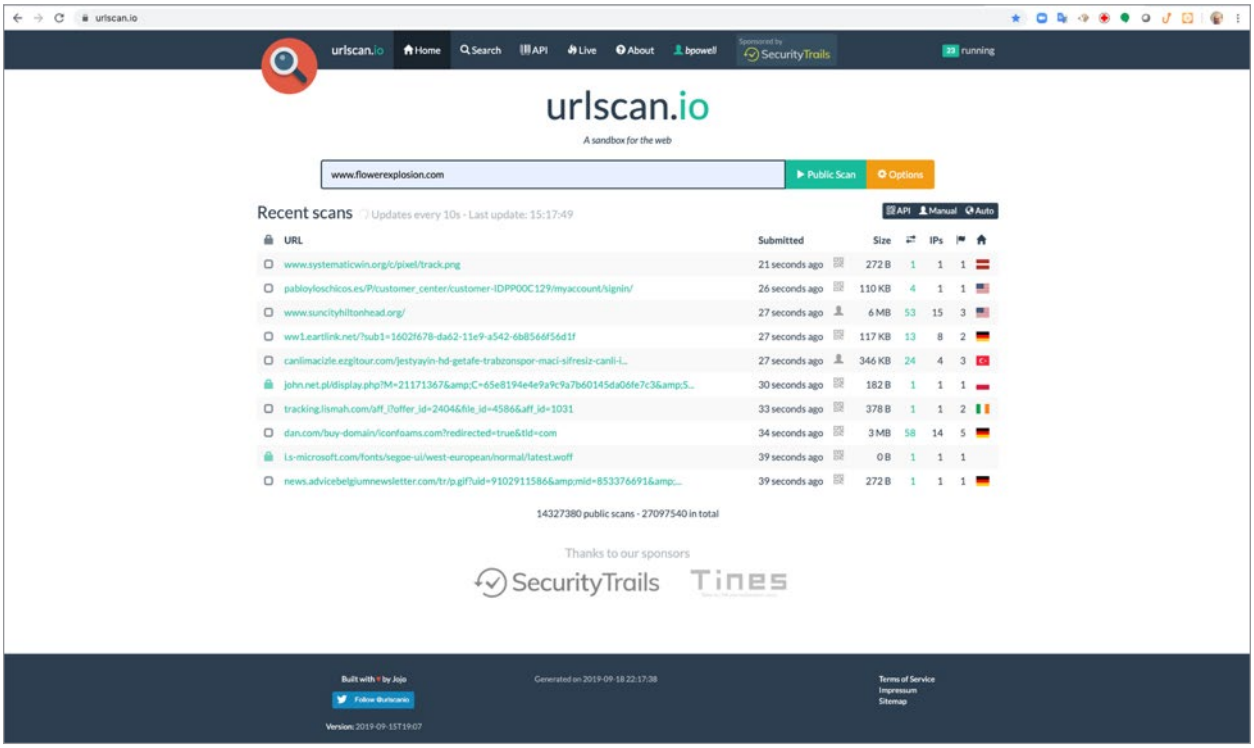
Step 17: Crawl the website www[.]flowerexplosion[.]com utilizing urlscan.io

We are now going to use a tool to visit and crawl the website www[.]flowerexplosion[.]com.

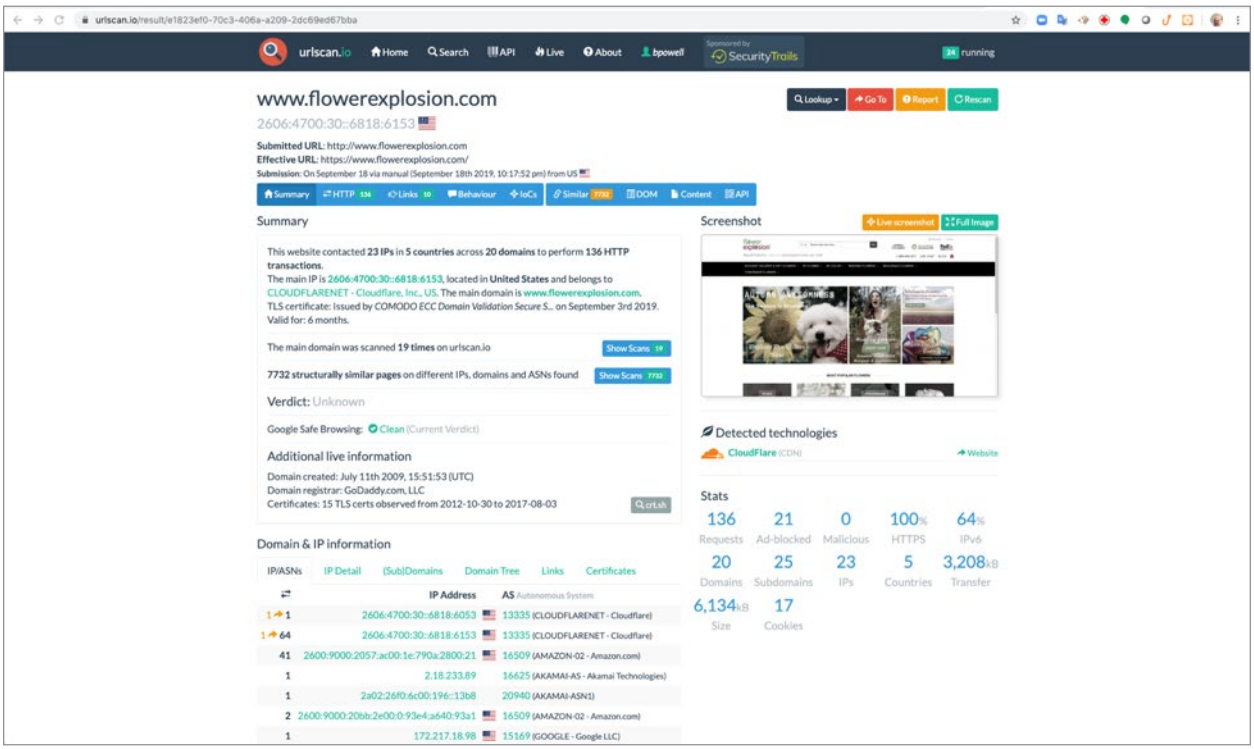
This is a safe way to investigate a website's content without directly going to the website. You can just view the results from the web crawl and understand what is happening. This will prevent your computer from potentially getting compromised and potentially tipping off the threat actor that you are investigating them.

In a new tab open the website https://urlscan.io

Search for www[.]flowerexplosion[.]com

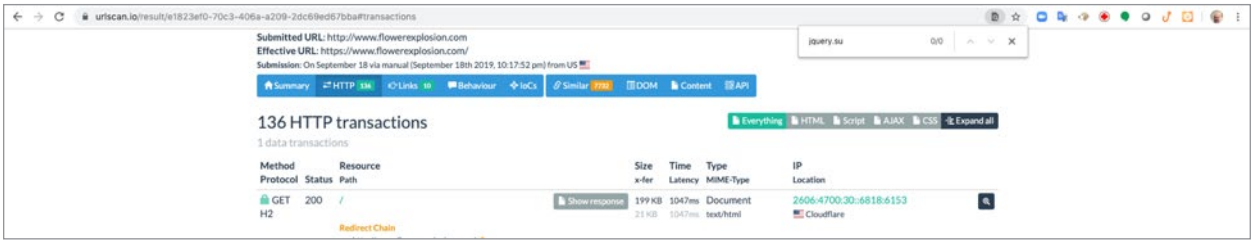


Step 18: Review the results



Click on the HTTP tab.

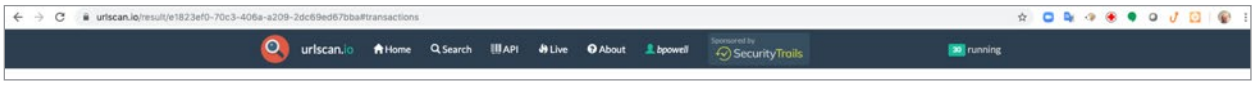
In your browser application search for “jquery.su”



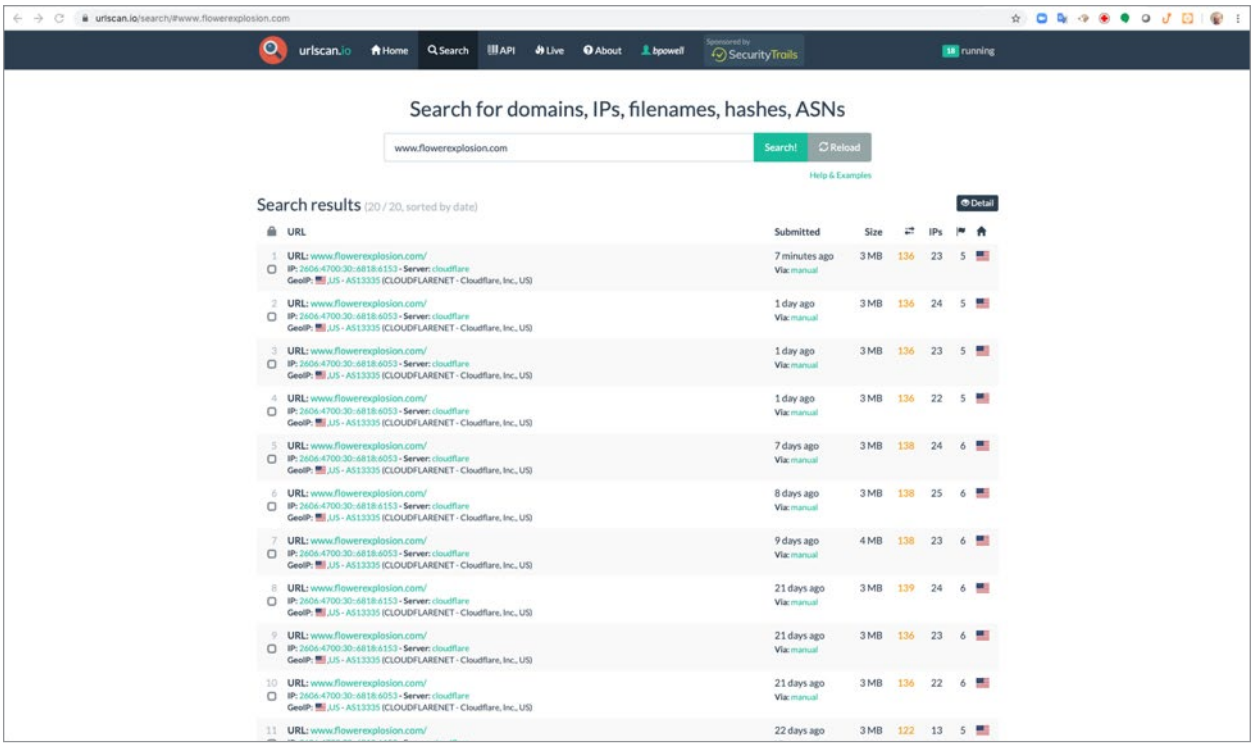
No results. Is the investigation over?

No, we can look at the previous crawls and see if they detected jquery[.]su on our server.

Click on Search on the title bar.



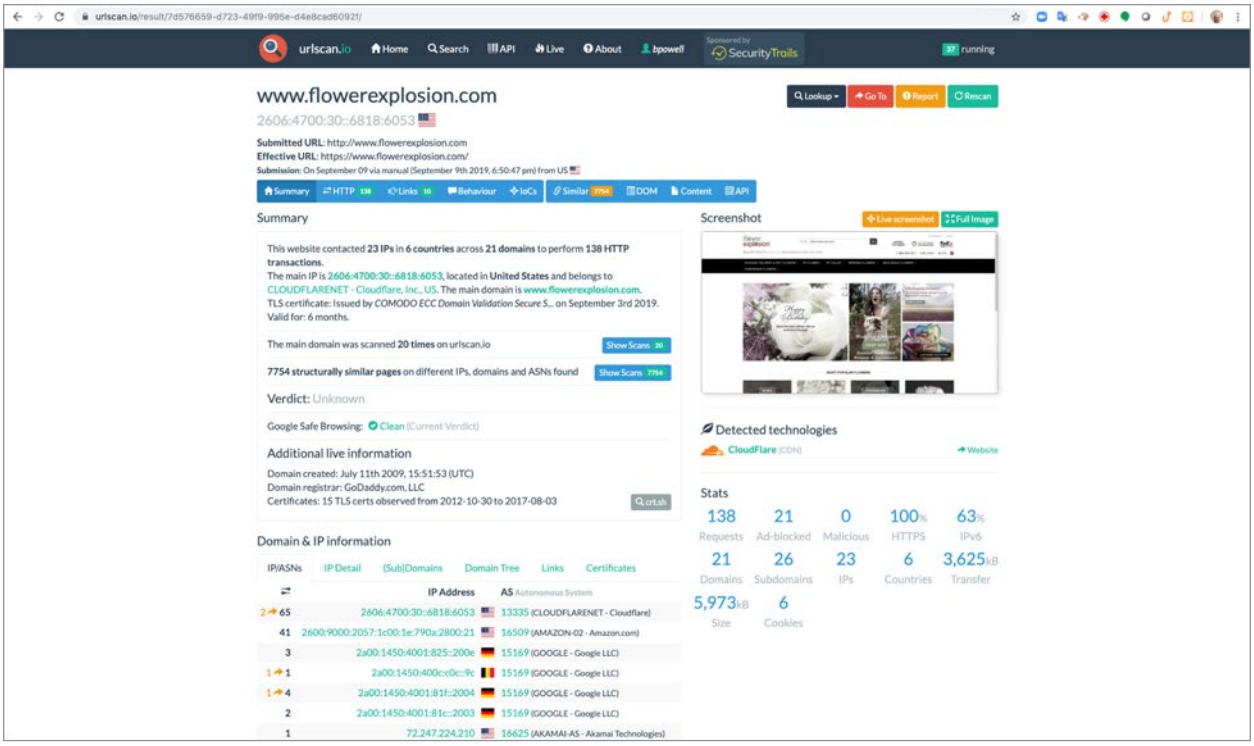
Search for www[.]flowerexplosion[.]com



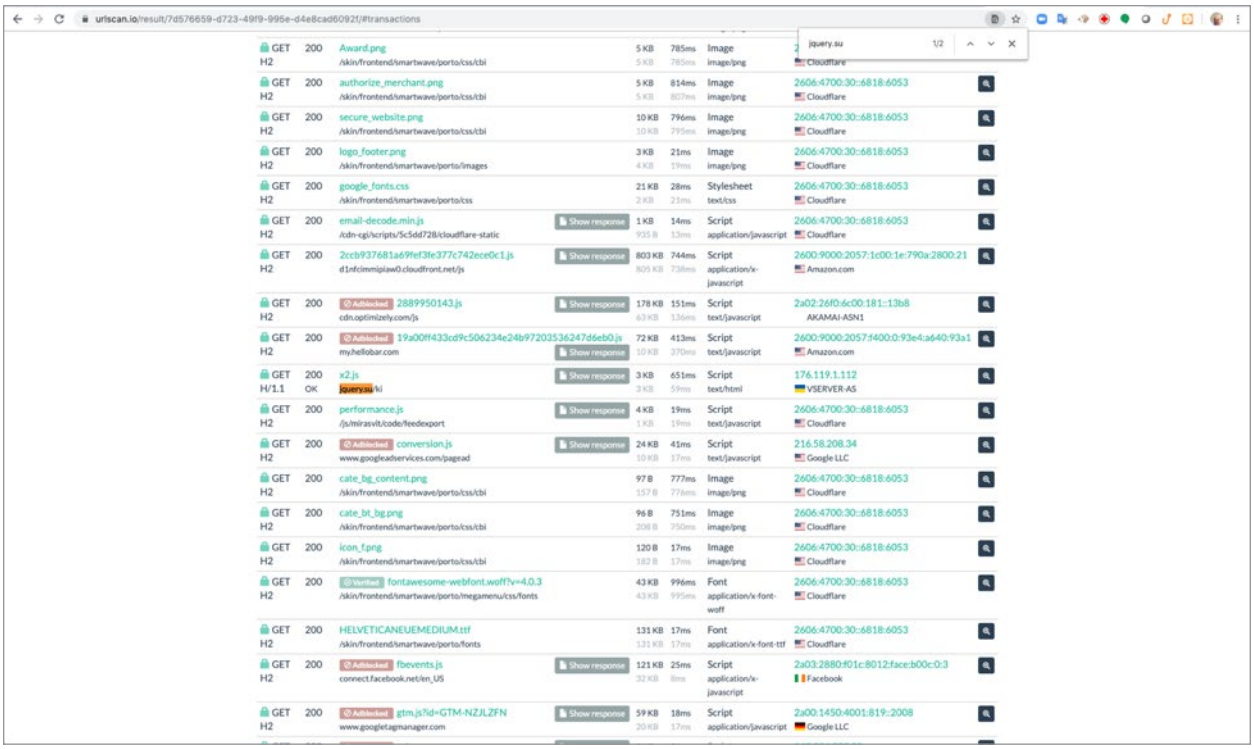
Results are from a few minutes ago to 22 days ago.

Click on a result from over a week ago.

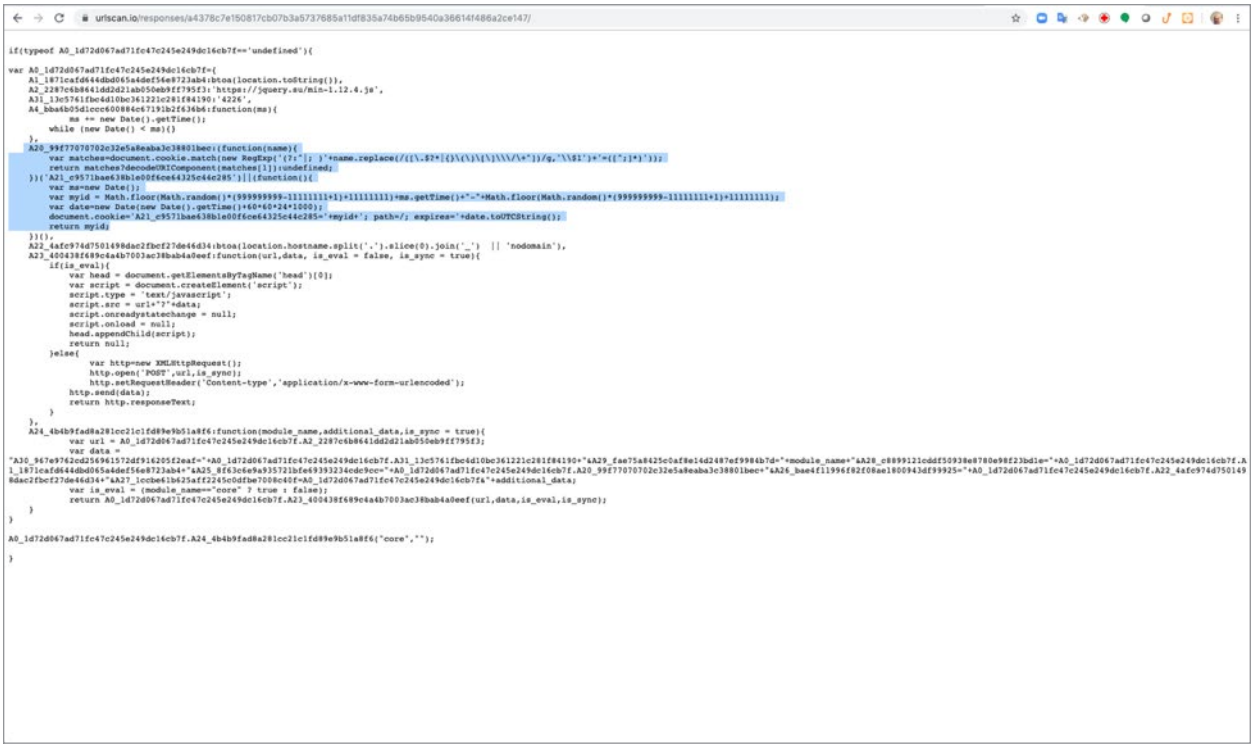
https://urlscan.io/result/7d576659-d723-49f9-995e-d4e8cad6092f/



Click on HTTP in the blue title bar, then search for “jquery.su”



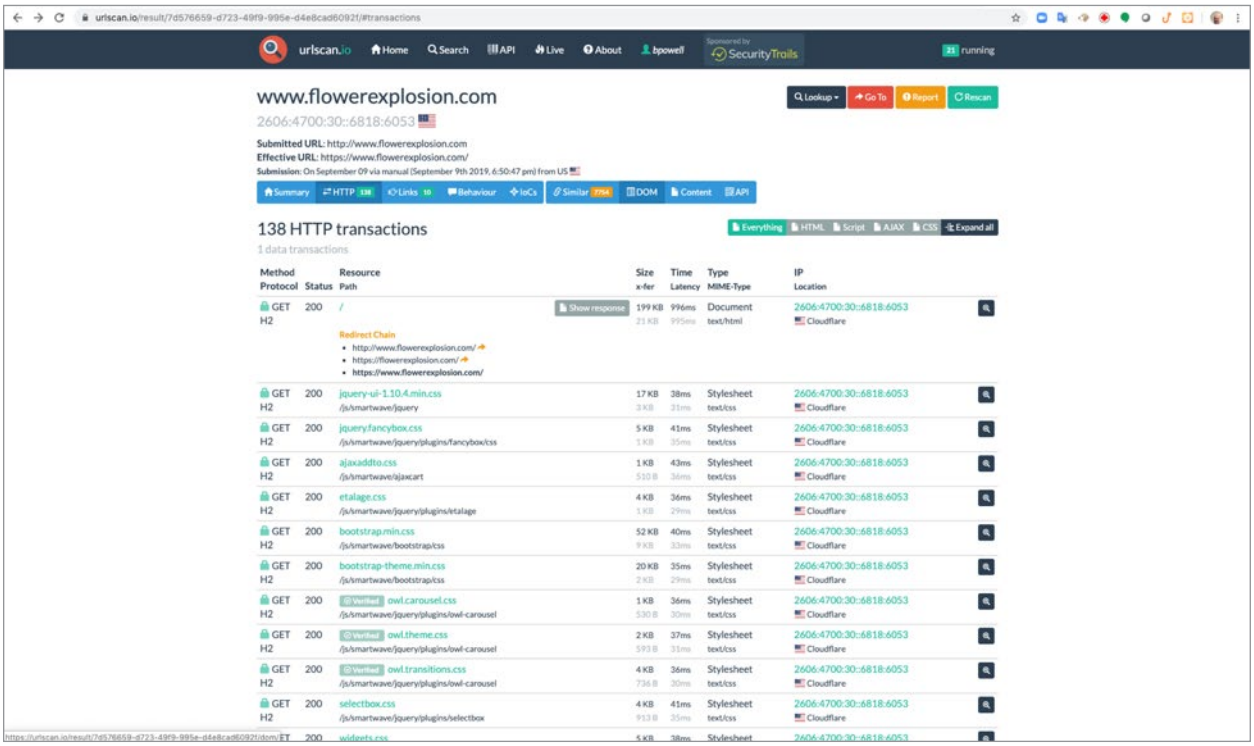
Click on the show response button. If you do not see a show response button next to jquery[.]su then hit the back button until you see the search results and choose a later result to view. Here we see the results for jquery[.]su/ki for the script x2[.]js.



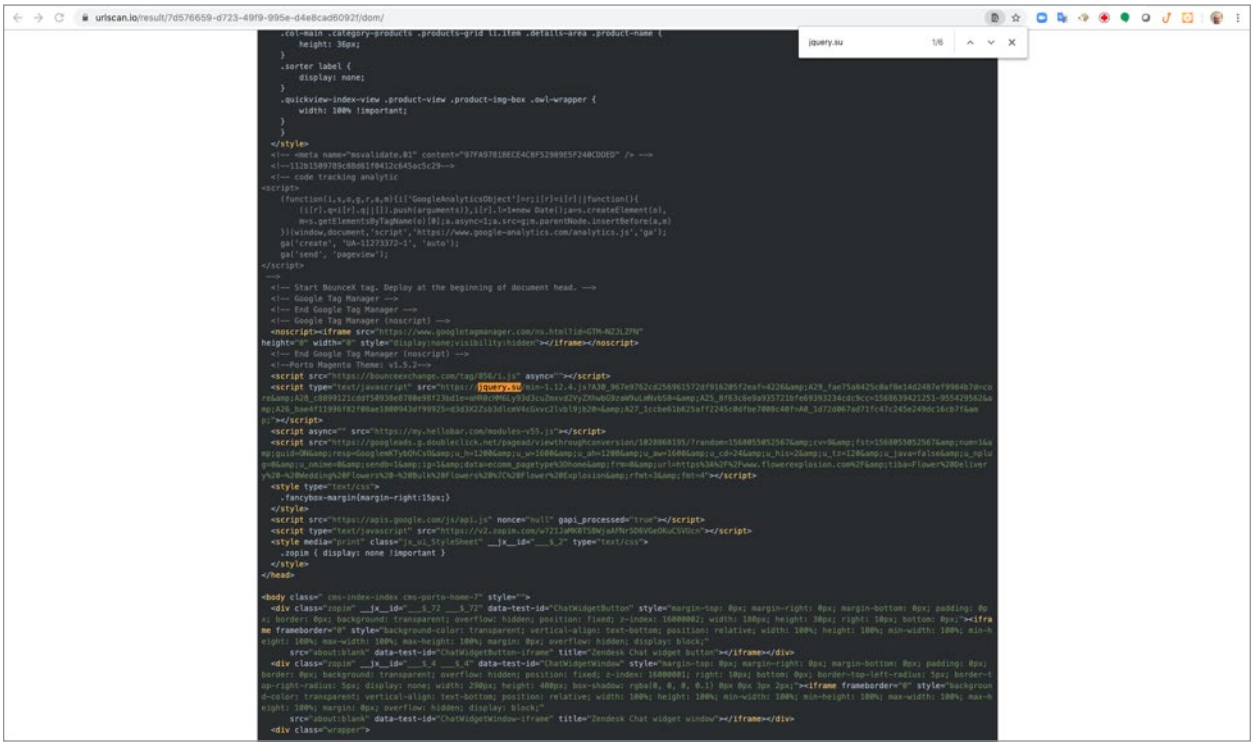
https://urlscan.io/responses/a4378c7e150817cb07b3a5737685a11df835a74b65b9540a36614f486a2ce147/

Now we can finally see the JavaScript. The highlighted areas show that the scrip was checking for a payment card and also executed another script called https[:]//jquery[.]su/min-1.12.4[.]js.

Click the back button, then click on DOM



Then in your browser search for “jquery.su”

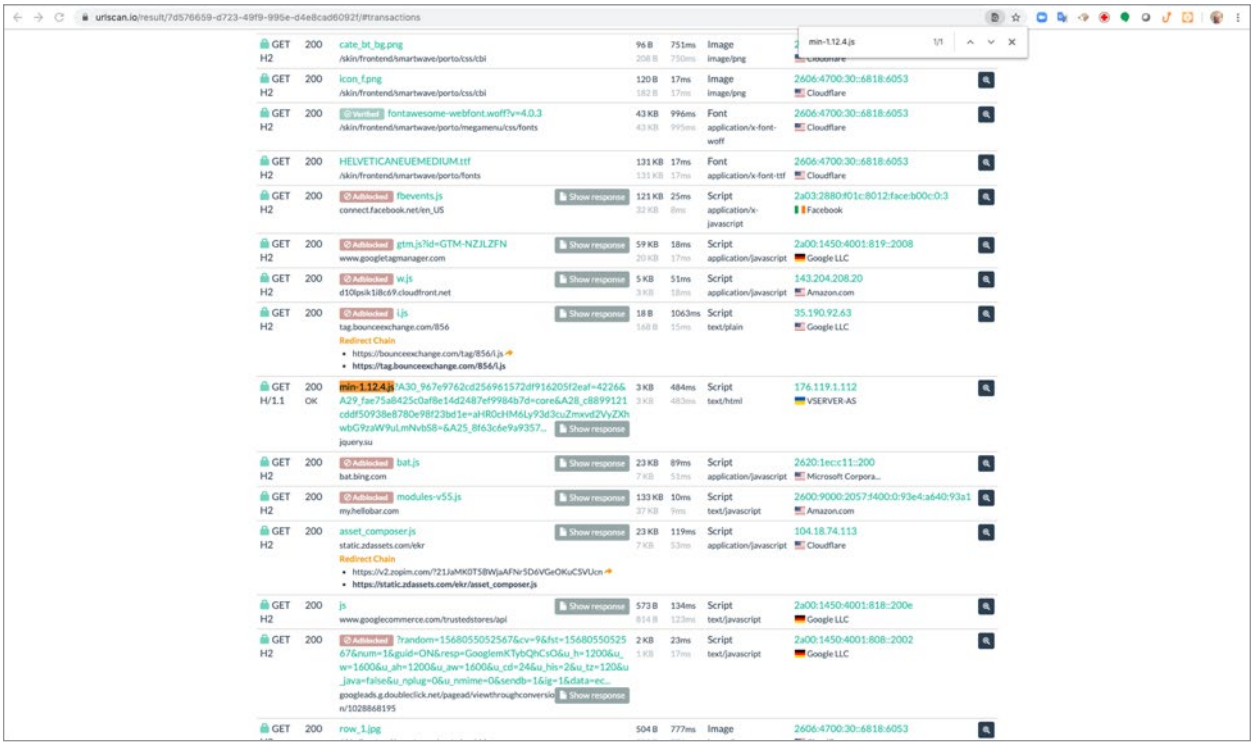


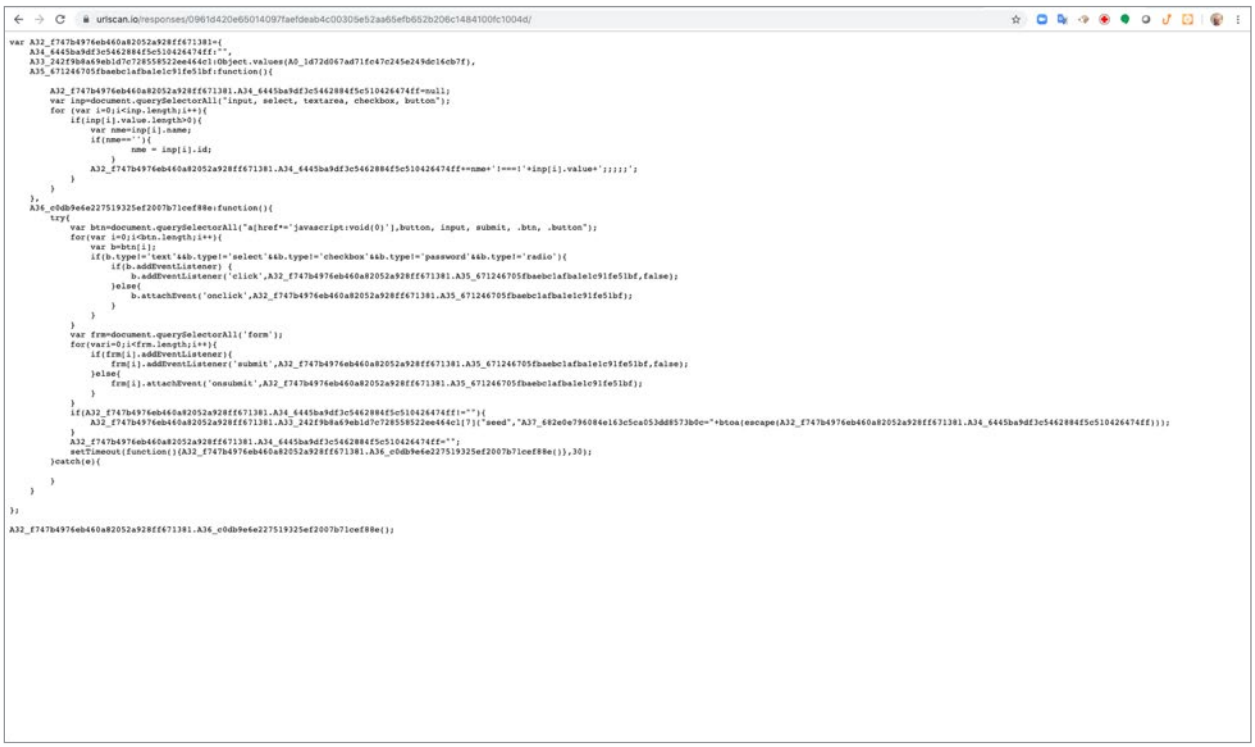
There are 6 separate entries in the DOM for jquery[.]su.

One entry for src="https[:]//jquery[.]su/min-1.12.4[.]js"
Five entries for src="https[:]//jquery[.]su/ki/x2[.]js"

If you go back to the HTTP results you can search for each script.

Here are the results for "min-1.12.4.js"





This appears to be looking for specific things on the checkout page, button, input, submit.

Now we have determined www[.]flowerexplosion[.]com was infected with Magecart from jquery[.]su. The threat actor appears to be in Ukraine or Russia. From Open Source Intelligence (OSINT) the threat actor targets Magento eCommerce platforms.

But we still have an issue. Server Admins blow away the website and restored a previous version that was known to be clean. We do not know if the server is vulnerable to attack and could get compromised again.

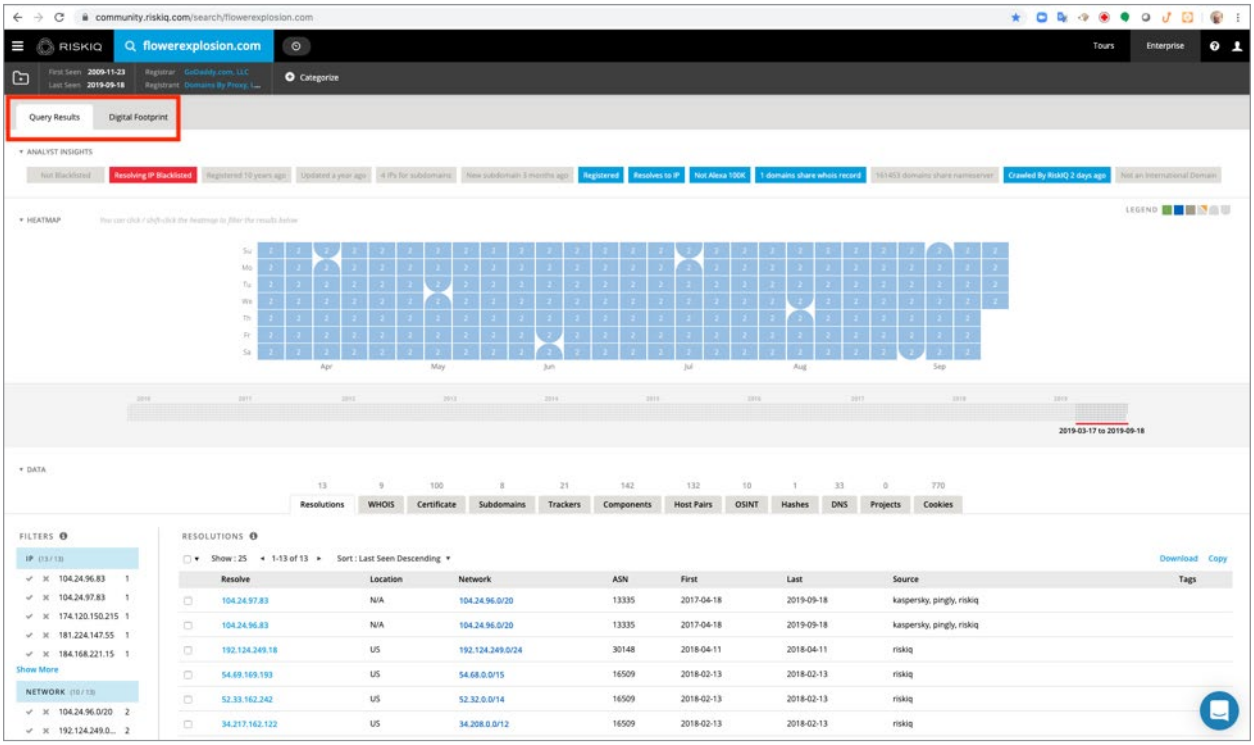
We need to investigate the www[.]flowerexplosing[.]com and determine what vulnerabilities are present.

Step 19: What vulnerabilities exist on my website?

Open a new tab and go to https://community.riskiq.com/.

Search for the domain flowerexplosion[.]com removing no www.

You should see an additional tab next to query results called Digital Footprint.



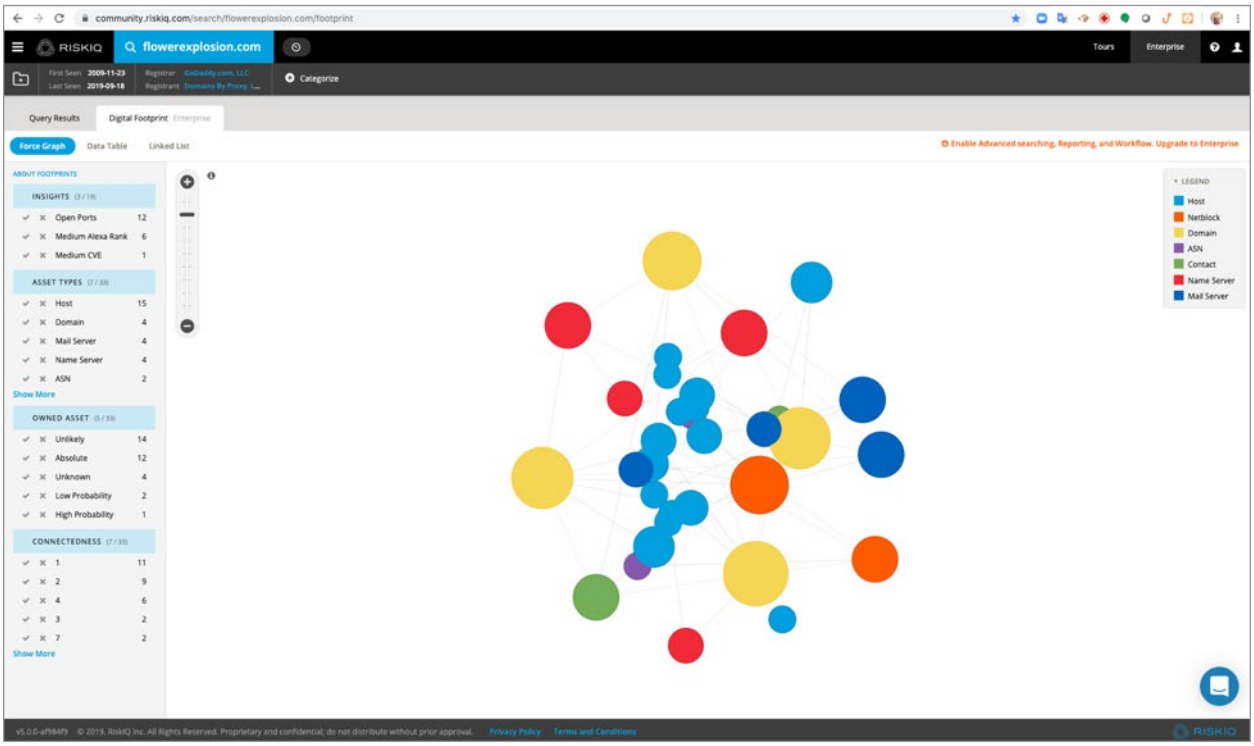
Step 20: Click on the Digital Footprint tab

https://community.riskiq.com/search/flowerexplosion.com/footprint

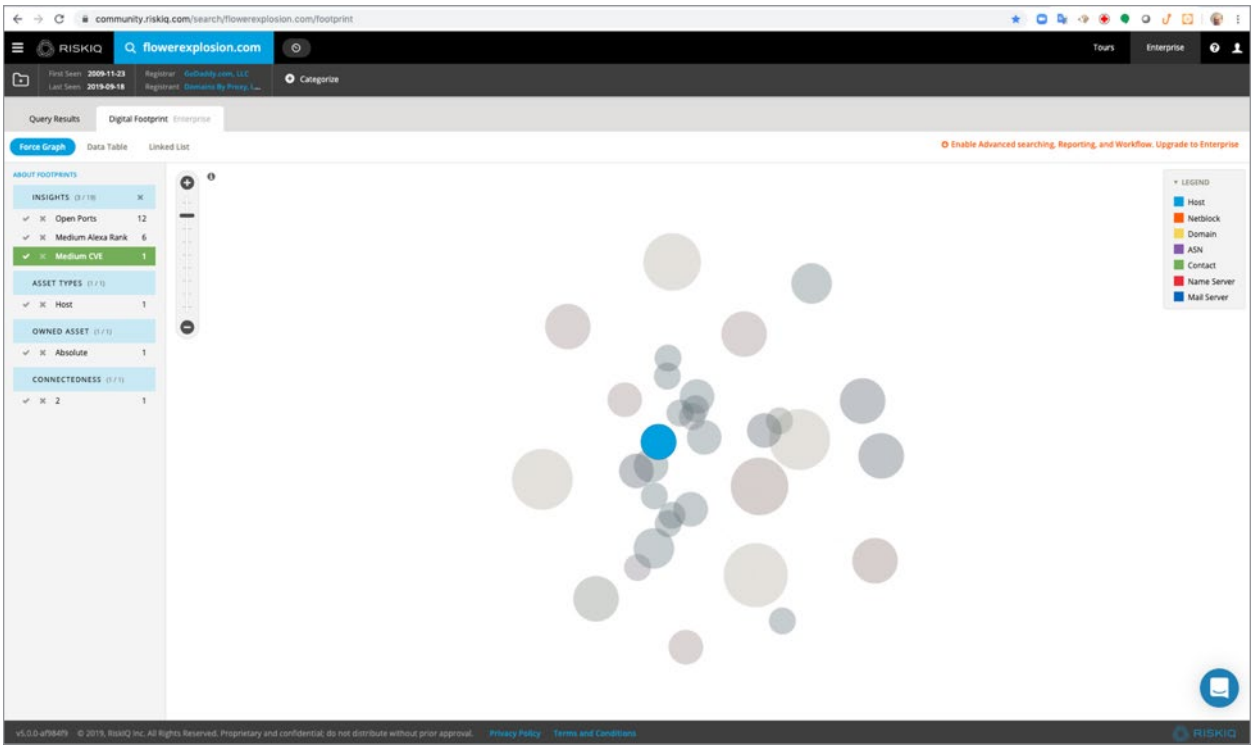
RiskIQ automatically created a Digital Footprints for all work domains when the PassiveTotal account is created. You cannot see a Digital Footprint if you used your Gmail or yahoo mail accounts.

We have allowed all organizations to view flowerexplosion[.]com Digital Footprint and see their internet attack surface. Community users will normally see hostnames obfuscated. Paid Enterprise users will see a non-obfuscated Digital Footprints with all hostnames. This is a non-curated view of your attack surface. You cannot add or remove hosts from your Digital Footprint. The Digital Footprint is machine created based upon your email domain. Digital Footprint Enterprise edition is fully customizable where you can add or remove hosts and domains from your Digital Footprint.

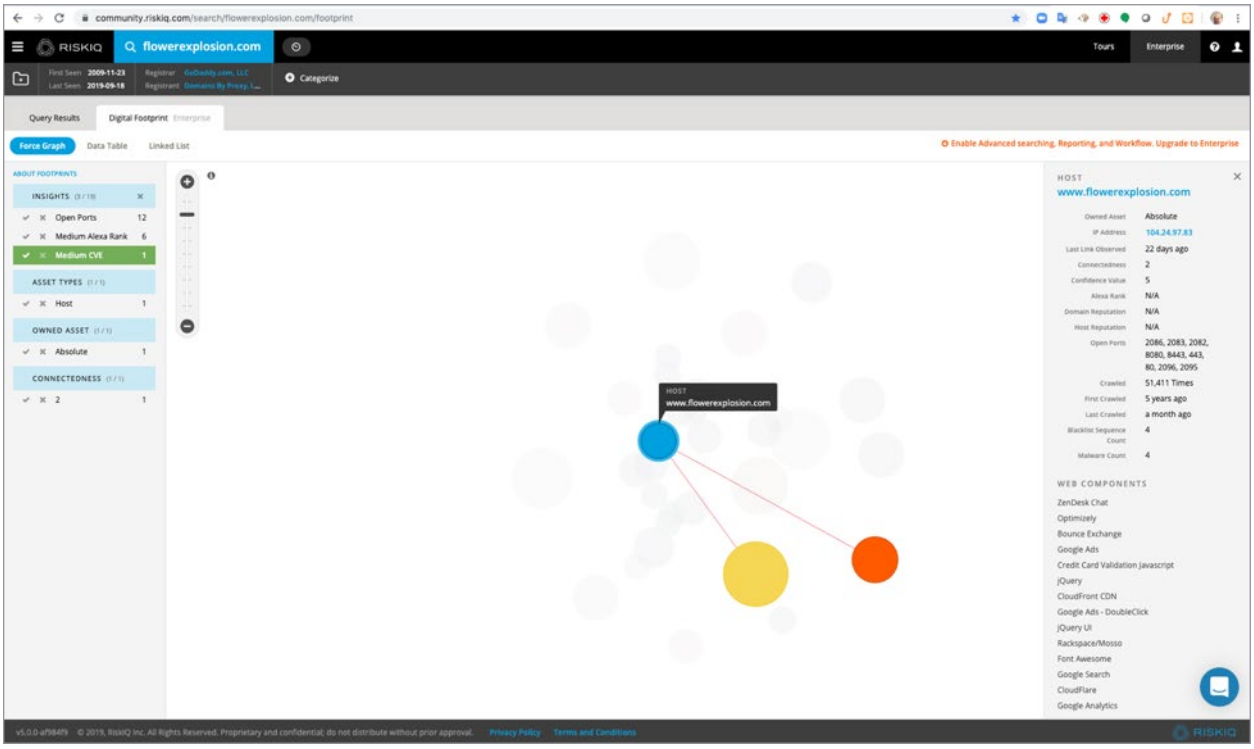
The vulnerabilities that are shown in the Digital Footprint are not tested they are based upon the version and banner information that is determined during RiskIQ's crawling of websites. This information might not be 100% accurate but it does point you in the right direction to determine where vulnerabilities may exist and what they might be. This information can be downloaded and used by your organization vulnerability scanning tool to determine what exact vulnerabilities exist and if any mitigating controls are in place to protect your organization.



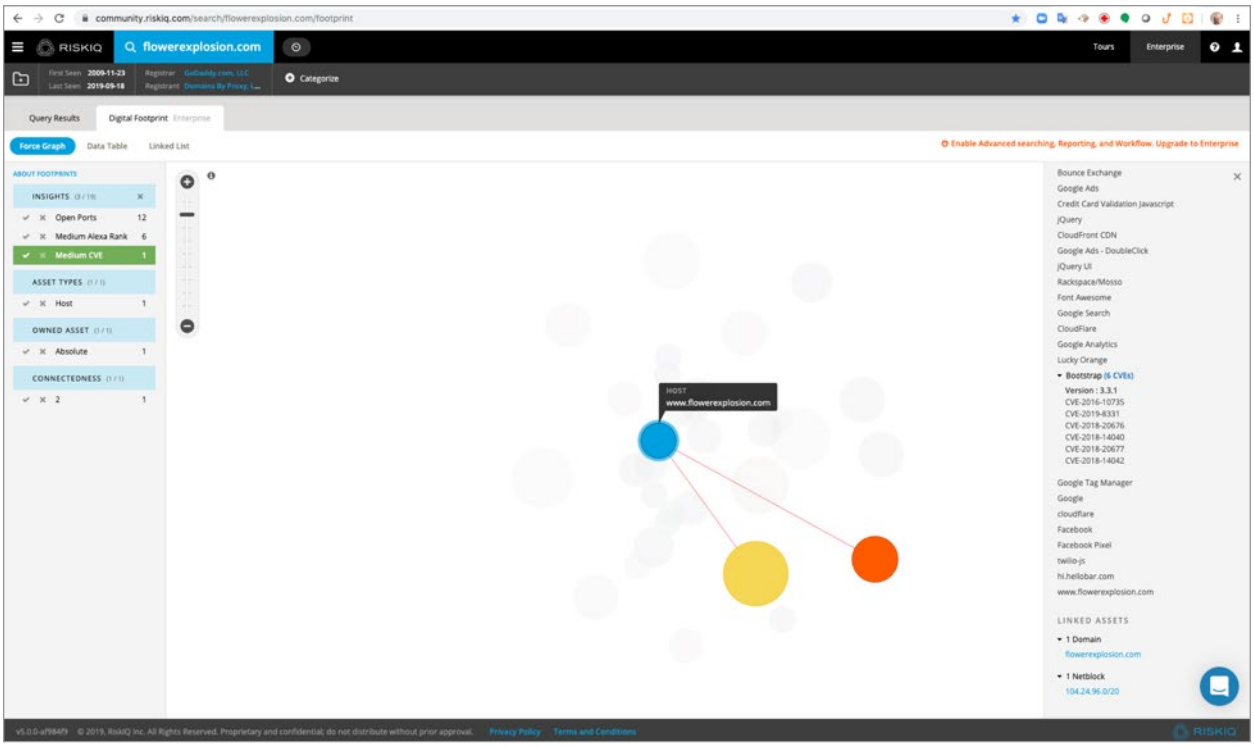
In the Insights section on the left side of the screen click on the check icon next to Medium CVE.



Only one item is shown in the graph. Click on the blue circle



In the right-hand side is now revealed and you will see information about the host www[.]flowerexplosion[.]com. Scroll down until you see the CVE information.

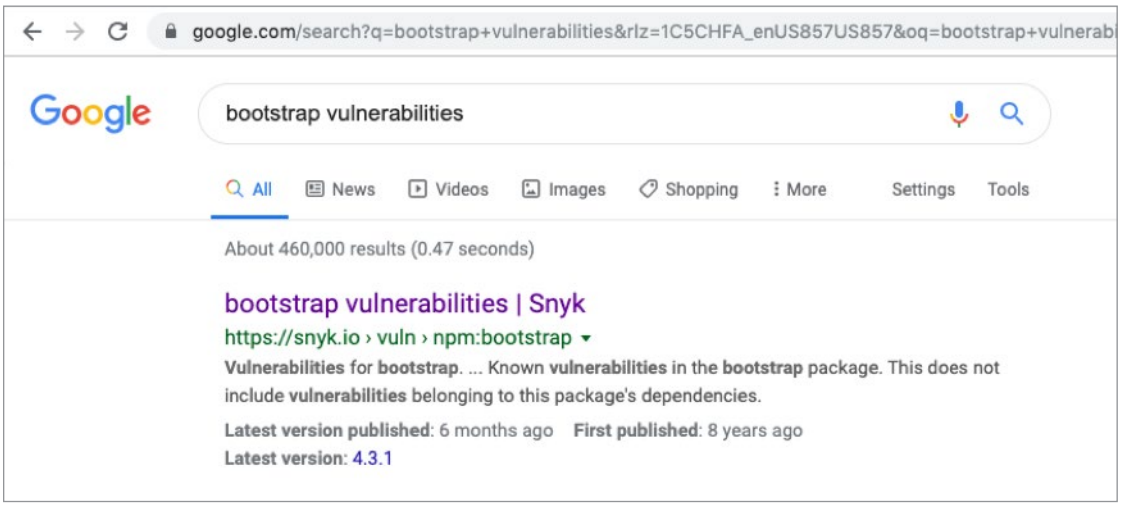


We see that this host has potentially 6 CVEs associated with Bootstrap version 3.3.1.
CVE-2016-10735
CVE-2019-8331
CVE-2018-20676
CVE-2018-14040
CVE-2018-20677
CVE-2018-14042

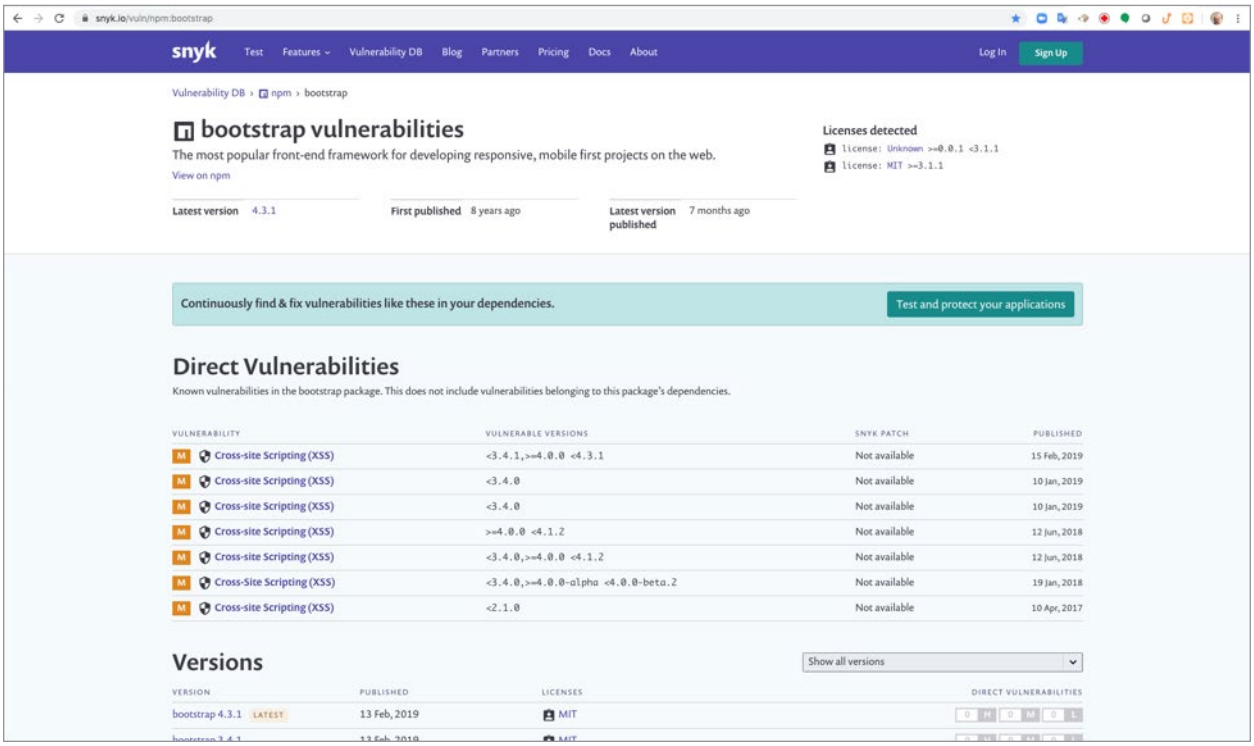
Step 21: Google search for bootstrap vulnerabilities

Now let's find out more information at bootstrap vulnerabilities by doing a google search.

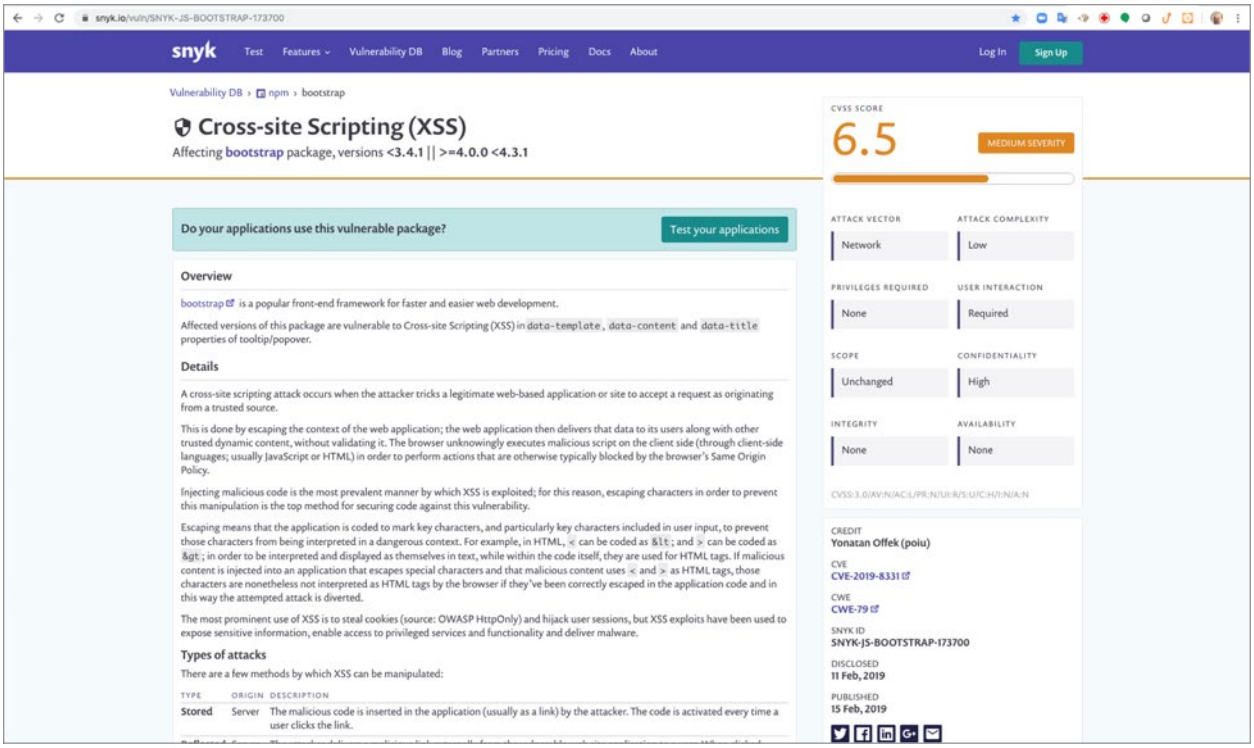
In a new tab, search https://google.com and search for bootstrap vulnerabilities.

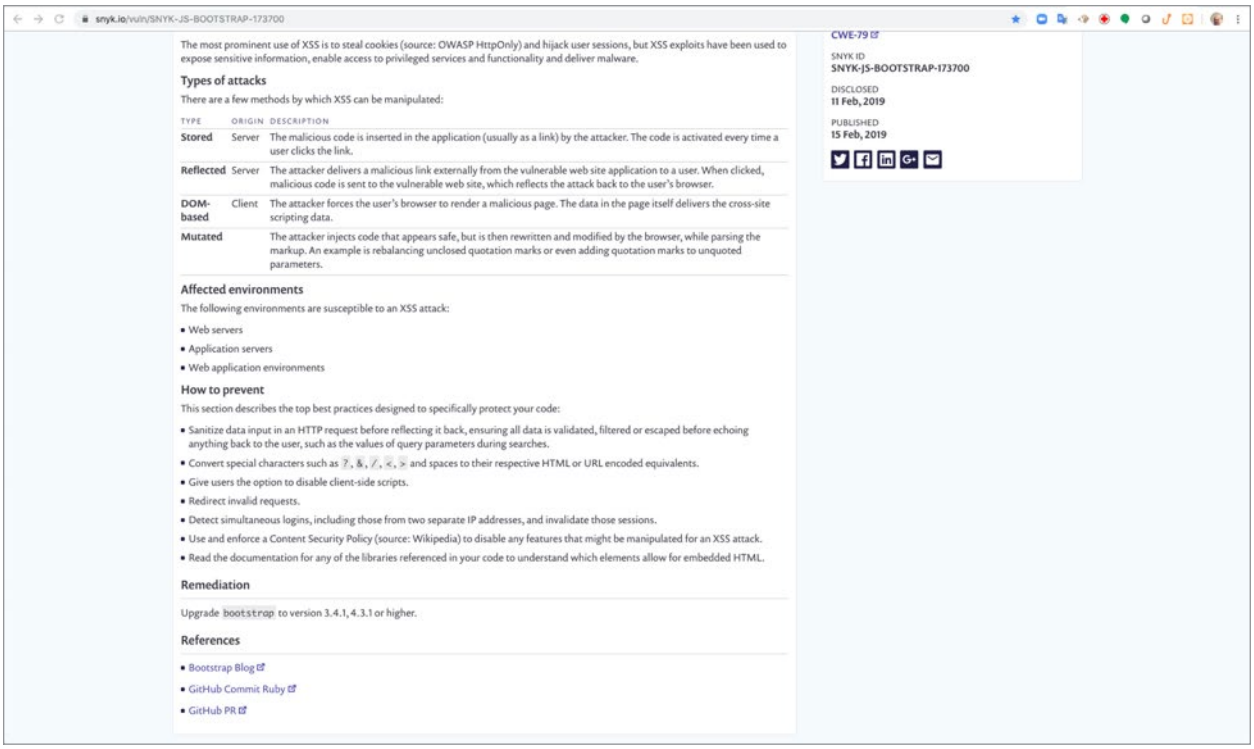


Click on the link https://snyk.io/vuln/npm:bootstrap



www[.]flowerexplosion[.]com is running version 3.3.1 click on the first link.





The most prominent use of XSS is to steal cookies (source: OWASP HttpOnly) and hijack user sessions, but XSS exploits have been used to expose sensitive information, enable access to privileged services and functionality and deliver malware.

Types of attacks

There are a few methods by which XSS can be manipulated:

TYPE	ORIGIN	DESCRIPTION
Stored	Server	The malicious code is inserted in the application (usually as a link) by the attacker. The code is activated every time a user clicks the link.
Reflected	Server	The attacker delivers a malicious link externally from the vulnerable web site application to a user. When clicked, malicious code is sent to the vulnerable web site, which reflects the attack back to the user's browser.
DOM-based	Client	The attacker forces the user's browser to render a malicious page. The data in the page itself delivers the cross-site scripting data.
Mutated		The attacker injects code that appears safe, but is then rewritten and modified by the browser, while parsing the markup. An example is rebalancing unclosed quotation marks or even adding quotation marks to unquoted parameters.

Affected environments

The following environments are susceptible to an XSS attack:

- Web servers
- Application servers
- Web application environments

How to prevent

This section describes the top best practices designed to specifically protect your code:

- Sanitize data input in an HTTP request before reflecting it back, ensuring all data is validated, filtered or escaped before echoing anything back to the user, such as the values of query parameters during searches.
- Convert special characters such as `?`, `&`, `/`, `<`, `>` and spaces to their respective HTML or URL encoded equivalents.
- Give users the option to disable client-side scripts.
- Redirect invalid requests.
- Detect simultaneous logins, including those from two separate IP addresses, and invalidate those sessions.
- Use and enforce a Content Security Policy (source: Wikipedia) to disable any features that might be manipulated for an XSS attack.
- Read the documentation for any of the libraries referenced in your code to understand which elements allow for embedded HTML.

Remediation

Upgrade `bootstrap` to version 3.4.1, 4.3.1 or higher.

References

- Bootstrap Blog
- GitHub Commit Ruby
- GitHub PR

CWE-79

SNYK ID
SNYK-JS-BOOTSTRAP-173700

DISCLOSED
11 Feb, 2019

PUBLISHED
15 Feb, 2019

Types of attacks

There are a few methods by which XSS can be manipulated:

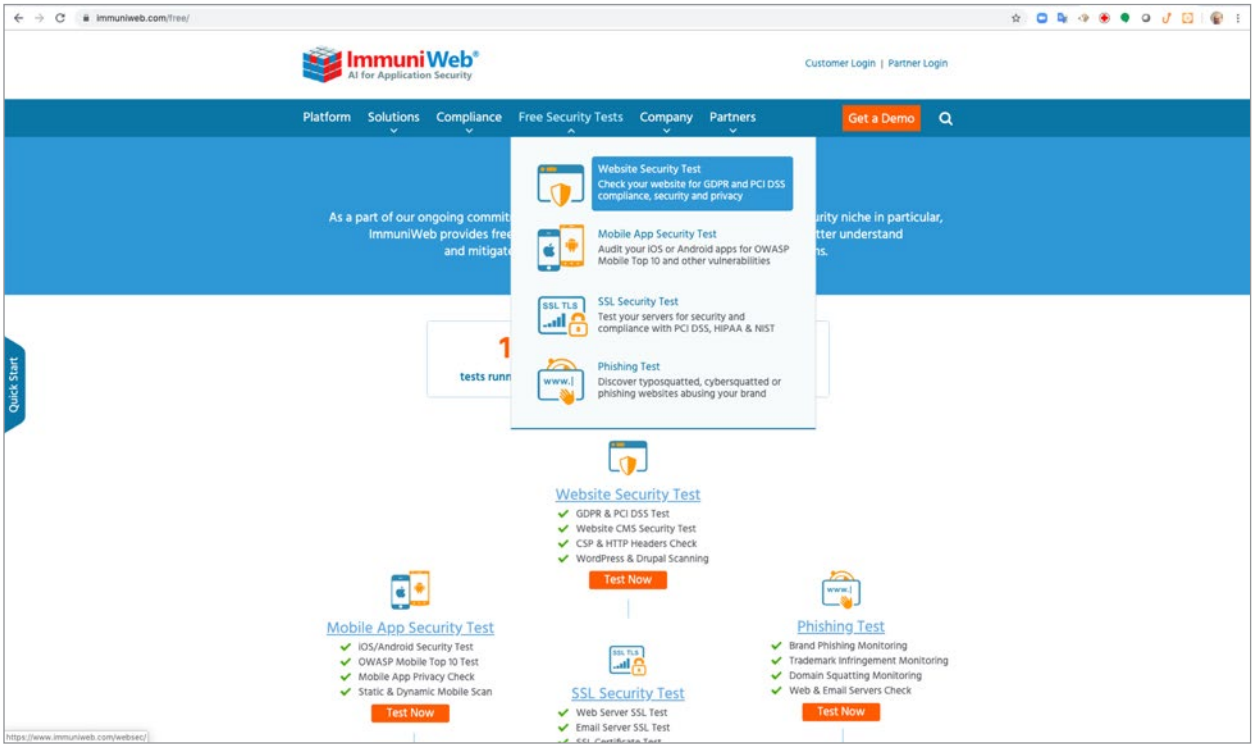
TYPE	ORIGIN	DESCRIPTION
Stored	Server	The malicious code is inserted in the application (usually as a link) by the attacker. The code is activated every time a user clicks the link.
Reflected	Server	The attacker delivers a malicious link externally from the vulnerable web site application to a user. When clicked, malicious code is sent to the vulnerable web site, which reflects the attack back to the user's browser.
DOM-based	Client	The attacker forces the user's browser to render a malicious page. The data in the page itself delivers the cross-site scripting data.
Mutated		The attacker injects code that appears safe, but is then rewritten and modified by the browser, while parsing the markup. An example is rebalancing unclosed quotation marks or even adding quotation marks to unquoted parameters.

Bootstrap 3.3.1 is vulnerable to cross-site scripting attacks and a threat actor might be able to insert code in your website. This is the most likely the vulnerability that was exploited for the threat actors to install Magecart. Now you need to upgrade your bootstrap to version 3.4.1 or 4.3.1 to prevent future Cross-site scripting attacks.

Step 22: Independent vulnerability assessment.

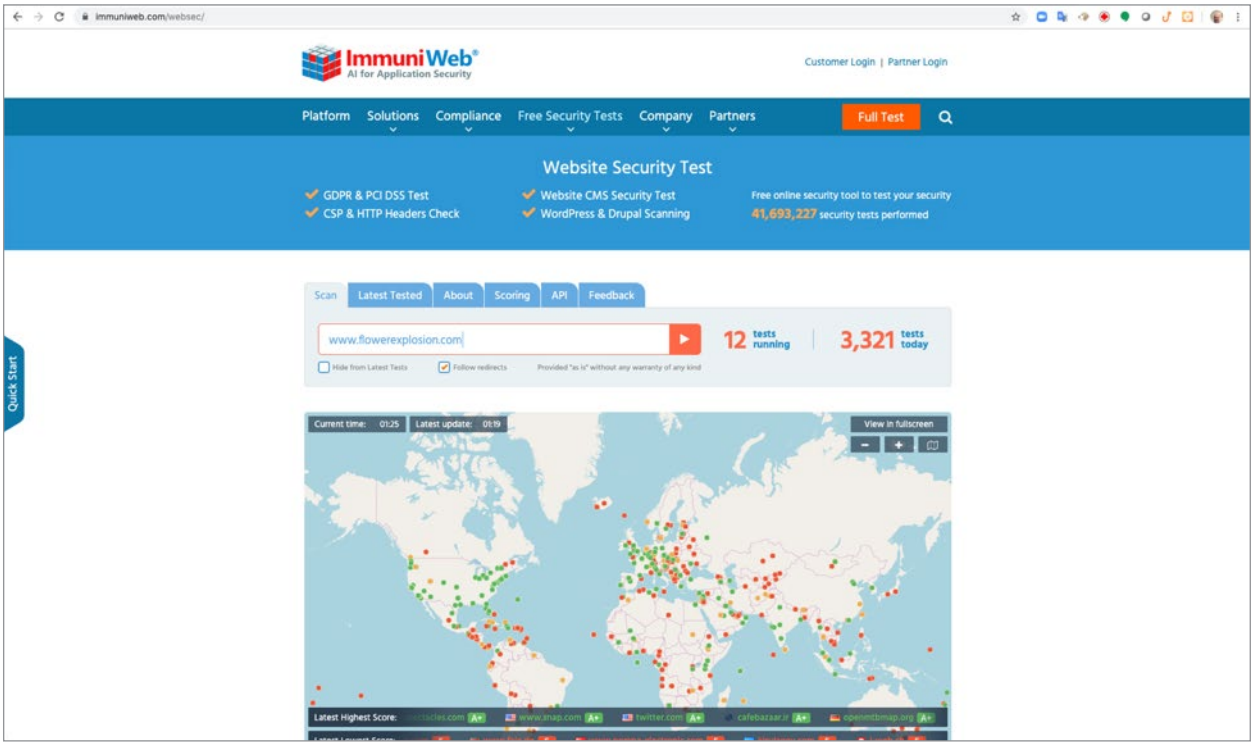
We are now going to perform a vulnerability scan utilizing a third-party to verify the vulnerability on your website.

In a new tab go to https://immuniweb.com

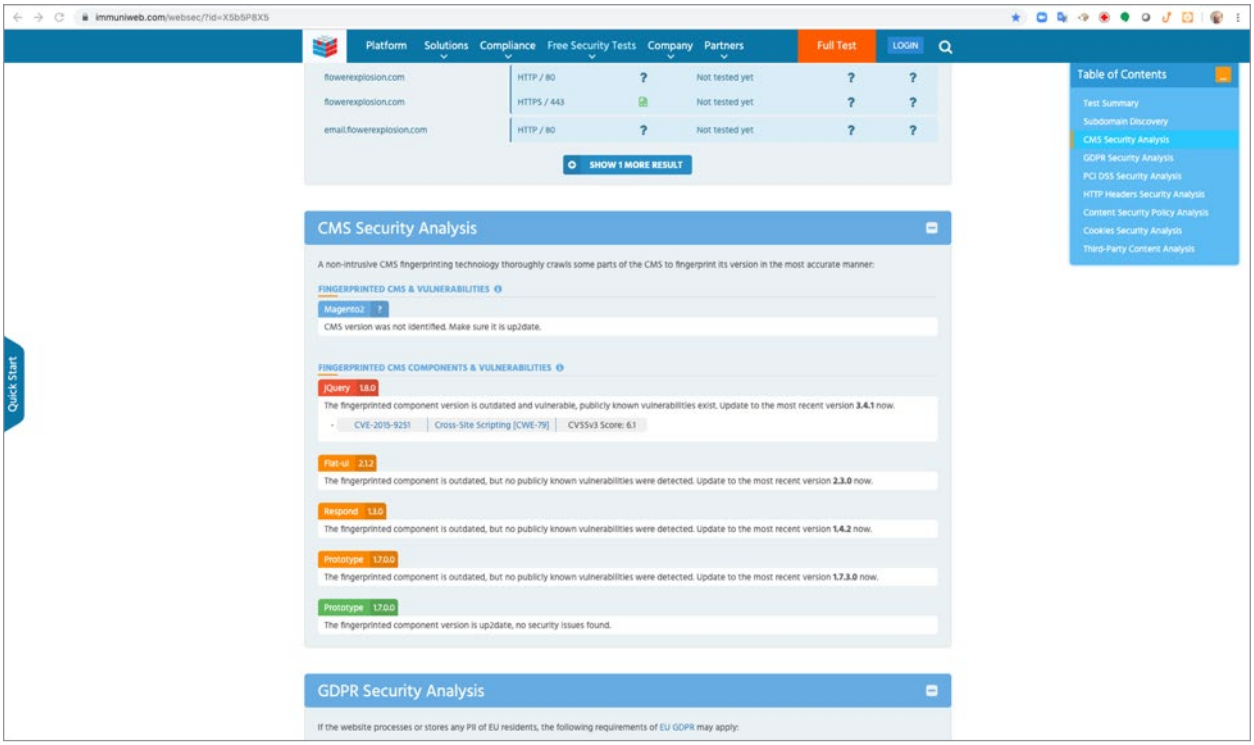


Click on the Free Security Test and click on Website Security Test
https://www.immuniweb.com/websec/

Scan the domain www[.]flowerexplosion[.]com.



Scan results



The results show a cross-site scripting vulnerability like we suspected and a Magento eCommerce Platform vulnerability.

Note:

The vulnerability scan represents how the website appeared at the time of the scan. If the organization has updated the server or new vulnerabilities have been announced your report may be different than what is reflected in this document.

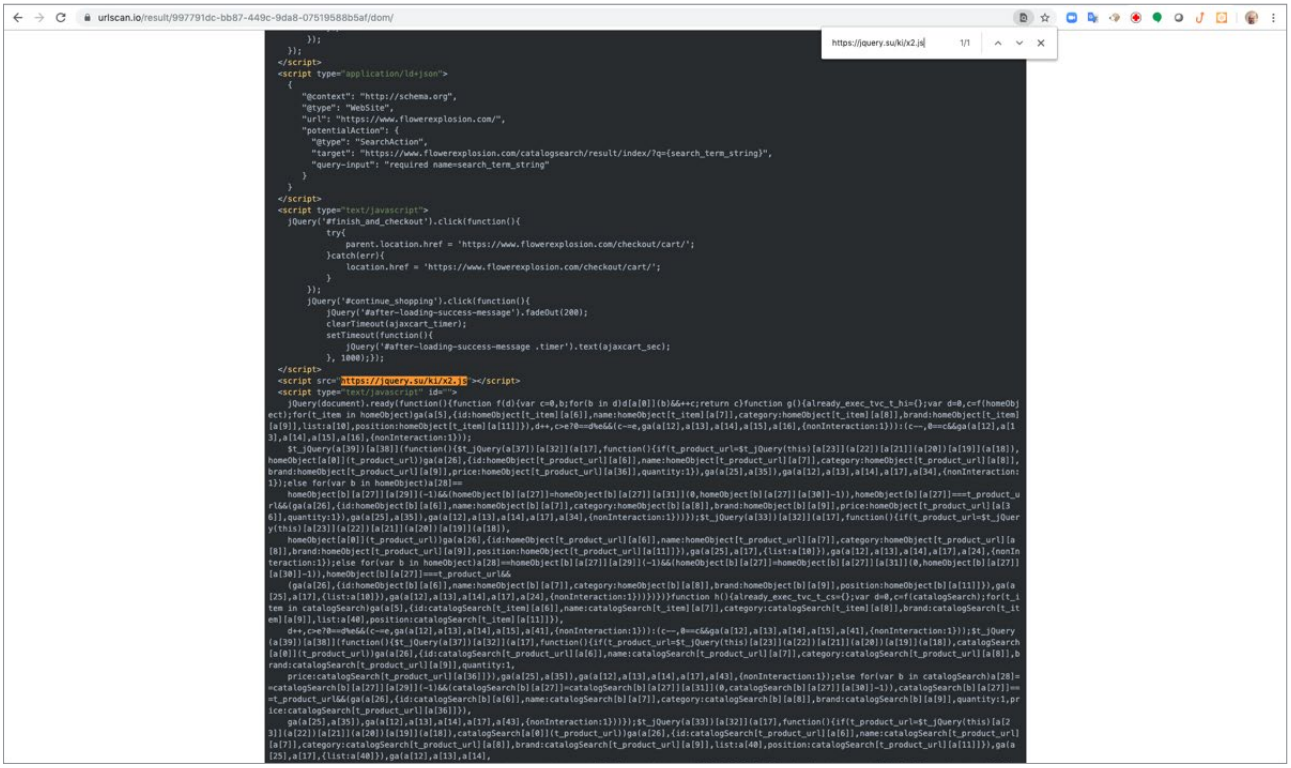
Conclusion:

While investigating it is best to utilize tools that safeguard your systems from possible compromised and limiting the threat actor from finding out you are investigating them.

PassiveTotal was able to show that a script was modified in late May of 2019. By examining the DOM from www[.]flowerexplosion[.]com (using https://urlscan.io) we were able to determine exactly what script the web site was calling from jquery[.]su. The file that was being called was /ki/x2[.]js. In the file we noticed the script was looking for patterns that appeared to be payment card information.

```
A20_99f77070702c32e5a8eaba3c38801bec:(function(name){

var matches=document.cookie.match(new RegExp('(?:^|; )'+name.replace(/
([\.$?*|{}\(\)\[\]\\\/\+^])/g,'\\$1')+'=([^;]*)'));

return matches?decodeURIComponent(matches[1]):undefined;

})('A21_c9571bae638b1e00f6ce64325c44c285')||(function(){

var ms=new Date();

var myid = Math.floor(Math.random()*(999999999-11111111+1)+11111111)+ms.
getTime()+"-"+Math.floor(Math.random()*(999999999-11111111+1)+11111111);

var date=new Date(new Date().getTime()+60*60*24*1000);

document.cookie='A21_c9571bae638b1e00f6ce64325c44c285='+myid+'; path=/;
expires='+date.toUTCString();

return myid;
```

Now we determined the malicious scripts to remove from your website.



But to prevent future compromises we needed to determine the vulnerabilities that exist on the web site so it can be patched or upgraded to prevent malicious code from being inserted back on the website.

By utilizing RiskIQ PassiveTotal's Digital Footprint we determine that the website had an old version of bootstrap 3.3.1 and it needs to be upgraded to either version 3.4.1 or 4.3.1.

1. Were payment cards being stolen from www[.]flowerexplosion.com?
   Yes, we have confirmed that scripts that were associated with Magecart were present on the server and financial institutions source fraudulent payment card transactions after users made purchases on www[.] flowerexplosion[.]com
2. How were the payment cards being stolen?
   Payment card information was stolen via a JavaScrip skimmer from jquery[.]com located in Ukraine and Russia.
3. How do you suspect the website was compromised?
   The website was compromised either by a vulnerability in Bootstrap 3.3.1 or Magento 0.74.
4. How can you prevent similar attacks in the future from occurring on www[.]flowerexplosion.
   The website's bootstrap 3.3.1 and Magento 0.74 needs to upgrade to a non-vulnerable version.

Investigations can be done even if the servers have been erased and no backups are available. The use of security tools that gather information all the time and have a rich history are vital to threat hunters and security researchers.

We hope you enjoyed this use case and share it with your friends and colleagues.

RISKIQ®

RiskIQ, Inc.
22 Battery Street, 10th Floor
San Francisco, CA. 94111

sales@riskiq.net
1 888.415.4447

Learn more at riskiq.com