Identifying Qakbot Servers with Regex and TLS Certificates

By Matthew

Published: 2023-11-30 · Archived: 2026-04-05 14:01:53 UTC

In this post we will leverage regular expressions and TLS certificates to capture 83 dispersed Qakbot servers.

These servers are well made and there are minimal traditional patterns (ports, service names, ASN's) that can be used for signaturing. Instead we will focus on commonalities within the `subject_dn` and `issuer_dn` fields to identify servers.

> This is a relatively advanced technique that will require a basic understanding of regular expressions, and also a paid/researcher license for Censys.

The final query is shown below. A link can be found here.

```
services.tls.certificates.leaf_data.subject_dn=/C=\w\w, OU=[a-zA-Z0-9 \.]+, CN=[a-z]+\.[a-z]+/ and services.tls
```

> Note that this post is primarily a demonstration of technical concepts that can be used to identify malware. We have not 100% validated that all results are Qakbot (although most appear to be), and we are relying on the initial ThreatFox tag being accurate.

Initial Server From ThreatFox

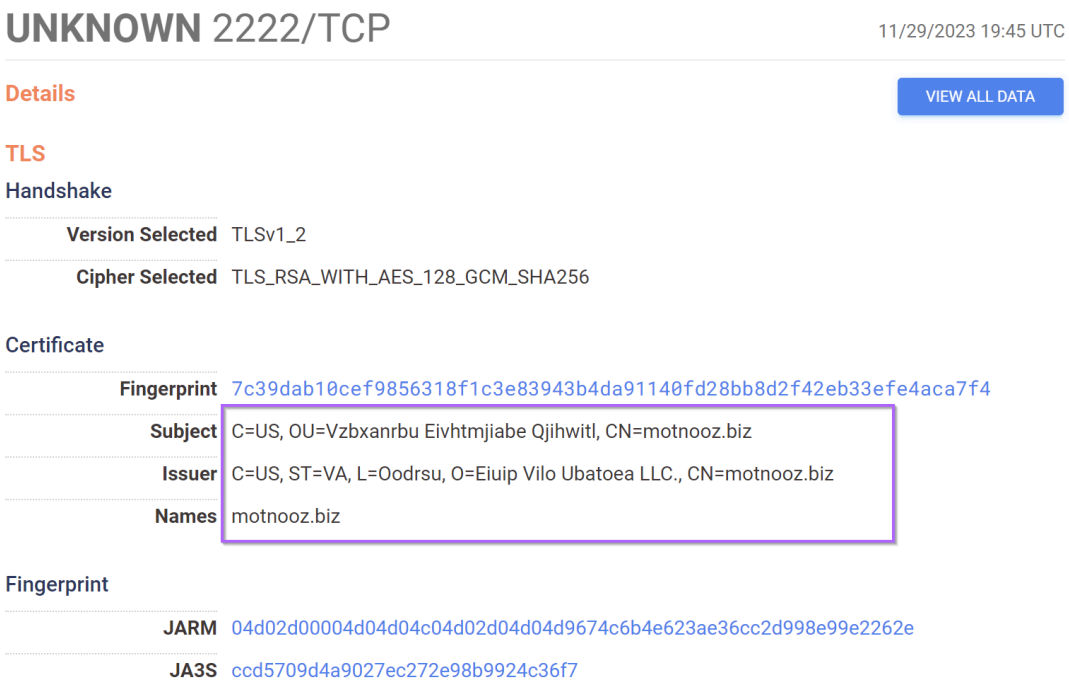
The initial server IP of `74.12.147[.]243:2222` was obtained from ThreatFox. Initially shared by the Twitter user @drb_ra.

Database Entry

IOC ID:	1207439
IOC:	74.12.147.243:2222
IOC Type:	ip:port
Threat Type:	botnet_cc
Malware:	QakBot
Malware alias:	Oakboat, Pinkslipbot, Qbot, Quakbot
Confidence Level:	Confidence level is moderate (50%)
First seen:	2023-11-30 06:54:15 UTC

By searching the IP address on Censys, we can quickly identify a suspicious certificate running on the reported `2222` port.

This certificate contains seemingly random text. With long values and only alphabetical characters.



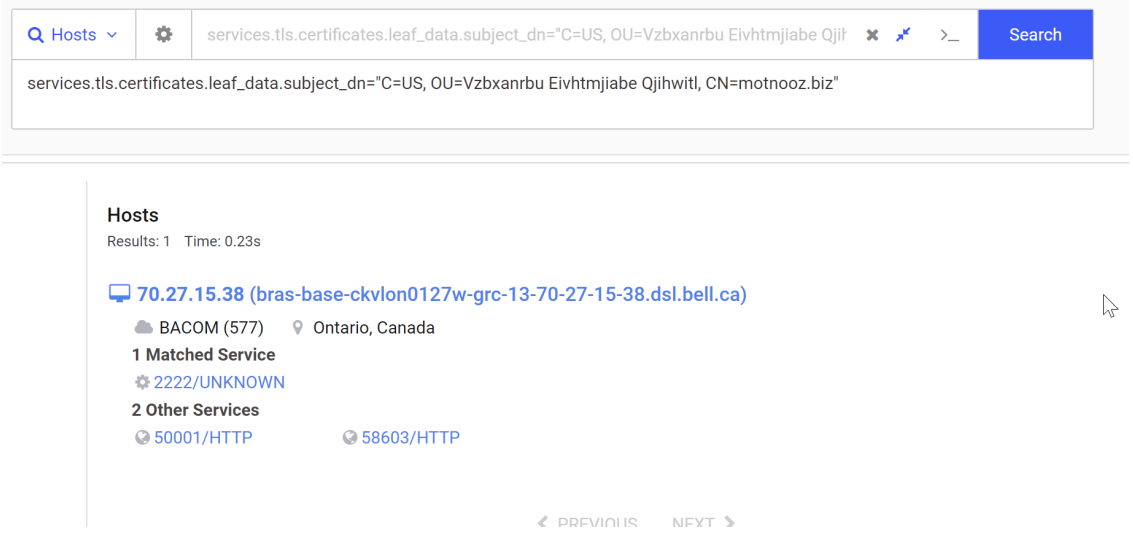
Using "View All Data", we can gather more information about the service running on `2222`.

This reveals an empty service banner that can be later used as a pivot point or as a field to narrow down search results.

The exact structure of the TLS Certificate can be established with this view.

The search box next to `services.tls.certificates.leaf_data.subject_dn` can be used to pre-build an exact query.

We will use this pre-built query as a base for our regular expression.



How To Convert Values Into Regular Expressions

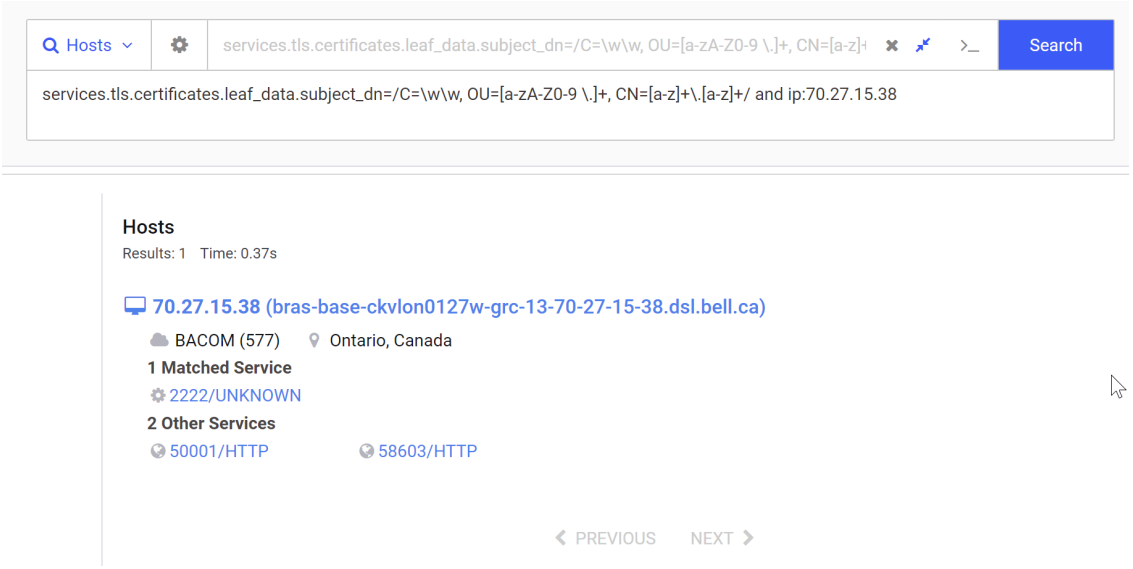
We can go ahead and modify the search parameter to a regular expression.

A summary of the changes can be found below.

- `C=US` -> `C=\w\w` - We will let the `C` field matches on any two characters
- `OU=Vzbxanrbu Eivhtmjiabe Qjihwitl` -> `OU=[a-zA-Z0-9 ]+` - We can let the `OU` field matches any sequence of alphabetical characters, allowing for a space in between.
- `CN=motnooz.biz` -> `CN=[a-z]+\.[a-z]+` - we will let the `CN` field match on any domain containing only lowercase letters.

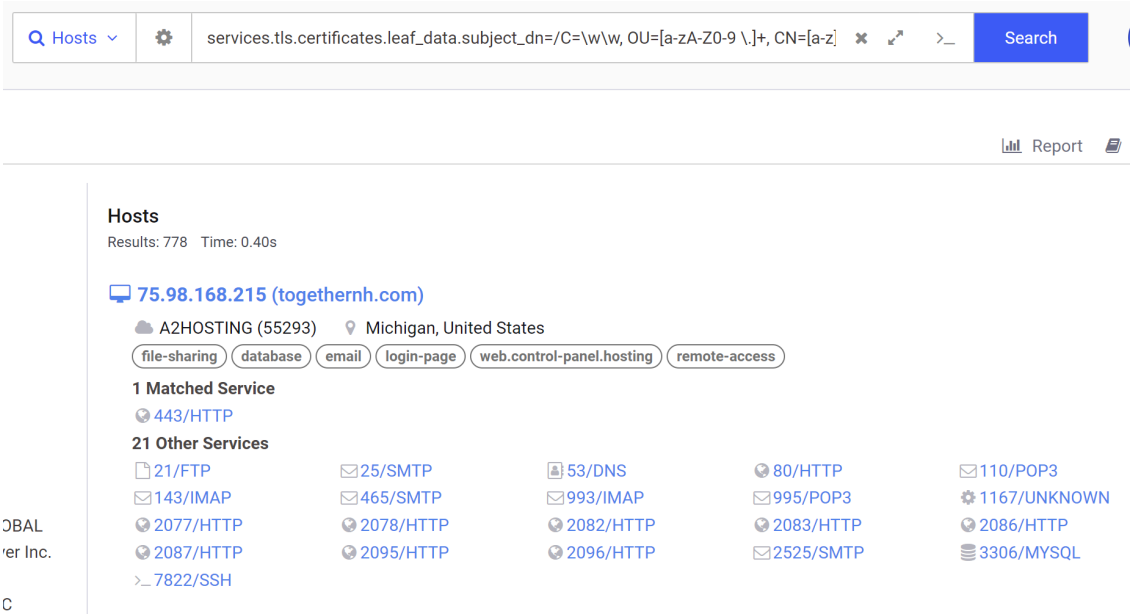
After modifying the query as above, we can also add a filter for our original IP. This ensures that the same IP is matched and hasn't been lost. This is a means of quickly verifying that a regex works as intended.

We can see below that the same Initial IP is matched, meaning that the regex probably works.



With the Regex validated, We can now go ahead and remove the IP Address, leaving only the `subject_dn` field.

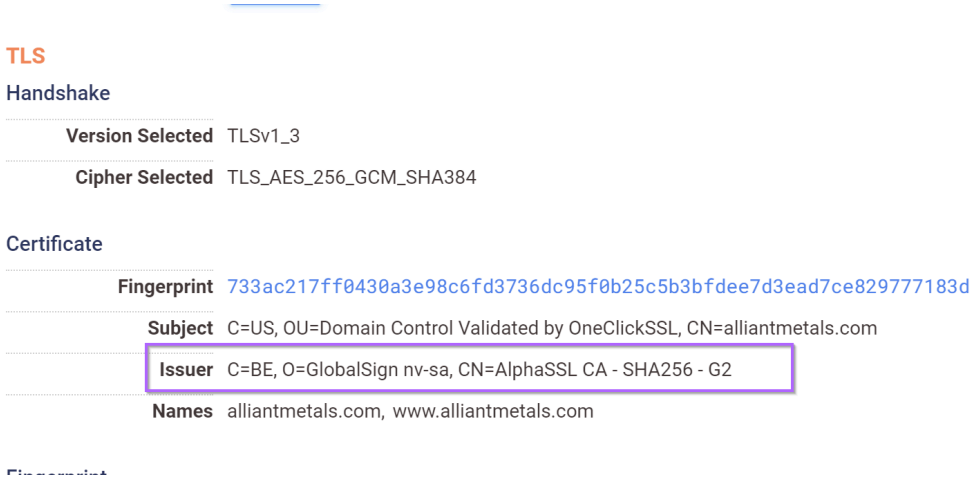
This modified search results in 778 servers, many of which don't completely follow the certificate structure we want.



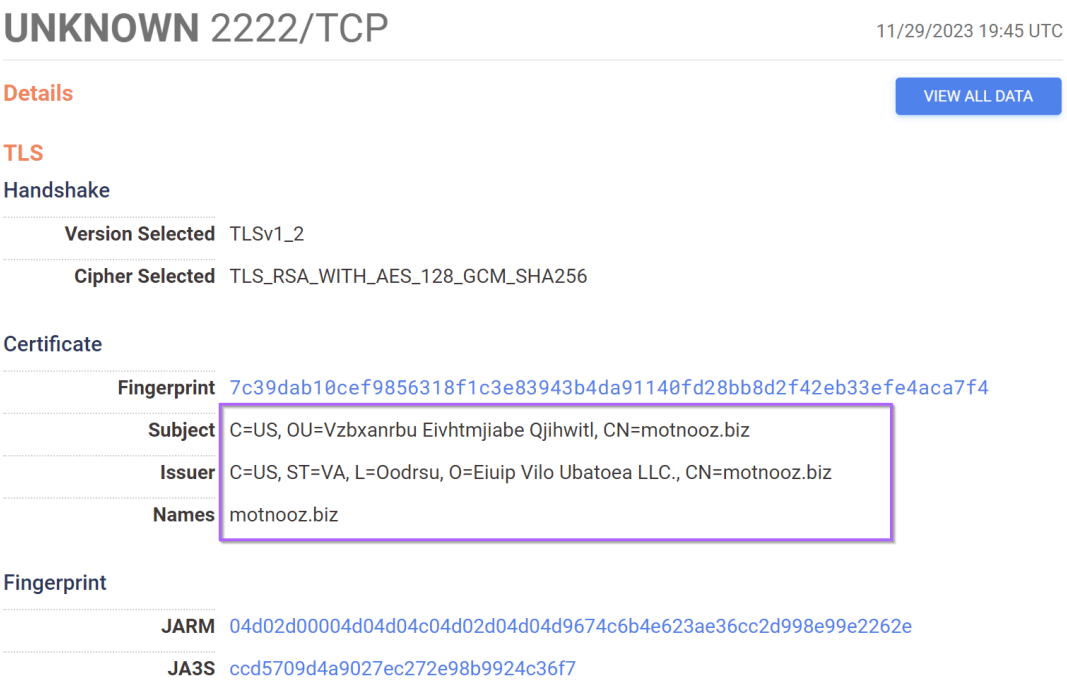
Validating Search Results

If we inspect the first returned result of `75.98.168[.]215`, we can see that the `subject_dn` matches our regular expression structure, but the `issuer_dn` is different to our initial Qakbot.

Below is the first returned result (Which does not match our pattern). Note that it contains the `-` character in the `CN` and `O` fields.



Below is the original Qakbot C2. Note the lack of special characters and numerical values.

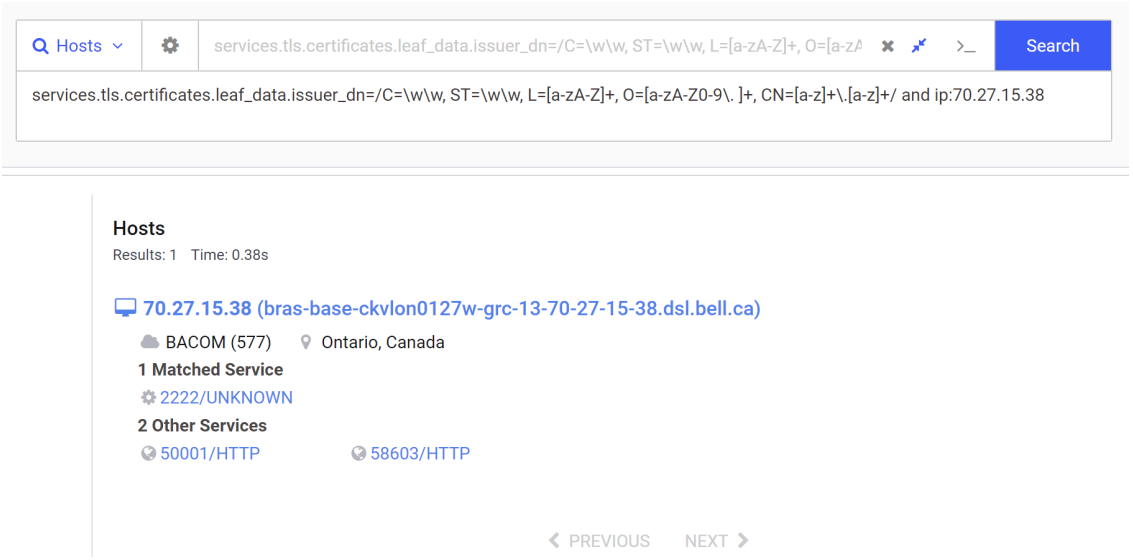


The initial search returns results that match our `subject_dn` regular expression.

But there are results with a completely different (and not matching) structure on the `issuer_dn`.

We can go back to our initial Qakbot C2 and follow the same process as before to build a regular expression on the `issuer_dn` field.

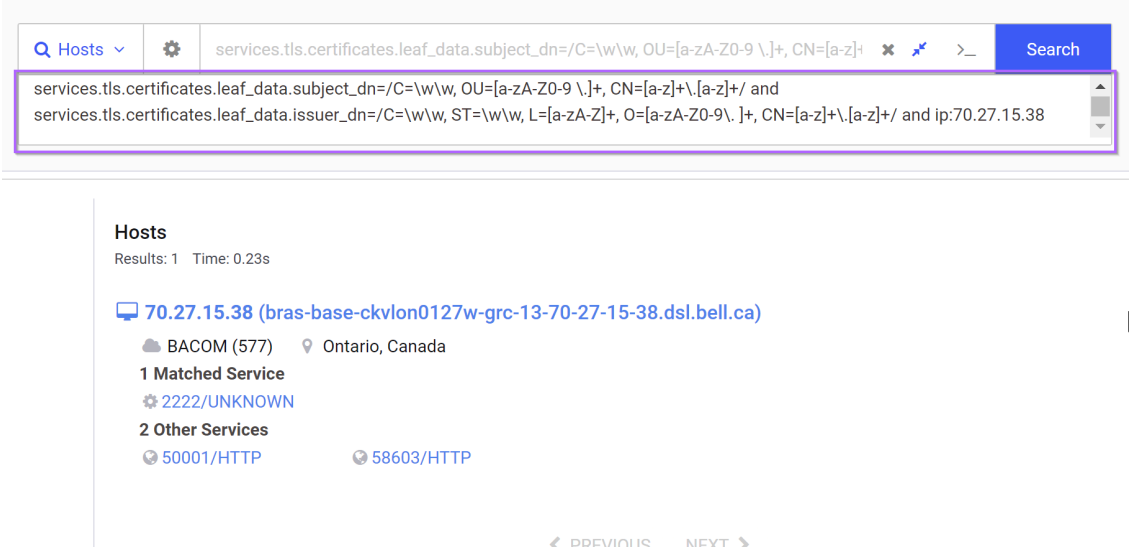
We can then validate the regular expression by including the initial IP address.



Since the `issuer_dn` field has not been validated; We can now go ahead and add the `issuer_dn` query to the initial `subject_dn` search. We can also include the initial IP for validation.

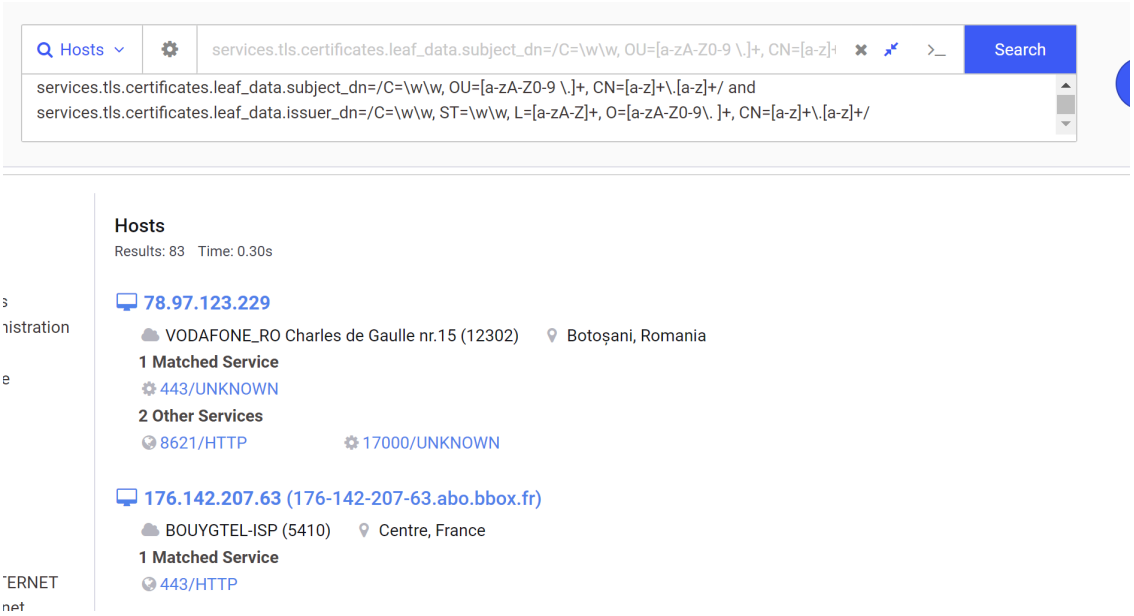
At this point, we have a total query of

```
services.tls.certificates.leaf_data.subject_dn=/C=\w\w, OU=[a-zA-Z0-9 \.]+, CN=[a-z]+\.[a-z]+/ and services.tls
```

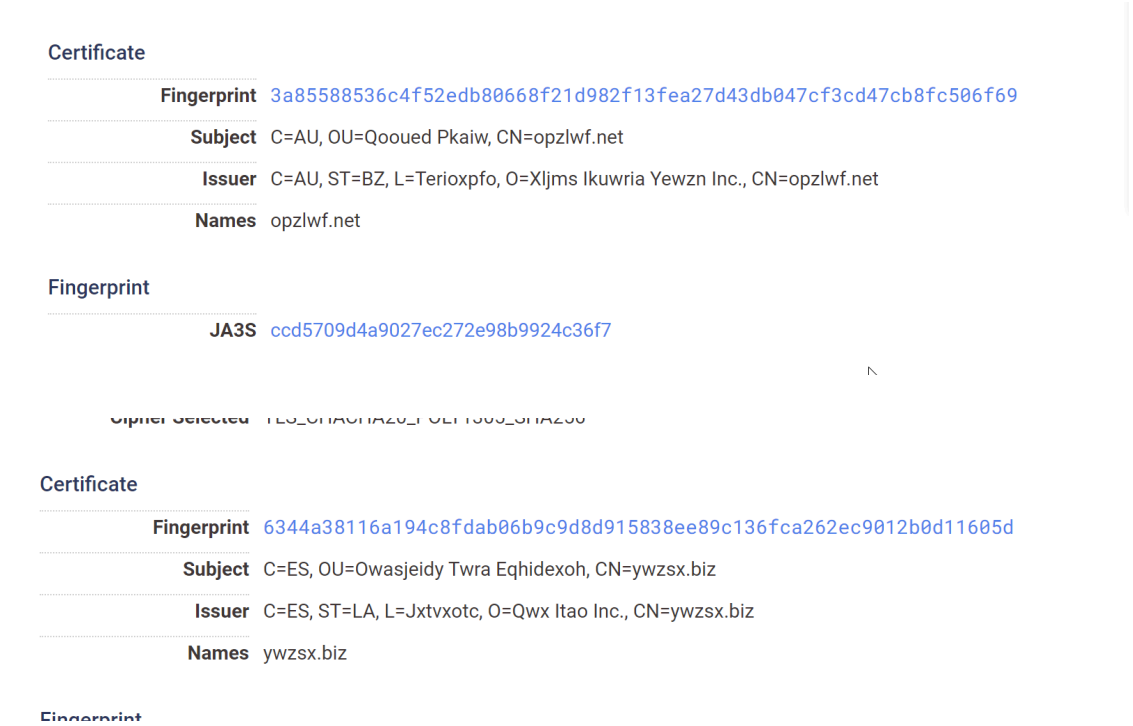


The above search confirms that we haven't lost our initial hit, meaning the regex is valid, and the initial IP can be removed.

By removing the Initial IP Address and including only the `subject_dn` and `issuer_dn`, we're now down to a manageable number of 83 results.

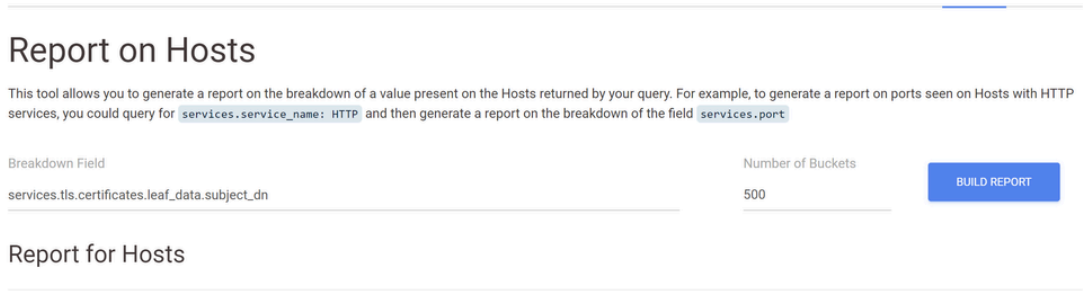


Inspecting the first two hits, we can confirm that we have matches on our intended certificate structure.

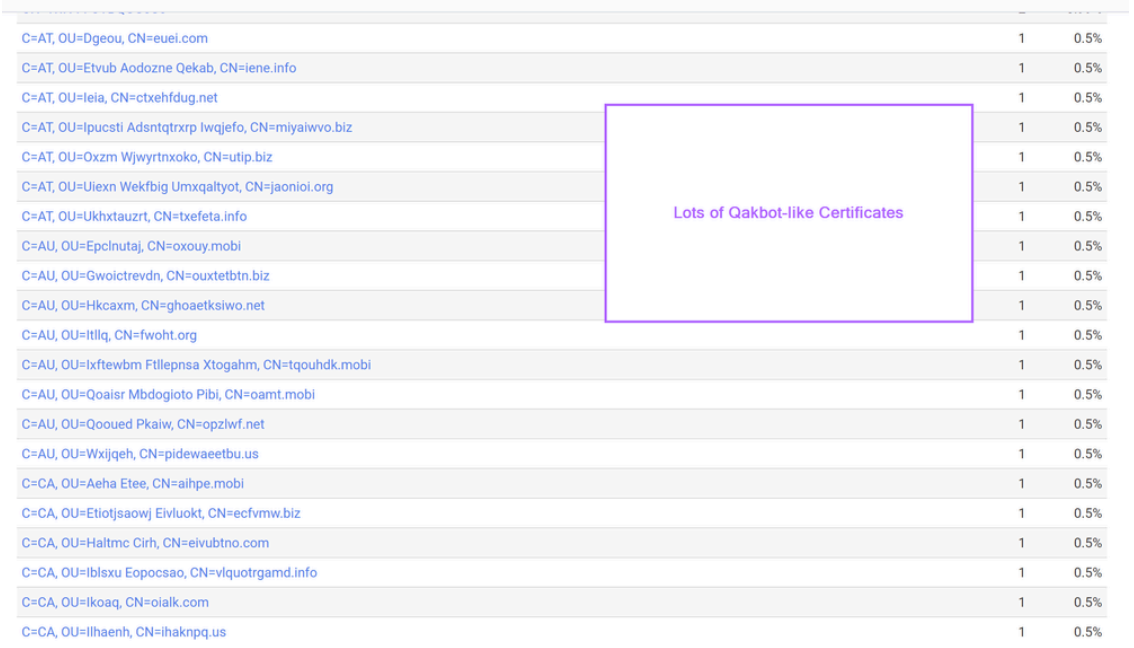


Further Validation With Report Building

To save time validating every result individually, we use the "build report" function of Censys to hone in on the `subject_dn` or `issuer_dn` fields.

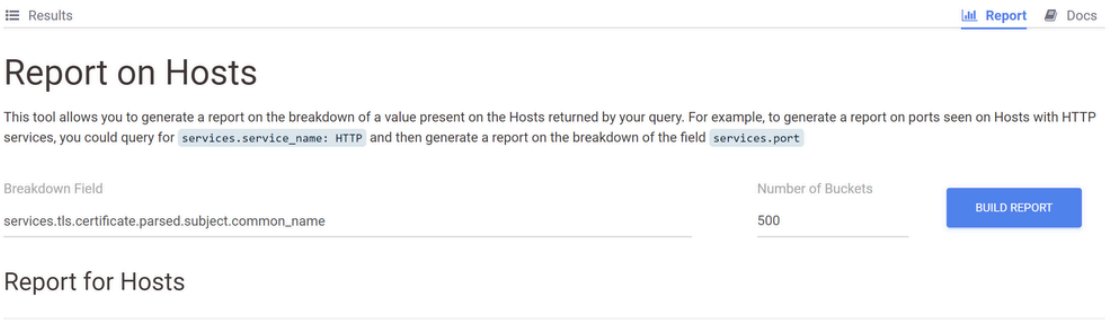


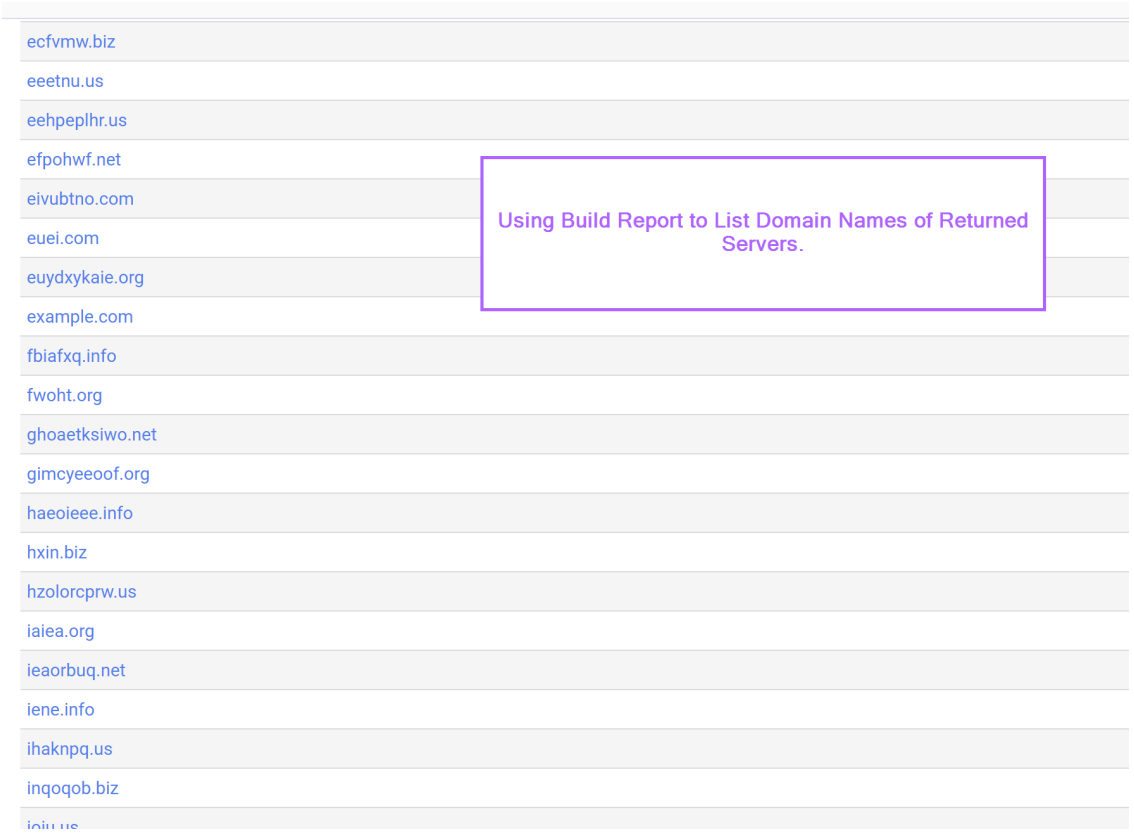
This confirms that most of the returned servers match our intended structure.



Honing in on Domain/Host Names

We can also use the "build report" function to hone in on `common_name` fields used in the TLS certificates.

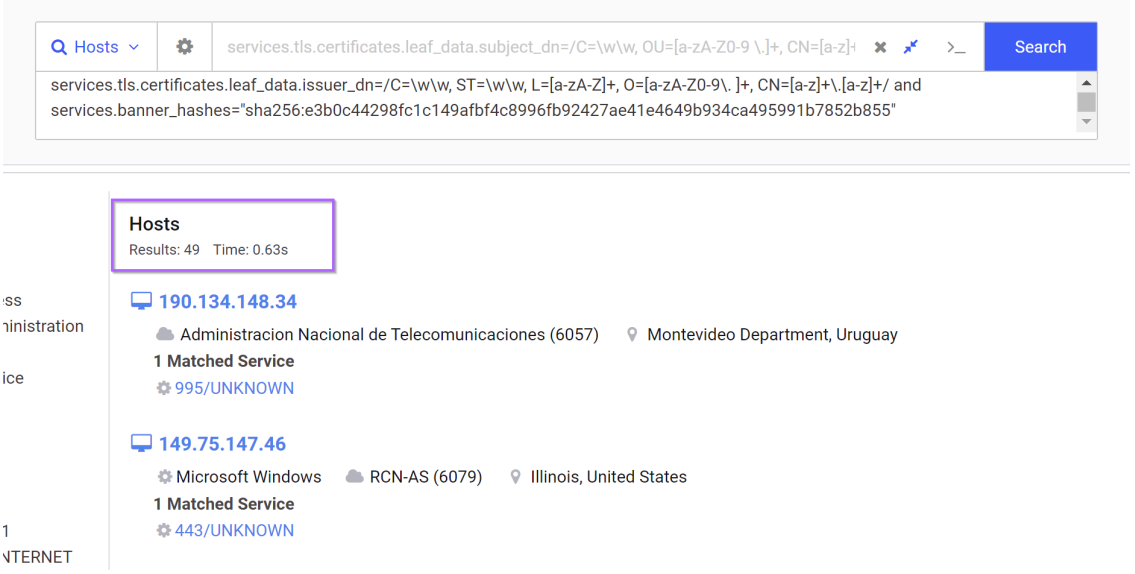




Query Refinement

There are potentially some false positives within the 83 returned results, so if we like, we can go ahead and add the empty banner hash from the initial IP.

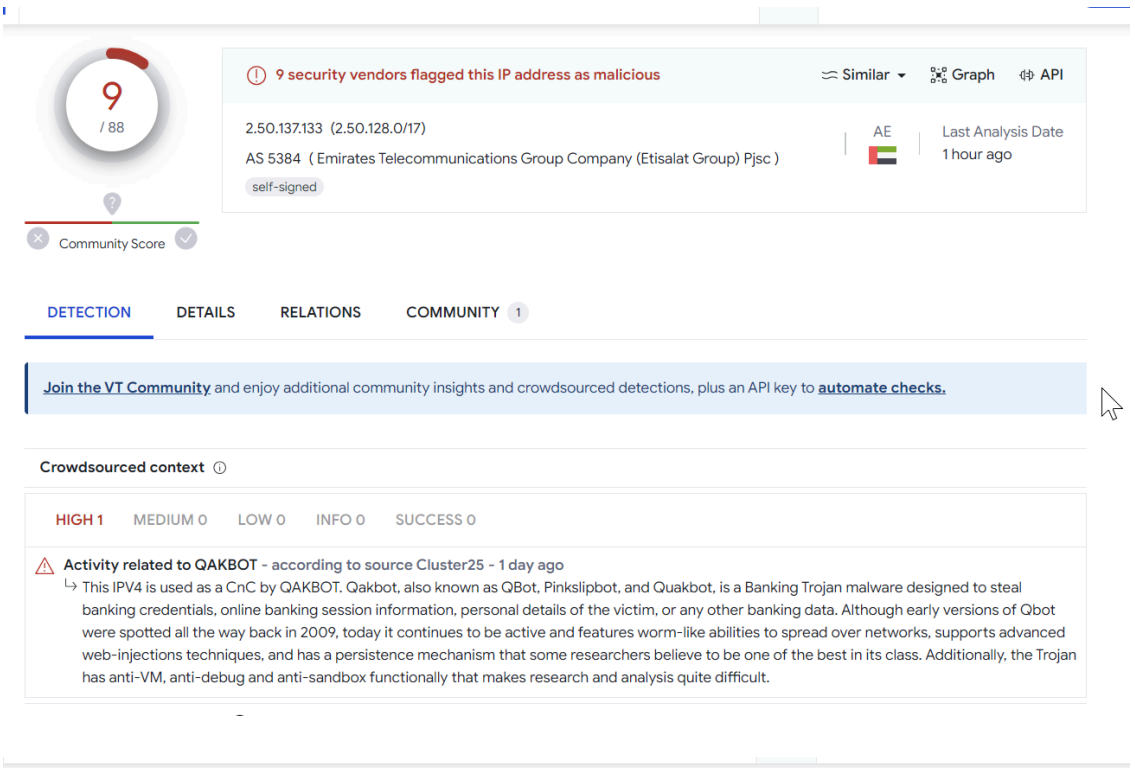
This will reduce the hits down to 49. But it's possible that this may remove some malicious results. I did not validate this as it's very time-consuming, and the majority of servers seem to be malicious either way.

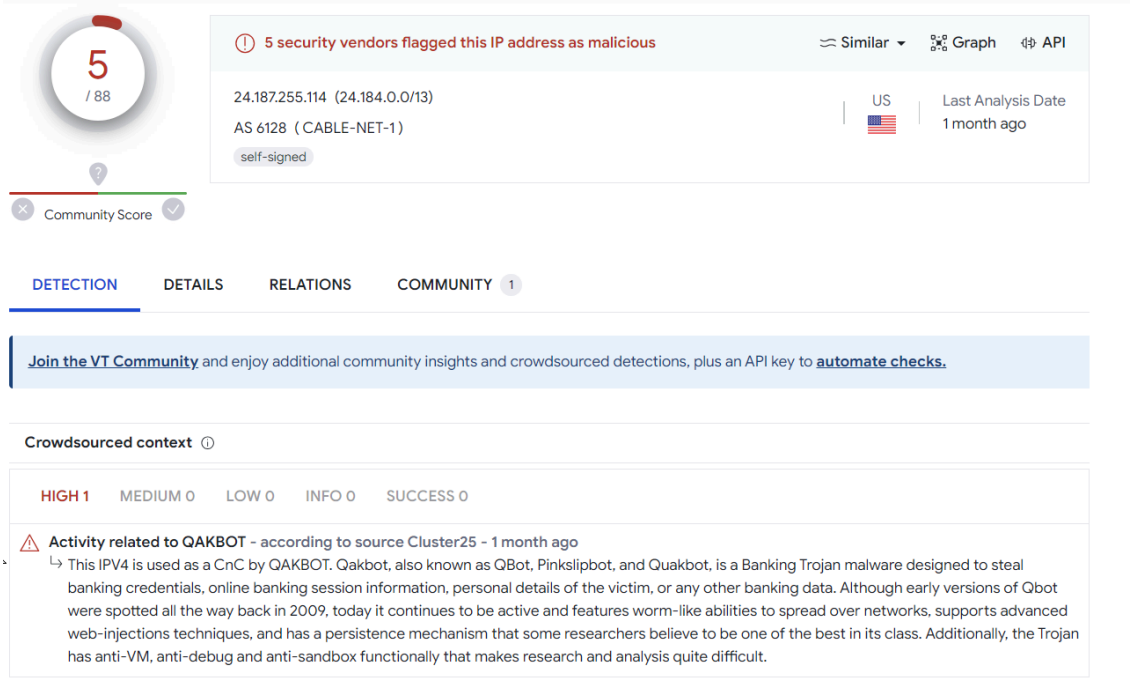


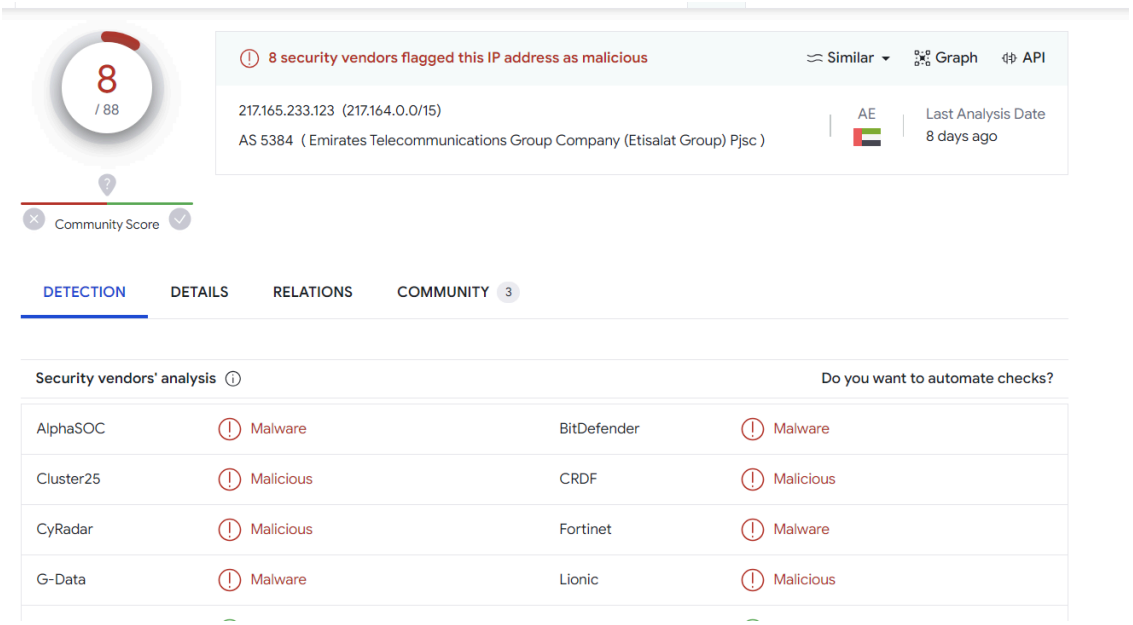
Validating Results With Virustotal

Performing a quick search on some of the returned hits on Virustotal.

Most seem related to Qakbot, although we have not confirmed this 100%.







Conclusion

We now have a [functioning query] that captures 83 servers. We have not had to rely on port numbers, port ranges, or ASN locations to hone in.

Here is another copy of our final query.

```
services.tls.certificates.leaf_data.subject_dn=/C=\w\w, OU=[a-zA-Z0-9 \.]+, CN=[a-z]+\.[a-z]+/ and services.tls
```

If we observe the returned results below, we can see that the ASNs and port numbers vary greatly between the results, meaning that many traditional query styles will not work.

Related Content

If you found this content useful, check out other related posts in the free Threat Intelligence Section.

- Advanced Regex Queries - BianLian
- Practical Queries for Identifying Malware - Part 3
- Combining Simple Pivot Points to Identify Infrastructure

Sign up for Embee Research

Malware Analysis and Threat Intelligence Research

No spam. Unsubscribe anytime.

Complete List of Qakbot Infrastructure

IP Addresses

2[.]50[.]137[.]133
23[.]93[.]65[.]180
24[.]187[.]255[.]114
24[.]187[.]255[.]116
24[.]187[.]255[.]117
24[.]255[.]174[.]187
31[.]117[.]63[.]201
35[.]134[.]202[.]121

37[.]210[.]162[.]30
39[.]40[.]144[.]179
41[.]38[.]97[.]237
41[.]99[.]46[.]66
45[.]65[.]51[.]130
46[.]251[.]130[.]164
47[.]16[.]64[.]215
47[.]149[.]234[.]6
50[.]99[.]8[.]5
60[.]48[.]77[.]48
64[.]46[.]22[.]26
64[.]229[.]117[.]137
67[.]60[.]147[.]240
68[.]160[.]236[.]23
68[.]163[.]65[.]72
70[.]27[.]15[.]38
70[.]29[.]135[.]118
70[.]49[.]34[.]218
70[.]52[.]230[.]48
70[.]121[.]156[.]34
72[.]190[.]100[.]201
74[.]12[.]145[.]202
74[.]12[.]145[.]207
74[.]12[.]147[.]243
76[.]142[.]13[.]8
77[.]124[.]85[.]166
78[.]97[.]123[.]229
79[.]130[.]51[.]242
80[.]192[.]52[.]128
81[.]151[.]251[.]196
82[.]76[.]99[.]171
83[.]110[.]196[.]111
83[.]110[.]223[.]89
84[.]155[.]8[.]44
84[.]215[.]202[.]8
85[.]49[.]243[.]230
85[.]243[.]247[.]137
86[.]97[.]84[.]192
86[.]207[.]26[.]60
86[.]236[.]11[.]235
87[.]223[.]92[.]180
88[.]249[.]231[.]161

90[.]4[.]74[.]222
95[.]76[.]193[.]223
95[.]149[.]166[.]38
96[.]43[.]115[.]158
96[.]248[.]1[.]183
97[.]118[.]24[.]246
100[.]2[.]41[.]26
102[.]157[.]101[.]136
102[.]157[.]244[.]251
104[.]157[.]102[.]161
108[.]4[.]77[.]65
108[.]49[.]159[.]2
109[.]48[.]28[.]129
121[.]121[.]101[.]31
124[.]13[.]232[.]162
125[.]209[.]114[.]181
136[.]232[.]179[.]26
141[.]164[.]249[.]90
149[.]75[.]147[.]46
151[.]48[.]137[.]184
161[.]142[.]99[.]88
168[.]149[.]47[.]164
172[.]77[.]204[.]25
172[.]91[.]3[.]194
173[.]30[.]189[.]100
174[.]164[.]68[.]180
179[.]158[.]101[.]198
186[.]182[.]15[.]91
187[.]147[.]137[.]67
188[.]48[.]72[.]229
189[.]253[.]235[.]140
190[.]134[.]148[.]34
197[.]2[.]11[.]142
201[.]103[.]222[.]151
201[.]244[.]108[.]183
217[.]165[.]233[.]123

Subject Common Names

epyhm[.]net
twmbelz[.]org
iene[.]info
ctxehfdug[.]net

utip[.]biz
jaonioi[.]org
vcivoqeqfh[.]us
ineieutzvt[.]mobi
tuayjhrdwg[.]mobi
oxouy[.]mobi
iemjmedtey[.]biz
ouxtetbtn[.]biz
ghoaetksiwo[.]net
fwoht[.]org
tqouhdk[.]mobi
pidewaeetbu[.]us
aihpe[.]mobi
zemureisir[.]info
oialk[.]com
ihaknpq[.]us
jqseote[.]us
gzfjtyr[.]com
aeztfeq[.]net
qbez[.]info
omloeceqiu[.]biz
ztiorhvb[.]net
lfad[.]mobi
egatcwojan[.]us
zcstobno[.]us
faexgkbimwe[.]org
bdae[.]info
xoehdsoeao[.]org
iekztmiw[.]com
oojfkdbgiec[.]info
ioiu[.]us
jaouem[.]info
xocsuioij[.]biz
euydxykaie[.]org
ipzc[.]net
lmatetu[.]mobi
woaitgja[.]info
kmeyihr[.]org
bvgfkdinjla[.]net
mrokouejcei[.]mobi
ztmt[.]org
epmsxuv[.]info

vsasikavjed[.]biz
yieziqg[.]biz
zvtilriljat[.]net
vzxei[.]net
fbiafxq[.]info
pmeooxard[.]org
gimcyeeoof[.]org
qocu[.]org
eeapissopx[.]biz
otihelb[.]biz
ewaguarw[.]org
haeoieee[.]info
gokeokaut[.]biz
czqphiwowf[.]biz
ieaorbuq[.]net
tcnzewxk[.]us
lynle[.]biz
hzlfitjo[.]net
alcvi[.]info
wcyoloy[.]mobi
temthdmeo[.]org
zufmpz[.]mobi
lijivtamo[.]mobi
kouxe[.]org
aidoxovuncx[.]mobi
rtouaxye[.]biz
zevjeo[.]mobi
aispzwot[.]biz
paod[.]org
iqtfotoe[.]mobi
twdifusycee[.]biz
frkneeatb[.]info
eehpeplhr[.]us
aodkhtecx[.]net
efpohwf[.]net
oesyahoixic[.]us
motnooz[.]biz
