**How to: Generate self-signed cert on the WAF, download a CSR, key stays on the waf, take CSR and get it signed, then upload the signed cert. Key never leaves waf for obvious reasons.**

I've got a few API calls in here, and some things I do in the GUI.

Step 1: Generate a Self-Signed cert for fastfood.com on the WAF. I actually did this in the GUI, because that's how we did it on the phone call yesterday, wanted to keep same workflow. In the postman collection sent yesterday are examples of generating it using the api if you want to use the api.

Step 2: Download the CSR. Can do in the gui of course, but here is the Api version:

Get login Token. This is just showing you my little token script, useful if I'm not using postman, because I don't want the long token in all my curl commands.

I cat it out here just so you can see it, copy it if you like, if it may be useful to you. It creates a file named token.txt that I can use with -K in subsequent curl calls.

brett@Azure:~$ cat bwaf\_login.txt  
curl <http://bwaf99.eastus.cloudapp.azure.com:8000/restapi/v3.1/login> -X POST -H Content-Type:application/json -d '{"username": "admin", "password": "Hello123456!"}' > blah.txt

sed 's/"/ /g' < blah.txt | awk '{print "-u \"" $4 ":\"" }' > token.txt

brett@Azure:~$ chmod +X bwaf\_login.txt

brett@Azure:~$ ./bwaf\_login.txt

Now do a PUT to actually download the CSR:

brett@Azure:~$ curl -K token.txt -X PUT "<http://bwaf99.eastus.cloudapp.azure.com:8000/restapi/v3.1/certificates/self-signed-certificate/fastfood.com?operation=download>" -H "accept:application/json" -H "Content-Type: application/json" -d '{"download-type": "csr"}'

-----BEGIN CERTIFICATE REQUEST-----

MIICKDCCAZECAQAwgYYxCzAJBgNVBAYTAnVzMRMwEQYDVQQIDApDYWxpZm9ybmlh

Etc etc etc etc etc

qdNepW6H/KBxbr4uIUPWt6WsXGqF57a9fz4fng==

-----END CERTIFICATE REQUEST-----.

Step 3: Take CSR to my CA and get it signed. The CSR is just ascii text, so I just copy and paste that text to a file on my linux box that has my CA on it. I am using openssl here, I don't have Venafi. We are working on talk to Venafi about an integration, by the way, and I know that's something you are looking for. Due to current events, it is slow-going getting a response from them. I hope you understand, but we do intend to partner with them. I completely understand the benefits of it as where I worked many of my customers had Venafi.

But for now, here is my openssl CA signing the CSR, and outputting a signed cert for fastfood.com

root@MSI:~/my\_ssl\_ca# openssl x509 -req -in fastfood.com.csr -CA rootCA.crt -CAkey rootCA.key -CAcreateserial -out fastfood.com.crt -days 500 -sha256  
Signature ok  
subject=/C=us/ST=California/L=Campbell/O=Barracuda Networks/OU=Barracuda Networks/CN=fastcood.com  
Getting CA Private Key  
Enter pass phrase for rootCA.key:  
root@MSI:~/my\_ssl\_ca# cat fastfood.com.crt  
-----BEGIN CERTIFICATE-----  
MIID1zCCAb8CCQD8GygReQKrjDANBgkqhkiG9w0BAQsFADBYMQswCQYDVQQGEwJV  
etc etc etc etc

WHLzNskR/D9+nNyAfPbThgNx2uWOc5h9uJRE  
-----END CERTIFICATE-----  
root@MSI:~/my\_ssl\_ca#

Step 4: upload the signed cert. I used the GUI. I called it fastfood.com.signed



Step 5: Associate the signed cert to my service, and test. I am just using curl with -V, for verbose, we see it is signed by my openssl CA which is named root.example.com

root@MSI:~/my\_ssl\_ca# curl -k <https://fastfood.com> -v  
\* Rebuilt URL to: <https://fastfood.com/>\* Trying 52.170.144.232...  
\* Connected to fastfood.com (52.170.144.232) port 443 (#0)  
\* found 148 certificates in /etc/ssl/certs/ca-certificates.crt  
\* found 592 certificates in /etc/ssl/certs  
\* ALPN, offering http/1.1  
\* SSL connection using TLS1.2 / ECDHE\_RSA\_AES\_256\_GCM\_SHA384  
\* server certificate verification SKIPPED  
\* server certificate status verification SKIPPED  
\* common name: fastfood.com (matched)  
\* server certificate expiration date OK  
\* server certificate activation date OK  
\* certificate public key: RSA  
\* certificate version: #1  
\* subject: C=US,ST=California,L=Campbell,O=Barracuda Networks,OU=Barracuda Networks,CN=fastfood.com  
\* start date: Fri, 27 Mar 2020 15:27:32 GMT  
\* expire date: Sun, 26 Apr 2020 15:27:32 GMT  
\* issuer: C=US,ST=CA,O=Internet Widgits Pty Ltd,CN=root.example.com  
\* compression: NULL  
\* ALPN, server did not agree to a protocol  
> GET / HTTP/1.1  
> Host: fastfood.com  
> User-Agent: curl/7.47.0  
> Accept: \*/\*  
>  
< HTTP/1.1 200 OK