## Implementation Notes: Guest Registration/Login using the DNS Enforce plugin

The first thing to do is configure the DHCP server for the Guest wifi network to set the DNS address to use an IP address that we’ll put on the managing Forescout appliance. Then we configure that IP address in the configuration for DNS Enforce. This needs to be an available IP address on your network of course. And by “managing appliance” I mean the appliance that manages the segment that will have the IP addresses given to the Guest wifi connections.

The Guest policy will have an action of “HTTP Login”. Hosts that fall into that subrule will get that action. There are a couple of things that can go wrong here so if it doesn’t look like it is working you can check a few things:

1. Enable debug level 6 on the hijack\_assist plugin (fstool hijack\_assist 6 1h) and have the action occur. Then check the logs for “action\_add”. This will show the URL that is going to be redirected-to:

hijack\_assist:141425:1646056201.117575:Mon Feb 28 07:50:01 2022: action\_add: ip=[10.102.16.16] url=[http://NOC-FSCT03.hidden.net:80/status?z=-3239115154639029625] host=[NOC-FSCT03.hidden.net]

1. Make sure that hostname can be resolved by the real DNS by running nslookup.
2. Add this hostname to the whitelist on the DNS Enforce plugin. Type it all in and leave the Regex checkbox unchecked.
3. fstool hijack\_assist restart whenever you change the whitelist
4. Verify what is in the whitelist (note that this is turned into a regular expression):

cat /dev/shm/hijack\_assist-exclude

NOC\-FSCT03\.hidden\.net

1. You’re supposed to have a signed/trusted certificate installed. It’s best to have a wildcard like \*.hidden.net in our example.
2. In the HTTP redirect configuration, check the “redirect using DNS name”
3. Also verify that this FQDN is in the hosts\_filter.conf file in /usr/local/apache2/conf. If it isn’t then use this to set the ‘extra’ hosts:

fstool set\_property fs.httpd.extra.hosts "guest NOC-FSCT03.hidden.net"

cat /usr/local/apache2/conf/hosts\_filter.conf

LoadModule rewrite\_module modules/mod\_rewrite.so

LoadModule allowmethods\_module modules/mod\_allowmethods.so

RewriteEngine on

ErrorDocument 500 "Error: Invalid request"

<If "req('Host') !~ /^noc-fsem01$|^\[::1\]$|^127.0.0.1$|^localhost$|^NOC-FSCT03.hidden.net$/i">

RewriteRule ^ https://%1%{REQUEST\_URI} [R=500,L]

</If>

Here is the desired flow, and you can use tcpdump to verify this:

1. Guest connects to Guest wifi and gets IP address plus DNS pointed to the Forescout appliance’s pseudo-DNS IP address
2. The laptop hits the Guest policy and we get the HTTP Login action
3. The various apps on the Guest’s laptop start firing off DNS requests. DNS Enforce responds to all of them with its own IP. For example, if the pseudo-IP is 172.16.10.30, DNS Enforce will keep saying that all FQDNs are at 172.16.10.30. \*Unless\* the FQDN being asked for is in the whitelist of course.
4. A browser on the Guest’s laptop will try to open a page. For example say it wants to go to [www.google.com](http://www.google.com).
   1. The browser looks up the IP by sending a DNS Request.
   2. The DNS sends back 172.16.10.30.
   3. The browser sends an HTTP GET to [www.google.com](http://www.google.com) which it thinks is at 172.16.10.30.
   4. The DNS Enforce plugin sees this HTTP GET and sends back a 302 Redirect to the HTTP Login page, such as the one given as an example above i.e. NOC-FSCT03.hidden.net…)
   5. The browser sends a DNS Request for NOC-FSCT03.hidden.net.
   6. The DNS Enforce plugin sees that this FQDN is in the whitelist and so forwards the DNS Request to the real DNS.
   7. The real DNS responds with the real IP for NOC-FSCT03.hidden.net.
   8. The DNS Enforce plugin forwards the response to the browser.
   9. The browser sends an HTTP GET to the real address (the management interface on the appliance). And so apache picks this up and we send the Guest Registration/Login page.