Working With APIs

Getting Started

The purpose of this guides is two fold:

- First, to provide an introduction to those unfamiliar with using the FileWave API
- Second, to be a reference for commands that can be used within the API

Command Line API refers to the original RESTful API. Recognisable both by the port used and URL paths commencing:

- /inv/

FileWave Anywhere API v2 refers to the newer web admin, FileWave Anywhere, API and recognisable by paths commencing:

- /api/

Purpose of an API

As described, APIs are designed to communicate with systems. As such, FileWave may be leveraged by other systems, e.g. SCCM data engines, in-house databases, etc. To maintain security, there must be an authentication method to allow such communication to take place. The API provides this kind of ability to provide in-depth integration on an as-needed basis.

Basic, non-specific command line examples:

macOS shell

```
curl -s -H "Authorization: $auth" \
    https://$server_dns:20445/inv/api/v1/query_result/ \
    --data $query \
    -H "Content-Type: application/json"
```

Windows PowerShell

```
$header = @{Authorization="$auth"}

Invoke-RestMethod -Method POST \
    -Headers $header \
    -ContentType application/json \
    -Uri https://$server_dns:20445/inv/api/v1/query_result/ \
    -Body $query
```

The macOS 'Authorization' or Window's 'Headers' make use of base64 tokens. From the above code, $auth would need to be set as one of these tokens.

> Each user generated will have their own unique base64 token automatically generated. This token can be revoked and new tokens created. Each user may have multiple tokens.

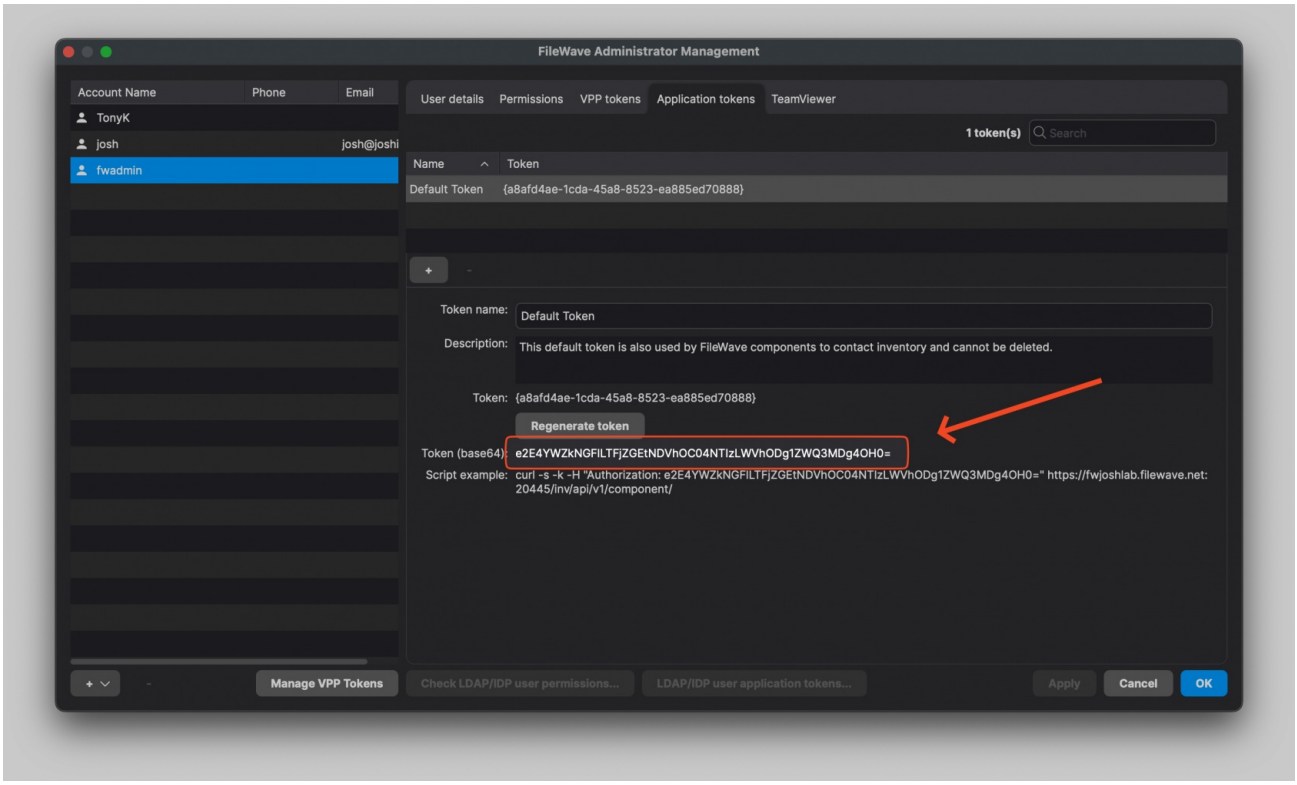
The Token is the Key

> The token allows access to any API calls from anything that has access to the FileWave Server. It should not be shared unnecessarily and otherwise should be kept secret.

Tokens may be viewed from FileWave Central App:

- Assistants > Manage Administrators

Select a desired Administrator and choose the 'Application Tokens' tab. The token value to copy is the 'Token (base64)'



Token value from above image:

```
e2E4YWZkNGFlLTFjZGEtNDVhOC04NTIzLWVhODg1ZWQ3MDg4OH0=
```

When choosing accounts and tokens for API interaction, consider making dedicated users for the task(s) required. Set the permissions of that user to limit their ability to the required task alone. If the token is compromised, this will limit not only the scope of how that token could be used by an attacker, but when revoking and generating a new token, minimal impact would be experienced.

For more information on the Application Tokens see the page: Managing FileWave Administrators (Application Tokens)

API Requests

Requests could be one of:

Command Type	Description
GET	Returns a resource value
PUT	Replaces a resource
PATCH	Updates a resource
POST	Creates a resource
DELETE	Removes a resource

Data sent with requests are in the form of a JSON, as are the responses from those requests.

JSON data is broken into keys/value pairs, where values could even be lists inside of lists.

Examples

Below examples are using a python tool to reformat the returned response. Python must be installed to benefit from this. If not, remove the piped section of the command or instal Python.

For example, actioning a GET to list of all FileWave Server Inventory Queries, could return a JSON similar to the below:

List Existing Inventory Queries

1) Get all Queries

```
curl -s  -H "Authorization: e2FjYzRkYmQzLTI3ZjYtNDEyMi1iMGVhLTI1YmY0OGNmYWM0NX0=" \
    https://myserver.company.org:20445/inv/api/v1/query/ | python3 -mjson.tool
```


```
[
    {
        "id": 1,
        "name": "All Windows",
        "favorite": true,
        "group": 1,
        "version": 1
    },
    {
        "id": 2,
        "name": "Mac OS X 10.7-10.11",
        "favorite": false,
        "group": 1,
        "version": 5
    },
...
    {
        "id": 103,
        "name": "All Computers to retire",
        "favorite": false,
        "group": 3,
        "version": 2
    }
]
```


Key	Value	Description
`id`	`integer`	The unique number for the query. To be used as reference
`name`	`string`	The name given to the query
`favorite`	`true/false`	Whether or not the query should show in the sidebar
`group`	`integer`	The group number given. built-in queries – for example – would be in the "Sample Queries" group, which is group 1. If the user made new groups
`version`	`integer`	The version for the query. How many times has the query been altered and saved, starting with 1

Return the definition of a chosen query

The query definition, not the results of the query. Using ID 1 as an example:

2) Get Query

```
curl -s -H "Authorization: e2FjYzRkYmQzLTI3ZjYtNDEyMi1iMGVhLTI1YmY0OGNmYWM0NX0=" \
    https://myserver.company.org:20445/inv/api/v1/query/1 | python3 -mjson.tool
```


```
{
    "criteria": {
        "expressions": [
            {
                "column": "type",
                "component": "OperatingSystem",
                "operator": "=",
                "qualifier": "WIN"
            }
        ],
        "logic": "all"
    },
    "favorite": true,
    "fields": [
        {
            "column": "device_name",
            "component": "Client"
        },
```

```
        {
            "column": "filewave_client_name",
            "component": "Client"
        },
        {
            "column": "name",
            "component": "OperatingSystem"
        },
        {
            "column": "version",
            "component": "OperatingSystem"
        },
        {
            "column": "build",
            "component": "OperatingSystem"
        },
        {
            "column": "edition",
            "component": "OperatingSystem"
        }
    ],
    "main_component": "Client",
    "name": "All Windows",
    "id": 1,
    "version": 1,
    "group": 1
}
```

Key	Value	Description
`criteria`	`array`	Expressions and logic of query
	Criteria Expressions (Each entry will require all of the below. Add multiple entries to the array as required):	

Key	Value	Description
`column`	Multiple values e.g. 'version', 'device_id', etc.	Chosen search component (Figure 1.2 #1)
`component`	Multiple values e.g. 'Client', 'OperatingSystem', etc.	Group containing above component (Figure 1.2 #2)
`operator`	Multiple values e.g.'is', 'begins', etc.	Method of comparison (Figure 1.2 #3)
`qualifier`	Multiple values (Either a: String, Integer, Date or Boolean value)	Value for comparison (Figure 1.2 #4)

Logic:

Key	Value	Description
`logic`	Multiple values 'all', 'none' or 'one'	How Components should be logically considered for correct return of results (Figure 1.2 #5)

`favourite`	`true/false`	Show/Hide from FileWave Central sidebar Inventory Queries
`fields`	`array`	Which components will be shown (ordered first to last)
	Fields to display (Each entry will require all of the below. Add multiple entries to the array as required):	

Key	Value	Description
`column`	Multiple values e.g. 'device_name', 'name', etc.	Component to display (Figure 1.2 #1)

	`component`	Multiple values e.g. 'Client', 'OperatingSystem', etc.	Group containing above component (Figure 1.2 #2)
`main_component`	Selection Box		Important this is set correctly (Figure 1.2 #6)
`name`	`string`		Inventory Query name shown in FileWave Central
`id`	`integer`		Inventory Query unique number, not already in use. (Each query has a unique number, starting at 1 and incremented with each new query generated when actioned through FileWave)
`version`	`integer`		Increment by 1 for each alteration
`group`	`integer`		The Inventory Query group which the query should be displayed within. E.g. 'group' value of 1 would be within the 'Sample Queries' group

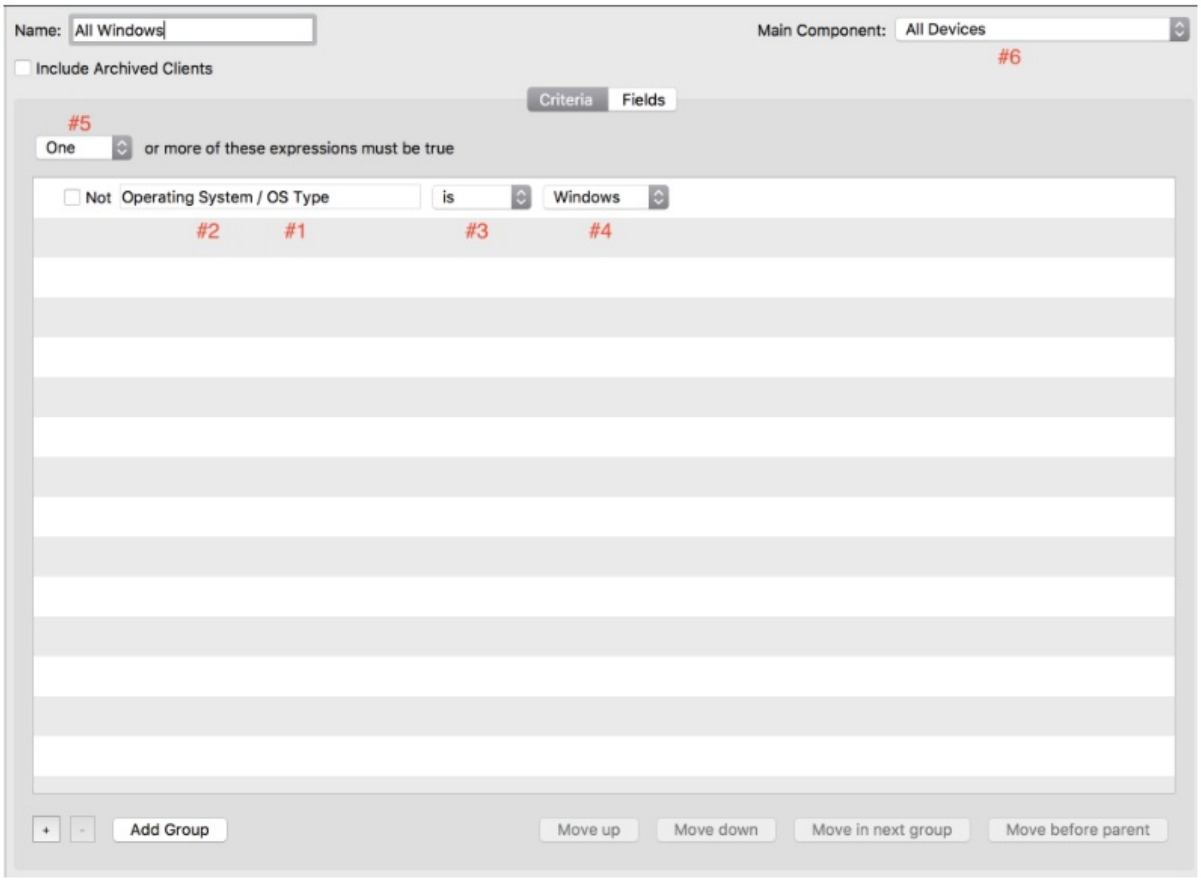


Figure 1.2 - Query Builder Criteria

Return the Inventory Query Results

3) Get Query Results

```
curl -s  -H "Authorization: e2FjYzRkYmQzLTI3ZjYtNDEyMi1iMGVhLTI1YmY0OGNmYWM0NX0=" \
  https://myserver.company.org:20445/inv/api/v1/query_result/1 \
  | python3 -mjson.tool
```


```
{
    "total_results": 13,
    "filter_results": 13,
    "offset": 0,
    "values": [
        [
            "FW-BLUE-02",
```

```
            "FW-Blue-02",
            "Windows 10.0",
            "10.0.0",
            "10240",
            "Microsoft Windows 10 Home"
        ],
        [
            "LAPTOP-C6LLFGH6",
            "FH-History3",
            "Windows 10.0",
            "10.0.0",
            "14393",
            "Microsoft Windows 10 Home"
        ],
...
    ],
    "version": 3
}
```

Key	Value	Description
`total_results`	`integer`	Total count of results
`filter_results`	`integer`	
`offset`	`integer`	
`values`	`array`	The results. Repeated for each result. Items depends on what your specified in the fields
`version`	`integer`	The version for the query. How many times has the query been altered and saved, starting with 1

JSON

Verify JSON Formatting:

- https://jsonlint.com/

API Application

- Postman https://www.getpostman.com/ (macOS, Windows, and Linux)

Browser Extensions

- Mod-Header https://mod-header.appspot.com/ Will allow you to use the Google Chrome Browser to view and interact with the FileWave API

Commands

Remember: All URLs start with

```
https://myserver.company.org:20445/inv/api/v1/
```

Must include the authorization header

Below are the possible options:

URLs

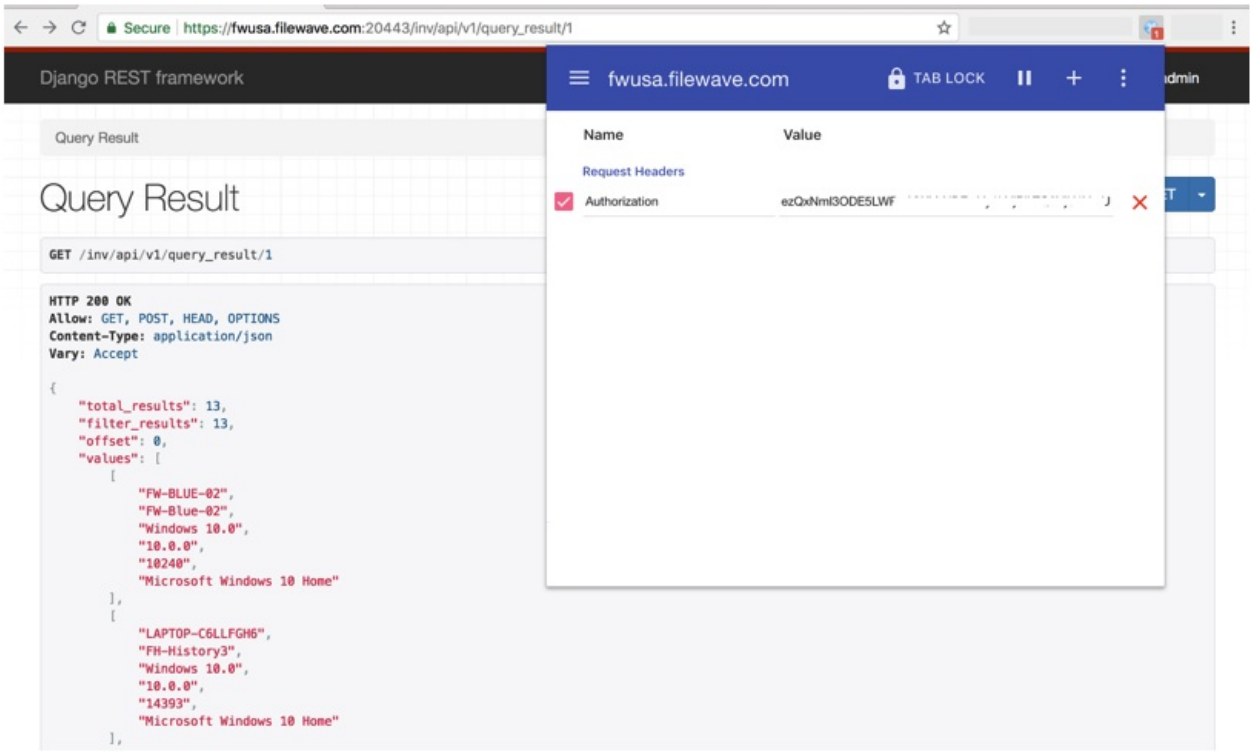
URL	Use	Options
	Inventory	
`query`	Show all queries	`GET POST`
`query/#`	Show information on a single query Where # is the query ID	`GET PUT DELETE`
`query_group/`	Show all query group	`GET POST`
`query_group/#`	Detail information on a single group Where # is the group ID	`GET PUT DELETE`
`query_result/#`	Show the results of one query	`GET POST`

	Where # is the query ID	
`query_count`		`POST`
`component`	Show all component options on your instance	`GET`
`field_type`	Show all fields on your instance	`GET`
License		
`license_definition`	Show all query	`GET`
`license_definition/#`	Show information on a single license Where # is the license ID	`GET`
Custom Fields		
`custom_field/`	Show all custom fields	
`custom_field/get_association`		`POST`
`custom_field/set_association`		`POST`
`custom_field/upload`		`POST`
`custom_field/usages/<Field_Name>`	Where <Field_Name> is the Internal Name (E.G "battery_cycle_count")	`GET`
`custom_field/values/`		`POST`
`custom_field/edit/`		`POST`

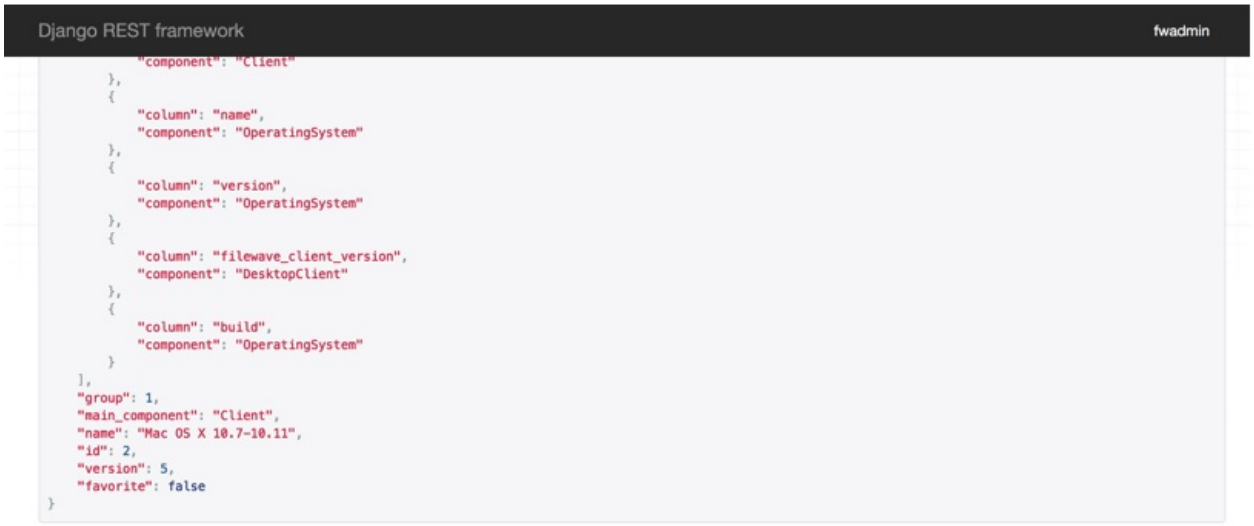
Examples

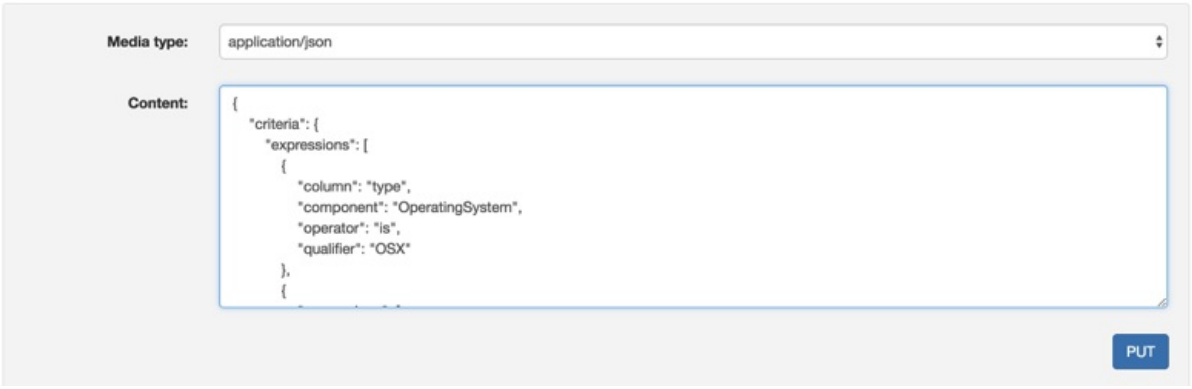
Using a browser extension

Using Mod-Header (see tools section), you can make Chrome a RESTful API browser by taking advantage of the FileWave Django Framework. Leveraging URLs and authorisation token to return Query Results.



Even if the URL is typically a POST, it provides an output similar to the following





Using the curl command

Viewing all available queries (GET)

```
curl -s -H "Authorization: e2FjYzRkYmQzLTI3ZjYtNDEyMi1iMGVhLTI1YmY0OGNmYWM0NX0=" \
  https://myserver.company.org:20445/inv/api/v1/query/ \
  | python3 -mjson.tool
```

Posting a new query (POST)

```
curl -s -H "Authorization: e2FjYzRkYmQzLTI3ZjYtNDEyMi1iMGVhLTI1YmY0OGNmYWM0NX0=" \
  --header "Content-Type: application/json" \
  -X POST \
  -d @<path/name of new query.json> \
  https://myserver.company.org:20445/inv/api/v1/query/
```

Removing a query (DELETE)

```
curl -s -H "Authorization: e2FjYzRkYmQzLTI3ZjYtNDEyMi1iMGVhLTI1YmY0OGNmYWM0NX0=" \
  -X DELETE https://myserver.company.org:20445/inv/api/v1/query/<id#>
```

For more curl help, see: Using the RESTful API to limit, sort, and offset values returned

Self-Signed Certificates

Hopefully everyone is using official certificates. However, if the FileWave Server does have a self-signed certificate, the above commands should fail. To ignore the warnings the following is required.

macOS

Add the -k option to the command. E.g.

```
curl -s -k -H "Authorization: e2FjYzRkYmQzLTI3ZjYtNDEyMi1iMGVhLTI1YmY0OGNmYWM0NX0=" \
  https://myserver.company.org:20445/inv/api/v1/query/ \
  | python3 -mjson.tool
```

Windows

PowerShell requires somewhat more code to ignore the warning. Add the below to the beginning of any script calling an API to a server with a self-signed certificate:

```
# Required for self-signed certs only
function Ignore-SSLCertificates
{
    $Provider = New-Object Microsoft.CSharp.CSharpCodeProvider
    $Compiler = $Provider.CreateCompiler()
    $Params = New-Object System.CodeDom.Compiler.CompilerParameters
    $Params.GenerateExecutable = $false
    $Params.GenerateInMemory = $true
    $Params.IncludeDebugInformation = $false
    $Params.ReferencedAssemblies.Add("System.DLL") > $null
    $TASource=@'
        namespace Local.ToolkitExtensions.Net.CertificatePolicy
        {
            public class TrustAll : System.Net.ICertificatePolicy
            {
                public bool CheckValidationResult(System.Net.ServicePoint
sp,System.Security.Cryptography.X509Certificates.X509Certificate cert, System.Net.WebRequest req, int problem)
                {
                    return true;
                }
            }
        }
'@
    $TAResults=$Provider.CompileAssemblyFromSource($Params,$TASource)
    $TAAssembly=$TAResults.CompiledAssembly
    ## We create an instance of TrustAll and attach it to the ServicePointManager
    $TrustAll = $TAAssembly.CreateInstance("Local.ToolkitExtensions.Net.CertificatePolicy.TrustAll")
    $AllProtocols = [System.Net.SecurityProtocolType]'Ssl3,Tls,Tls11,Tls12'
    [System.Net.ServicePointManager]::SecurityProtocol = $AllProtocols
    [System.Net.ServicePointManager]::CertificatePolicy = $TrustAll
}

Ignore-SSLCertificates
```

Using PHP

Saved as a php file (like inv.php), update the URL and auth code, then place the file on a web server where PHP has been enabled. This creates a 'view only' web page version of the inventory. People can enter the URL for the query, hit refresh as many times as they like and will always see the latest information in inventory. All without having to hassle IT for the latest data.

The output of the query results isn't fancy, but this is to illustrate what can be achieved.

▼ PHP Inventory Viewer

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd"><html xmlns="http://www.w3.org/1999/xhtml">
<html>
<?php
$baseurl="myserver.company.org";
$port="20445";
$authcode="e2FjYzRkYmQzLTI3ZjYtNDEyMi1iMGVhLTI1YmY0OGNmYWM0NX0=";


ini_set('display_errors', 'On');
### do not edit below ###
if (!isset($_GET["qid"])){
        $url = "https://".$baseurl.":".$port."/inv/api/v1/query/";
} else {
        $url = "https://".$baseurl.":".$port."/inv/api/v1/query_result/".$_GET["qid"];
}
//  Initiate curl
$ch = curl_init();
// Set the url
```

```
curl_setopt($ch, CURLOPT_URL,$url);
// Disable SSL verification
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, FALSE);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, FALSE);
curl_setopt($ch, CURLOPT_SSLVERSION, 1);
// Will return the response, if false it print the response
curl_setopt($ch, CURLOPT_RETURNTRANSFER, TRUE);
//authenticate
curl_setopt($ch, CURLOPT_HTTPHEADER,array('Authorization:<'.$authcode.'>='));
// Display errors if any
if (curl_errno($ch)) {
   print curl_error($ch);
}
// Execute
$result=curl_exec($ch);
if (curl_errno($ch)) {
   print curl_error($ch);
}
$output=json_decode($result, true);
curl_close($ch);

//create function for looping unknown dimensional array
function printAll($a) {
  if (!is_array($a)) {
    echo $a, ' <br/>';
    return;
  }
echo "<br/>";
  foreach($a as $v) {
    printAll($v);
  }
}

// Start html Page
echo '<head>'
.'<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />'
.'<title>'.$baseurl.'Inventory Page</title>'
.'</head>'
.'<body>'
.'<style type="text/css">'
."body {font-family:'Helvetica Neue Light', 'Helvetica Neue', Helvetica, Arial, 'Lucida Grande', sans-serif;}"
.'h1,h2,h3,h4,{font-weight:100;}'
.'div {padding:10px; color:#fff; background:#333;}'
.'div.output{border:1px solid rgba(0,0,0,0.1); background: rgba(0,0,0,0.03); color:#555;-webkit-border-
radius:3px;border-radius:3px;margin:15px 25px; padding:10px;}'
.'tr:nth-child(even) {background: rgba(255,255,255,0.85);}'
.'tr:nth-child(odd) {background: rgba(0,0,0,0.05);}'
.'a, a:hover {color:#ce1300; text-decoration:none;}'
.'</style>'
.'<div><h1><img src="https://www.filewave.com/images/filewave-1024x788.png" height="60" />'.$baseurl.'
Inventory</h1></div>'
.'<br/>';

// Default homepage
if (!isset($_GET["qid"])){
        echo "<table>"
        ."<thead>"
        ."<tr>"
        ."<th>&hearts;</th>"
        ."<th>Query Name</th>"
        ."<th>Query ID</th>"
        ."</tr>"
        ."</thead>"
        ."<tbody>";
        foreach ($output as &$value) {
                if ($value['favorite'] == true) { $fav="&hearts;";} elseif ($value['favorite'] == false)
{$fav=" ";}
                echo "<tr><td>".$fav."</td><td>".$value['name']."</td><td><a href='".$_SERVER['PHP_SELF']."?
qid=".$value['id']."&n=".$value['name']."'>".$value['id']."</a></td></tr>";
        }
```

```
        echo"</tbody></table>";
}
//If an individual query has been selected
elseif (isset($_GET["qid"],$_GET["n"]))  {
        echo "<h3>Home &gt; Query: "
        .$_GET["n"]
        ."</h3><hr/>"
        ."<strong>Total Results: </strong>"
        .$output['total_results']
        ."<br/>"
        .'<strong>First Column results: </strong>';

        foreach ($output['values'] as &$value) {
                echo $value['0'].' <strong>   |   </strong>';
        }
echo "<br/>"
."<strong> All Results: </strong><br/><div class='output'>";
printAll($output['values']);
echo "</div>";
#var_dump($result);
}
else {
        echo "<h1 style='color:#ff0000;'> An error has occurred</h1> Parhaps you used a bookmark and the URL
has changed";
}
?>
<hr/>
<center><font style=" font-size:9px;"><a href="http://filewave.com" target="_blank">&copy; BenM@ FileWave</a>
</font></center>
</body>
</html>
```

Revision #13
★Created 4 July 2023 15:18:20 by Sean Holden
Updated 28 March 2024 14:46:13 by Josh Levitsky