

Basic Orbital Search Queries for Threat Analysis

Contents

[Introduction](#)

[Prerequisites](#)

[Requirements](#)

[Components Used](#)

[Background Information](#)

[Access](#)

[Custom Queries](#)

[1. Startup Items](#)

[2. Sha256 Hashes of Running Processes](#)

[3. Process with Network Connections](#)

[4. Privileged Process with Non-Localhost Network Connection](#)

[5. Backup/Restore Registry Monitoring](#)

[6. File Search](#)

[7. Powershell History Monitoring](#)

[8. Prefetch Query](#)

[9. Address Resolution Protocol \(ARP\) Cache Inspection](#)

Introduction

This document describes the basic orbital search queries for threat analysis.

Prerequisites

Requirements

Cisco recommends that you have knowledge of the interest in the understanding of threats and malware and a basic understanding of Structured Query Language (SQL) tables.

Components Used

The information in this document is based on these software and hardware versions:

- Secure Endpoint Connector version 7.1.5 or later for Windows
- Secure Endpoint Connector version 1.16 or later for Mac
- Secure Endpoint Connector version 1.17 or later for Linux
- Secure Endpoint user must be assigned the role of **admin to deploy Orbital**

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Background Information

The Custom Queries are leveraged which must help you quickly learn the power of Orbital and osquery for threat hunting.

Orbital makes use of osquery's stock tables in addition to Orbital-specific tables. The results returned through Orbital can be sent to other applications, such as Secure Endpoint, Secure Malware Analytics, and SecureX Threat Response, and can be stored in remote data stores (RDS), such as Amazon S3, Microsofts Azure, and Splunk.

Use the Orbital Investigate page in order to construct and execute live, live queries on endpoints in order to gather more information from them. Orbital uses osquery, which allows you to query your devices like a database with basic SQL commands.

Here is a simple example: `SELECT column1, column2 FROM table1, table2 WHERE column2='value'`.

In this example, column1 and column2 are the field names of the table from which you want to choose data. In order to choose all fields available in the table, use this syntax: `SELECT * FROM table1`.

Access

Open Orbital directly at these sites:

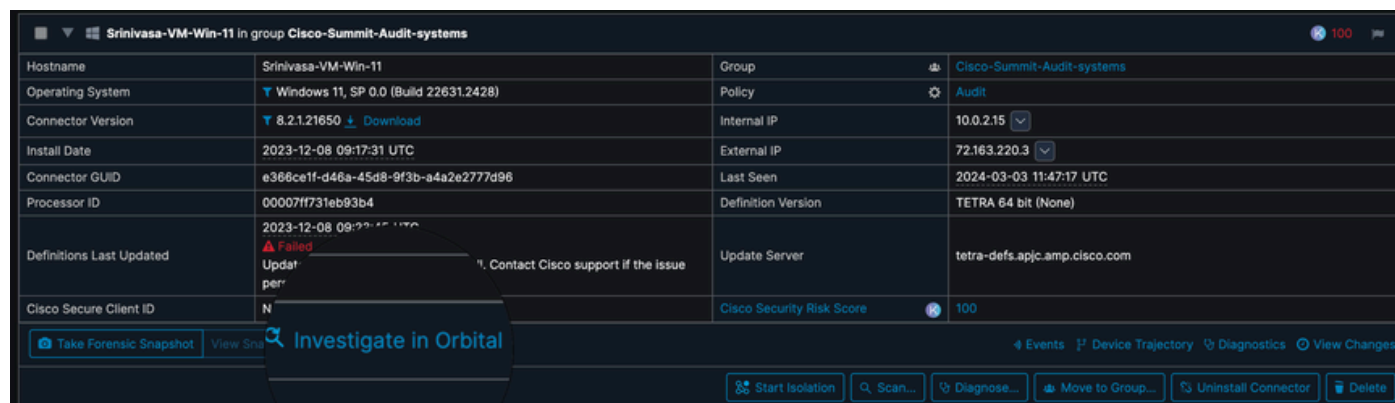
North America - <https://orbital.amp.cisco.com>

Europe - <https://orbital.eu.amp.cisco.com>

Asia Pacific - <https://orbital.apjc.amp.cisco.com>

Or

On the Secure Endpoint Console, choose the impacted Host system and click **Investigate in Orbital**.



There are options for using the Orbital Catalogue (Click **Browse**) or Enter the Custom Queries under Custom SQL section as mentioned:



Investigate

Clear



0 rows from 1 endpoint

Endpoints *Add host:hostname, IP, MAC, node ID, or Connector GUID*

.amp:2450c73e-3c60-40fd-9ba8-91fd67697397 x



Query

Script

Search Catalog



Browse

Custom SQL *ex. SELECT column_name FROM table_name;*



Run Query

Schedule Query

ENDPOINT dnGYB0xoj
No Result. Last Seen 2024-0

Custom Queries

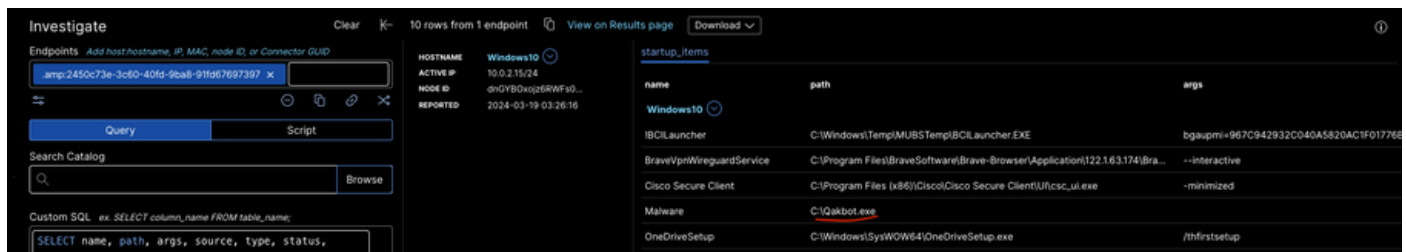


Note: Host system is in the lab network and it is tried to keep the system/network unharmed.

1. Startup Items

Startup items can be exploited by attackers to maintain persistence on a compromised system, which means that the malicious software will continue to run or be re-launched automatically with each system restart. In the next example, **Qakbot.exe** is running in the host system.

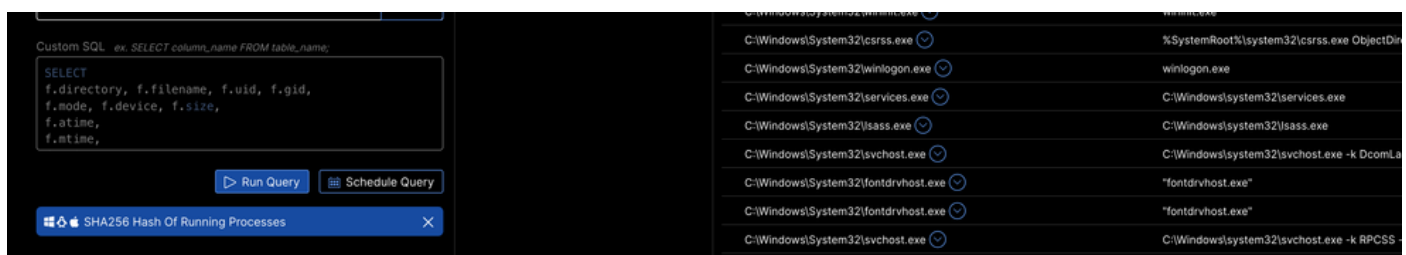
```
SELECT name, path, args, source, type, status, username  
FROM startup_items;
```



2. Sha256 Hashes of Running Processes

SHA256 hashes are not inherently associated with running processes in their natural state. However, security software and system monitoring tools can calculate the SHA256 hash of a running process of the executable file in order to help verify its integrity and authenticity.

```
SELECT
p.pid, p.name, p.path, p.cmdline, p.state, h.sha256
FROM processes p
INNER JOIN hash h
ON p.path=h.path;
```



STILL_ACTIVE	4865366ea2c4a60d4f6d3c8bcd345fa15c5ae5270163043582972632246f0a54	✓
STILL_ACTIVE	43ec773e0ec626bf6d8a7fd04e64dc36afa6801444a3c36ef4da2a909fa0d83f	✓
STILL_ACTIVE	652607db7763f423419fd98807a2436f22007e0a54965f24c671bbd1a20197d6	✓
STILL_ACTIVE	f13de58416730d210dab465b242e9c949fb0a0245eef45b07c381f0c6c8a43c3	✓
STILL_ACTIVE	f71d6bcd8e1440f39c0f5ed88e5edd66833987126366f9d12e136199af90f1d9	✓
STILL_ACTIVE	f71d6bcd8e1440f39c0f5ed88e5edd66833987126366f9d12e136199af90f1d9	✓
STILL_ACTIVE	f13de58416730d210dab465b242e9c949fb0a0245eef45b07c381f0c6c8a43c3	✓

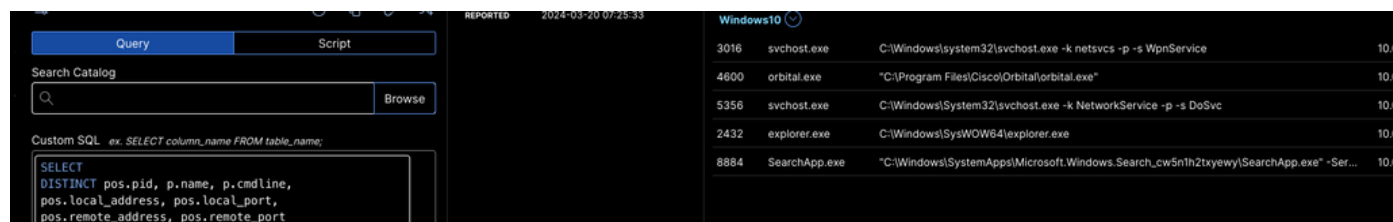
if the associated hash of a File is malicious, you will be able to identify with this query.

3. Process with Network Connections

Processes with network connections are programs or system services that are actively using the network interface in order to communicate with other devices on a network or over the internet.

```
SELECT
DISTINCT pos.pid, p.name, p.cmdline, pos.local_address, pos.local_port, pos.remote_address, pos.remote_port
FROM processes p
JOIN process_network pos
ON p.pid=pos.pid;
```

```
JOIN process_open_sockets pos USING (pid)
WHERE
pos.remote_address NOT IN ("", "0.0.0.0", "127.0.0.1", ":::", ":::1", "0");
```



The screenshot shows a network monitoring tool interface. On the left, there's a 'Query' tab with a 'Search Catalog' field and a 'Browse' button. Below it, a 'Custom SQL' field contains a query. On the right, a table lists processes with columns for PID, process name, command line, and IP address.

PID	Process Name	Command Line	IP Address
3016	svchost.exe	C:\Windows\system32\svchost.exe -k netsvcs -p -s WpnService	10.0
4600	orbital.exe	"C:\Program Files\Cisco\Orbital\orbital.exe"	10.0
5356	svchost.exe	C:\Windows\System32\svchost.exe -k NetworkService -p -s DoSvc	10.0
2432	explorer.exe	C:\Windows\SysWOW64\explorer.exe	10.0
8884	SearchApp.exe	"C:\Windows\SystemApps\Microsoft.Windows.Search_cw5n1h2txyewy\SearchApp.exe" -Ser...	10.0

4. Privileged Process with Non-Localhost Network Connection

Running program or service that has elevated permissions (like those of an administrator or system account) and is communicating over the network with an external device or service—meaning any IP address other than 127.0.0.1 (localhost) or ::1 (IPv6 localhost).

```
SELECT DISTINCT p.name, p.cmdline, pos.pid, pos.local_address, pos.local_port, pos.remote_address, pos.
FROM processes p JOIN process_open_sockets pos USING (pid)
WHERE pos.remote_address NOT IN ("", "0.0.0.0", "127.0.0.1", ":::", ":::1")
```



The screenshot shows a network monitoring tool interface. On the left, there's a 'Search Catalog' field and a 'Browse' button. Below it, a 'Custom SQL' field contains a query. On the right, a table lists processes with columns for PID, process name, command line, and IP address.

PID	Process Name	Command Line	IP Address
4			169.254.65.122
4			169.254.65.122
1436	C:\Windows\system32\svchost.exe -k LocalServiceNetworkRestricted -p -s Dhcp		0.0.0.0
1616	C:\Windows\System32\svchost.exe -k NetworkService -p -s NlaSvc		127.0.0.1
2216	C:\Windows\system32\svchost.exe -k NetworkService -p -s Dnscache		0.0.0.0
2216	C:\Windows\system32\svchost.exe -k NetworkService -p -s Dnscache		0.0.0.0
2216	C:\Windows\system32\svchost.exe -k NetworkService -p -s Dnscache		::
2216	C:\Windows\system32\svchost.exe -k NetworkService -p -s Dnscache		::

Once you have the Packet Identifier (PID) list, you can add it accordingly in the Custom Queries.

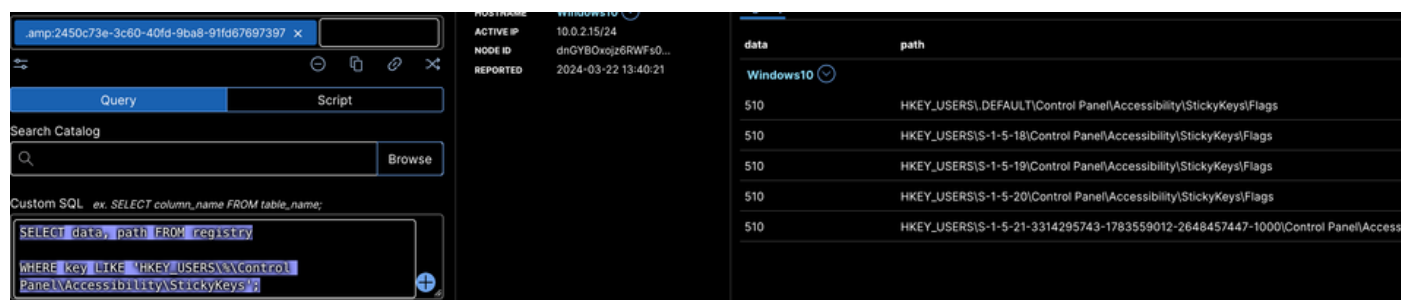
```
SELECT DISTINCT p.name, p.cmdline, pos.pid, pos.local_address, pos.local_port, pos.remote_address, pos.
FROM processes p JOIN process_open_sockets pos USING (pid)
WHERE pos.remote_address NOT IN ("", "0.0.0.0", "127.0.0.1", ":::", ":::1") and p.uid=1436
```

5. Backup/Restore Registry Monitoring

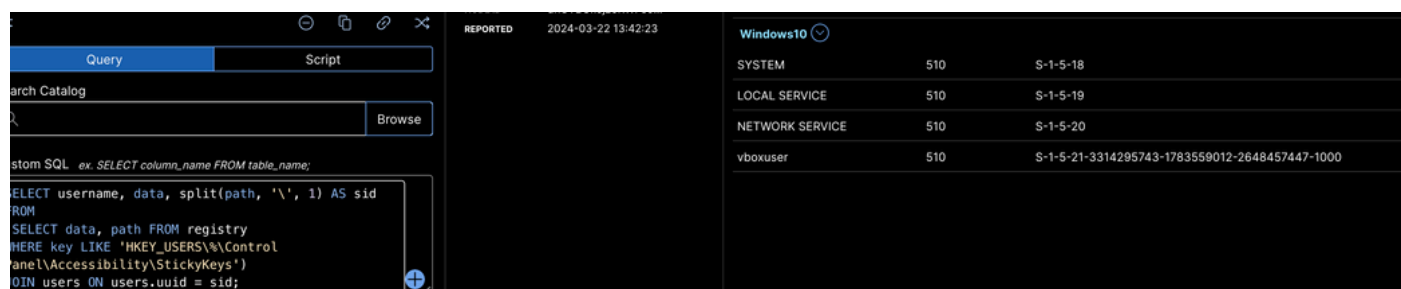
Tracking of events where changes are made to the Windows Registry through backup or restore operations. The Windows Registry is a hierarchical database that stores configuration settings and options on Microsoft Windows operating systems.

```
SELECT key AS reg_key, path, name, data, DATETIME(mtime, "unixepoch") as last_modified
FROM registry
WHERE key LIKE "HKEY_LOCAL_MACHINE\system\currentcontrolset\control\backuprestore\filesnottosnapshot";
```

```
SELECT data, path FROM registry
WHERE key LIKE 'HKEY_USERS\\%\Control Panel\Accessibility\StickyKeys';
```



```
SELECT username, data, split(path, '\', 1) AS sid
FROM
(SELECT data, path FROM registry
WHERE key LIKE 'HKEY_USERS\\%\Control Panel\Accessibility\StickyKeys')
JOIN users ON users.uid = sid;
```



6. File Search

Allows users to locate files and folders on their computer using various criteria such as file name, content, properties, or metadata.

```
SELECT
f.directory, f.filename, f.uid, f.gid,
f.mode, f.device, f.size,
f.atime,
f.mtime,
f.ctime,
f.btime,
f.hard_links, f.symlink, f.file_id, h.sha256
FROM file f
LEFT JOIN hash h on f.path=h.path
WHERE
f.path LIKE (SELECT v from __vars WHERE n="file_path") AND
f.path NOT LIKE (SELECT v from __vars WHERE n="not_file_path");
```

Navigate to **PARAMETERS > File Path** and click %.dll or %.exe or %.png.

Custom SQL ex. <code>SELECT column_name FROM table_name;</code>		C:\Windows\system32	CredentialEnrollmentManager.exe	2271478464	2271478464	-1
SELECT f.directory, f.filename, f.uid, f.gid, f.mode, f.device, f.size, f.atime, f.mtime,		C:\Windows\system32	CredentialUIBroker.exe	2271478464	2271478464	-1
Run Query Schedule Query		C:\Windows\system32	CustomInstallExec.exe	2271478464	2271478464	-1
File Search		C:\Windows\system32	DFDWiz.exe	2271478464	2271478464	-1
PARAMETERS ⓘ		C:\Windows\system32	DTUHandler.exe	2271478464	2271478464	-1
File Path		C:\Windows\system32	DWWIN.EXE	2271478464	2271478464	-1
Not File Path		C:\Windows\system32	DataExchangeHost.exe	2271478464	2271478464	-1
TEXT		C:\Windows\system32	DataStoreCacheDumpTool.exe	2271478464	2271478464	-1
		C:\Windows\system32	DataUsageLiveTileTask.exe	2271478464	2271478464	-1
		C:\Windows\system32	Defrag.exe	2271478464	2271478464	-1
		C:\Windows\system32	DeviceCensus.exe	2271478464	2271478464	-1

7. Powershell History Monitoring

Practice of keeping track of the commands that have been executed in PowerShell sessions. Monitoring PowerShell history can be particularly important for security and compliance reasons.

```
SELECT time, datetime, script_block_id, script_block_count, script_text, script_name, script_path
FROM orbital_powershell_events
ORDER BY datetime DESC
LIMIT 500;
```

Search Catalog		Set-ExecutionPolicy Bypass				
Custom SQL ex. <code>SELECT column_name FROM table_name;</code>		Set-ExecutionPolicy Bypass				
SELECT time, datetime, script_block_id, script_block_count, script_text, script_name, script_path FROM orbital_powershell_events ORDER BY datetime DESC LIMIT 500;		set -executionpolicy Bypass				
		Set-ExecutionPolicy Bypass				
		# Copyright © 2008, Microsoft Corporation. All rights reserved. #Common utility functions I...				
		# Copyright © 2008, Microsoft Corporation. All rights reserved. #Common utility functions I...				

8. Prefetch Query

Performance feature that speeds up the loading of applications. Prefetching involves analyzing the way software is loaded and run on a system and then storing information about this in specific files.

```
select datetime(last_run_time, "unixepoch", "UTC") as last_access_time,*
from prefetch
ORDER BY last_access_time DESC;
```

Search Catalog		2024-03-22 08:59:31	C:\Windows\Prefetch\FILECOAUTH.EXE-87F9F8AC.pf
Custom SQL ex. <code>SELECT column_name FROM table_name;</code>		2024-03-22 08:57:41	C:\Windows\Prefetch\SVCHOST.EXE-C5371482.pf
select datetime(last_run_time, "unixepoch", "UTC") as last_access_time,* from prefetch ORDER BY last_access_time DESC;		2024-03-22 08:50:15	C:\Windows\Prefetch\WMIPRVSE.EXE-43972D0F.pf
		2024-03-22 08:45:33	C:\Windows\Prefetch\SVCHOST.EXE-1616013E.pf
		2024-03-22 08:45:30	C:\Windows\Prefetch\MOUSOCOREWORKER.EXE-8C0B73B1.pf
		2024-03-22 08:45:30	C:\Windows\Prefetch\SVCHOST.EXE-C157FE85.pf
		2024-03-22 08:44:59	C:\Windows\Prefetch\WMIAPSRV.EXE-576286C3.pf

Prefetch is a mechanism with which SQL Server can fire up many I/O requests in parallel for a Nested Loop join.

9. Address Resolution Protocol (ARP) Cache Inspection

Involves examining the contents of the ARP cache on a computer or network device. The ARP cache is a table that stores mappings between IP addresses and their corresponding MAC addresses.

```
SELECT address, mac, count(*) as count
FROM arp_cache GROUP BY mac,address;
```

Search Catalog		Custom SQL ex. <i>SELECT column_name FROM table_name;</i>	
<input type="text"/>		<input type="text" value="SELECT address, mac, count(*) as count FROM arp_cache GROUP BY mac,address"/>	

224.0.0.251	01:00:5E-00:00:FB	2
224.0.0.252	01:00:5E-00:00:FC	2
239.255.255.250	01:00:5E-7F:FF:FA	2
10.0.2.2	52:54:00:12:35:02	1
10.0.2.255	FF:FF:FF:FF:FF:FF	1
169.254.255.255	FF:FF:FF:FF:FF:FF	1

The next example figures out the Suspicious MAC address and its count from the ARP Cache.

```
SELECT address, mac, count(*) as count
FROM arp_cache GROUP BY mac,address
HAVING COUNT(mac) >= (SELECT count FROM arp_cache WHERE count>=1)
AND mac LIKE (SELECT mac FROM arp_cache WHERE mac="52:54:00:12:35:02");
```

HOSTNAME Windows10		arp_cache		
ACTIVE IP 10.0.2.15/24				
NODE ID dnGYBOxojz6RWFs0...				
REPORTED 2024-03-22 14:21:02				
		address	mac	count
		Windows10		
		10.0.2.2	52:54:00:12:35:02	1