Uncovering Persistence with Osquery

Queries compatible with all supported platforms unless otherwise noted.

ATT&CK Technique & ID	Questions to Ask & Misc Notes	Example Query

Create Account - T1136	**Any recent, abnormal local users?**
	The two WHERE clauses help filter down results.
		SELECT uid,username,shell,directory FROM users
		WHERE type = ‘local’; → Windows Domain Joined systems
		WHERE shell NOT LIKE ‘%/bin/false’; → MacOS & Linux

Create Account - T1136	**What users have administrative privileges?**
	Default Admin group IDs:
	Windows [Administrators] = 544
	MacOS [admin] = 80
	Ubuntu Linux [sudo, root] =27,0
		SELECT users.uid,users.username,users.shell FROM user_groups
		INNER JOIN users ON user_groups.uid = users.uid
		WHERE user_groups.gid = @groupid;

New Service - T1050 ⊞	**Any abnormal services?**
	Only displays services that are set to auto start, and filters out legit svchost services.
		SELECT name,display_name,user_account,path FROM services
		WHERE start_type = ‘AUTO_START’
		AND path NOT LIKE ‘C:\Windows\system32\svchost.exe -k %’;

Daemons - T1160
Agents - T1159
	Any abnormal Daemons or Agents?
	If using osqueryi, may need to change the output mode as these columns will have very lengthy strings.
		SELECT name,program,program_arguments
		FROM launchd WHERE disabled != 1
		AND run_at_load = 1;

Scheduled Task - T1053 ⊞	**Any abnormal tasks?**
	May need to add ‘path’ column for further context.
		SELECT hidden,name,action
		FROM scheduled_tasks WHERE enabled = 1;

Local Job Scheduling - T1168 🐧	**Any abnormal jobs?**	SELECT minute,hour,path,command
		FROM crontab;

User Login/Startup Items - T1165	**Any startup items?**
	Lots of stuff can be filtered out. for eg:
	Windows = desktop.ini for each user profile
		SELECT name,path,source,status,username
		FROM startup_items;

Browser Extensions - T1176	**Any abnormal extensions?**
	Joined with the users table, to get the username;
	Useful to filter for all extensions for a particular user.
		SELECT users.username,chrome_extensions.name,
		chrome_extensions.identifier,chrome_extensions.path
		FROM users CROSS JOIN chrome_extensions USING (uid);

Browser Extensions - T1176	**Any abnormal extension identifiers?** Fuzzy search for extension name and compare against known good identifier/s.	SELECT users.username,chrome_extensions.name FROM users CROSS JOIN chrome_extensions USING (uid) WHERE name LIKE ‘%lastpass%’ AND identifier <> ‘hdokiejnpimakedhajhdlcegeplioahd’;

Application Shimming - T1138 ⊞	**Any suspicious entries in the AppCompat shims?**
	Web searching the SDB ID can provide lots of context to decide whether the shim is legitimate or not.
		SELECT executable,path,description,sdb_id
		FROM appcompat_shims;

Once an intruder gains an initial foothold on a system, they will need to establish some type of persistence so that they can return to the system even after it has been restarted. There are many different techniques to accomplish this - the chart above outlines some of the most common, as well as how to uncover them using osquery.

Feedback? Josh@DefensiveDepth.com

LearnOsquery.com

NetworkDefense.io

Process Interrogation with Osquery

Queries compatible with all supported platforms unless otherwise noted.

Process Attribute	Questions to Ask & Misc Notes	Example Query

Resource Usage	Abnormal CPU or Memory usage?	SELECT pid,name,user_time,system_time,resident_size FROM processes
		ORDER BY user_time; -- Time spent in Userspace
		ORDER BY system_time; -- Time spent in Kernel
		ORDER BY resident_size -- Private memory

Binary Name	Review binary names for misspellings of key processes. (eg scvhost instead of svchost)	SELECT pid,nameFROM processes
		WHERE name like 's%host.exe';

Path	Review paths for suspicious executions (eg /tmp) or abnormal paths for certain binaries (lsass outside of system32)	SELECT pid,name,path FROM processes
		WHERE name like 'l%a%s.exe';

Command Line Arguments	Review command line arguments for abnormalities (eg svchost without a valid -k switch)	SELECT pid,name,path,cmdline FROM processes
		WHERE name like 's%host.exe';

Parent Name & Path	Review Parent Process & Path for abnormalities (eg lsass parent process should be wininit.exe)	SELECT proc.pid, proc.name, proc.path,
		parent.name AS parentname, parent.path AS parentpath
		FROM processes proc, processes parent
		WHERE parent.pid = proc.parent AND proc.pid = @TargetPID;

Listening Ports	Review listening ports for suspicious listening process/ports (eg svchost listening on TCP/8080)	SELECT DISTINCT processes.name, processes.path, listening_ports.port
		FROM listening_ports
		JOIN processes USING (pid)
		WHERE listening_ports.family = 2 – Filters out IPv6
		AND listening_ports.address <> '127.0.0.1';

Examination of running processes can reveal much when trying to understand what is happening on a suspect system. Use the chart above to gain a better understanding of how to utilize osquery to slice and dice the processes on your system, looking for suspicious activity. The example queries focus on a modern Windows system and a few of its key system processes – svchost.exe and lsass.exe.

Feedback? Josh@DefensiveDepth.com

LearnOsquery.com

NetworkDefense.io