VT LIVEHUNT CHEAT SHEET

VirusTotal HUNTING

VirusTotal provides to malware researchers two hunting services based on Yara rules:

Livehunt - (Future): Continuously scans incoming samples, notifying you of files matching your rules. Ideal for monitoring ongoing campaigns, tracking leaked data, and ensuring brand protection.

Retrohunt - (Past): Investigates a year-long dataset of 500+ million files, and reports those matching your rules. Ideal for precise attribution and uncovering historical activity.

The VT module

The YARA VT module was created specially for VirusTotal Hunting services to operate with all available VT context data:

Metadata (AVs rate, ExifTool, submissions, type, signature, VT tags, etc)

Behaviour (network, file system, registry, sandbox verdict, Android/Windows specific, etc)

METADATA

FIELDS USAGE EXAMPLES

Files submitted to VirusTotal for the first time.	vt.metadata.new_file
Files detected by 10 or more antivirus engines.	vt.metadata.analysis_stats.malicious >= 10
LNK files that are executing PowerShell from Exif CommandLineArguments metadata field.	vt.metadata.exiftool["CommandLineArguments"] icontains "powershell"
File's name as it was last submitted to VirusTotal.	vt.metadata.file_name icontains "invoice"
File with a size greater than 100KB.	vt.metadata.file_size > 100KB
Files that are DLL executables. Full types list here.	vt.metadata.file_type == vt.FileType.PE_DLL
Files downloaded in the wild from discordapp.com.	vt.metadata.itw.url.raw startswith “https://cdn.discordapp.com"
Files containing “cmd.exe” in the Content-Disposition header, possible download of a file with that name in the wild.	vt.metadata.itw.url.response_headers["Content-Disposition"] icontains "cmd.exe"
JARM fingerprint of the domain where the file was downloaded in the wild.	vt.metadata.itw.domain.jarm == "2ad2ad16d00000022c2ad2ad2ad2adc048c697e0d6d0c91c6bf49b0695f45c"
Files that contains the ChatGPT logo as an icon to impersonate. The dhash is a hash of the image.	vt.metadata.main_icon.dhash == "b2f1d4eaa8d470b2"
Files associated with the "Redline" family and containing configuration data extracted from Mandiant Backscatter.	for any family_name in vt.metadata.malware_families: (family_name == "redline")
Downloads occurring in the wild, where the IP's ASN is associated with the REGRU-RU provider.	vt.metadata.itw.ip.ip_as_owner == "REGRU-RU"
Files that contains the keyword ransom in the Kaspersky Antivirus engine.	for any engine, signature in vt.metadata.signatures: (engine == "Kaspersky" and signature icontains "ransom")
Files having specific imphash.	vt.metadata.imphash == "85f60041d9bc9a44bdc4c312071802cb"
Files that were submitted from Russia.	vt.metadata.submitter.country == "RU"
Files that exploit CVE-2023-38831.	for any tag in vt.metadata.tags: (tag == "cve-2023-38831")
Files that are downloaded in the wild from an URL with the tag downloads-pe, which means that is downloading a PE executable.	for any itw_url_tag in vt.metadata.itw.url.tags: (itw_url_tag == "downloads-pe")

See the full list of metadata fields here

Livehunt rule: malicious docx files with macros

```
rule malicious_docx_macros {
meta:
 description = “Malicious documents using macros”
condition:
 vt.metadata.file_type == vt.FileType.DOCX and
 vt.metadata.analysis_stats.malicious > 5  and
 for any tag in vt.metadata.tags:(tag == “macros”)
}
```

Livehunt rule: files with CLI powershell execution

```
rule LNK_metadata_execution_powershell {
meta:
 description = “Possible LNK execution through
 CommandLineArguments Exif metadata field”
condition:
 vt.metadata.exiftool["CommandLineArguments"]
 icontains "powershell" or
 vt.metadata.exiftool["RelativePath"] icontains
 "powershell"
}
```

virustotal.com twitter.com/virustotal linkedin.com/company/virustotal

VIRUSTOTAL

VT LIVEHUNT CHEAT SHEET

BEHAVIOUR

FIELDS USAGE EXAMPLES

Registry key created in RunOnce with the name “OperaSetups” for possible persistence.	for any registry_key in vt.behaviour.registry_keys_set : (registry_key.key icontains "\\CurrentVersion\\RunOnce\\OperaSetups")
Value of the registry key which contains a VBS file stored in \AppData\Roaming\ path.	for any registry_value in vt.behaviour.registry_keys_set : (registry_value.value icontains "\\AppData\\Roaming\\" and registry_value.value endswith ".vbs")
Files with RANSOM verdict extracted from the behaviour.	for any v in vt.behaviour.verdicts : (v == vt.BehaviourVerdict.RANSOM)
Files with CryptoCurrencyMiner verdict label by sandboxes.	for any label in vt.behaviour.verdicts_labels : (label contains "CryptoCurrencyMiner")
Files that are doing Process Injection (T1055).	for any technique in vt.behaviour.mitre_attack_techniques : (technique.id == "t1055")
Files that have in memory URLs and the pattern /recovery/store.php? in the path.	for any memory_url in vt.behaviour.memory_pattern_urls : (memory_url icontains "/recovery/store.php?")
Samples that have created \AppData\Local\Temp\coeghglefsmk.sys during execution.	for any files_written in vt.behaviour.files_written : (files_written icontains "\\AppData\\Local\\Temp\\coeghglefsmk.sys")
DNS resolutions against us-zephyr.miningocean.org during execution.	for any dns_lookups in vt.behaviour.dns_lookups : (dns_lookups.hostname == "us-zephyr.miningocean.org")
Samples that have this particular Sigma rule identified during the execution. Full list of Sigma rules.	for any sigma in vt.behaviour.sigma_analysis_results : (sigma.rule_id == "8b5db9da5732dc549b0e8b56fe5933d7c95ed760f3ac20568ab95347ef8c5bcc")
Samples communicating with 5432 port.	for any com in vt.behaviour.ip_traffic : (com.destination_port == 5432)
Files executing cmd.exe with specific params.	for any cmd in vt.behaviour.command_executions : (cmd icontains "cmd.exe /Q /c")
Files creating the HGL345 mutex during execution.	for any mutex in vt.behaviour.mutexes_created : (mutex icontains "HGL345")
Files creating a service called eckwIIMB during execution.	for any svc in vt.behaviour.services_created : (svc == "eckwIIMB")
Strings that are either encoded or decoded. Possible configuration identified of the sample.	for any text_decoded in vt.behaviour.text_decoded: (text_decoded icontains "\"Ports\"" and text_decoded icontains "\"Server\"")
Samples containing specific string related to DDNS service in the IDS alert generated by the sample.	for any ids_alert in vt.behaviour.ids_alerts: (ids_alert.rule_msg icontains "dyndns.org")
Suspicious API calls identified during the execution of the sample. Possible process injection.	for any api_call in vt.behaviour.calls_highlighted: (api_call == "ResumeThread" and api_call == "SetThreadContext" and api_call == "WriteProcessMemory" and api_call == "VirtualAllocEx" and api_call == "SuspendThread")
Processes in which some kind of code was injected.	for any p_injected in vt.behaviour.processes_injected: (p_injected == "powershell.exe")
HTTP communications using OPTION method. List of available methods.	for any conv in vt.behaviour.http_conversations : (conv.request_method == vt.Http.Method.OPTIONS)
Files that contains the string “Moxilla” in the User-Agent header during communications.	for any conv in vt.behaviour.http_conversations : (conv.request_headers["user-agent"] == "Moxilla")

See the full list of behaviour fields here

Livehunt rule: files modifying Registry Run Keys for persistence

```
rule persistence_runonce_vbs {
meta:
 description = “Detect persistence by establishing a VBS
 file in the runonce key”
condition:
 for any registry_key in
  vt.behaviour.registry_keys_set: (registry_key.key
  icontains "\\CurrentVersion\\RunOnce\\") and
  registry_key.value icontains “.vbs”
}
```

Livehunt rule: Files created within the "profile.d" directory to maintain persistence.

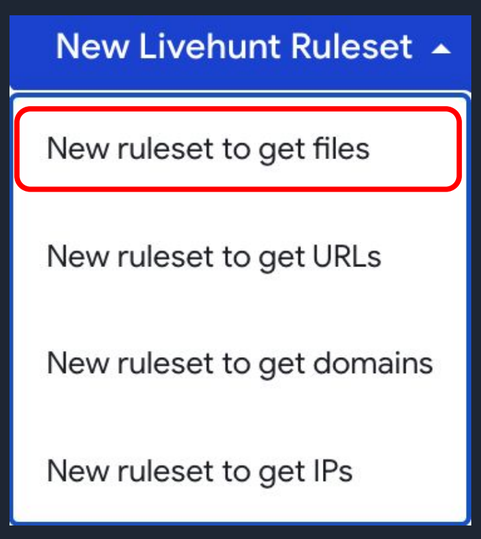
```
rule profile_folder_shell_script {
meta:
 description = “Detects Linux shell scripts creation in
 profile.d path”
condition:
 for any dropped in vt.behaviour.files_dropped :
  (dropped.path contains "/etc/profile.d/"
  and dropped.path endswith ".sh") or
 for any file_path in vt.behaviour.files_written :
  (file_path contains "/etc/profile.d/"
  and (file_path endswith ".sh") )
}
```

VIRUSTOTAL

VT LIVEHUNT CHEAT SHEET

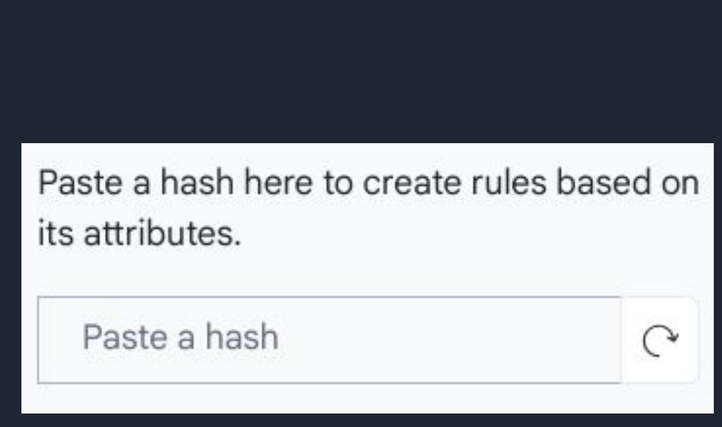
HUNTING WITH STRUCTURE

A simple way to create a Yara rule for Hunting is to use the Structure panel. Navigate through the attributes in the VT module and click on the values to populate a Yara rule condition. You can select metadata and behavior attributes and start using information related to signatures, processes, registry, files, network traffic and more.

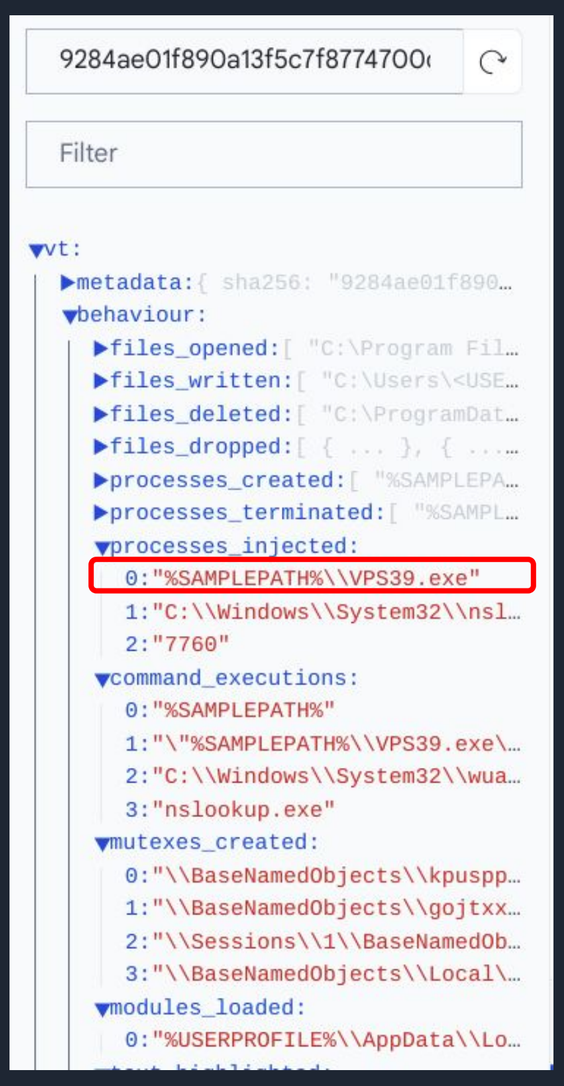


Step 1:
Create a new Livehunt rule for files

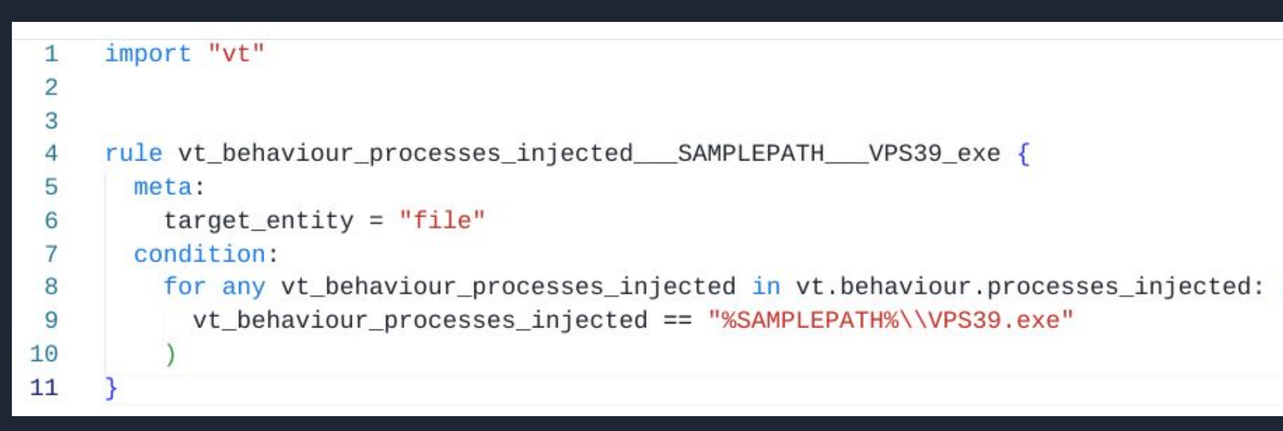
Step 2:
Click “Structure” Icon on the left menu



Step 3:
Paste the hash of a file that match a specific metadata or behaviour field that you are looking for your rule.

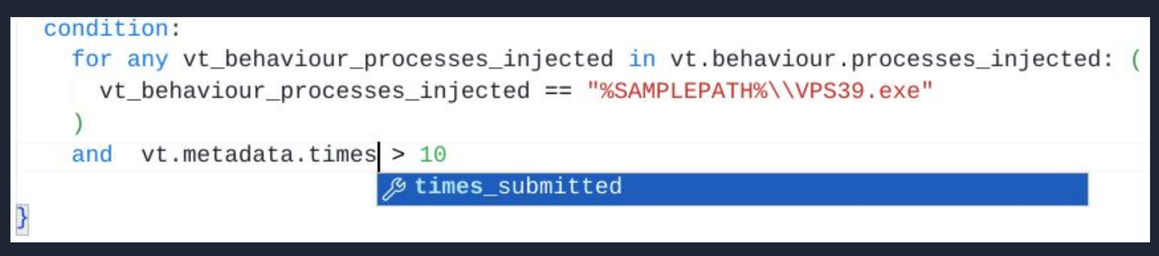


Step 4:
Optionally you can now use the filter view to find the attribute of the report you are interested in to create your Livehunt rule or navigate and click directly on the attribute.



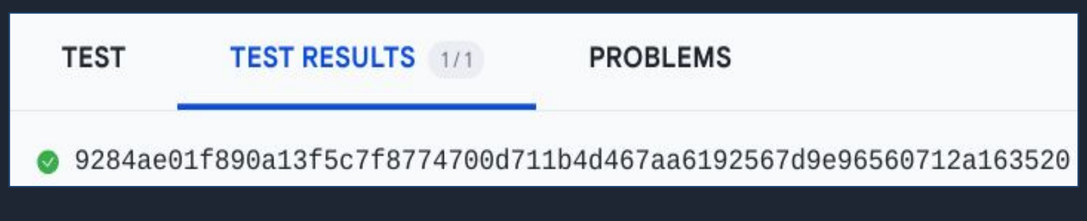
```
1  import "vt"
2
3
4  rule vt_behaviour_processes_injected___SAMPLEPATH___VPS39_exe {
5    meta:
6      target_entity = "file"
7    condition:
8      for any vt_behaviour_processes_injected in vt.behaviour.processes_injected: (
9        vt_behaviour_processes_injected == "%SAMPLEPATH%\\VPS39.exe"
10     )
11 }
```

Step 5:
You will have automatically your rule based on that condition you selected.



```
  condition:
    for any vt_behaviour_processes_injected in vt.behaviour.processes_injected: (
      vt_behaviour_processes_injected == "%SAMPLEPATH%\\VPS39.exe"
    )
    and  vt.metadata.times > 10
                           times_submitted
}
```

Step 6:
Customize your rule by adding more conditions. The visual editor offers auto-complete to guide you through available options.



Step 7:
Test your rule by providing more hashes that should match. This will confirm if it's working correctly before putting rule into production.

Download here: virustotal.com/go/livehunt-cheatsheet

virustotal.com twitter.com/virustotal linkedin.com/company/virustotal