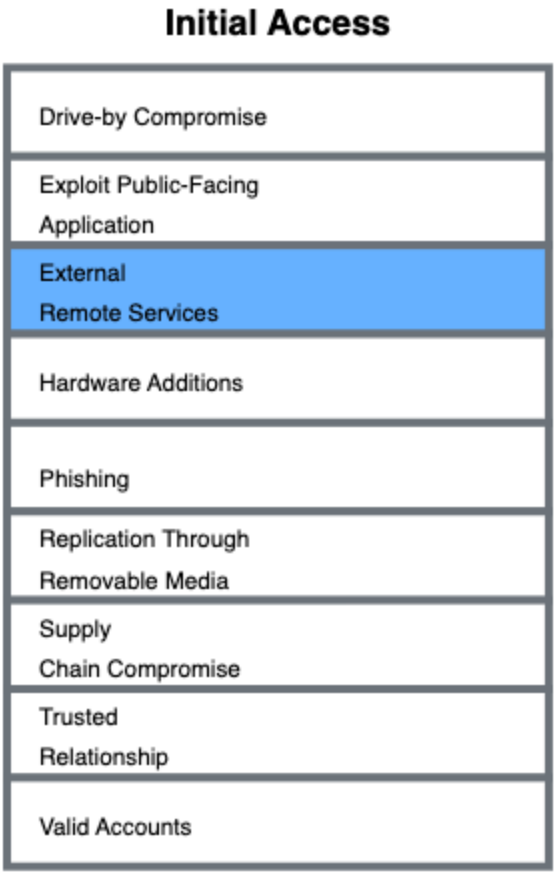
Cloud Adversary TTPs

TeamTNT is a threat group active since Oct. 2019, and is one of the only APT groups to focus efforts primarily on cloud and containerized environments. Due to the cloud focus, we will be modeling our simulated attack after their efforts. Please note that additional TTPs will be added in that have not been seen by TeamTNT.

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Discovery	Lateral Movement	Command and Control	Impact
Exploit Public-Facing Application	Scripting	Kernel Modules and Extensions	Exploitation for Privilege Escalation	Obfuscated Files or Information	System Network Configuration Discovery	Exploitation of Remote Services	Standard Application Layer Protocol	Malware Detected
	Third-party Software	Local Job Scheduling		System Information Discovery	File and Directory Discovery		Direct Communication with an Explicit IP	Resource Hijacking
				Data Encoding	Network Service Scanning		Web Service	
				Hidden File System	Security Software Discovery		Standard Encoding	
				Disable or Modify Tools	System Information Discovery			
				Disable or Modify System Firewall				
				Clear Command History				
				File Deletion				
				Masquerading				
				File and Directory Permissions Modification				

- Initial Access
    - Cetus - Docker Worm
    - Black-T - Linux Malware
    - Hildegard - Kubernetes Malware
    - Takeaways:
- Execution
    - Takeaways:
- Persistence
    - Takeaways:
- Privilege Escalation
    - Takeaways:
- Defense Evasion
    - Takeaways:
- Credential Access
- Discovery
- Lateral Movement
- Collection
- Command and Control
- Exfiltration
- Resources

Initial Access



For Initial Access, TeamTNT uses masscan to seek exposed Docker API ports and Kubernetes Kubelets which allow anonymous access to the kubelet.

Cetus - Docker Worm

This worm targets exposed Docker daemon APIs, then impersonates Portainer, deploys Docker containers to mine Monero, then scans both internal and external Docker daemon instances to spread.

Black-T - Linux Malware

Same initial access of targeting exposed Docker daemon APIs, but they also add additional secondary scanning techniques and collect AWS credential files from compromised hosts.

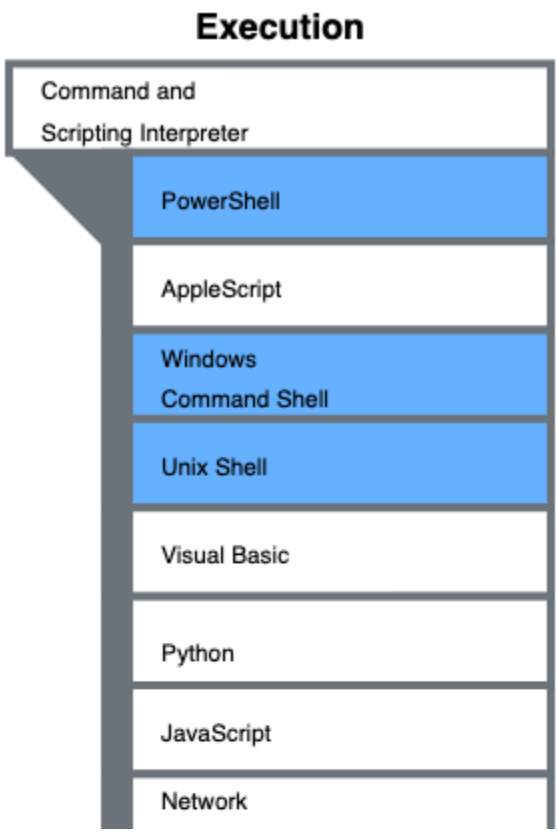
Hildegard - Kubernetes Malware

Azure Kubernetes Service (AKS) is claimed to enforce proper authentication by default, but the default standard Kubernetes config allows anonymous access.

Takeaways:

- Misconfigurations and exposed APIs are a common entry path. Can we develop analytics alerting on AKS API requests and Docker on Azure? API requests coming from anything other than the few “normal” development or DevOps rollout machines would be good to alert on, but I don’t know how general the analytics would be.
- Malware often tries to hide itself by impersonating the management software (i.e. Portainer) If we focus one layer up on the Docker on Azure or AKS, are we more likely to be able to monitor new images being downloaded or containers being provisioned?
- Most of these seem to be focused on T1133 - External Remote Services. I think that’s a solid way to go for our exercise.

Execution



Execution was mostly covered in the previous section, so I’ll be brief. The majority of the execution efforts are focusing on T1609 - Container Administration Command and T1610 - Deploy container. Once any of the worms are able to gain remote access to the Docker or Kubernetes API via remote API calls, they download and deploy XMRig to mine the XMR cryptocurrency.

T1204.003 - User Execution: Malicious Image was another TTP that was used for additional execution. TeamTNT appeared to have collected credentials to an account on DockerHub when the credentials were accidentally committed to GitHub. TeamTNT then hosted several malicious images on this account.

Takeaways:

- Cryptomining is big here. Can we develop analytics that alert on cryptomining? Common protocols or outgoing ports?
- Malicious Images are the other focus here. I know that version pinning is a good practice and defends against some of these attacks. Are there analytics that we can develop that could monitor dependency version numbers? (A common attack is to host a really high version number on public repos, so when automated pipelines check for “latest” version, they see the malicious one is version 9999, which is newer than version 2.3, for example.)

Persistence

For persistence, the majority of the observed TTPs were related to adding new accounts or keys to remote authentication services. (i.e. adding SSH keys to authorized keys on compromised hosts, creating local privileged accounts on compromised hosts, etc.) They also add crypto miners to systemd services and Windows services to persist through reboots.

Takeaways:

- Persistence via cloud does not seem to be a priority for TeamTNT. Many of the techniques used to persist already have host-based analytics. This makes sense as a worm would be more focused on spreading quickly and getting as many nodes mining as possible.
- I am adding the following TTPs to our pool to draw from for our story, regardless of what TeamTNT does. They just seem to fit really well as a potential cloud focused actor:
- T1136.003 - Cloud Account
- T1098.003 - Account Manipulation: Add Office 365 Global Administrator Role
- T1098.001 - Account Manipulation: Additional Cloud Credentials

Privilege Escalation

Again, Privilege Escalation does not seem to be as much of a priority for TeamTNT. Their main goal seems to be cryptomining, so as long as they can get the miners working, they wouldn't likely need to seek Domain Admin permissions. Even the use of a container escape to host seems mainly to be there to just give the attackers more options.

T1547.001 - Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder

T1543.002 - Create or Modify System Process: Systemd Service

T1543.003 - Create or Modify System Process: Windows Service

T1611 - Escape to Host

Takeaways:

- We are unlikely to be able to find cloud-focused Privilege Escalation techniques in the wild. Most of TeamTNT's operation is not cloud-focused by this point in their cyber kill chain.
- I'm adding the following techniques from the ATT&CK Cloud Matrix to our pool:
- T1484 - Domain Policy Modification
- T1078 - Valid Accounts
- Focusing on analytics coming from the Azure AD login seems like the way to go forward. Container escape might be a stretch goal, but it's less cloud-focused. Happy to discuss further.

Defense Evasion

T1610 - Deploy Container

T1222.002 - File and Directory Permissions Modification: Linux and Mac File and Directory Permissions Modification

T1562.001 - Impair Defenses: Disable or Modify Tools

T1562.004 - Impair Defenses: Disable or Modify System Firewall

T1070.002 - Indicator Removal on Host: Clear Linux or Mac System Logs

T1070.003 - Indicator Removal on Host: Clear Command History

T1070.004 - Indicator Removal on Host: File Deletion

T1014 - Rootkit

T1027.002 - Obfuscated Files or Information: Software Packing

Takeaways:

- I am adding the following TTPs to our pool to draw from for our story, regardless of what TeamTNT does. They just seem to fit really well as a potential cloud focused actor:
  - T1484 - Domain Policy Modification
  - T1564 - Hide Artifacts
  - T1578 - Modify Cloud Compute Infrastructure
  - T1078 - Valid Accounts

Credential Access

T1552.005 - Unsecured Credentials: Cloud Instance Metadata API

T1552.001 - Unsecured Credentials: Credentials In Files

T1552.004 - Unsecured Credentials: Private Keys

Discovery

Discovery seems like it might be a rich category for analytics. My working theory is that admins with familiarity with their cloud infrastructure (Or those cool enough to have DevOps'd away their need to interact with much of the infrastructure directly) will be less likely to trigger the discovery alerts than attackers trying to take inventory. Most of TeamTNT's discovery seems to take place after the initial compromise, so they solidly fall into more of the enterprise environment focus rather than Azure, so I will be adding quite a few additional things to look at from the cloud matrix.

T1613 - Container and Resource Discovery

T1046 - Network Service Scanning

T1518.001 - Software Discovery: Security Software Discovery

T1016 - System Network Configuration Discovery

T1049 - System Network Connections Discovery

Additionally, I am adding in the following from the Cloud Matrix:

- T1580 - Cloud Infrastructure Discovery
- T1538 - Cloud Service Dashboard
- T1526 - Cloud Service Discovery
- T1619 - Cloud Storage Object Discovery
- T1201 - Password Policy Discovery
- T1069 - Permission Groups Discovery

Lateral Movement

The only technique used by TeamTNT for lateral movement is SSH. For our purposes, I don't see this helping much for cloud-specific analytics.

I am adding:

- T1080 - Taint Shared Content

From the Cloud matrix. The other 2 techniques don't seem to relate to Azure. We should evaluate:

- T1550 - Use Alternate Authentication Material

For Google. There are alternate authentication methods potentially available. Trying to log in using an auth token might be a good thing to look for.

Collection

Nothing to report from TeamTNT.

This does warrant a discussion about if there is a good way to be able to monitor Cloud repositories. I'm thinking this might have a lot of false positives, and I can't think of a good way to do this. Happy to be proven wrong here.

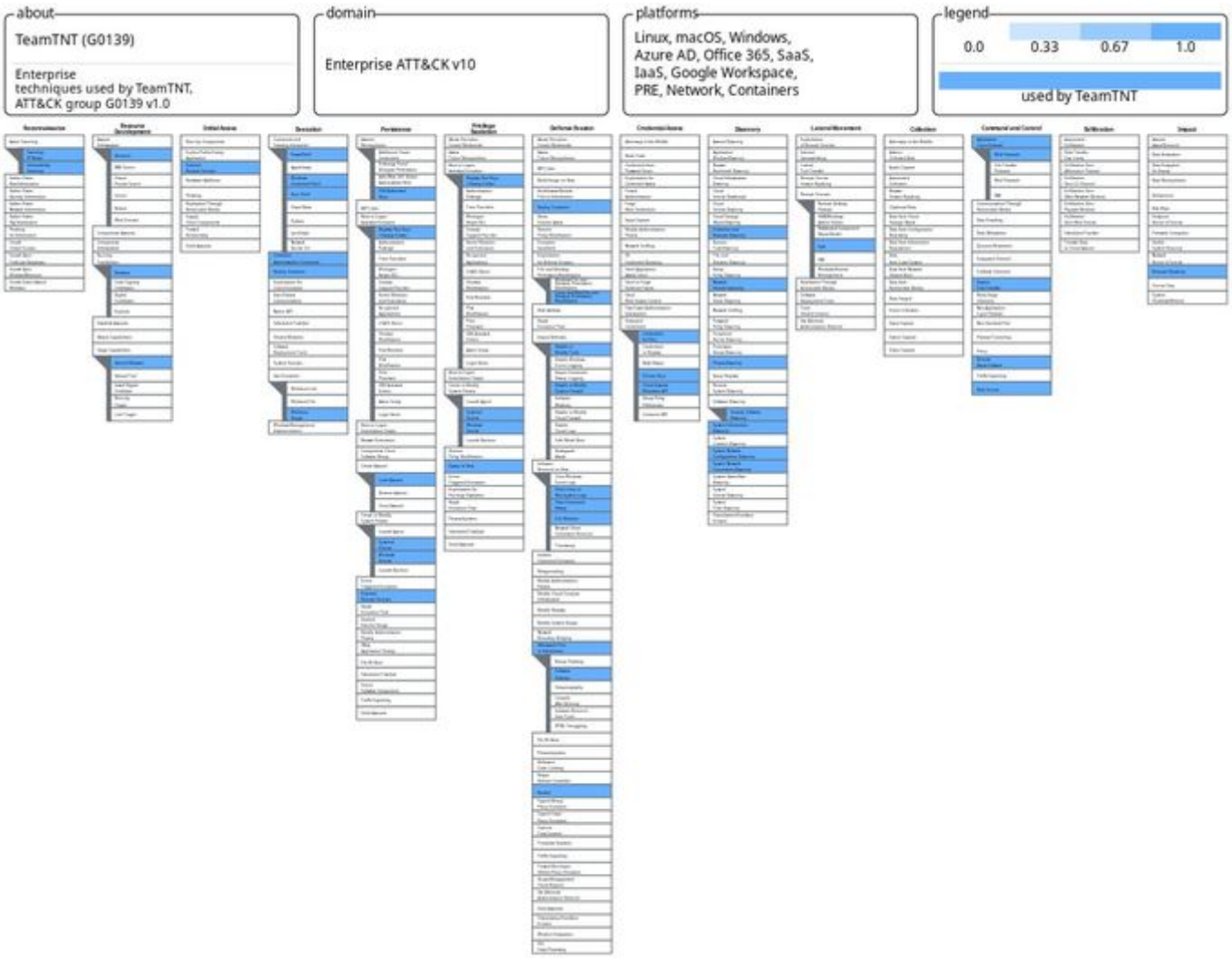
Command and Control

C2 for TeamTNT seems like mostly Curl and IRC. I don't know that any of this is relevant to Cloud analytics. Nothing is Azure specific. There is not even a C2 category in the Cloud matrix, so unless there is a good reason to include this, my thought is that we ignore this.

Exfiltration

Nothing to report from TeamTNT.

Same point as before with the Collection category. If we can monitor cloud repos, we might be able to write some analytics. Otherwise, I think we should ignore this category as well.



Resources

MITRE ATT&CK APT Group Overview - https://attack.mitre.org/groups/G0139/

Trend Micro Whitepaper - https://documents.trendmicro.com/assets/white_papers/wp-tracking-the-activities-of-teamTNT.pdf

TeamTNT on Malpedia - https://malpedia.caad.fkie.fraunhofer.de/actor/teamtnt

TeamTNT_(G0139) (2).xlsx

TeamTNT_(G0139).json