Talos Vulnerability Report

TALOS-2021-1247

Microsoft Azure Sphere mount namespace unsigned code execution vulnerability

APRIL 13, 2021

CVE NUMBER

CVE-2021-27074

Summary

An unsigned code execution vulnerability exists in the mount namespace functionality of Microsoft Azure Sphere 21.01. A specially crafted shellcode could allow an adversary to execute an arbitrary binary in a tmpfs mount, leading to unsigned code execution. An attacker can switch to a new mount namespace to trigger this vulnerability.

Tested Versions

Microsoft Azure Sphere 21.01

Product URLs

https://azure.microsoft.com/en-us/services/azure-sphere/

CVSSv3 Score

6.2 - CVSS:3.0/AV:L/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N

CWE

CWE-284 - Improper Access Control

Details

Microsoft's Azure Sphere is a platform for the development of internet-of-things applications. It features a custom SoC that consists of a set of cores that run both high-level and real-time applications, enforces security and manages encryption (among other functions). The high-level applications execute on a custom Linux-based OS, with several modifications to make it smaller and more secure, specifically for IoT applications.

A namespace is an abstraction provided by the Linux kernel that allows for modifying the execution context of a process (or thread).
Currently, there exist 8 kinds of namespaces: Cgroup, IPC, Network, Mount, PID, Time, User, UTS. An unprivileged user can create a new user namespace (using the `CLONE_NEWUSER` flag) and have a full capabilities (root user with all caps) in that namespace. From `man user_namespaces(7)`:

```
User namespaces isolate security-related identifiers and
attributes, in particular, user IDs and group IDs (see
credentials(7)), the root directory, keys (see keyrings(7)), and
capabilities (see capabilities(7)).  A process's user and group
IDs can be different inside and outside a user namespace.  In
particular, a process can have a normal unprivileged user ID
outside a user namespace while at the same time having a user ID
of 0 inside the namespace; in other words, the process has full
privileges for operations inside the user namespace, but is
unprivileged for operations outside the namespace.
```

Once in the new user namespace, the user has the `CAP_SYS_ADMIN` capability in the namespace. This means that it's possible to use the `CLONE_NEWNS` flag to switch to a new mount namespace. From `man mount_namespaces(7)`:

```
Mount namespaces provide isolation of the list of mount points
seen by the processes in each namespace instance.  Thus, the
processes in each of the mount namespace instances will see
distinct single-directory hierarchies.
...
Subsequent modifications to the mount point list (mount(2) and
umount(2)) in either mount namespace will not (by default) affect
the mount point list seen in the other namespace (but see the
following discussion of shared subtrees).
```

For more details on namespaces, see the `man namespaces(7)`.

Since the user can modify the mount point list in the new namespace, a subset of the usual mount operations is available.
For example, it's not possible to mount an existing block device (e.g. `/dev/mtdblock0`) in the new namespace, since that requires root privileges on the parent namespace. It is however possible to create a new `tmpfs` mount within the new namespace, since that won't affect the parent namespace.

One of the security features provided by Azure Sphere is the protection against unsigned code: only the code already present in the device, or signed code that has been deployed to the device via the cloud, and marked as executable can ever be executed. This is enforced by Linux kernel patches around `mprotect` and `mmap` that make sure that memory that has ever been writeable can't be executable. Moreover, this is enforced at the userspace level by ensuring all executable mountpoints are not writeable and conversely that all writeable mountpoints are not executable.

By using namespaces however, it’s possible to alter this situation. The `CLONE_NEWUSER` and `CLONE_NEWNS` flags can be used together in a single invokation of the `unshare` syscall to change the current (unprivileged) namespace. Next, it’s possible to mount a `tmpfs` filesystem anywhere (e.g. to `/tmp`), create a binary file inside the mountpoint with arbitrary contents, and execute it.

Timeline

2021-02-02 - Vendor Disclosure

2021-04-13 - Public Release

CREDIT

Discovered by Claudio Bozzato and Lilith >_> of Cisco Talos.

VULNERABILITY REPORTS

PREVIOUS REPORT

TALOS-2020-1202

NEXT REPORT

TALOS-2021-1250

© 2022 Cisco Systems, Inc. and/or its affiliates. All rights reserved. View our Privacy Policy.