Vulnerabilities with blob verification

Moderate **cpanato** published **GHSA-8gw7-4j42-w388** on Sep 14

Package

cosign (sigstore)

Affected versions	Patched versions
<=1.11.1	1.12.0

Description

Summary

A number of vulnerabilities have been found in `cosign verify-blob`, where Cosign would successfully verify an artifact when verification should have failed.

Vulnerability 1: Bundle mismatch causes invalid verification.

Summary

A cosign bundle can be crafted to successfully verify a blob even if the embedded rekorBundle does not reference the given signature.

Details

Cosign supports "bundles" which intend to allow offline verification of the signature and rekor inclusion. By using the --bundle flag in cosign sign-blob, cosign will create a JSON file called a "bundle". These bundles include three fields: base64Signature, cert, and rekorBundle. The desired behavior is that the verification of these bundles would:

- verify the provided blob using the included signature and certificate
- verify the rekorBundle SET
- verify the rekorBundle payload references the given artifact.

It appears that step three is not being performed, allowing "any old rekorBundle" to pass validation, even if the rekorBundle payload does not reference the provided blob or the certificate and signature in the rekorBundle do not match those at the top level.

Steps to reproduce

Enable keyless signing:

```
export COSIGN_EXPERIMENTAL=1
```

Create two random blobs:

```
dd bs=1 count=50 </dev/urandom >blob1
dd bs=1 count=50 </dev/urandom >blob2
```

Sign each blob:

```
cosign sign-blob blob1 --bundle bundle1
cosign sign-blob blob2 --bundle bundle2
```

Create a falsified bundle including the base64Signature and cert fields from bundle1 and the rekorBundle from bundle2:

```
jq --slurpfile bundle2 bundle2 '.rekorBundle = $bundle2[0].rekorBundle' bundle1 >
invalidBundle
```

Now, the falsified bundle can be used to verify blob1:

```
$ cosign verify-blob blob1 --bundle invalidBundle
tlog entry verified offline
Verified OK
```

Patches

Users should update to the latest version of Cosign, `1.12.0`.

Workaround

If you extract the signature and certificate from the `bundle`, you may use it for verification as follows and avoid using an invalid bundle:

```
$ cosign verify-blob blob1 --signature $(jq -r '.base64Signature' bundle1) --certificate $(jq
-r '.cert' bundle1)
```

Note that this will make a network call to Rekor to fetch the Rekor entry. However, you may then be subject to Vulnerability 4.

Vulnerability 2: Certificate Identities are not checked in some cases

Summary

When providing identity flags, the email and issuer of a certificate is not checked when verifying a Rekor bundle, and the GitHub Actions identity is never checked.

Details

Users who provide an offline Rekor bundle (`--bundle`) when verifying a blob using `cosign verify-blob` and include flags that check identity such as `--certificate-email` and `--certificate-oidc-issuer` are impacted. Additionally, users who provide the GitHub Actions verification flags such as `--certificate-github-workflow-name` when running `cosign verify-blob` without a bundle, key reference, or certificate are impacted.

When providing these flags, Cosign ignored their values. If a certificate's identity did not match the provided flags, Cosign would still successfully verify the blob.

Patches

Users should update to the latest version of Cosign, `1.12.0` .

Workarounds

There are no workarounds, users should update.

Vulnerability 3: Invalid Rekor bundle without the experimental flag will result in successful verification

Summary

Providing an invalid Rekor bundle without the experimental flag results in a successful verification.

Details

Users who provide an offline Rekor bundle (`--bundle`) that was invalid (invalid signed entry timestamp, expired certificate, or malformed) when verifying a blob with `cosign verify-blob` and do not set the `COSIGN_EXPERIMENTAL=1` flag are impacted.

When an invalid bundle was provided, Cosign would fallback to checking Rekor log inclusion by requesting proof of inclusion from the log. However, without the `COSIGN_EXPERIMENTAL` flag, Cosign would exit early and successfully verify the blob.

Patches

Users should update to the latest version of Cosign, `1.12.0`.

Workarounds

There are no workarounds, users should update.

Vulnerability 4: Invalid transparency log entry will result in successful verification

Summary

An invalid transparency log entry will result in immediate success for verification.

Details

Users who provide a signature and certificate to `verify-blob` will fetch the associated Rekor entry for verification. If the returned entry was invalid (invalid signed entry timestamp, invalid inclusion proof, malformed entry with missing verification), then `cosign` exits early and succeeds unconditionally.

Patches

Users should update to the latest version of Cosign, `1.12.0`.

Workarounds

There are no workarounds, users should update.

For more information

If you have any questions or comments about this advisory:

- Open an issue in cosign
- Send us a message on Slack.

Severity

Moderate 5.5 / 10

CVSS base metrics

Attack vector	**Local**
Attack complexity	**Low**
Privileges required	**Low**
User interaction	**None**
Scope	**Unchanged**
Confidentiality	**None**
Integrity	**High**
Availability	**None**

CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:N

CVE ID

CVE-2022-36056

Weaknesses

No CWEs

Credits

 **codysoyland**

 **asraa**

 **haydentherapper**