Richtlinie zum „Software-Sprint“

SiC – Signierte Container

Schlussbericht

Zuwendungsempfänger:

Lucas Humfeldt

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS23S01 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

Kurze Darstellung der Aufgabenstellung und Motivation

In den letzten Jahren hat sich gezeigt, dass die Cloud zukünftig nicht mehr wegzudenken ist. Gleiches gilt für den öffentlichen Sektor, dessen Digitalisierung gerade stattfindet. Zumindest im privaten Sektor hat sich gezeigt, dass die Absicherungen von Systemen, die in der Cloud betrieben werden, in vielen Fällen mangelhaft ist und teilweise grundsätzliche Best Practices im Bereich der IT-Sicherheit fehlen.

Viele dieser Konzepte sind komplex und lassen sich schwer ad-hoc umsetzen. Daher ist es bei der Digitalisierung der Verwaltung notwendig, vor bzw. während der Umsetzung Lösungsansätze mitzudenken. Vor allem dann, wenn keine einfache Insellösung, sondern eine föderale Lösung gewünscht ist.

Ziel dieses Projekts war es daher, den grundlegenden Pfeiler der Software-Supply-Chain-Sicherheit Software-Signaturen im Kontext von aktueller Cloud-Technologie zu untersuchen. Dieser Bereich wurde gewählt, da es für Software-Signaturen im Cloud-Bereich immer noch keine allgemein verbreitete Lösung gibt und eine Untersuchung daher wertvoll ist. Speziell sollte untersucht werden, inwiefern der vielversprechende Kandidat Sigstore in ein beispielhaftes Fachverfahren integriert werden kann. Hierzu wurden die folgenden Meilensteine zu Beginn des Projekts definiert:

1. Entwicklung eines MVP, der alle notwendigen Technologien abbildet
2. Integration von IRIS-Connect [d] in den MVP
3. Härtung der Infrastruktur nach gängiger Praxis, bspw.:
    a. BSI Grundschutz [e]
    b. BSI Container SYS 1.6 [f]
4. Inbetriebnahme des IRIS-Prototypen in einer Testumgebung
5. Aufforderung der Sicherheits-Community zum Pentest
6. Migration von extern genutzten Funktionalitäten zu selbst gehosteten Lösungen

Beitrag des Projektes zu den Zielen der Förderinitiative „Software-Sprint“

Das Konzept der Supply-Chain-Sicherheit taucht in der Öffentlichkeit meist nur in Zusammenhang mit großen Unternehmen auf. Diese Wahrnehmung ist gefährlich, da es in der Natur von Software im Allgemeinen und Open-Source-Technologie bzw. Civic Tech im Speziellen liegt auf der Software anderer Projekte aufzubauen.
Eine Technologie, die es erlaubt die Integrität und Authentizität von Software zu beweisen ist hierfür axiomatisch. Dies liegt darin begründet, dass sie es Nutzer*innen und Entwickler*innen erlaubt zum einen sicherzustellen, dass die Software die verwendet wird, von den Entwickler*innen gebaut worden ist. Zusätzlich dazu ist es aber auch möglich, damit spezielle Eigenschaften von dieser Software zu beweisen. Solche Eigenschaften können sicherheitsrelevant sein. Sicherheitsrelevante Eigenschaften sind z.B. Software Bill of Materials (SBOMs). SBOMs beschreiben alle Abhängigkeiten, die für die Erstellung einer Software verwendet wurden und deren Version.

Die Kopplung solcher Eigenschaften an die Software, wenn korrekt durchgeführt, ermöglicht es anderen Projekten, die diese Software verwenden wollen, diese besser einschätzen zu können. Dies ist ein direkter Vorteil gegenüber Closed-Source-Software, schafft Transparenz und kann die Zusammenarbeit im Civic-Tech Bereich fördern.

Ausführliche Darstellung der Ergebnisse

Während des Projekts konnten ausschließlich der erste und letzte Meilenstein erreicht werden. Das heißt, es wurde ein Proof-of-Concept (PoC) in Form eines Kubernetes-Clusters aufgesetzt, der aus zwei Teilen bestand. Der erste Teil ist ein Namespace mit einem Beispiel-Deployment, mit dem gezeigt wird, dass das Deployment in der Lage ist, ein signiertes Container-Image zu überprüfen. Hierzu wurde ein separater Namespace erstellt, der das Deployment vom restlichen Teil des Clusters abtrennt. Der zweite Teil besteht aus mehreren Namespaces und enthält alle für ein Eigen-Hosting notwendigen Sigstore-Komponenten. Auch wenn von Sigstore eigentlich als externe Komponente angenommen, soll hier erwähnt sein, dass dieser PoC die von Sigstore unterstützen OpenID Connect (OICD) Identitäts-Anbieter verwendet. Im Fall dieses Projekts wird Github als Identitäts-Anbieter verwendet. Dieser Teil wird später genauer beschrieben.

Das Deployment existiert in zwei Ausführungen, die jeweils eine der beiden existierenden Möglichkeiten umsetzen, die Signaturen von Container-Images zu überprüfen. Die erste ist die Integration mithilfe der Container Engine Cri-O. Cri-O ist eine Entwicklung von Red Hat und ist (stand heute) die einzige Container-Runtime, die in der Lage ist die Signaturen von Container-Images auf Container-Runtime-Ebene zu überprüfen. Die Integration in der Runtime reduziert die Angriffsoberfläche bzw. Möglichkeiten von Fehlkonfiguration (zumindest im Kontext von Kubernetes-Objekten). Dazu muss aber jeder Node des Clusters, der in der Lage sein soll Signaturen zu überprüfen, die Runtime verwenden und mit dementsprechenden Regeln konfiguriert sein.

Die zweite Ausführung des Clusters verwendet den von Sigstore entwickelten Admission Controller: Policy Controller. Ein Admission-Controller erlaubt es, API-Anfragen zum Cluster abzufangen und diese entweder zu ändern oder zu blockieren. Die über den Policy Controller definierten Regeln werden auf Namespace-Ebene angewendet, was im Falle des PoCs, für den Namespace mit dem Beispiel-Deployment geschehen ist.

Die für den PoC benötigten Signaturen werden mittels Github Actions – Githubs CI/CD-Pipelines – erstellt. Diese Action baut ein Container-Image und signiert dieses nach der Erstellung mittels des OICD-Tokens, der durch Github über eine Umgebungsvariable zur Verfügung gestellt wird. Für das Identitätsfeld der Signatur wird dann anstelle der E-Mail-Adresse der/des Entwickler*in die URL des sog. Github Actions Workflows verwendet. Das so erstellte Image wird von der Action in die Github Image-Registry zusammen mit der erstellten Signatur hochgeladen und die Signatur direkt überprüft. Eine Nutzung der selbst betriebenen Sigstore-Komponenten mit der Action hat nicht stattgefunden. Dafür gab es zwei Gründe: zum einen hätte die Infrastruktur öffentlich gehostet werden müssen. Zum anderen gab es einen Bug in der Verwendung des Sigstore-Clients, der es nicht erlaubt hat, mit dem Client Signaturen in der lokalen Infrastruktur zur veröffentlichen, dessen Grund bis zum Ende des Projekts leider nicht entdeckt werden konnte.

Mit dem Projekt wurde festgestellt, dass ein Projekt mit mehreren voneinander unabhängig zu entwickelnden Komponenten für den Zeitraum zu komplex ist. Im Projekt wurde die Menge an Zeit unterschätzt, die nötig war, um Teilkomponenten des Projekts funktionsfähig zu bekommen, was primär in Problemen begründet war, die nicht vorhergesehen worden sind. Exemplarisch dafür ist beispielsweise der im vorherigen Absatz erwähnte Bug, der es nicht erlaubt mit dem Sigstore-Client Signaturen in der selbst gehosteten Infrastruktur zu erstellen und in den dafür vorgesehenen Logs zu veröffentlichen. Der Grund dafür konnte bis zum Ende des Projekts nicht ergründet werden. Des Weiteren wurde festgestellt, dass die externen Abhängigkeiten zu zeitlichen Verzögerungen im Projekt aber auch, wie in unserem Fall, dazu geführt haben, dass ganze Meilensteine nicht erfüllt werden konnten. Im Falle dieses Projekts hat sich zum Ende herausgestellt, dass das Fachverfahren, dessen Integration als Meilenstein geplant war, nicht mehr existiert hat. Das heißt, dass die Infrastruktur des Verfahrens nicht mehr verfügbar ist und eine Integration daher nicht vorgenommen werden kann.

Zielgruppe, Nutzen und mögliche Weiterentwicklungen

Auch wenn kein real nutzbares System als Ergebnis des Projekts vorliegt, hat die Arbeit zu Erkenntnissen geführt, die für die genannten Zielgruppen von Relevanz sind. Dazu zählen, unter anderem, die Untersuchung von:

- der Validität von Sigstore als Lösungsansatz für die genannten Probleme
- der generellen technischen Umsetzung Sigstore
- unterschiedlichen Ansätzen, Sigstore in die eigene Infrastruktur zu integrieren.

Diese und weitere Erkenntnisse sind in einen Vortrag auf der FrosCon 2023 eingeflossen, in dem diese aufgearbeitet dargestellt worden sind.

Die Open-Source-Stellung des Projekts ermöglicht es, eine Vertrauensbasis aufzubauen. Diese ist notwendig, da das Projekt, mittels der Technologie auf der sie basiert, programmatisch Vertrauenswürdigkeit ermöglichen soll, die wiederum bei dem Projekt und seiner Arbeitsweise beginnt. Speziell für die Zielgruppe der öffentlichen Verwaltung ist eine offene Entwicklung darüber hinaus sinnvoll, um den Nutzer*innen von Fachverfahren die Transparenz zu geben, die sie von einer staatlichen Dienstleistung erwarten können sollten und um den Entwickler*innen die Möglichkeit zu

geben, auf Basis der oben genannten Eigenschaften Fachverfahren zu implementieren, die Secure by Design sind.

Derzeit gibt es keine Pläne einer direkten Weiterentwicklung des Projekts. Jedoch gibt es bekundetes Interesse aus der Verwaltung, die Ergebnisse des Projekts vorzustellen, was wiederum zu einer Weiterentwicklung in Zusammenarbeit mit der Verwaltung führen könnte.

Persönlich ermöglichte das Projekt eine Vertiefung des Verständnisses von Kubernetes als Technologie: vor allem Kubernetes-Interna, wie die Container-Interfaces und den damit zusammenhängenden alternativen Container-Runtimes, sowie eine tiefe Einarbeitung in Software-Signatur-Systeme, wobei hier Sigstore als Basistechnologie des Projekts im primären Fokus war. Über die technischen Erkenntnisse hinaus hat das Projekt zu einer Verbesserung des eigenen Zeit- und Prioritätsmanagements geführt.

Kurze Darstellung der Arbeiten, die zu keiner Lösung geführt haben

Zu Beginn des Projekts war der Plan, die Helm-Charts für die einzelnen Sigstore-Komponenten selber zu entwickeln, womit auch begonnen wurde. Bei der Recherche nach weiterer technischer Dokumentation hat sich herausgestellt, dass es schon ein Projekt gibt, was dies für die Sigstore-Infrastruktur tut: Scaffolding. Das Projekt wurde dementsprechend auf Nutzbarkeit überprüft und dessen Nutzung für sinnvoll befunden, obwohl das Projekt explizit keine Produktions-Deployments, sondern ausschließlich Test-Deployments zur Verfügung stellt. Dies lag primär begründet in der hohen Qualität der Helm-Charts und der Aktivität des Projekts. Die Zielsetzung des Projekts wurde dahingehend angepasst, dass eine Konfiguration für ein Produktions-System Teil der Integration des Fachverfahrens ist und entsprechend der Ergebnisse, diese entweder zu einem Upstreamen der Änderungen oder einem Fork als Teil des Projekts geführt hätten.
Ein weiteres Problem während des Projekts war, dass der für den lokalen PoC verwendete Kubernetes-Cluster aus bis heute unbekannten Gründen keine funktionierenden DNS hatte, obwohl sichergestellt werden konnte, dass die dafür notwendigen Komponenten des Clusters verfügbar und funktionsfähig waren. Die Annahme war bzw. ist, dass das Problem in der Networking-Engine der Container Runtime lag. Als Lösung wurde auf Minikube umgestellt, das alternativ zu einer Container-Runtime eine Virtuelle Maschine für den Kubernetes-Cluster verwendet, was das Problem gelöst hat.

Zuletzt war ein weiteres Problem, dass die Infrastruktur Fachverfahren, dass zur Integration vorgesehen war, zu dem Zeitpunkt in dem das Projekt an einem Punkt war die Integration vornehmen zu können, nicht mehr existiert hat. Dies wurde aber erst bekannt, als Fragen zur Open-Source Code Base an die Entwickler*innen des Projekts notwendig wurden, die dann im Gespräch darauf hingewiesen haben. Dies war zum Ende des Projekts, weshalb entschieden wurde nicht mit der Integration fortzufahren, sondern, anstelle dessen den Fokus auf die Verfügbarmachung der Projektergebnisse zu setzen, weshalb der Talk auf der FrOSCon eingereicht und erarbeitet wurde.

Kurze Angabe von Präsentationsmöglichkeiten für mögliche Nutzer

Das Projekt präsentiert sich auf der Website https://sic-signierte_container.gitlab.io . Eines der Ergebnisse diese Projekts war ein Vortrag auf der 18. FrOSCon 2023 (Free and Open Source Software Conference), der unter folgender URL zu finden ist: https://media.ccc.de/v/froscon2023-2901-the_new_old_supply_chain_security . Alle zum Projekt gehörenden Git-Repositories inklusive der

Slides für den FrOSCon- und den Demo-Day-Vortrag sind in folgender Gitlab-Gruppe zu finden: https://gitlab.com/sic-signierte_container. Eine Ausnahme stellt das Github-Projekt mit der für den Proof-of-Concept erstellten Github-Action dar. Die URL zu dem Projekt ist https://github.com/Delet0r/workflow-test.

Kurze Erläuterung zur Einhaltung der Arbeits- und Kostenplanung

Wie in den vorherigen Antworten ausführlich beschrieben, haben die folgenden Dinge zu einem Mehraufwand innerhalb des Projekts geführt:

- Umstellung der Eigenentwicklung von Sigstore Helm-Charts auf die Verwendung des Scaffolding-Projekts
- Diverse Probleme mit dem Aufsetzen des PoC (DNS, Client, etc.).

Abgesehen davon hat die Einstellung des Fachverfahrens zu einer theoretischen Verminderung des Aufwands geführt, die dann mit dem Vortrag ausgeglichen wurde.

Generell wurde die verfügbare Arbeitszeit voll ausgenutzt. Ein darüber hinaus zeitlicher Mehraufwand wäre nötig gewesen. Jedoch war es nicht möglich diesen zu erfüllen.

Kurze Darstellung von etwaigen Ergebnissen bei anderen Stellen

Abgesehen von dem Scaffolding-Projekt, was, wie oben beschrieben, ein Teil der Arbeit an diesem Projekt obsolet gemacht hat und in dieses Projekt integriert worden ist, gab es keine Entwicklungen, die Einfluss auf die Arbeit bzw. Zielsetzung hatten.