Accessible Telemetry Streams using a Zero Trust Architecture for the Flight Operations Directorate

Paul Shoemaker
APPDAT
NASA Johnson Space Center
Mathematical Research, Inc
paul.shoemaker@nasa.gov

Collin Estes
APPDAT
NASA Johnson Space Center
Mathematical Research, Inc
collin.j.estes@nasa.gov

***Abstract* - As a result of information technology based work becoming increasingly distributed, unique challenges have been presented within the realm of defined network perimeters, namely with respect to secure access to resources. Historically, and from a simplistic abstract perspective, the common approach has been to adopt the, so-called, moat model whereby a physical network perimeter (or interconnected perimeters) is defined to encapsulate resources behind a boundary protected by a firewall. Users are provisioned access through a virtual private network (VPN) and may be further constrained to resources through specific firewall allow and disallow rulesets. Virtual Private Networks and firewall rulesets lead to common problems, particularly at scale and, as a result, perimeter-less architectures provided over the public internet are increasingly becoming prevalent, particularly with its more popular implementation, the Zero Trust Architecture. We present a proposed implementation of the Zero Trust Architecture with a particular concrete example utilizing a de-perimeterized network that requires authentication and authorization for each action between nodes and does not operate within an implicit trust boundary. It should be noted that this paper is not an attempt at providing comprehensive resolutions for the specific problem space with respect to perimeter based security and is more directed at providing information with regard to our proposed implementation of a Zero Trust Architecture for the Flight Operations Directorate. We direct the reader to our Introduction and Background section for more details on specific documentation and where it can be located as it relates to de-perimeterization and Zero Trust.**

I. Introduction and Background

Network de-perimeterization was first presented by the Jericho Forum[1] in 2005 resulting in the further expansion and implementation as well as coining of the term Zero Trust by John Kindervag from Forrester Research in 2010[2]. In 2014, Google published their implementation of Zero Trust Architecture called BeyondCorp[3]. Each of these publications go into great detail concerning the need for removal of the traditional model of information security whereby nodes within the network perimeter are implicitly trusted while those nodes residing outside of the perimeter are untrusted. Zero Trust Architecture (ZTA) establishes the concept of trustless operating environments where there is no implicit trust between any node within the network and all traffic must be authenticated and authorized. For the purposes of providing a demonstration of capabilities, we chose to protect a resource consisting of publicly available telemetry data from the International Space Station. Our environment consisted of container-based applications running within a Google Kubernetes Engine (GKE) managed Kubernetes environment with a self-managed Istio Service Mesh layer providing for mutual TLS encryption, traffic interception, and traffic routing to individually protected resources. With a custom application specifically written for this purpose, we subdivided the telemetry data into specific ElasticSearch indexes representing varying levels of access to data based upon the requesting user's explicit level of trust corresponding to authentication and authorization rules defined for each protected ingress point. Additionally, we designed a custom token exchange middleware application deployed into the same cluster to demonstrate the capabilities of in-cluster traffic routing based on underlying service mesh rules in cooperation with application-based token authentication capabilities (namely that of the ElasticSearch system) that allowed for a Single Sign On passthrough into

the terminating application Kibana. Kibana was used as a visualization and index management platform as part of the ELK stack (ElasticSearch, Logstash, and Kibana). Finally we developed custom dashboards inside Kibana in order to demonstrate at a glance that varying users with varying authentication and authorization rules could access only specific indexes for which their application role allowed. Section II provides context for this implementation by laying out specifically defined Use Cases that are addressed by this architecture. Section III goes into more detail as to how the architecture is defined and how specific components within the architecture work. Section IV concludes our paper following with acknowledgements and references.

II. Use Cases

These use cases helped guide the implementation for the proof of concept and served to illustrate several plausible scenarios that might be directly applicable to the Flight Operation Directorate's needs. It should be noted that there are a few use cases that indicate the user's use of VPN-based networking resources. While we briefly noted that there are challenges with existing VPN infrastructures, we also note that, "It is unlikely that any significant enterprise can migrate to zero trust in a single technology refresh cycle. There may be an indefinite period when ZTA workflows coexist with non-ZTA workflows in an enterprise."[5] As such we included these use cases to demonstrate the power of CIDR based IP boundaries that take advantage of Context Aware rulesets for enforcement.

A. Public User

As a public user, I should be able to access publicly available data without authenticating. The data that I have access to will be very limited in scope as a result of my trust level. Authentication and Authorization will be forfeited *for the individual user* as a result of this very limited scope of trust.

B. External Partner with External Partner Identity Platform

As an external partner, I should have access to a higher trust level of data as an authenticated user. Authentication and Authorization will be accomplished using my external partner identity platform and relies on a trusted relationship between NASA and the external partner.

C. NASA Employee with PIV on the Public Internet

As a NASA employee with a PIV card (badge), when I am on the public internet, my trust level will be slightly higher than the external partner due to the presence of a vetted credential such as the PIV. Authentication and Authorization will be completed using Identity Aware Proxy and Context Aware Access using my NASA G Suite account.

D. NASA Employee with PIV on VPN and GFE

As a NASA employee providing a PIV credential while on VPN will gain me a higher trust level than previous trust levels. Authentication will be completed using Identity Aware Proxy and Context Aware Access will provide the IP based filtering to allow for authorization constraints to be placed based on incoming IP.

E. NASA Employee with PIV on VPN on Agency Device

As a NASA employee providing a PIV credential while on VPN and utilizing a NASA agency device will gain me the highest trust level. Authentication will be completed using Identity Aware Proxy and Context Aware Access will provide the IP based filtering as well as agency-managed device verification, in cooperation with the Device Management component discussed later (see section below).

III. Implementation

We present our implementation of a Zero Trust Architecture built around the vendor solution provided by Google through their BeyondCorp offering. It should be noted that while the Google framework was chosen for our proof-of-concept (POC), the abstract components within this infrastructure can be swapped for non-Google counterparts from other vendors. We will first discuss the abstract components of the Zero Trust Architecture and then further discuss the architecture as it is implemented by BeyondCorp.

A. Abstract Components

The abstract components that adequately describe a Zero Trust Architecture are illustrated in Figure 1.

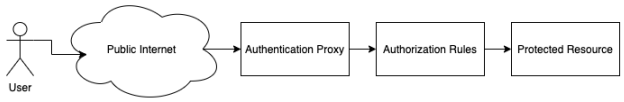

Fig. 1. Abstract process flow diagram of ZTA protected resources

1. Proxy

Any resource that is accessible within the network requires Authentication and Authorization prior to granting access to that resource. As such a universal proxy is required as a provided minimum level of access control to that resource. The proxy is responsible for providing a challenge to the requesting user (be it a physical person or machine) which will require some credential in order to pass Authentication.

2. Authentication

The Authentication mechanism is typically provided for by an Identity Provider (IdP) such as Google, Microsoft, etc. The IdP will present the challenge discussed above and provide for allowance or disallowance based on the criteria of the challenge. For example, if the challenge requires a username and password, those components must be correct in order to pass the authentication gate.

3. Authorization

After authentication has been granted, Authorization steps must be performed in order to provide for a context-based decision as to whether the requesting user will have access to the requested resource in spite of providing proper authentication credentials. Authorization is typically implemented as a series of rules consisting of sets of logical AND, OR, and NOT operations with respect to a variety of parameters.

B. BeyondCorp

The Zero Trust Architecture as it is implemented by BeyondCorp consists of two primary components: Identity Aware Proxy and Context Aware Access (with further utilization of Device Management capabilities). We utilized these individual components in concert in order to provide concrete implementations of the abstract components discussed above.

1. Identity Aware Proxy

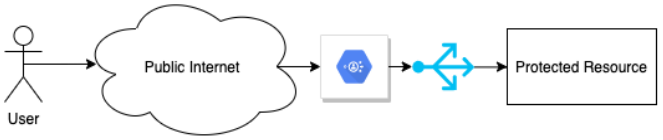

Fig. 2. Architectural flow diagram for IAP terminated ahead of Load Balancer

Identity Aware Proxy (IAP) is a centralized proxy that can be terminated on demand in front of an HTTPS load balancer resource that is serving traffic to within a compute environment such as Google Kubernetes Engine (Fig. 2).
Its purpose is to provide a centralized authentication mechanism whereby you can provision access to your protected resource utilizing a variety of identity platforms, including Google's own Cloud Identity (Fig. 3).

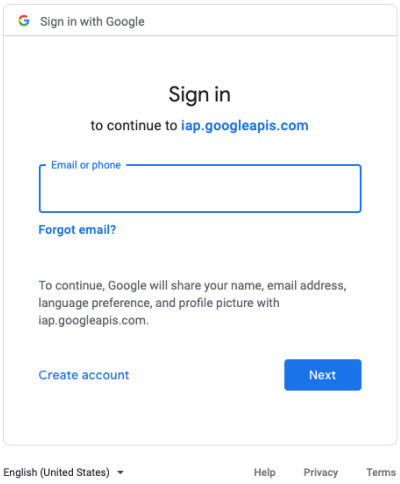

Fig. 3. Example of a authentication form for IAP

Additional external identity platforms can easily be utilized including custom Security Assertion and Markup Language (SAML) and OpenID Connect (OIDC) connections. This means that IAP can provide the proxy access point specifically while an external Identity Provider (IdP) actually provides the core authentication mechanism. It should be noted that with respect to Google's BeyondCorp offering specifically, IAP manages user-based access controls through its Identity and Access Management system (IAM) as long as you are using Google Cloud Identities. IAM based access controls have the added benefit of providing capabilities of the Context Aware Access

controls discussed in the next section. If the user wishes to utilize an external identity platform, that user will forgo IAM based control mechanisms as well as Context Aware Access capabilities. Additionally, IAP is terminated in front of an HTTPS Load Balancer, where, within Google Kubernetes Engine (GKE), it is prescribed upon a single Ingress Custom Resource Definition (CRD) which utilizes the Google Ingress Controller within the cluster. As a result, it is possible to control access to a single resource utilizing multiple Identity Providers by defining several discrete Ingress resources.

2. Context Aware Access

Context Aware Access is a mechanism by which a user might provide for authorization to a protected resource after the step of authentication. Once the authentication step has completed and is successful, there is an authorization step that must be completed in order to ultimately gain access to the desired resource. Context Aware Access provides for this capability by having the ability to segment based on a variety of properties including IP boundary (by Classless Inter-Domain Routing or CIDR), regional location, as well as specific properties of managed devices (please see more information below on Device Management). Based on this complex rules engine, a decision is made to allow or disallow access to your protected resource. These rules can be chained together via the web UI or through APIs to the underlying IAM system.

3. Device Management

Within the BeyondCorp suite of tools, you are able to provision devices within your enterprise utilizing a Chrome-based client that will synchronize device-specific information to a centralized location that is manageable within your Google account. This information includes operating system patch level, administrative privileges, model number, serial number, etc. This information can be used to require, for example, that only agency approved devices that meet specific device-based criteria will be authorized to access a particular resource on each request. Should anything about that device change which changes its compliance state, that device's permission will be immediately revoked and access will no longer be granted until the device is compliant once more.

C. Bringing It All Together

As part of our proof-of-concept for implementing a Zero Trust Architecture targeting varying levels of access to non-sensitive telemetry data, our implementation consisted of four discrete application bundles, including a landing page, a telemetry listener and indexing application, a token exchange middleware, and a GKE application stack that included ElasticSearch, Kibana, and a variety of ingress and service mesh virtual service manifests. The landing page served as a way for the user to choose their desired authentication and authorization mechanism. The user had a choice between public, NASA, and External in order to demonstrate the ability to authenticate using several different IdPs and for the results of that authentication and authorization route to have an impact on the eventual authentication to the protected resource (and, by extension, the amount of data that user has access to). The telemetry application was responsible for providing data from a publicly available feed containing various ISS telemetry data points. In order to subdivide telemetry into increasingly more privileged levels of access, the data was placed into individual ElasticSearch indices where they would be used to constrain access by role within Kibana (it should be noted that it is possible to constrain access by column within ElasticSearch which represents another vector by which this could be accomplished). Once a user chose their authentication IdP, they would enter their credentials (if required; in the case of Public, this was not required). Upon successful authentication and authorization, the user would be forwarded through to the protected resource, however in this case, that resource required a specific token for authentication itself and was thus further protected. IAP provides a JSON Web Token (JWT) [4] as the method of passing cryptographically signed non-sensitive user data in a portable manner. Within the JWT payload is a list of claims that are arbitrary and quite service specific, meaning, in the case of IAP, there were specific pieces of data within the payload that pertained to within the Google ecosystem specifically. As such, the JWT token claims were incompatible with those required by the protected resource. It is posited that this particular scenario will be relatively normal particularly if you are attempting to protect either a Commercial Off the Shelf (COTS) or Open Source Software (OSS) resource where the particular mechanisms of

authentication may vary. As a result of this particular implementation, a combination of the Istio Service Mesh and a custom token exchange middleware application served as the mechanism by which a token from IAP would be exchanged for a token by ElasticSearch through the protected ElasticSearch API. Istio Service Mesh was used as a control plane to an Envoy proxy based data plane automatically injected as a sidecar container to each pod deployed in the cluster. The service mesh served as a mechanism to route traffic to the desired terminating service which, in this case, was simply Kibana, as well as filtering for the IAP header which contained the JWT needed to perform the token exchange with ElasticSearch. Once the token exchange was completed, a valid Authorization Bearer header was written to the request and forwarded to the Kibana application for verification and authentication. This process flow provides for a seamless Single Sign On (SSO) experience so that the user is forwarded to their application without the need to authenticate through several different mechanisms.

Once the user has arrived in Kibana, they will notice that they are able to utilize the application as normal and, importantly, only have access to the index for which their role allows (directly related to the Use Cases outlined previously).

IV. Conclusion

We have discussed a bit of the history behind de-perimeterization and the Zero Trust Architecture implementation as well as some of the abstract components that comprise it. Additionally, we have presented our proof of concept as it relates to a component of the mission under the purview of the Flight Operations Directorate namely with respect to ISS telemetry. Our solution utilized many components of the BeyondCorp vendor offering in order to achieve its goals, however, many of the abstract components of proxy, authentication, authorization, device management, and context aware access can be implemented utilizing other offerings from other vendors. The advantage of using a comprehensive set of tools from a single vendor is mainly in speed of implementation and time-to-solution. For BeyondCorp specifically, it is necessary to ensure that those members participating within the Zero Trust network possess identities from Google Cloud Identity or from an externally approved IdP with OIDC or SAML integration mechanisms (such as Microsoft). The disadvantage of a single-vendor solution is in that same comprehensive nature where it becomes necessary to adopt all required components that comprise the solution. In conclusion, we hope to have introduced the core concepts to those unfamiliar and to have provided enough detail to those parties of interest that may be seeking to implement similar architectures or use it as context or inspiration for their own solutions.

Acknowledgements

Thank you to the Flight Operations Directorate for the opportunity to design this proof of concept and to Jennifer Morehead, Peter Mossbacher, and Aaron Goldenthal for providing resources, context, and answers to organizational questions along the way. Also thank you to Rudis Muiznieks and David Kelldorf for providing valuable feedback during the process of documentation.

References

[1] "Visioning White Paper What is Jericho Forum?" (2005)

[2] Kindervag J "No More Chewy Centers: Introducing The Zero Trust Model Of Information Security" (2010)

[3] Beyer B, Ward R "BeyondCorp - A New Approach to Enterprise Security." (2014)

[4] Wikipedia contributors. "JSON Web Token." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 18 Aug. 2020. Web. 24 Aug. 2020.

[5] Rose S, Borchert O, Mitchell S, Connelly S "NIST Special Publications 800-207 Zero Trust Architecture" pp 36
https://doi.org/10.6028/NIST.SP.800-207

Disclaimer

Trade names and trademarks are used in this report for identification only. Their usage does not constitute an official endorsement, either expressed or implied, by the National Aeronautics and Space Administration.