Secure Cloud Architecture for Protecting the Software of Space Vehicles

Keshav Raviprakash, Cameron Bonesteel, Sydney Whilden, Deepak R. Mishra
kr00436@uga.edu, cameron.bonesteel@uga.edu, sydney.whilden25@uga.edu, dmishra@uga.edu
Small Satellite Research Laboratory, University of Georgia

Abstract

As the number of space vehicles launched into outer space increases over time, the need to determine and counter possible cyber attacks from malicious agents will drastically increase. A potential solution to this problem is to create a cloud architecture that a spacecraft can use for autonomous detection or recovery from cyber attacks when out range of a ground station. For this work, the development of a new cloud architecture for space vehicles is proposed which allows users to control access to information or applications within the system while using the self-healing capabilities of containers. The cloud architecture will first be tested without the use of any containerization to establish a control. Afterwards, experiments will be conducted to test the effectiveness of different proposed security measures such as security vulnerabilities, cyber-hardening of the architecture, and logging. This research will then use qualitative and quantitative metrics, such as a pass or fail test, self-healing time of containers, and the number of vulnerable access points to determine the effectiveness of each of these measures.

Figure 1: Example of Cyber Threats to Space Systems [5]

Motivation

Space assets that are on the ground and within space are a part of critical infrastructure used by a variety of organizations around the world. Thus, the need to implement cybersecurity measures to protect them is important. While there are many different ways that organizations can implement these measures, these organizations suffer from challenges including a lack of standard cybersecurity standards and regulations for space, increased usage of off-the-shelf technology in spacecraft development, resource constraints, and potential single point of failures within space systems[1]. This body of work aims to determine solutions to fixing these problems through self healing, reduced resource usage, decentralization, cyber hardening, and anomaly detection. Each of the following ideas are currently being developed on a sample space vehicle software system to determine the outcomes of using the proposed cloud architecture in real-time.

Methodology

To address the criteria required to effectively implement this process, the following architectural choices were made:

1. Self healing
   Self healing is the ability of software to redeploy software applications to a desired state when a software application is down. Self healing is implemented via a software architecture known as K3s due to its ease of access and ability to update software quickly. Implementing self healing will allow the system to destroy pods that contain the sample satellite software when cyber attacked and return pods back to a desired state for operators of the proposed cloud architecture.
2. Reduced resource usage
   Current satellite missions have a variety of software, power, and hardware requirements to successfully complete the mission tasks. However, meeting these requirements can be costly and prevent cybersecurity measures from being applied on the satellite. Thus, this proposed cloud architecture will satisfy requirements, reduce costs, and utilize cybersecurity measures against cyberattacks in an optimal manner that can be decided by operators of the architecture.In order to achieve this, K3s will be used due to its lightweight Kubernetes distribution and its ability to be implemented on embedded devices.
3. Decentralization
   Decentralizing components of the software using the cloud architecture is important step to avoid having the whole software system from being shut down. The use of Docker allows cloud images to be created that run separate components of the sample satellite system independently and then be combined together. This reduces vulnerabilities to cyber attacks over the whole system.
4. Cyber Hardening
   Cyber hardening is the process of keeping out cyber attackers and reducing potential vulnerabilities from cyberattacks. Cyber hardening of the cloud architecture will reduce cyber vulnerabilities across the entire cloud architecture. For the proposed cloud architecture, K3s will be cyber hardened through various policy controls available through its architecture. Secure Docker images and software packages were also made to reduce cyber vulnerabilities. Finally, an authentication service will also be used in conjunction to protect the overall cloud architecture and reduce access to the system.
5. Anomaly Detection
   Anomaly detection is important to providing information about potential cyberattacks when implementing cybersecurity for software systems. Many satellite software systems already utilize this idea as shown by the publicly released NASA cFS and Fprime software systems. Thus, logging will continue to be used by the proposed cloud architecture to provide relevant information to the operators of the architecture while also providing updates about the status of the cloud architecture when in use.

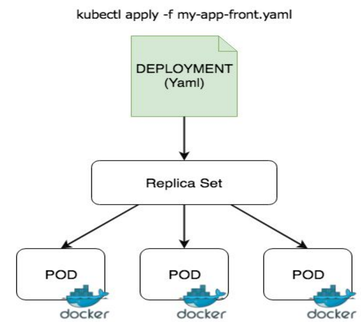

Figure 2: Example Kubernetes Architecture with Self Healing [6]

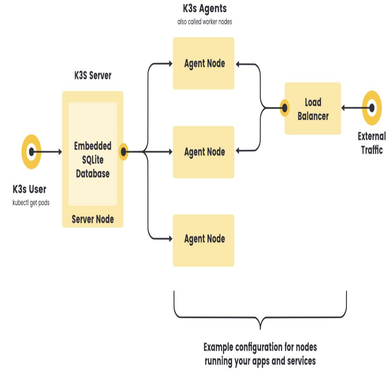

Figure 3: K3s Software Architecture [7]

Analysis

While results are still being generated, the table below demonstrates the advantages that the system is able to implement. The results presented in this table are predictions that could change once the cloud architecture is fully developed. Numerical estimates are made based on individual analysis of the authors.

	Sample Software System	Proposed Cloud Software System
Can Cyber-hardening be implemented?	Yes	Yes
Is Logging Used?	Yes	Yes
Estimated Minimum Required Satellite Onboard Computers	1	1
Estimated Number of Cyber Vulnerabilities	20	10
Self-Healing Time	N/A	1-10 Minutes

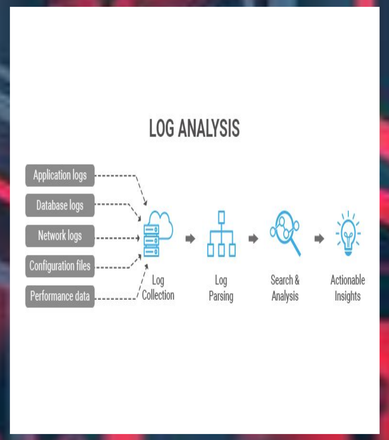

Figure 4: Log Analysis Process [8]

Future Objectives

Currently, the proposed cloud architecture can be improved in a variety of ways. We propose that the following considerations be made for future development:

1. Creating automated anomaly detection
   Automating the Anomaly detection process will be important to implement resilience while acting like an active barrier to cyberattacks.
2. Deploying onto hardware for testing
   Implementing the cloud architecture onto hardware such as Raspberry Pi or Nvidia Jetson Nano would be important step to proving whether this architecture work in space.
3. Increasing cyber hardening of K3s components
   K3s is lightweight distribution of Kubernetes which has had many known published cyber vulnerabilities. Increasing cyber hardening of K3s is important to avoid the whole system from being attacked and operation while allowing the cloud architecture to actively counter cyber threats as they are discovered.

References

1. Falco, G. (2019). Cybersecurity principles for space systems. *Journal of Aerospace Information Systems*, *16*(2), 61-70.
2. *Lightweight Kubernetes*. K3s. (2023, July 12). https://docs.k3s.io/
3. *Kubernetes Documentation*. Kubernetes. (n.d.). https://kubernetes.io/docs/home/
4. *Reliable, self-healing kubernetes explained*. Kublr. (n.d.). https://kublr.com/blog/reliable-self-healing-kubernetes-explained
5. *Why space cyber?* . Center for Space Cyber Strategy and CyberSecurity - University at Buffalo. (2021, March 4). https://www.buffalo.edu/space-cybersecurity/center/why-space-cyber.html
6. Borja, P. (2018, September 21). *From Docker to elastic self-healing infrastructure*. Medium. https://medium.com/jeff-tech/from-docker-to-elastic-self-healing-infrastructure-67a48830dc02
7. *Architecture*. K3s. (2023a, July 12). https://docs.k3s.io/architecture
8. Tek-Tools. (2020, June 3). *Log analysis: How to choose and use a log analyzer tool*. TekTools. https://www.tek-tools.com/apm/choosing-log-analyzer-tool