Fortanix®

Leader in Runtime Encryption

Solution Brief

Managing Secure Containers using Fortanix Confidential Computing Manager and Runtime Encryption™(RTE)

Problem: Trust and Integrity in the Cloud

Container based software development and deployment has become the next big thing in technology. Container technology, such as Docker, is redefining cloud computing and offers tremendous benefits to companies and developers, including consistency, reliability, efficiency, cost savings and scalability across the entire DevOps process. However, for enterprises running sensitive applications at scale using Docker, securing and maintaining the integrity of the containers in the cloud is an important challenge.

Data at Rest

Data in motion

Runtime/ Data in use?

Today, when it comes to safeguarding “data at rest” and “data in transit”, there are various encryption schemes and strategies. However, there is a bigger question: what about “data in use”? Even if data is carefully safeguarded while at rest and when exchanged over secure channels like TLS, data does get decrypted while it is used or marshalled. This makes software containers a good attack target for malicious entities. None of the container security technologies today can guard against vulnerabilities in the OS, hypervisor, or other software that runs with root privileges.

Fortanix

Intel® Software Guard Extensions

Intel® Software Guard Extensions (SGX) is a technology which allows software developers to encrypt their applications' data at runtime. It achieves this by allowing applications to launch special protected software containers known as enclaves. These enclaves can be used for protecting applications' data from adversaries with root privileges or even physical access to the memory bus.

Running security critical services in SGX enclaves poses various challenges to developers:

• Since running in SGX involves making "OCALL"s instead of system calls, the applications run in an enclave must be heavily refactored, which is almost impossible for the applications written in high level programming languages like Java or Python.

• Just running an application (or its sensitive parts) within an enclave is not sufficient. To fully utilize the SGX security guarantees, the user must also verify that the application which they have built is running unmodified inside a secure enclave. Intel provides a way for users to achieve this using remote attestation.

• **Intel's remote attestation is a complex flow which poses its own usability challenges:**

- Every time an instance of a microservice is deployed, the flow of remote attestation must be repeated, which includes an external call to the IAS.

- Most microservices use TLS for exchanging sensitive information. To benefit from the security guarantees of SGX, the TLS stack should verify SGX attestations too.

- If the IAS API changes, all application enclaves need to be upgraded.

- A monitoring/controlling service also needs to verify SGX attestation.

- IAS will attest any securely deployed enclave signed by the registered user. If the user wants to have policies controlling when an enclave should be allowed to join their cluster of services, additional instrumentation is needed.

Fortanix

Solution – Fortanix Confidential Computing Manager

Fortanix Runtime Encryption technology allows you to run your applications inside SGX enclave unmodified. What's more, Fortanix Confidential Computing Manager helps you overcome all the above challenges in a matter of few clicks.

Let us look at what Confidential Computing Manager provides and what is the users' flow while using its services:

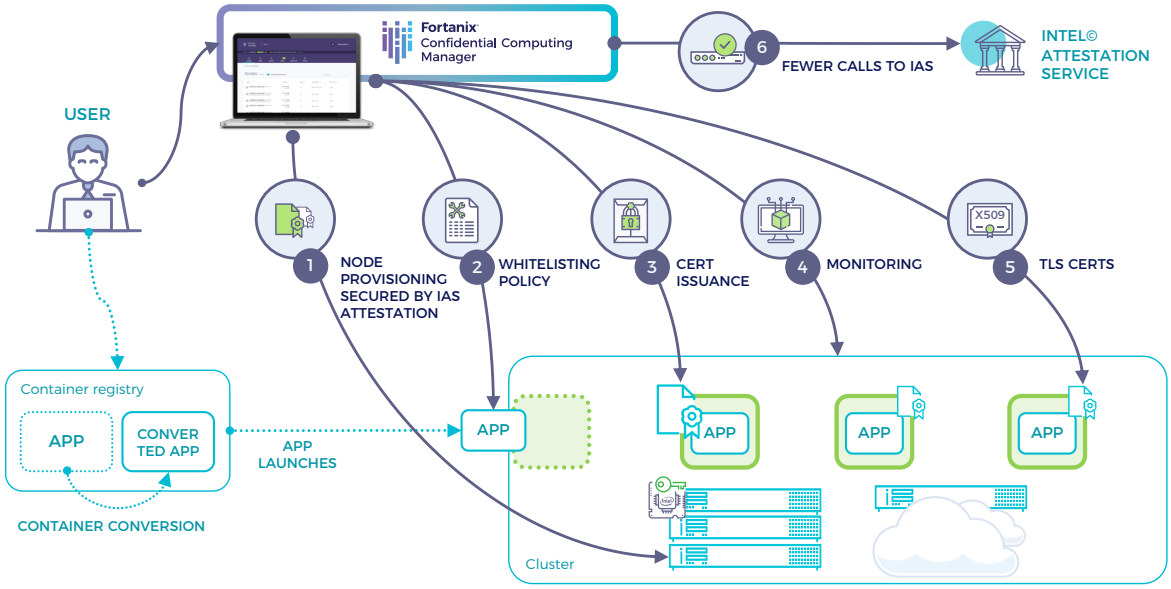
The Confidential Computing Manager Node Agent is deployed on users' nodes. It guarantees that the nodes are valid SGX machines with all the necessary security critical microcode patches applied.

• The Confidential Computing Manager converter allows users to convert their application containers to run inside the SGX environment in a matter of a few clicks or API calls.

• Once the converted applications are deployed, RTE and EM automatically perform a local attestation and provide a TLS certificate to the application without having to call IAS for each application. As TLS using X.509 certificates is a widely used protocol, the microservices can use the Confidential Computing Manager's root certificate as the root of trust and secure inter-enclave communication without having to change their crypto stack. This ensures that only enclaves running in a secure SGX environment get the required TLS certificates.

• Confidential Computing Manager also requires that the enclave measurements (specifically, the hash of the enclave's memory at load time) must be whitelisted prior to certificate issuance ensuring that only enclaves that the user intends to deploy are issued TLS certificates. Once we can guarantee that an enclave was deployed as intended, the processor hardware guarantees that the enclave cannot be tampered with, and its run time memory cannot be scraped at runtime.

• RTE enclaves can also issue periodic heartbeats to the Confidential Computing Manager back end for secure monitoring.



A high-level flow of Confidential Computing Manager usage is described below:

1. The user deploys Confidential Computing Manager.

a. Upon deployment, the back end enclaves obtain IAS attestation and use that to securely establish a highly available service cluster (for multi-node back end deployment).

b. The back end issues an X.509 certificate to each instance of the node agent if it can obtain a valid IAS attestation for the node. This flow is called node provisioning and happens only once for the lifecycle of node agent process.

c. Node provisioning will happen automatically post deployment.

2. The user whitelists their enclaves (either from the UI or using the Confidential Computing Manager APIs) and then deploys them.

3. The application enclave communicates with the node agent to obtain a local attestation which the back end service verifies before issuing a TLS certificate. For enclaves deployed using EnclaveOS, this flow is automated and triggered using certificate-related manifest options.

4. The application enclave can also choose to provide periodic heartbeats signed from within the enclave for secure enclave monitoring. For EnclaveOS applications, this is triggered using the heartbeat-related options in the application manifest.

LEARN MORE ABOUT RUNTIME ENCRYPTION