Platform Security Overview

EDITORS
Jeff Andersen, Google
Yigal Edery, NVIDIA

CONTRIBUTORS
Andres Lagar-Cavilla, Google
TODO: Feel free to add yourselves

Revision History

Revision	Date	Guiding Contributor(s)	Description
0.1	2021.09.07	Jeff Andersen, Yigal Edery	First Draft

Table of Contents

Executive Summary

[editor's notes] Summarize the whole document in one page when it's ready.

Purpose

The Open Compute Security project has been on a journey to define the various aspects for how to build security into the hardware, and how to design devices and platforms that can be trusted to be secure. Work on this has been going on since 2018. Several requirements document have already been published in that context, and the overall work of the group can be followed on the OCP-Security wiki (https://www.opencompute.org/wiki/Security)

This document is intended to serve as the high level overview of OCP's approach to platform & device security, and help put the various OCP security documents in context. It should be seen as the authoritative top level document which explains how to get OCP systems to comply with the various OCP security requirements and checklists, and the OCP-Security badge levels.

In short, this should be the first document to be read by anyone looking into joining the OCP security project, or building systems that are compliant with OCP security requirements.

Audience

The audience for this document includes, but is not limited to, system and system component designers, who want to build secure systems that comply with OCP security, and for customers/cloud operators that want to understand how OCP platforms are built for security.

Overview

In 2018 NIST published a Special Publication 800-193 titled *Platform Firmware Resiliency Guidelines,* which outlines the three main principles that are required to support platform security and resiliency:

- **Protection** - Mechanisms for ensuring that Platform Firmware code and critical data remain in a state of integrity and are protected from corruption, such as the process for ensuring the authenticity and integrity of firmware updates.
- **Detection** - Mechanisms for detecting when Platform Firmware code and critical data have been corrupted or otherwise changed from an authorized state.
- **Recovery** - Mechanisms for restoring Platform Firmware code and critical data to a state of integrity in the event that any such firmware code or critical data are detected to have been corrupted, or when forced to recover through an authorized mechanism.

The NIST publication provides guidance on meeting those requirements via the use of three Roots of Trust. The Root of trust for Update (RTU) which is responsible for authenticating firmware updates, and supports the platform protection capability; The Root of Trust for Detection (RTD), responsible for detecting firmware and data corruption, and the Root of Trust for Recovery (RTRec) which is responsible for bringing a corrupt platform back to a trusted state.

In addition, Root of Trust must adhere to the requirements of NIST SP 800-161 - Supply Chain Risk Management Practices for Federal Information Systems and Organizations.

These requirements and guidelines serve as the basis for the set of specifications of the OCP platforms security model.

High Level Architecture of an OCP System

OCP implementation of holistic system security is architected around the model where each component in the system needs to have Root of Trust elements covering the RTU, RTD and RTRec functions, for Updates, Detection and Recovery. A system typically has a main RoT of the system (referred to as the PA RoT - The Platform Active Root of Trust), and several devices/components/peripherals (e.g. NICs, SSDs, or even the CPU itself, etc), each with their own RoT referred to as the AC RoT (Active Component Root of Trust). A chain of trust is established between the PA RoT and the AC RoTs. The PA RoT is generally described as a component on the main board. Some platforms, however, could have the PA RoT subsystem separated out, such as in the case of the OCP DC-SCM design.

In the OCP model, each component in a system boots intrinsically with integrity, thanks to having an AC RoT component on it. The PA RoT serves as the holistic RoT for the whole system, collecting attestations from all devices in the system, attesting them using a secure protocol, and can represent an inclusive and holistic system security state to some external management infrastructure, in a process we refer to as platform attestation (or remote attestation), which in turn helps the operator of the system gain trust in the end-to-end security state of a whole fleet of systems.

The following is a high level canonical example of a simplified architecture meant to ease the explanation of the various functions in a system. In this example architecture, the PA RoT is

shown as separate from the CPU and BMC, attesting them just like any other component in the system. It is definitely possible to envision scenarios where the PA RoT would interpose the system flash and therefore would not need to attest the BMC and CPU (because it provides the protect/detect/recover functionality for them), or where the BMC would act as an extension of the PA RoT, taking a transitive role in attesting other components in the system. Both PA and AC RoT components could either be discrete (i.e. a RoT chip such as Titan or Cerberus), or be implemented internally in an ASIC or FPGA, or both approaches could be used complementary -- an internal RTM and a discrete chip providing RTU/RTRec capabilities. As long as the combination of RoTs complies with the set of requirements defined in OCP security specs, it provides the necessary assertions about system security that helps an operator know that they are running secure and resilient platforms which comply with the NIST 800-193 guidelines.

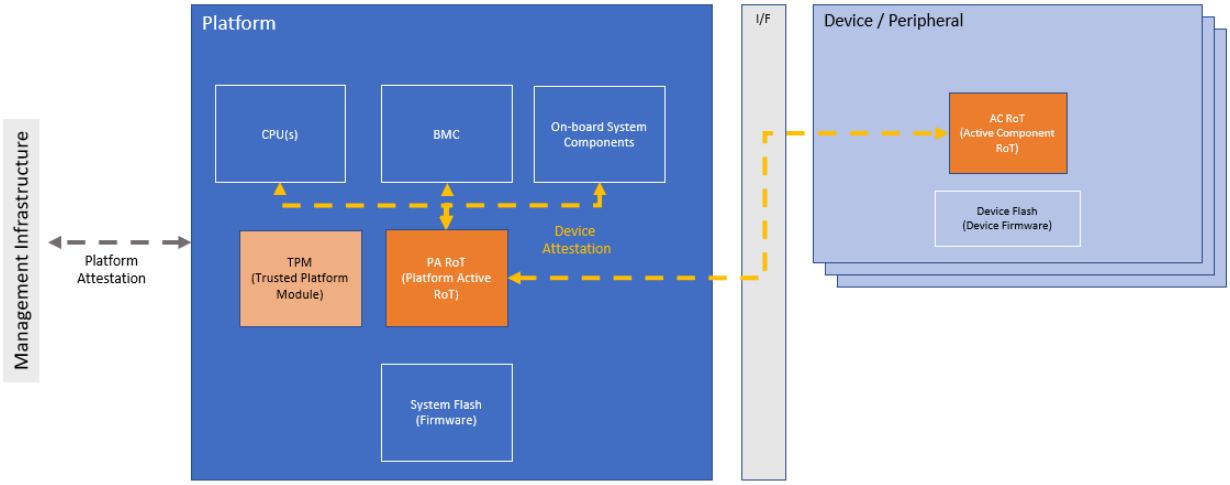


Figure 1 : A typical OCP system, per the OCP-security model

An OCP system can be compliant as long as there's a way to measure and attest all of its components, and to provide Protection and Recovery capabilities. Several additional detailed views of the various admissible architectures are described later in this document.

The OCP-Security Requirements Documents

In order to build a system around the architecture outlined above, the OCP security project has been on a journey to define the various requirements documents, each contributing its' respective part to building a secure system:

Common Threat Model - outlines the threat model that OCP follows. Specifically, what are the various security risks & threats that can make a system compromised, and which features in each requirements document are meant to mitigate these risks.

Secure Boot - outlines the requirements from a RoT (acting either as the PA RoT of a platform/motherboard, OR as the AC RoT for a device/component), that are needed in order to be considered as booting with integrity, at least until the point where a first measurement is taken. This is critical in order to start from a safe ground and be able to trust measurements and chain them back to the RoT.

Attestation of System Components - outlines the protocol for internal attestation between AC RoT's (on devices) in the system, and the PA RoT (of the platform). This is the protocol that enables the PA RoT in the system to attest all components and devices in it, before admitting them into the system and enabling them to operate and access internal system busses.

Note that storing measurements and reporting platform measurements to some external management plane can still leverage regular TPMs, as defined by TCG, or other similarly secured methods. OCP security does not define a specific protocol for how to do that, and customers can still use standard TPM APIs as they do today. Basically, remote platform attestation is currently considered out of scope for OCP-Security.

Secure Firmware Development Best Practices - Integrity & resilience are not enough This document complements the story by providing guidelines and a checklist for how to develop secure firmware. This should assist developers in implementing the necessary features called out in the OCP specs, in a robust & secure way, using industry-recognized best practices.

More Requirements documents (still in the works)

Anyone in the security industry knows that security is a never ending journey, constantly pushing the envelope and improving. OCP-Security is not different, and more requirements documents have been evolving and are in continuous development.

The following are the specs in various stages of development, as for 2021:

Secure Recovery - will cover how to perform recovery from a corrupt state, and what is needed from a good RTRec that complies with the NIST requirements for data center operators.

Ownership transitions - will cover how what it means to "own" a system, and how to support OCP's circular economy and balance between security needs (where a given customer "owns" a system and wants to ensure it only executes firmware authorized by them), and the desire to enable reselling these platforms to some other customers and securely transition ownership.

Platform Security Overview - this document.

Secure Firmware Updates - will cover how to perform secure and resilient firmware updates, and basically cover the RTU aspects of the NIST requirements.

Additional backlog items can be found here and we encourage and welcome new members to join the effort and contribute and help raise the bar!

OCP Compliance and Security Badges

OCP has a certification program that's designed to enable customers to choose OCP compliant solutions. There are two levels of general OCP compliance, called OCP-Inspired™ and OCP-Accepted™ . Customers can then use the OCP marketplace (https://www.opencompute.org/products) to choose from OCP compliant systems.

OCP-security adds a layer of security requirements to the compliance program, which is designed to help customers know that a system has implemented the right set of security features, and can be deployed in a data center in a secure way.

As of 2021, a new security checklist has been integrated into the OCP compliance program. This should further help customers choose the products that demonstrate having the right level of security for their needs. Suppliers can now claim compliance with the set of security requirements, and be granted a security badge on top of their OCP compliance badge.

OCP-security has defined three levels of security badges. At a high level, these are:

- **Bronze** - focuses on the core security features, for systems that have implemented at least the core secure boot and attestation “MUST” features.

- **Silver** - Builds on top of the Bronze level, and adds support for more advanced features, including secure firmware update, secure recovery, and providing collateral to describe the threat model that is applicable for the compliant product. It also expands the depth of implementation required to include the “SHOULD” features in each spec and requires making source code available to the customers.

- **Gold** - Is the most advanced and most open badge, signifying products that not only implemented the complete set of features in all specs (including the optional ones), but that are also open sourced publically, and are certified appropriately by formal certification programs such as FIPS.

This compliance program is designed to keep pushing the industry’s security posture to improve. As of today, not all systems implement the necessary security features, or follow secure development lifecycles. The hope and intent is that the security badge program will help recognize products who do implement strong security, and to further allow differentiation by also being open and enable community scrutiny to push security upwards.

It is expected that the security badge definition will change year over year, to include more requirements and keep pushing the security levels up.

Admissible Architectures

As discussed in the overview section, systems architecture can vary, and OCP does not mandate a specific implementation, as long as the architecture enables meeting the uber-goal of gaining trust in the system as a whole.

This section touches on several categories of admissible architecture, for RTD, RTU, and RTRec. These are enumerated separately, as the requirements of an RTD are distinct from an RTU or RTRec. That being said, a given RoT may fulfill multiple roles in these architectures.

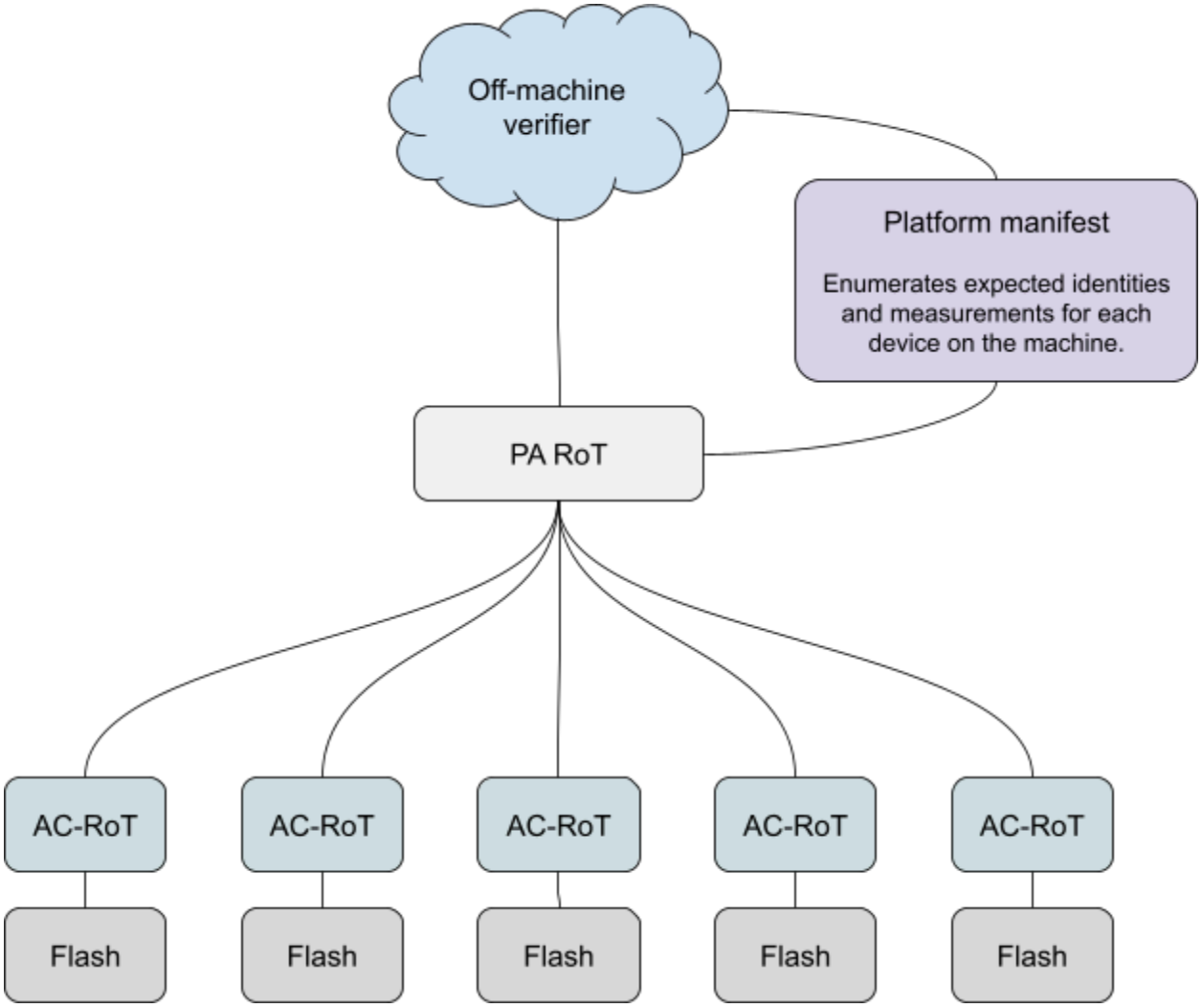
RTD architecture: pre-boot attestation

In this architecture, the PA RoT is empowered to hold peripheral devices in reset and to challenge their AC RoTs for measurements. Only once measurements have been obtained, are the peripherals released from reset. The PA RoT can later be challenged by an off-machine verifier to attest to the measurements it obtained at boot.

The PA RoT may be an embedded controller chip (e.g. Cerberus), or a BMC. If the former, the PA RoT may either retrieve measurements itself, or a BMC may act as a broker, obtaining measurements and passing them to the PA RoT. If this is done, the PA RoT must ensure that the measurements it receives from the BMC are fresh. In the case BMC acts as RoT, it has to adhere to the same rigorous security principles as a dedicated RoT and implementation of the entire security stack, and not mere forwarding of the measurements elsewhere.

The PA RoT is provisioned with a policy that enumerates valid measurements for peripheral devices, and makes decisions based on whether or not each device's attested measurements conforms to the policy. These decisions may range from raising an alert to an off-machine verifier, to holding the peripheral device in reset. The PA RoT may also decide to recover the device if its measurements do not conform to the policy.

The PA RoT should be empowered to detect when a component has reset, and be able to repeat the process of holding that component in reset while it challenges the component's AC RoT.



RTD architecture: post-boot aggregated attestation

In this architecture, the PA RoT gathers measurements from each AC RoT. Unlike in the pre-boot architecture, this may happen after the devices have completed their boot sequences. PA RoTs may aggregate measurements (a) soon after boot, (b) upon sensing that a component has reset, (c) on-demand from an off-machine verifier, or some combination of the above.

This architecture is suitable for platforms that feature components that cannot attest while they are being held in reset, and avoids adding complexity in early platform boot.

In this architecture, the PA RoT may be provisioned with a local policy that it enforces. Alternatively, the PA RoT may only be tasked with aggregating measurements and reporting them to an off-machine verifier.

Like pre-boot attestation, the PA RoT may be an embedded controller or a BMC.

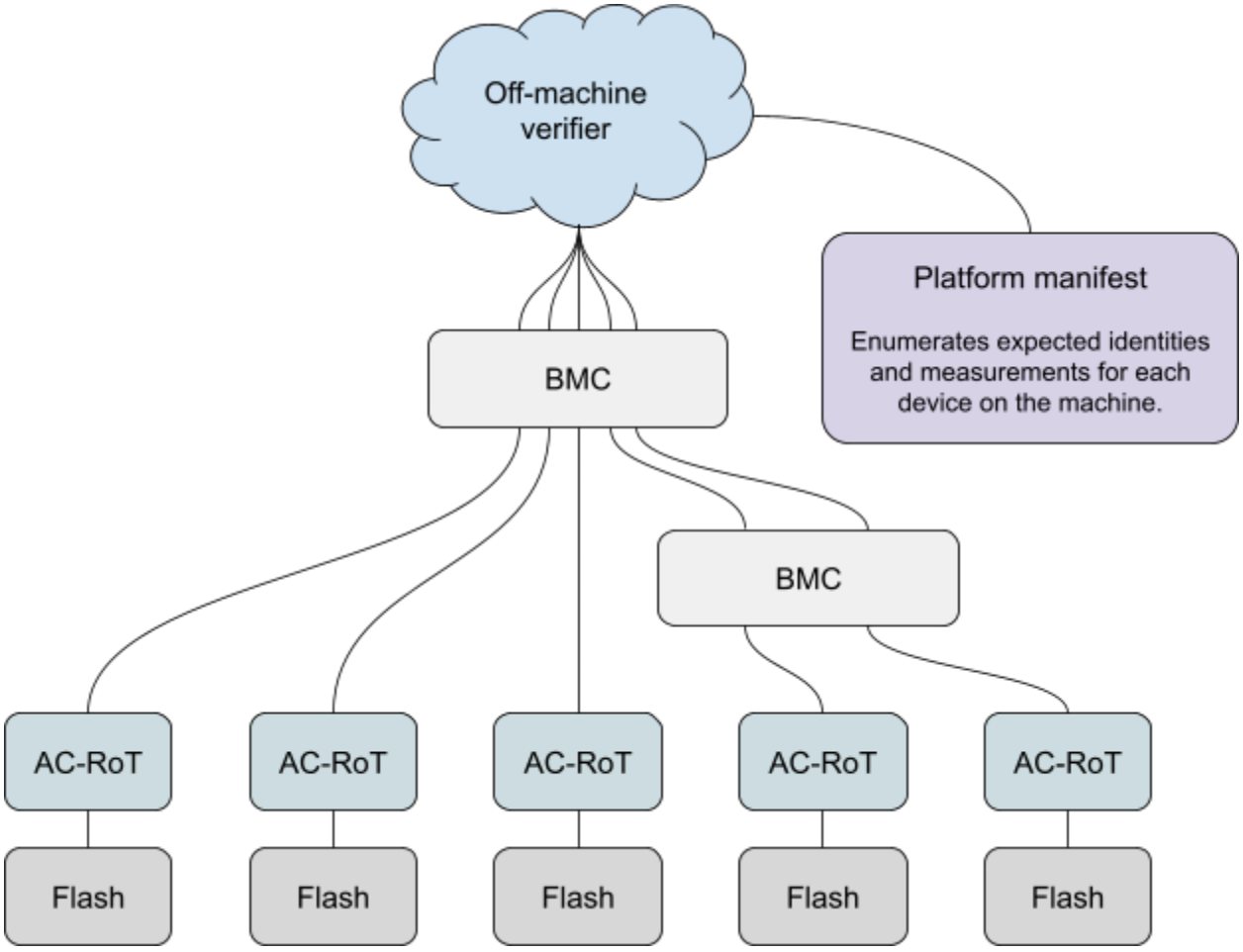
RTD architecture: on-demand discrete component attestation

In this architecture, a BMC forwards attestation challenges from an off-machine verifier to each AC RoT on the platform. The BMC is not trusted to aggregate the AC RoT measurements nor to evaluate whether they conform to a policy; the BMC merely passes the AC-RoT-signed measurements back to the off-machine verifier.

This architecture is suitable for platforms that do not feature a component that both (a) has the needed connectivity to aggregate measurements from every AC RoT, and (b) is considered trustworthy enough to do so. A platform may feature a BMC that is able to aggregate measurements, but platform operators may wish to trust the BMC as an availability-only actor; thus the BMC would not be suitable to act as a PA RoT. Likewise, a platform may feature an embedded-controller that is trustworthy enough to aggregate AC RoT measurements, but that lacks the intra-machine connectivity to do so.

The set of attested components may be managed by multiple BMCs, in the case of complex multi-tray machines or disaggregated hardware.

This architecture requires that each AC RoT be reachable out-of-band by way of the BMC.



RTU architecture: distributed enforcement

In this architecture, each device features an AC RoT that guards the device's update path. When a firmware update is requested, the AC RoT makes local policy determination as to whether the update should be allowed. This is typically done by ensuring that the update has been cryptographically signed by a public key provisioned to the AC RoT.

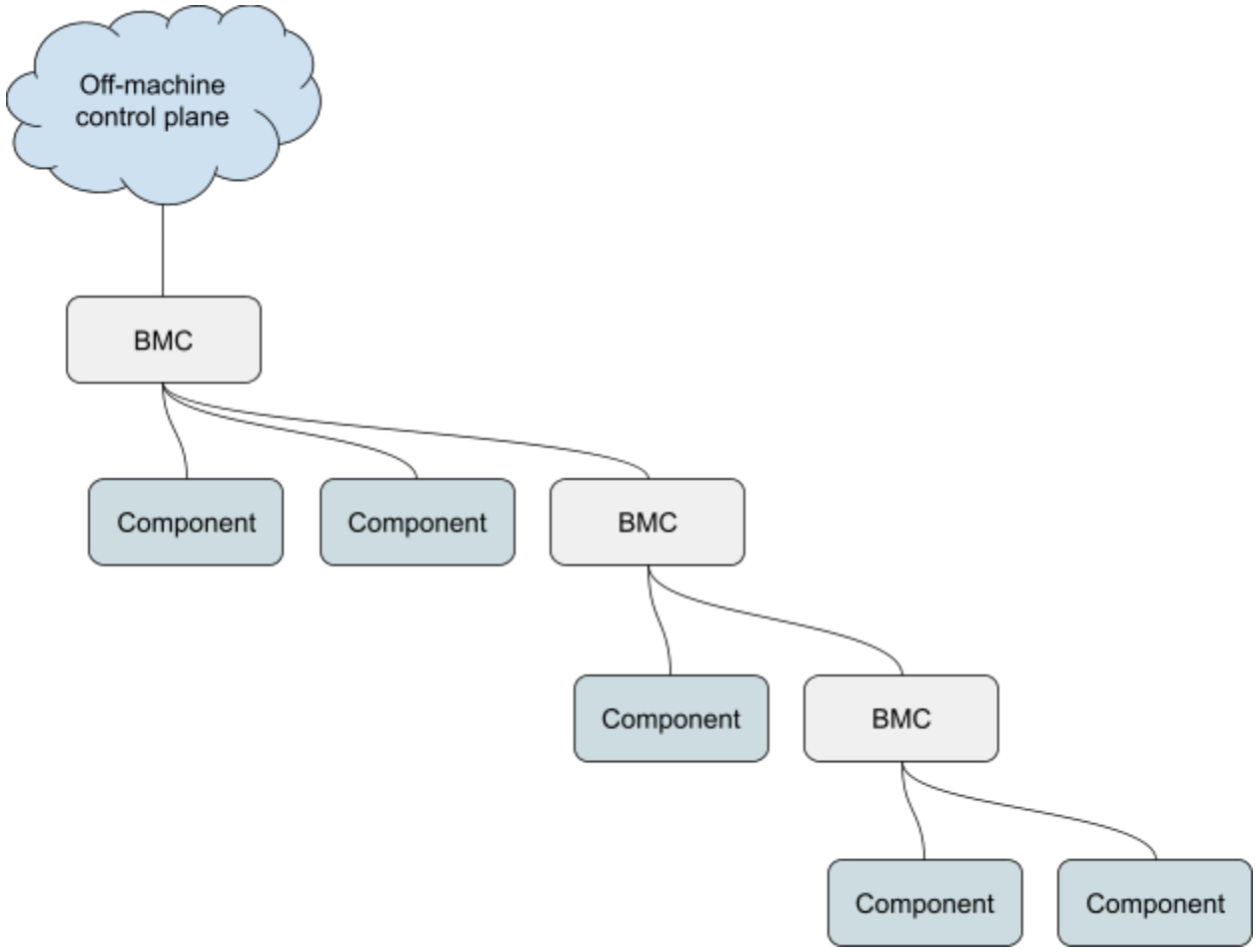
The AC RoT may also enforce anti-rollback, through use of an SVN. The minimum SVN may be stored in the AC RoT's internal fuse or flash bank.

RTRec architecture: hierarchical recovery

RTRec architectures are closely tied to RTU architectures, as recovery may be seen as a special flavor of update. It should not be possible to leverage recovery flows to put a device into a degraded security state as compared to what would be allowed in a standard update flow.

In this architecture, a machine's components are organized into a hierarchy, based on who is empowered to recover whom. The root of this hierarchy is the minimal foothold that must be functional in order to remotely recover the rest of the machine from a non-functional state. This minimal foothold typically consists of a BMC, as well as a discrete PA RoT if present. In complex machines, further BMCs may be present, with recovery authority over their respective boards.

TODO: symbiont devices (From SP 800-193: Due to lack of capability or functionality, some platform devices may not have their own root(s) of trust to perform an update, detection, or recovery. We refer to devices needing assistance as symbiont devices and those lending assistance as host devices. A depende



ncy may be established whereby a host device and a symbiont device jointly fulfill the

guidelines for protection, detection and/or recovery that the symbiont device cannot fulfill independently.)

Platform-Level Requirements

TODO: Discuss requirements from platforms in order to support having a RoT in them. (note: this might better be served as a separate spec, and point to it in the section above).

Break it down to system, motherboards, cards, etc, and what's required from each in order to get secured.

Interoperability?

Running OCP Servers in a Data Center

Establishing Trust in a System

TODO's:

- *Discuss platform boot behavior, when trust is established, point to the attestation spec to cover early/late attestation.*
- *Discuss system attestation by some external management plane in more details, and how this should work.*
- *How to balance secure boot and attestation to get to a trustworthy server*
- *Verifiers and what they need to do to attest systems and how do you know good from bad?*
- *How to reconcile with other ways to do that (e.g. TCG, what do you do if you have a TPM in the system), interaction between RoT and TPMs.*
- *Where to get trusted manifest values of good measurements.*
- *Different models of verifying.*
- *Implicit vs explicit attestation*
    - *Secret sealing, e.g., integrated RoT / discrete RoT that implements SPDM can also implement core TPM commands?*

Data Center Policies

TODO: Discuss how having various policies (automatic or ad-hoc recovery, what to do when secure boot fails, what to do when device attestation fails), can help manage systems at scale.

Which entity in the system needs to support which policies (note: this might feed requirements into other documents, or end up as requirements in this doc).

Circular Economy Considerations

TODO: Discuss the balance between secure boot and ownership, and how customers can achieve both.

Terminology / Glossary

[editor's notes] create a full glossary of terms used in this doc and other OCP docs, to avoid ambiguity and misinterpretations.

Maybe point to this glossary and get it done:
Glossary of Terms

E.g. terms to be well defined:
System
Platform
Component
Device / Peripheral
Secure Boot
Device Attestation
Platform attestation
Ownership
…

License

OCP encourages participants to share their proposals, specifications and designs with the community. This is to promote openness and encourage continuous and open feedback. It is important to remember that by providing feedback for any such documents, whether in written or verbal form, that the contributor or the contributor's organization grants OCP and its members irrevocable right to use this feedback for any purpose without any further obligation.

It is acknowledged that any such documentation and any ancillary materials that are provided to OCP in connection with this document, including without limitation any white papers, articles, photographs, studies, diagrams, contact information (together, “Materials”) are made available under the Creative Commons Attribution-ShareAlike 4.0 International License found here: https://creativecommons.org/licenses/by-sa/4.0/, or any later version, and without limiting the foregoing, OCP may make the Materials available under such terms.

As a contributor to this document, all members represent that they have the authority to grant the rights and licenses herein. They further represent and warrant that the Materials do not and will not violate the copyrights or misappropriate the trade secret rights of any third party, including without limitation rights in intellectual property. The contributor(s) also represent that, to the extent the Materials include materials protected by copyright or trade secret rights that are owned or created by any third-party, they have obtained permission for its use consistent with the foregoing. They will provide OCP evidence of such permission upon OCP’s request. This document and any "Materials" are published on the respective project's wiki page and are open to the public in accordance with OCP's Bylaws and IP Policy. This can be found at http://www.opencompute.org/participate/legal-documents/.
If you have any questions please contact OCP.

About Open Compute Foundation

The Open Compute Project Foundation is a 501(c)(6) organization which was founded in 2011 by Facebook, Intel, and Rackspace. Our mission is to apply the benefits of open source to hardware and rapidly increase the pace of innovation in, near and around the data center and beyond. The Open Compute Project (OCP) is a collaborative community focused on redesigning hardware technology to efficiently support the growing demands on compute infrastructure. For more information about OCP, please visit us at http://www.opencompute.org