Palantir —Apollo

Palantir

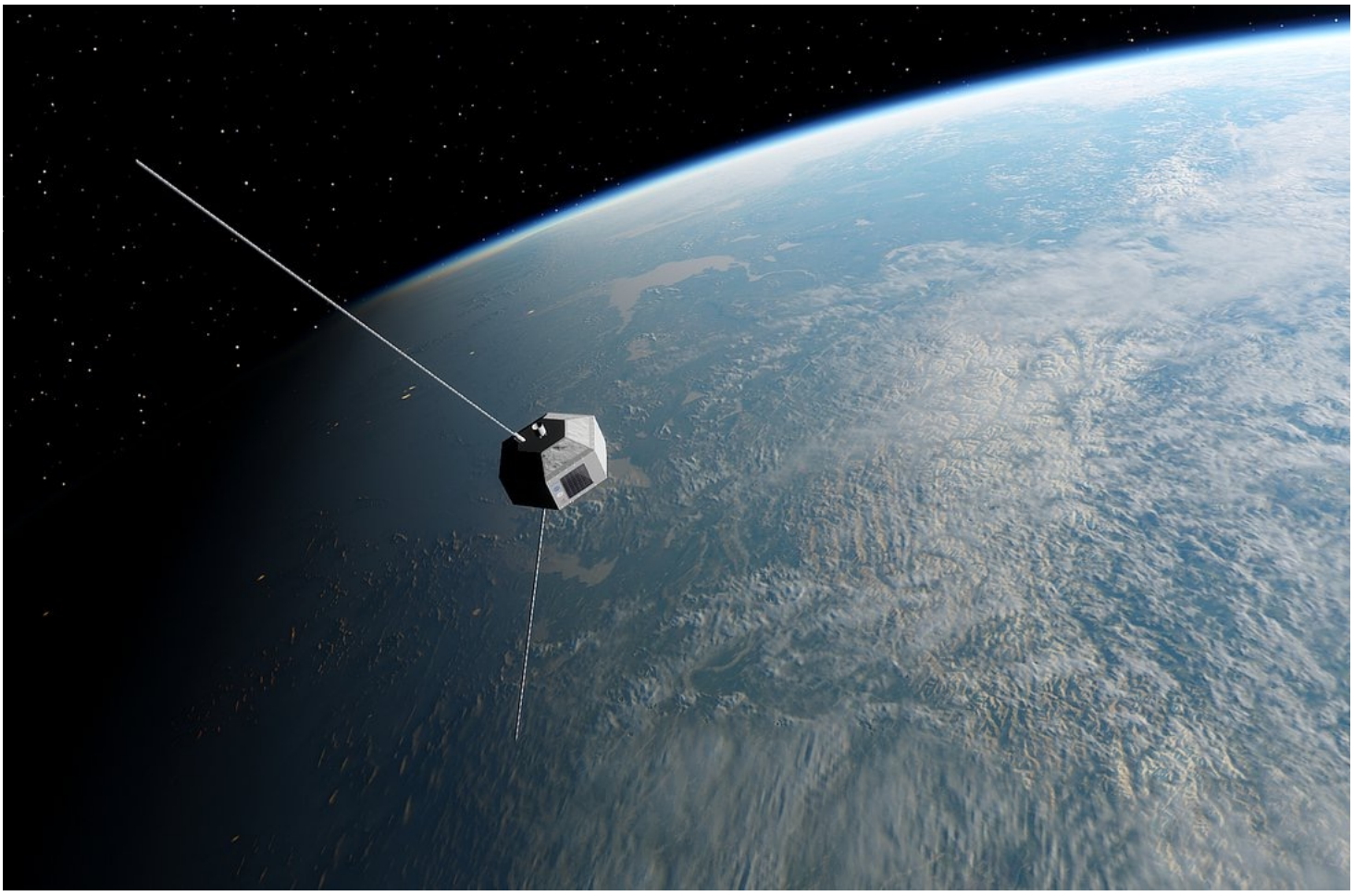
SOLUTION OVERVIEW

—WHITEPAPER

PALANTIR TECHNOLOGIES

PALANTIR.COM

COPYRIGHT © 2022



Copyright © 2022 Palantir Technologies Inc. and/or affiliates (“Palantir”). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Palantir Apollo —Solution Overview

Copyright © 2022 Palantir Technologies Inc. and/or affiliates (“Palantir”). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Introduction

An Operating System for Continuous Deployment →

From on-premise data centers to edge hardware and across clouds, Apollo enables the secure, autonomous delivery of your software to any environment, anywhere on Earth.

With Apollo, teams have a single pane of glass to monitor deployment health, coordinate the delivery of new features, customize platform configurations, and rapidly remediate issues.

Each layer of the Apollo platform serves to ease and expedite the journey of code from keyboard to operations; engineers write code once that works for all environments. This way, developers can focus on building new capabilities, not the nuance of deployment and management.

Maximize the reach of your software infrastructure with the flexibility of Apollo.

Problem Statement

An Exponential Expansion of Deployment Environments →

Historically, enterprise software was designed for and deployed to a single environment – on-premise data centers. Every organization had one (or a few) and it represented the totality of their computing footprint. Once software was installed, upgrades were infrequent because of the manual work and support burden required.

By 2016, the rise of cloud computing had changed this deployment paradigm dramatically. Companies rapidly embraced the SaaS model and micro-service architecture for the ease of only having to operate and deploy updates to a single production environment — powered by hyper-scale commercial cloud providers. “The Cloud” was discussed as a singular entity, with a singular grip on enterprise software.

However, six years later, and there is no single cloud. Instead, organizations are looking up and seeing a sky full of clouds: some public, some private and many that are hybridized. Several macro-level trends are combining to drive this expansion of delivery environments and change how software is deployed.

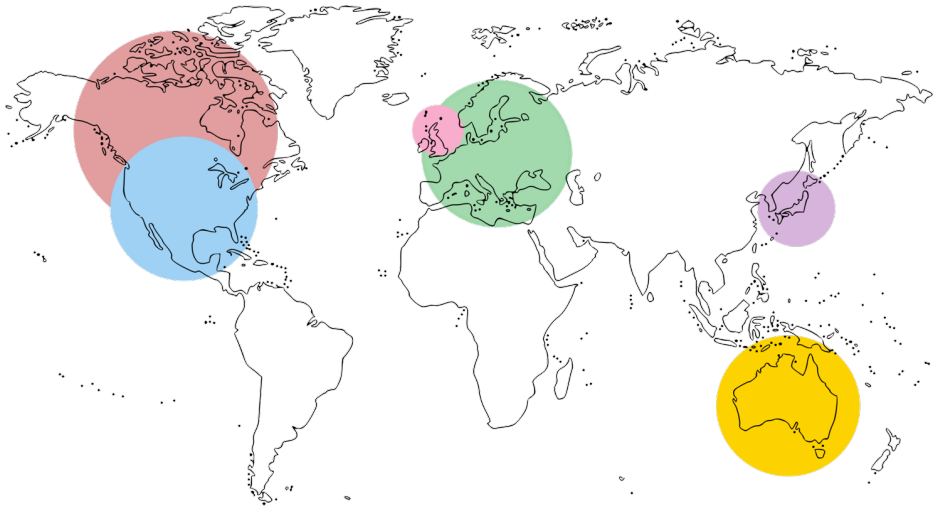
Here are some of those macro-level changes →

Copyright © 2022 Palantir Technologies Inc. and/or affiliates (“Palantir”). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Problem Statement

MACRO-LEVEL TREND	DESCRIPTION
Commercial Cloud Fragmentation	As organizations transitioned their software infrastructure to a cloud-first paradigm, they were met with ballooning fees from hyper-scalers. As the cloud infrastructure market becomes more saturated and competitive, customers are resisting vendor lock-in and opting for software that supports a multi-cloud paradigm.
Sovereign Clouds	Increasingly, government customers are requiring their software to run in purpose-built government-only or classified clouds that live separately from standard public cloud infrastructure; France, Germany, Canada, Australia, UK, U.S., and others are creating unique rules around how to deploy software for these environments. In some cases, these have been formalized into rigorous accreditations, such as the FedRAMP and DoD IL programs in the U.S. and Protected B in Canada.
Data Residency Restrictions	In response to stringent data residency legislation, such as GDPR, CCPA, and GxP, companies are requiring data to be hosted in-country – whether in physical data centers, private clouds, or region-specific public clouds – to assure regulatory compliance.
Need for On-Premise	An increasing number of companies are adopting cloud principles but need to stay on-premise for security, privacy, or financial reasons – but they still want the advantages of cloud speed delivery.
Rise of the Edge & IoT	Distributed computing is still in its infancy, but already organizations are having to adapt their deployment models to account for the unique challenges of delivering software to disaggregated, low-connectivity environments. This boom in physical infrastructure and hardware must be met with sophisticated solutions for software management.



Taken together, this expansion of delivery environments has narrowed the opportunities available to companies relying on traditional SaaS architectures. Organizations need flexible deployment capabilities to capture emergent opportunities, enabling them to evolve alongside their customers and support robust product growth — all while maintaining the speed and convenience customers have come to expect from the SaaS model.

Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Why Apollo?

Requirements to do CD at Scale →

An end-to-end CD platform

A change movement engine

An OS that aligns incentives

To keep up with expanding delivery environments, some organizations build continuous integration/continuous delivery (CI/CD) capabilities in-house. While this is altogether possible, doing so pulls valuable resources away from innovation and continued feature development, as time is spent stitching together a set of open-source CI/CD tools that were designed for a singular cloud world.

Instead, organizations need an integrated, end-to-end solution that can evolve in tandem with their business strategy. This is Apollo: capable of deploying your software wherever it's needed, whenever you need it.

REQUIREMENTS FOR CD AT SCALE	PALANTIR APOLLO
01 **An end-to-end Continuous Deployment (CD) platform** for centrally managing heterogenous versions of software across independent environments, regardless of what and where those environments are	No matter how remote, how many separate environments you have, or how complex your fleet is, Apollo will ensure all your latest features can get to any and all environments.
02 **A change management engine** for orchestrating software upgrades and changes safely across connected & disconnected environments	Apollo allows you to translate operational needs into specifications codified in the platform, and then does the hard work for you of figuring out what, where, and how everything across your fleet should be safely upgraded. It allows you to deploy with speed *and* stability, giving your customers both the newest features and 24/7 reliability, and giving your engineers rapid feedback loops.
03 **An operating system for your engineers and operators** to manage your software platforms across your fleet	Apollo gives you a 360° visibility into what is deployed where and a toolkit to rapidly respond to issues, and aligns incentives and practices across developers, operators, and security professionals.

Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

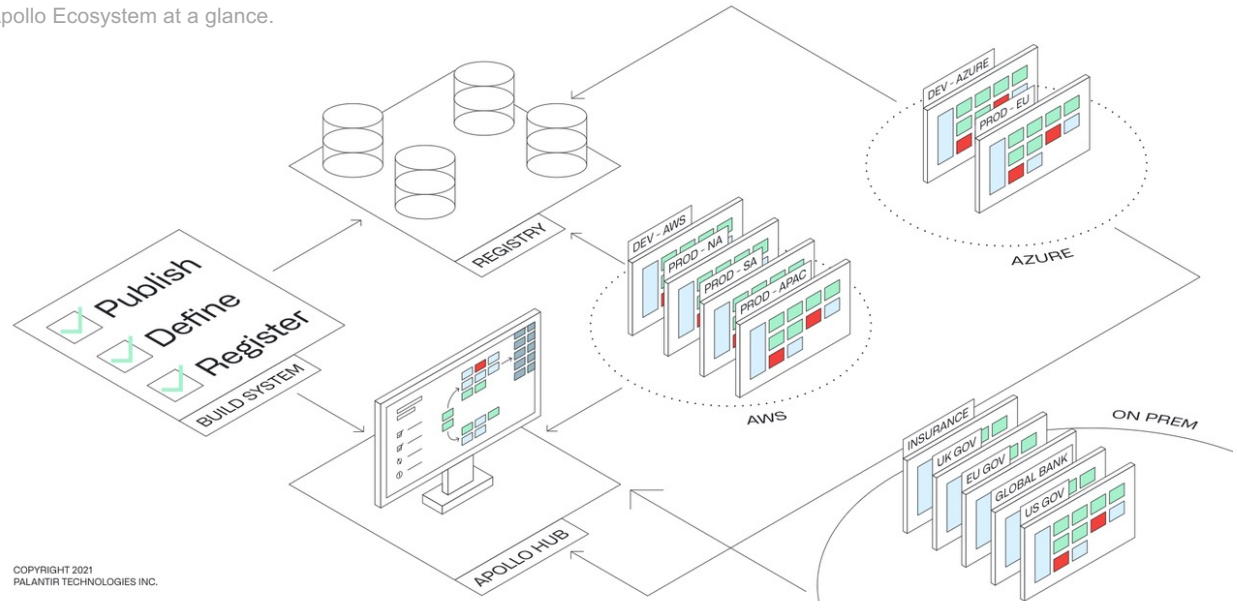
Components of the Apollo Platform

01—
The Building Blocks of the Apollo CD Platform →

So — how does it all work? Apollo is made up of a few key components.

- The ***Apollo Platform*** consists of an Apollo Hub deployed centrally and the Apollo Deployment Platform deployed per managed environment.
- The ***Apollo Hub*** is the central environment that maintains the status and unique requirements of each deployment environment and proposes plans to *Apollo Deployment Platform* for each of the environments it manages.
    - The ***Main Apollo Hub*** is the SaaS hub which manages all of your *connected environments*.
    - ***Remote Apollo hubs*** are Apollo Hubs that are designed for usage within remote or isolated networks to manage *remote environments.* Remote Hubs are kept in sync with the Main Hub using the *Apollo bundle.*
- The ***Apollo Deployment Platform*** runs alongside managed software in each environment and sits on top of Kubernetes (but also supports a container-less world). The Deployment Platform's primary function is to deploy and manage your software in the environment you would like to manage. To do that, it communicates with an Apollo hub to report the current state of managed software and receives work plans from it to execute.
- The ***Apollo SDK*** defines a standard framework to communicate metadata about each piece of software you want to manage. ***No code changes are required.*** The metadata provided via this framework is one of several ways that developers interact with and are able to customize the Apollo Platform to fit your business needs.

FIGURE 01
The Apollo Ecosystem at a glance.



Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Components of the Apollo Platform

01— The Building Blocks of the Apollo CD Platform [Cont.] →

— ***The Apollo Catalog*** is a metadata layer that bridges the gap between what's contained in your artifact store and your managed environments, connected or otherwise. The Catalog acts as a source of truth for all available software packages, so that it knows when new versions/patches are available and where to find them within your artifact store. This model gives your organization the comfort and security of knowing that **your code and software always stays within your existing source control and artifact repositories.**

  — This is where Apollo seamlessly integrates to your existing build systems / CI tooling. Developers simply merge changes into a given product repository and your CI build runs as usual, building a distribution containing the contents of the artifact and sending it to your existing artifact repository.

    — For example, you might use Github Enterprise as your version control system. A user merges a change into a Git repository, your CircleCI build runs, builds the distribution, and sends the artifact to Artifactory.

  — To start managing a product with Apollo, all you need to do is add a step to build process that pushes the release metadata to a service running in the Apollo Hub. **Your code and software always stay within your existing source control and artifact repositories; the Apollo Catalog simply gets the maven coordinates of the new release and other metadata defined in the Apollo SDK.**

02— The Apollo Orchestration Platform

Safe, granular coordination of upgrades across all environments →

The ***Apollo Orchestration Engine*** is at the heart of the Apollo platform, residing in the main Apollo Hub, and is what allows for autonomous orchestration and central management of change across heterogenous environments. It does the hard work for you of figuring out what, where, and how everything across your fleet should safely upgrade, and allows you to deploy with speed and stability.

— Thanks to the Apollo Catalog, the Orchestration Engine knows as soon as new versions and configuration changes are available. It computes what upgrades and actions should be performed based on the observed state reported by the Deployment Platform Services, available versions, preconfigured environment and service constraints, and any commands issued by the environment owners. It does so by evaluating where the target and observed state disagree and recommending change to converge the current state with the target state.

Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Components of the Apollo Platform

02— The Apollo Orchestration Platform

Safe, granular coordination of upgrades across all environments [Cont.] →

— Changes are proposed in the form of an Apollo Plan, consisting of a Plan Type and the necessary information for the plan to be actuated (such as the configuration change being made or the version of software an environment needs to upgrade to). Before the plan is executed, both the product constraints (such as the appropriate dependencies installed) or environment constraints (such as specific downtime windows) must be met.

— Once all of these conditions are met, the orchestration platform autonomously instructs an environment's deployment platform services to execute a given Apollo Plan.

— This happens **autonomously, after desired constraints and specifications are configured**.

Apollo's Orchestration Platform is especially powerful because it enables you to encode business and operational needs into the way it performs changes. The customizations and features give you unmatched flexibility in managing your software. **The features of the Apollo Orchestration Platform include:**

FEATURE	DESCRIPTION
Canary Analysis	Developers can benefit from dramatically shortened feedback loops by releasing new features to canary environments directly from the develop branch, which can be defined and customized to fit the target environment. Customers benefit from increased testing and rapid, safe feature releases.
Adjudication	Apollo automatically adjudicates releases by observing performance metrics and error states; then, gradually rolls out passing releases to the fleet, starting with internal installations and eventually reaching more conservative, mission-critical environments. For certain services and environments, it's pertinent to optimize for the latest features as soon as they're available, while others cannot risk de-bugging or downtime. With Apollo, each environment can be assigned an individual risk tolerance level to account for this balance.
Dependency Resolution	Developers can specify the dependencies and compatible version range for each service they deploy in the Apollo SDK. Apollo orchestrates upgrades in accordance with these compatibilities. Apollo also understands which releases require database migrations and chooses upgrade paths compatible with such constraints, instead of simply fast-tracking to the latest release which would leave the database in an incompatible state. — This means that even complex upgrades and migrations are hands-off for developers: they merely indicate the presence of a service or database migration in their release metadata and Apollo takes care of rolling out the release in a safe way.

Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Components of the Apollo Platform

FEATURE	DESCRIPTION
Blue/Green Deployment	Apollo employs a zero-downtime upgrade strategy by deploying a second instance of an application and slowly shifting traffic from the old instance to the new one. Apollo's release adjudication mechanism starts collecting performance and error signal as soon as the first node has been upgraded, yielding early indicators for the quality of a release. In this sense, blue/green deployment is not only a zero-downtime rollout mechanic, but also an extra safety net that allows Apollo to further accelerate the feedback loop for developers.
Release Channels	Environment owners can subscribe to a specific 'release channel' in accordance with their individual risk tolerance and appetite for new features. For example, environments subscribed to the 'develop' release channel might get new features the moment they're available. As Apollo gathers data on the deployment of this feature to 'develop' environments, it adjudicates whether a release is ready for promotion to the next release channel, e.g., 'stable.' Apollo comes with a set of predefined release channels and automated adjudication sequence between them, but it also allows you to create release channels and define the adjudication sequence according to your needs.
Maintenance & Suppression Windows	Developers and operators can specify when their products and environments should and should not take upgrades. — Developers and operators can define these windows at both the product and the environment level to encode certain operational constraints. They establish when changes should actually roll out to the fleet. For example, developers may want changes to a risky product to rollout only when there are people online and available to field any issues. On the other hand, environment owners may want to ensure that large upgrades only happen after hours on a critical production environment so users aren't affected. — To improve release safety, Apollo also automatically creates suppression windows when customized failure thresholds are met, such as surprisingly high disk space or memory usage by several services in an environment.
Recalling	If a particular release of an application has been identified as buggy, unstable, or slow, Apollo can recall the release and force the upgrade or downgrade to a known good version across all environments. — Recalls can be done by administrators and developers, or by Apollo itself: Apollo adjudication system monitors the health of each component over time (for instance by observing error logs and performance metrics) and **automatically recalls releases that appear unhealthy.**

Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Components of the Apollo Platform

03 — The Apollo Control Panel

Understand and manage software deployment across all services and environments →

The Apollo Control Panel is the user interface of Apollo that developers and operators interact with to perform workflows related to management, configuration, operations, and remediation of their software.

It enables the evaluation of rollouts through a powerful suite of tools to help operators and developers understand the risks associated with each rollout, surface problems, and ship code better.

Here are common workflows developers and operators love using Apollo.

FEATURE	DESCRIPTION
Management	Create and modify settings for: — Teams: membership, contact information, product ownership, who is on call — Products: ownership, adjudication configuration, soak time for blue/green configuration, default canary environments, maintenance windows — Environments: accreditation/compliance regime (affects how approvals and change management works within the system), ownership, operational responsibility (if 'on' then the responsible product teams will get paged for all products installed in the environment), maintenance windows, default release channel
Configuration	Manage service configurations on one or many environments — Edit the deployability properties of your installations (set a separate release channel, dependency overrides, update specific maintenance windows, etc.) — Service configuration can either be changed at the product level (across all installations), or at the environment level. At the per product level, developers can bulk edit existing installation configurations by seeing where they are deployed across the fleet, with what overrides, and edit multiple at once. — Configuration changes go through adjudication just like new features. This enables you to meet various audit requirements, quickly debug, and improve the stability of your deployed software. — The ability to compare, preview, and bulk edit installation configurations and the adjudication of all configuration changes takes significant error and overhead out of managing configuration.

Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Components of the Apollo Platform

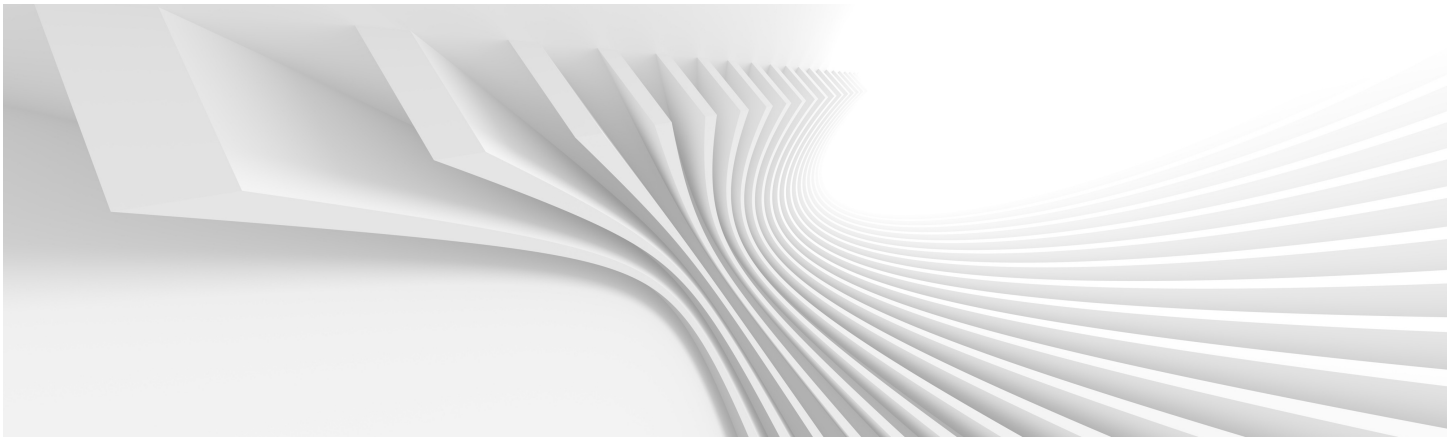
FEATURE	DESCRIPTION
Operations	Ensure actuation of configuration changes to individual environments and to the entire fleet: — Apollo will compute what changes should take place, try to perform them, and then report back if they were successful or why not. It calculates this based on information from health checks, metrics, logs, configuration, and its catalog of releases. Sometimes, changes are not successful for good reason; a human may need to take action to safely unblock an upgrade. — If you need to investigate further, the Apollo Control Panel helps you remove blockers to rollout to installations across fleet. Developers can check to ensure progress being made. One way Apollo helps with this is by reporting a ‘stale’ status in the UI. It indicates that a release is on a ‘stable’ release channel but hasn’t rolled out in more than five days. Some examples of what could cause this: — A dependency isn’t on an acceptable version for this product to be upgraded and its upgrade has also been blocked. The path forward is to reach out to the given environment or product owners (whose identity and contact information can be found in the UI) to get dependency upgraded. — There’s an active suppression window on the environment. Environment owners should be contacted in order to determine if this upgrade can be pushed through, or if the suppression window should be honored.

Copyright © 2022 Palantir Technologies Inc. and/or affiliates (“Palantir”). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Components of the Apollo Platform

FEATURE	DESCRIPTION
Remediation	Respond to production issues by tracking monitors and recalling bad product versions allowing your operators to take the appropriate actions to remediate. — **Handle potentially complex tasks with ease –** Apollo's visibility into how your software is deployed enables your operators to easily take action either across your fleet or a particular installation and quickly remediate issues in your production environments. Some common remediation actions include: — Graceful service restarts — Upgrade to latest version of product – There might just be a known bug for which there is already a fix — Recalling configuration and software versions — **Recall** – When you've found a bad product version, a recalled version will never be installed. If an environment is running one, then it will be transferred to a roll-off version defined by its roll-off strategy: — Stay on current version – often the first step in a high priority incident. Pauses all new installations and blue/green upgrades from accepting this version but keeps current ones on it as we investigate an issue. — Stay on current version with exceptions – can define specific environments that will get the newer than or equal to strategy. Helpful when testing hypotheses, when only seeing issues in specific environments. Usually, a temporary measure when debugging. — Any version newer than recalled version – often used in high priority incidents when rolling back isn't an option, and installations will stay in place until that version is available. — Any version equal to or newer than x.x version – roll back fleet to a previously known safe version.



Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Components of the Apollo Platform

04— Apollo Observability Platform

An optional observability platform for logging, metrics, monitoring and alerting of production services →

With Apollo's integration with popular observability and DevOps services such as Prometheus, DataDog, and PagerDuty, Apollo is able to seamlessly integrate with your organization's existing observability tooling and incident response processes. In addition, Apollo also offers its own suite of observability tooling that allows your organization to quickly bootstrap an observability platform if needed.

- **Break silos between Dev and DevOps –** With Apollo Monitors, your developers are able to codify what conditions should alert across all your environments where a given product is deployed, breaking down a traditional information silo that exists between Dev and DevOps teams.
- **Customize your alerting framework –** With Apollo's concept of Teams and Products, organizations are able to easily customize how alerts are triaged and routed to the appropriate team in a central place. So even if you have one team on PagerDuty and another one utilizing Slack or even JIRA, Apollo meets those teams where they are and ensures that they're able to receive the alerts they need.
- **Investigate using the Apollo Control Panel –** When a monitor fires an alert, users can go into the Monitors view for an environment to see what is actively firing and follow it to the Events view to see events related to the alert. Users can also view related logs and the latest results of queried health checks. From here, developers or operators may want to dive deeper into specific logs.
- **Debug** – Apollo has its own log exploration tool for analyzing product logs shipped to Apollo. It allows you to view logs, build charts, explore traffic, and diagnose.
    - View request, service, and other log content and drill down by Trace and Error IDs, types of logs, environments, and products
    - Build charts from request or service logs based on fields of interest with arbitrary filtering and grouping
    - Visualize and explore aggregated traffic between services with dependency graph views to trace issues across multiple services

Apollo can also *enhance* your existing observability platform. Industry standard observability tools are pretty good for looking at trends from metrics across your environments, and standard incident response platforms are good at notifying a set of people and following a schedule. However, a great incident response platform not only intelligently notifies the right people at the right time, but also arms them with the right information during all stages of the response process.

The centralization that Apollo provides, allowing system changes to be viewed at various levels alongside metrics and alerts, makes it far easier to answer questions such as, "was a metric spike due to a configuration change, an upgrade, or something environment-specific?" Intelligent, developer-defined notification also optimizes and streamlines the incident response process.

Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Security and Governance

Drive business growth and fortify software security →

Apollo integrates best-in-class security and governance controls into the same platform where software is delivered and managed.

This aligns incentives between developers, operators and security professionals — enabling organizations to ship code safer and further and driving business growth in the new disaggregated, digital risk-laden world.

Compliance-aware change management →

Apollo is built with the intention to deploy software to mission critical environments where privacy and security are of utmost concern, such as government and regulatory-heavy environments. These environments require strict controls which necessitate organizations to implement heavy processes and incur far more operational overhead, and often necessitate parallel engineering efforts which quickly start to lag behind public cloud offerings.

— Apollo enables you to unlock access to new business opportunities faster and more efficiently than ever before. It does the hard work so you don't have to — with its **compliance-aware change and operator access management**, your organization can auto-enforce the specific controls needed for any given environment. It also has a fully auditable history.

— Apollo is key to Palantir's ability to accredit and maintain Foundry & Gotham SaaS for mission critical National Security Systems (IL5, IL6, and higher). To earn these authorizations, organizations must comply with very strict change management and security controls. Palantir uses Apollo to meet a large subset of them.

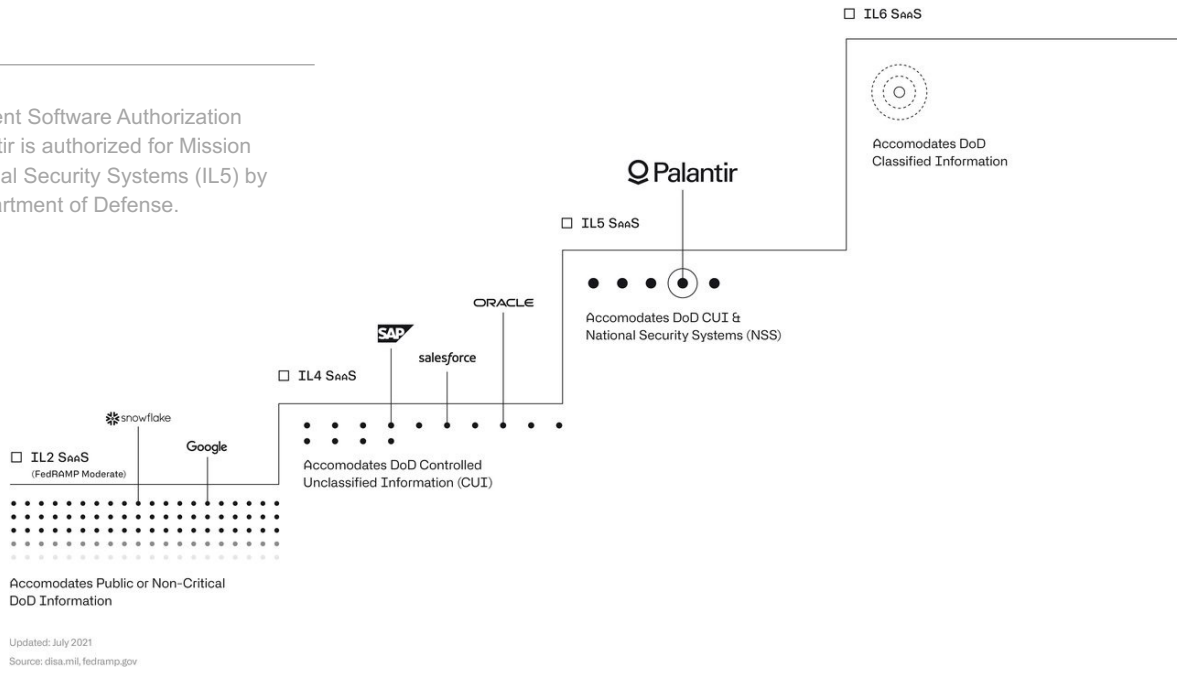


FIGURE 02
US Government Software Authorization Levels. Palantir is authorized for Mission Critical National Security Systems (IL5) by the U.S. Department of Defense.

Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Security and Governance

Encode security policies →

Apollo makes security and quality first-class concepts in continuous deployment, turning InfoSec, Quality, and governance compliance into a DevOps problem solved through Apollo. By encoding security and quality principals and controls into the Apollo Platform, developers no longer have to break out of their flow to rely on memory, emails, and runbooks to comply with InfoSec and Quality policies, and InfoSec/Quality teams no longer have to worry about security or quality being treated as an afterthought. As a result, they can focus on identifying new ways of fortifying your organization from the next attack and raising the quality bar, not enforcing established policies.

— An example of Apollo's functionality as an integrated DevSecOps platform is encoding InfoSec or compliance regime vulnerability remediation SLAs into the platform. Apollo has an optional service that is deployed in the main Apollo Hub which communicates with whichever industry standard scanner you use to scan new container images added to your registry. This service will be notified when containers fail scans, and it will automatically recall products with known vulnerabilities in Apollo. If a developer's product gets recalled, they'll be able to view this in the Apollo Control Panel and can click on a link in the recall message to see the results of the scan.

— Your InfoSec organization can define SLAs for remediating these vulnerabilities based on the severity of the CVE, e.g., 24 hours to patch an emergency CVE vs. 60 days to patch a medium CVE. This clock starts when a fix or mitigation is made available by the vendor, not when the issue happens to be discovered. Developers can also file suppression requests with InfoSec if they believe the failed scan is a false positive or the vulnerability is reasonable to leave unremediated. These features are valuable because they help you strike a balance between strict security controls and agility; if there is no available fix yet or there is a valid reason against it, engineering progress is not unreasonably blocked.

Software Supply Chain Security strategy →

Apollo should also form an integral part of any Software Supply Chain Security strategy. An important component of a Supply Chain Security strategy is a comprehensive software bill of materials (SBOM). Since the Apollo Catalog knows what you are deploying, and through the information provided via the Apollo SDK framework knows the dependencies for a given piece of software, the Apollo Catalog is a key enabler for quickly building out SBOMs. Supply Chain Security is about more than just what's in a package though — a critical component is also about what has gone where. The Apollo Platform gives you both: you have access to not only your entire SBOM, but you also know exactly where every part of it is deployed.

Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Interoperability

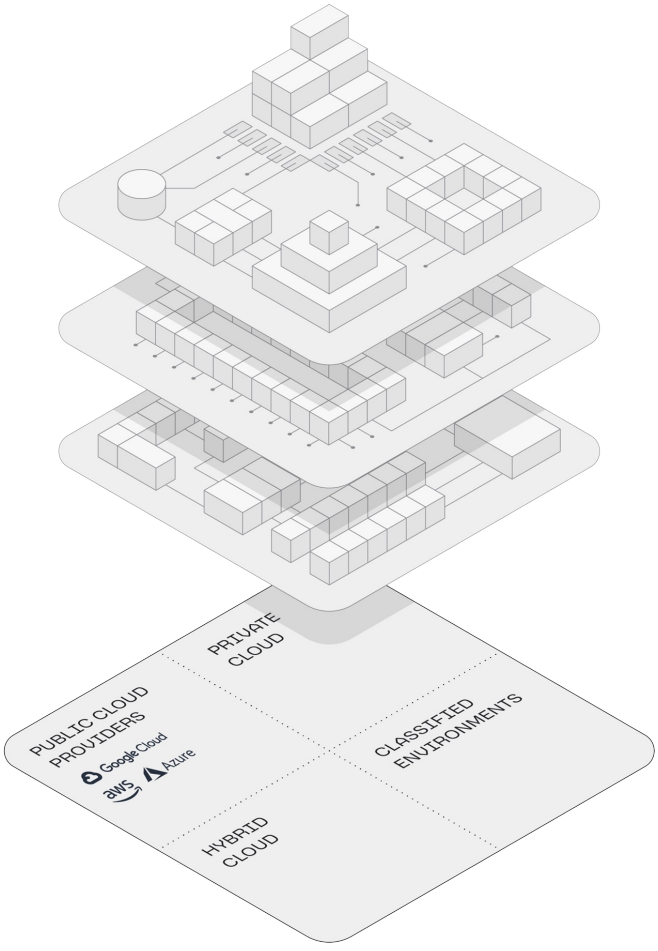
Seamless integration with existing version control →

Apollo was built to be both end-to-end and extensible, connecting seamlessly with existing version control, CI, and artifact repository tools in organizations' technical landscapes. Organizations can plug-and-play, selecting whichever platforms work best for pertinent workflows. For example, operators can use GitHub for version control, CircleCI for CI builds, and Artifactory as their artifact repository.

Apollo's interoperability extends not only to existing technical infrastructure, but also into the future as software transitions to a multi-cloud world. Apollo is cloud-agnostic, liberating organizations from vendor lock-in and providing unparalleled flexibility to meet changing needs and regulatory requirements.

Palantir Apollo can help organizations maximize the cost-efficiency of their cloud investments, while maintaining optionality for the future.

FIGURE 03
Infrastructure-agnostic architecture: Apollo connects with existing infrastructure and tooling today, and enables a flexible future.



Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.

Palantir

Apollo in Action

The Platform Behind our Platform →

We can attest to the utility and efficacy of Apollo because we use it to deploy instances of Foundry and Gotham worldwide, across a wide range of hosting environments.

Each platform is made up of hundreds of individual services, each owned by a development team that writes and releases product features continuously and independently. This approach allows us to roll out updates across services asynchronously, meaning we can meet our customers where they are, wherever they are.

Apollo enables this concurrent development without adding significant overhead or requiring specialized workflows: our software developers write code, Apollo deploys it, and our centralized operations team monitor the whole fleet from a single pane of glass.

FIGURE 04
By the numbers: Apollo's reach across environments and services.

250+
engineering teams at Palantir deploying with Apollo

300+
deployment environments, across on-premise, public, and private clouds

250+
services managed and shipped

Log4j Remediation →

This level of visibility into fleet-wide deployment health and continuity is critical for optimizing day-to-day operations, but it becomes existentially important during times of crisis.

With a click, Apollo autonomously remediated log4j vulnerabilities for Palantir and our customers – managing thousands of production service upgrades across 200+ environments, including on-premise data centers, edge hardware and classified networks.

All within hours.

While the severity and scope of log4j was unprecedented, such vulnerabilities are becoming more norm than exception. Apollo provides a single pane of glass for organizations to operate their infrastructure and strengthen their incident response plans at scale and pace.

Copyright © 2022 Palantir Technologies Inc. and/or affiliates ("Palantir"). All rights reserved. The information in this document is proprietary and provided for informational purposes only and shall not create a warranty or guarantee of any kind. Unless otherwise specified, all data contained herein is notional.