Platform Autopilot

Handbook

# Intent

This document aims to provide an overview of the *Platform Autopilot* service offering - what it is, the problems it aims to solve, and how it works.

Platform Autopilot is an offering in the *Platform Ops* space. It is designed to provide the outcomes of a patched, available, and up to date platform. Platforms that are regularly patched are protected from security vulnerabilities, providing a safe place for workloads to run. In order for the platform to be a suitable place for these workloads, it must also meet the appropriate level of availability for both developers and end-users. Lastly, platforms that are up to date provide the latest and greatest features and efficiencies to application developers. There are many ways to achieve these outcomes; Platform Autopilot is tailored towards customers who would prefer that *we* do the operations work to achieve these outcomes for them, leaving customer people the time and space to focus on building out the developer experience on top of a stable platform.

# Customer Onboarding

Platform Autopilot is a subscription service where we perform the patching, updating, and incident response for customer platforms **running on customer infrastructure.** As such, there are some prerequisites and onboarding steps that must be completed in order to prepare a new customer for Platform Autopilot.

## Platform Installation

Installation of the platform is not in scope for Platform Autopilot engagements. As such, there are several scenarios where we will recommend a *Platform Rapid Deploy* engagement to precede the Platform Autopilot service:

* New customers who do not yet have a platform installed
* Existing customers who are 2 or more minor versions behind the latest release and want help getting caught up
* Existing customers who are looking to replace their current platforms (data center migration, new IaaS provider, etc.)

## Sandbox Environments

We require that Platform Autopilot customers have one sandbox environment per IaaS. This environment must be free of customer workloads, as it will be used to validate and roll out the change in the customer environment. For example, a customer who has 4 deployments of our platform installed on vSphere must have 1 sandbox deployment that runs on the same vSphere environment. A customer who has 10 deployments split across vSphere and AWS must provide 2 sandbox environments, 1 in vSphere and 1 in AWS.

## Management InfrastructureTelemetry

Platform Autopilot customers are required to opt into our Telemetry Program for the platforms that are managed by the service. The data collected under this program is specific to product usage and does not include intellectual property or sensitive information. In the case of Platform Autopilot, this data helps us to proactively identify issues that may affect availability (e.g. certificate expiration) and prevent them from occurring. For more details on the data that is collected, how it is used, and how it is secured, please refer to <INSERT THE LINK HERE>.

Onboarding into the Telemetry Program is a three-step process. First, we will introduce the customer to the program and answer any questions that arise. Next, we’ll issue the customer a unique API key via secure file transfer. Lastly, we will begin to collect the data on a regular basis.

## Infrastructure Access

We require a service account with read-only access to the customer’s IaaS account in order to facilitate troubleshooting and incident response activities.

## Kickoff

We plan to initiate Platform Autopilot engagements with a visit to the customer site for a [2-3 day] kickoff. The goal of this visit is for the Pivotal team to leave with everything needed to begin delivering the service. This includes a mutual understanding of how we reason about change, initial service level objectives for patching, updating, and availability, and various logistical concerns around remote access, points of contact, and ongoing activities.

<> prep work required, visitor access, a room to set up in, the right attendees included, IaaS accounts?

<what activities we will do> SRE intro, SLO workshop, infrastructure deployment/validation (concourse, teleport, etc), support onboarding?,

# Delivery Team Staffing

Platform Autopilot is designed to be staffed by a shared pool of individuals. These individuals will not be dedicated to a single customer, but will split their time across multiple customers.

* What does this mean for onboarding in customer systems?
* How do we handle background checks, drug tests, etc?
  + Who is the vendor
  + Nature of the checks that are done
* Who will have access, for how long (how do we revoke it)
* Employee workstations and security thereof
* Other Geos?

# Remote Access

The Platform Autopilot service is delivered remotely. There are generally two scenarios where remote access will be required for our employees to work with customer infrastructure:

1. Planned activities (i.e. patching and updates)
2. Unplanned maintenance and troubleshooting, such as during incident response

Planned activities such as patching are performed via automation, which is driven by changes to a source code management system. This provides reproducibility in the environment(s), an audit log of who initiated changes (along with commentary as to why they were made), and a consistent approach to rolling out said changes

<graphic showing source control repo, automation, etc.>

<where does the git repo live? Our account?>

Unplanned incident response or other activities that require an interactive session will be performed via a jump host in the customer environment. We leverage a tool called Teleport for access to this host, which ensures that all sessions are performed from individual accounts with strong multi-factor authentication, and also captures the full session history for auditing purposes.

It is a design goal of Platform Autopilot that all connections between the customer environment and our network are initiated from the customer environment. In other words, we will not initiate connections into a customer network. For the remote access activities that are automated via CI/CD tooling, we will leverage Concourse’s remote worker feature to achieve this objective. Teleport provides similar functionality for remote jump host access.

<add graphic showing network architecture>

# Credential Management

Because we will rely heavily on automation to perform operational tasks, a secure credential management solution is necessary to ensure that the automation tooling has the secrets and access required to do this work. We plan to rely on our CredHub product for credential management. Additionally, we will deploy a CredHub instance for each customer *in the customer environment*. This ensures that all credentials needed to access customer environments are stored in customer environments and not on shared VMware infrastructure.

# Monitoring / Alerting

## Monitoring Data and Visualizations

The most important data that we will collect for the purposes of monitoring and alerting is that of our service level objectives:

* Patch Level
* Update Level
* Application Availability
* API Availability

This data is high level status (up/down), and does not contain any sensitive or customer identifiable information. We plan to store this data in a time series database that lives in VMware infrastructure, so that we can apply our dashboard and alerting tooling on top of it. We will deploy a separate database per customer so that this data is isolated from other customers.

<TODO: retention, encryption at rest, etc.>

<TODO: choice of TSDB>

If the customer is also interested in access to this data (for either auditing purposes or developer transparency) we can deploy a separate visualization tool (Grafana) in the customer environment that pulls from the data source, and instantiate the same set of dashboards and queries.

## Logging

Logs emitted from customer platforms will not be exported to VMware systems. We prefer to leave this data in the customer environment, because:

1. The volume of this data can be quite high, and
2. Logs may contain sensitive information that should not leave the customer environment

In order for the delivery team to effectively respond to incidents and remediate any issues, we may require that the customer use one of our approved logging systems. This will ensure that the team is familiar with the logging tool and can be effective in using it to triage and remediate issues as quickly as possible. The approved logging tools are: X, Y, and Z.

# Patching and Updating

The Platform Autopilot team uses Site Reliability Engineering (SRE) practices as a way to reason about balancing the availability of the system with the rate of change introduced to the system. We will work with each customer to establish an error budget that allows us to define and achieve the appropriate level of availability, and [extend this model](https://www.usenix.org/conference/srecon19americas/presentation/thomson) to include a vulnerability budget (patch level) and a legacy budget (update level). We will do this for each product and/or service plan that we offer in the environment.

We believe that for most customers, protection from the latest security vulnerabilities (patch level) is more important than access to the latest features (update level) or availability. As such, we will generally roll out patches automatically in order to meet our customers’ goals for patch level. For minor version updates we will plan the date at which updates are applied based on the customer update goal and the amount of error budget remaining.

We validate change by first rolling it out to a sandbox environment that is running one or more “canary” apps - workloads that are representative of real customer workloads but take no production traffic. After validating <TODO promotion path…>

* Is there an “incubation period” in each environment before promotion?
* Is there ever a manual approval needed (prod for example)

We understand that in some industries there are times when availability becomes extra important, and customers may elect to prioritize availability even if it means failing to achieve their update goal.. For example, in the retail industry most customers will do whatever is necessary to keep revenue-generating systems available, even if it means that their developers will have to wait an extra month for access to the new features in the next platform update. We accommodate these periods by allowing Platform Autopilot customers to request that we cease updates for *up to 3 months,* once per year.

## Additional Customization Work

Platform Autopilot is scoped to the patching, availability, and updates of the platform. It includes an additional 20 hours per month for any “on-demand” work outside these three goals. For example, customers could request that we use this time to:

* Install and configure new tiles or isolation segments
* Integrate with additional third party systems for logging, identity, etc
* Modify the configuration of the platform to support new features or workloads

TODO:

* How will we track this work?
* How do we handle requests that exceed the 20 hours per month?

# Incident Response

In order to maintain the availability of the system, we will need to step in to diagnose and remediate any unplanned issues that may occur. There are two ways such activities could be initiated.

## Customer Initiated Incident Response

If the customer identifies an issue, they will leverage their [existing 24x7 support services agreements](https://pivotal.io/support/support_terms_conditions). Our support team will be able to page the on-call members of the Platform Autopilot staff who will perform the incident response.

* Work with support to come up with a plan for how much triage they do before looping us in, how they identify a customer as a Platform Autopilot subscriber, etc.

## Pivotal Initiated Incident Response

Monitoring and alerting will be configured to page members of the Platform Autopilot team when the availability of the system is threatened. When this occurs, the team will perform the incident response and will notify the customer as necessary.

## Incident Handling

An incident is any *unplanned* disruption (or degradation) of service that impacts users’ ability to interact with the system. In the case of a platform, the user could be a developer trying to deploy new workloads, or an end user attempting to access an application that is running on the platform. Our incident response process prioritizes restoring service as quickly as possible by establishing an environment where ...

TODO: we can probably be high level here and leave the finer details to a separate effort..

* Initiating an incident
* Identifying severity
* During the incident (emergency mode)
* Describe minimum roles
* To what extent will we update the customer? Will we permit them to be present?
* Describe the need for a path to loop customer people in if we discover/require off-platform troubleshooting

# Postmortems

After an incident is resolved, we will conduct a *postmortem*. The purpose of this process is to analyze the data from the incident in order to learn from the experience and prevent such an event from recurring. The output of this process is a document that details an analysis of what went wrong, how the issue was detected, what steps were taken to resolve the issue, and what actions will be taken to prevent a recurrence. We aim to complete all postmortems (including a review with customer stakeholders) within 3 weeks of the incident resolution.

* How we Identify an owner to drive the process to completion
* Document: incident duration, impact, etc.

# On-Call Policy

In order to maintain the availability of customer systems, members of the Platform Autopilot delivery team will be required to take part in on-call rotations. When on-call, an employee must be able to be contacted at any time in order to diagnose, mitigate, or fix any incidents that may occur.

The goal of our on-call policy is to ensure that we’re able to achieve our availability goals for the systems under management. It is important that we do this without sacrificing the health of our team members, so we aim to minimize the amount of pages the team receives. This ensures that true emergencies are treated with the appropriate sense of urgency, and that the team is set up to perform well when incidents do happen.

* What tooling will be used for paging, what integrations does this require on the customer side?
  + Can we leverage Pivotal’s PagerDuty subscription?
  + Does VMware have other tooling?
* Logistics
  + How long are on-call shifts? (cap at 8-12 hours?)
  + How many people on-call at any given time?
  + How will on-call staff acknowledge a page?
  + What happens if someone fails to answer a page, what’s the backup plan?
  + Tracking shifts, etc.
* Escalation path? (support, R&D, etc.) - understand existing mechanisms
* Compensation: several options
  + Incentivized: extra time off, flexible hours, higher salaries, etc.
  + Paid (scheduled): direct compensation for on call shifts, even if no issues arise
  + Paid (issues): compensated only for time actually spent working on issues (hourly, per incident, etc.)
  + Often compensation is tied to the “time to keyboard” requirement. 5 minutes is more demanding than 30 minutes.

TODO: create and link to separate internal on call guide?

* How to prepare for a shift:
  + what to validate ahead of time (notification settings, credential access, etc.)
* Helpful info:
  + Escalation paths
  + Overview of monitoring systems, common queries, etc.

# Interactions

* How does a customer platform team interact with the platform autopilot staff?
  + What should they focus on (feedback, dev onboarding, path to prod)
  + What if they want to also be involved in ops work?
* Is there an Anchor dedicated to a customer, or a CL, or what? Who should the customer reach out to?
* What are the regular activities expected of the customer (weekly sync, review of post-mortem, etc.)
* How does a customer interact with us when they wish to use their customization budget?
* Shared slack channel or other communication tool? (describe desire to avoid ticket-based system)

# SOC-2 Compliance

We intend to achieve [SOC-2 certification](https://www.aicpa.org/interestareas/frc/assuranceadvisoryservices/aicpasoc2report.html) for our Platform Autopilot offering. The purpose of SOC-2 is to evaluate and report on an organization’s information systems with respect to security, availability, processing integrity, and confidentiality or privacy..

TODO: type 1 vs type 2 SOC-2 engagement?

* Type 1: describes a vendor’s systems, whether the design is suitable to meet principles
* Type 2: details the operational effectiveness of those systems

# Needs

The needs we must satisfy are derived from the [ASEC *Trust Services Criteria*](https://www.aicpa.org/content/dam/aicpa/interestareas/frc/assuranceadvisoryservices/downloadabledocuments/trust-services-criteria.pdf)*,* which is broken down into five principles:

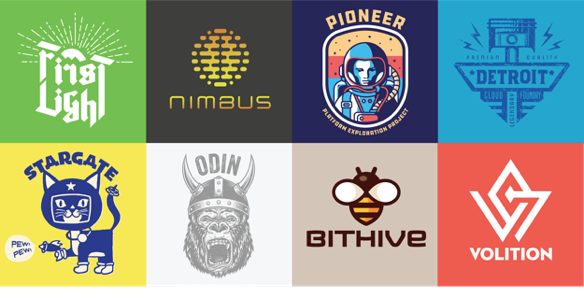
* Privacy: collection, use, retention, disclosure, and disposal of personal information
* Security: access controls, multi-factor authentication, intrusion detection, etc.
* Confidentiality: data protection, encryption, firewalls, etc.
* Availability: accessibility of the system (monitoring, failover, incident handling)
* Processing Integrity: data processing is complete, valid, accurate, timely, and authorized

Note that not all elements of these criteria are suitable or relevant to the services performed under Platform Autopilot. For example, Platform Autopilot operates on systems that run on customer-provided infrastructure.

* Remote access
* Authentication/authorization (MFA
* Traceability/auditing
* Credential management (isolation)
* Alerting
* Log retention?

# Branding?

If the customer does not have a Platform Brand, will we consider creating one as part of the Platform Autopilot?



### 

### HEADING 3

Some medicinal phenols show through toward the finish. Finish dries the palate, with a lingering wheat and barley husk.

#### **HEADING 4**

Yo

# 

Here’s a pull quote:

“Pours fizzy, creating a beige/eggshell-colored foamy head thats dense image and wonderfully laced, sticking to the glass and also retaining magnificently. In fact, the lacing stays until the end. Beneath, a deep brown brew with rich tawny hues. This beer has some serious legs. Complex aromas: soft and powdery on the nose, with aromas of malt, chocolate chip cookie dough and a deep-rooted fruitiness, notes of plum skins, spicy phenols and a soft bready yeast character..”

– Beer Ipsum

Tasty,

# Heading 1

Sometimes we need numbered lists:

1. The first item
   1. A sub-item
   2. Another subitem
2. The second item
   1. A sub-item

# Heading 1

Sometimes we need bullets:

* An item
* Another item

The bulleted lists use squares!