salesforce platform

Salesforce Heroku Enterprise:

A Cloud Security Overview

June 2016

Contents

Introduction

Heroku Enterprise, a key component of the Salesforce Platform, is a cloud application platform used by organizations of all sizes to deploy and operate applications throughout the world. The Heroku platform is one of the first cloud application platforms delivered entirely as a service, allowing organizations to focus on application development and business strategy while Salesforce and the Heroku division of Salesforce focus on infrastructure management, scaling, and security.

Heroku applies security best practices and manages platform security so customers can focus on their business. Heroku inherently protects customers from threats by applying security controls at every layer from physical to application, isolating customer applications and data with its ability to rapidly deploy security updates without customer interaction or service interruption.

Heroku Enterprise delivers key enterprise capabilities to the Heroku developer experience:

- **Collaboration**. Heroku helps teams build apps together via shared workspaces.
- **Trust**. Enhanced-support SLAs and a solutions architecture team help guide customers through major traffic events, application architecture design, white-glove onboarding, and more.
- **Visibility**. Unified tracking across application portfolios improves resource management and consumption.
- **Control**. Seamlessly integrated permission sets give developers and administrators the access privileges they need without slowing productivity.

Salesforce Trust Model

Salesforce understands that the confidentiality, integrity, and availability of our customers' information are vital to their business operations and our own success. We use a multilayered approach to protect that key information, constantly monitoring and improving our application, systems, and processes to meet the growing demands and challenges of security.

Security responsibilities and awareness are communicated on multiple levels. Both Salesforce and Heroku ensure that customer data protection are a core principle in our service delivery. Salesforce has multiple organizations, teams, and individuals responsible for security and security-related matters. The chief trust officer is responsible for Salesforce's security program and personnel, including: information, product, and corporate security; enterprise risk management; and technology audit and compliance. The Global Privacy Counsel is responsible for Salesforce's privacy program, including compliance with applicable privacy and data protection laws. Additionally, all Salesforce personnel are required to follow Salesforce confidentiality, privacy, and information security policies.

For more information, please see the following:
trust.salesforce.com
heroku.com/policy/security

Cloud Computing and the Shared Security Model

As defined by the National Institute of Standards and Technology (NIST), cloud computing is “a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.” There are three standard service models: software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS). Although Salesforce is often associated with helping define the SaaS category, the Salesforce Platform and Heroku Enterprise are examples of a PaaS offering.

With a PaaS offering, there is a shared responsibility between the service provider (Salesforce) and the end customer in order to provide end-to-end security and compliance. The graphic below shows a high-level abstraction of the line of control between the provider and the customer or tenant.

	IaaS	PaaS	SaaS
Applications	Tenant	Tenant	Provider
Data	Tenant	Mixed	Provider
Runtime	Tenant	Provider	Provider
Middleware	Tenant	Provider	Provider
OS	Tenant	Provider	Provider
Virtualization	Provider	Provider	Provider
Servers	Provider	Provider	Provider
Storage	Provider	Provider	Provider
Networking	Provider	Provider	Provider

Salesforce hosts the Heroku cloud application and data services platform on Amazon (a third-party infrastructure provider), and utilizes Amazon Web Services (AWS) to compose part of the underlying infrastructure layer for Heroku. In this configuration, Heroku is responsible for the complete provider obligation to a customer, although it leverages AWS capabilities in some areas to help deliver on these capabilities.

Provider Responsibilities

As the service provider, Heroku controls the primary management of the following areas and uses reasonable efforts to provide:

Infrastructure and application security. Heroku designs, deploys, maintains, and is responsible for securing many of the underlying compute services, including the virtual machines and application microservices that support a customer's runtime. Heroku is responsible for the elements to which it has sole administrative access on behalf of the customer.

Security monitoring. Heroku runs a comprehensive set of security monitoring and event-logging tools and policies for the infrastructure and platform services to which it has sole administrative control.

Patch management. Heroku maintains the operating systems and applications it uses to deliver the cloud application platform, including the application of patches it deems critical to these systems and applications.

Network security. Heroku protects networks containing the information systems that run customer applications. Customers assume responsibility for the level of access, control, and configuration of their application environments which they directly manage.

Physical security. Heroku, both directly and through the third-party cloud provider that it leverages for IaaS, is responsible for protecting the physical data center, both in terms of physical as well as environmental controls.

Tenant Responsibilities

As the tenant, a customer using the Heroku platform retains critical responsibilities to complete the end-to-end security of tenant applications:

- *Application design.* The customer is responsible for the secure design of applications running on Heroku, including access to the data and system configurations that are controlled by the application. This includes, but is not limited to: proper identity access and management; end-to-end encryption beyond TLS termination; security handling and storage of certifications provided by Heroku; and the roles and permission granted to an application's end users.

- *Security monitoring.* The customer is responsible for the detection, classification, and remediation of all security events occurring within an application or database environment that is configured and controlled by the customer, including any monitoring tied to compliance or certification programs that the customer is participating in and that are not covered by the infrastructure and services managed by Heroku. Note that Heroku provides security monitoring of Heroku-managed data services such as Heroku Postgres and Heroku Redis.

“ *I don't need to worry about patching, adding additional storage, or security vulnerabilities. The Heroku platform is managed and supported by the Heroku team.* ”

LEELA PARVATHANENI
Senior Manager, Doctor Portal and Business Support
Align Technology

Infrastructure and Application Security

Server Hardening

Heroku hardens its OS images (stacks) by turning off all services and components by default, and only enabling those that have a justified business and operational purpose.

Customer Applications

Each application on the Heroku platform runs within its own isolated environment and cannot interact with other applications or areas of the system. This restrictive operating environment is designed to prevent security and stability issues. These self-contained environments isolate processes, memory, and the file system using Linux containers (LXC), while host-based firewalls restrict applications from establishing local network connections.

> For additional technical information, see:
> **devcenter.heroku.com/articles/dyno-isolation**

Dyno Container Hardening

Heroku takes several precautions within the kernel to reduce the risk that a kernel bug could lead to the compromise of a host. Heroku filters syscalls using secure computing mode (seccomp), a security facility that provides application sandboxing in the Linux kernel. Containers are configured to explicitly deny root inside containers and drop several capabilities while executing inside containers, including the ability to mount filesystems or load kernel modules.

There are some distinctions between the Heroku Common Runtime environment and the Heroku Private Spaces Runtime environment. Heroku Common Runtime is a blend of both multitenant and single-tenant architecture, and containers leverage AppArmor – a mandatory access control (MAC) system that uses Linux Security Module to restrict programs and protect Linux services. Heroku Private Spaces Runtime provides complete isolation between tenants, with a single container per compute instance and the entire runtime operating on dedicated instances.

Application Security

Heroku undergoes extensive internal penetration tests, vulnerability assessments, and source code reviews to assess the security of the application, architecture, and implementation. Third-party security assessments cover all areas of the Heroku platform, including testing for OWASP Top Ten web application vulnerabilities, and customer application isolation. Heroku works closely with external security assessors to review the security of its platform and applications and apply best practices.

Heroku obtains vulnerability data through a variety of sources to detect vulnerabilities and deploy patches. These include periodic red-team audits, external and internal penetration tests, a bug bounty program (bugcrowd.com/heroku), and a collection of open-source community vulnerability data. Heroku is also implementing internal and external scanning to supplement these programs.

Issues found in Heroku applications are risk ranked, prioritized, assigned to the responsible team for remediation, and Heroku's security team reviews each remediation plan to ensure proper resolution.

> Customers may scan their own applications on Heroku, but are requested to notify Heroku prior to any external testing:
> **devcenter.heroku.com/articles/pentest-instructions**

Heroku practices a secure development lifecycle, and the Heroku Security team works closely with engineering teams from architecture review through source code review and penetration testing of all high-risk features. The security team is included in quarterly planning to track the security implications of upcoming feature work and identify the highest-risk projects. Projects identified as high risk are tracked for security team review. The process includes an initial security assessment, guidelines for secure coding, periodic red-team audits, and external and internal penetration tests, all performed or supervised by the security team.

Heroku leverages a platform integration testing service that runs end-to-end tests from the perspective of external Heroku users, a process that exercises the different services needed to deliver user-facing functionality. This service supports both one-off test runs and continuous running of tests for monitoring purposes. Heroku also uses third-party testing tools for repository-level testing of platform software components.

Heroku Flow

Heroku Flow is a continuous delivery (CD) toolset that streamlines the application development process and automates the promotion of code between stages, leveraging Heroku Pipelines. A pipeline is a group of Heroku apps that share the same codebase. Apps in a pipeline are grouped into one of four distinct stages – review, development, staging, or production. Each stage represents different deployment steps in a continuous delivery flow. There are a rich set of administrative controls that allow distinct control between stages of the software that are managed in Heroku Pipelines. Each app can have its own designated set of users with manage, deploy, and operate permissions. This allows broad collaboration for apps in the review and development stage, for example, but tighter governance and control of apps being promoted to staging and production environments.

Identity and Access Management

Account Types

There are two account types with access to the Heroku platform:

- *Heroku application accounts.* Heroku customer users, typically software developers, use Heroku application accounts to manage their applications.
- *Platform administration accounts.* Heroku employees involved with platform operations use accounts with elevated privileges to operate and maintain the system.

Note that because many of the components that make up the platform are Heroku apps themselves, Heroku administrators regularly use both application and platform accounts to perform their daily administration tasks.

Heroku application account passwords require a minimum of eight characters, with a combination of letters, numbers, and symbols. Platform administration accounts are limited to authorized Heroku staff and administered centrally by the security team, based on job responsibility. These accounts are assigned elevated privileges as needed by Heroku to perform system administration of the platform. Heroku enforces two-factor authentication for all access to the platform.

Identity and Access Management (Cont.)

Salesforce IT manages access to all workstations, and requires all employees to change their network password every 90 days. The corporate password policy requires:

- *Minimum length*: 15 characters
- *Use*: uppercase letters, lowercase letters, numbers, symbols, and emoji (at least three out of five)
- *Multifactor authentication*

Passwords must be changed completely, and previous passwords cannot be reused. Operating systems do not allow password authentication, in order to prevent password brute-force attacks, theft, and sharing. Heroku administrators employ secure SSH connections when remotely working on the system.

Identity Federation via Single Sign-On

Heroku easily integrates with customers' existing identity providers (IdPs) so customers can provide their employees with single sign-on (SSO) to Heroku using the same credentials and login experience as their other service providers. Heroku's SSO integration supports SAML 2.0.
The following major identity providers offer built-in support for Heroku:

- Auth0
- Azure Active Directory
- Bitium
- Okta
- Ping Federate
- Ping Identity
- Salesforce Identity

Organizations, Roles, and Permissions

Organization accounts, which are currently available only as a part of Heroku Enterprise, help customers manage Heroku applications as a business or other kind of group. Organization accounts let customers treat apps as a shared collection, giving a group of developers selective access to each app in the collection and monitoring resource usage across the entire organization.

Organization users can be assigned one of two roles: admin or member. An organization can have any number of each role, but must have at least one admin user. In addition to their member role in the organization, members can be assigned app permissions on specific apps owned by the organization.

Admins

The admin role allows users to:

- Add/remove admins and members in the organization
- Grant permissions on apps to org members and non-org users
- View resources and access billing for the organization
- List and access all apps in the organization, even if locked
- Perform any operation (deploy, scale dynos, manage add-ons, delete, etc.) on any app in the organization
- Transfer apps in or out of the organization

Each organization must have at least one admin user, therefore the last administrator in the organization cannot be removed. Admin users can only be added by other org admins.

Organizations, Roles, and Permissions (Cont.)

Members

Assigning a user the member role gives them read-only access to all apps within an organization. Member users can only be added by org admins. Members can:

- List all apps in the organization
- Access any app that is unlocked and view basic information including who has what permissions on the app
- Be assigned permissions to deploy, operate, or manage specific apps
- Create apps in the organization
- Transfer personal apps into the organization
- Perform any operation on apps they create or transfer in
- View resources, admins, and members in the organization

Non-Members

For users who are not formally members of the organization, per-app permissions may be granted so that they can access specific applications. An org admin or member with manage permissions on an app can give these individuals any combination of permissions on that app. Such users who are not org members but have been granted permissions on specific apps will be unable to:

- List access other org apps
- View other org users
- Create or transfer apps to the organization

Network Security

Secure Network Architecture

Network devices, including firewall and other boundary devices, are in place to monitor and control communications at the external boundary of the network and at key internal boundaries within the network. By default, all access is denied and only explicitly allowed ports and protocols are allowed based on business need. Each system is assigned to a firewall security group based on the system's function. Security groups restrict access to only the ports and protocols required for a system's specific function to mitigate risk.

Host-based firewalls restrict customer applications from establishing localhost connections over the loopback network interface to further isolate customer applications. Host-based firewalls also provide the ability to further limit inbound and outbound connections as needed.

Secure Access Points

Our data center provider has placed a limited number of access points to the cloud to allow for a more comprehensive monitoring of inbound and outbound communications and network traffic. These customer access points allow secure HTTP access (HTTPS), which allows you to establish a secure communication session with your runtime in Heroku.

In addition, our data center provider has implemented network devices that are dedicated to managing interfacing communications with internet service providers (ISPs). The data centers employ a redundant connection to multiple communication services at each internet-facing edge of the data center network. These connections each have dedicated network devices.

Data in Motion

All data in transit between instances is encrypted by default via SSL/TLS. This means that all traffic between single-tenant dynos and the Private Spaces Runtime are encrypted, but communication between multitenant dynos running within the same instance may be unencrypted. Encryption algorithms are chosen by browser/endpoint negotiation. All internet traffic is encrypted by default.

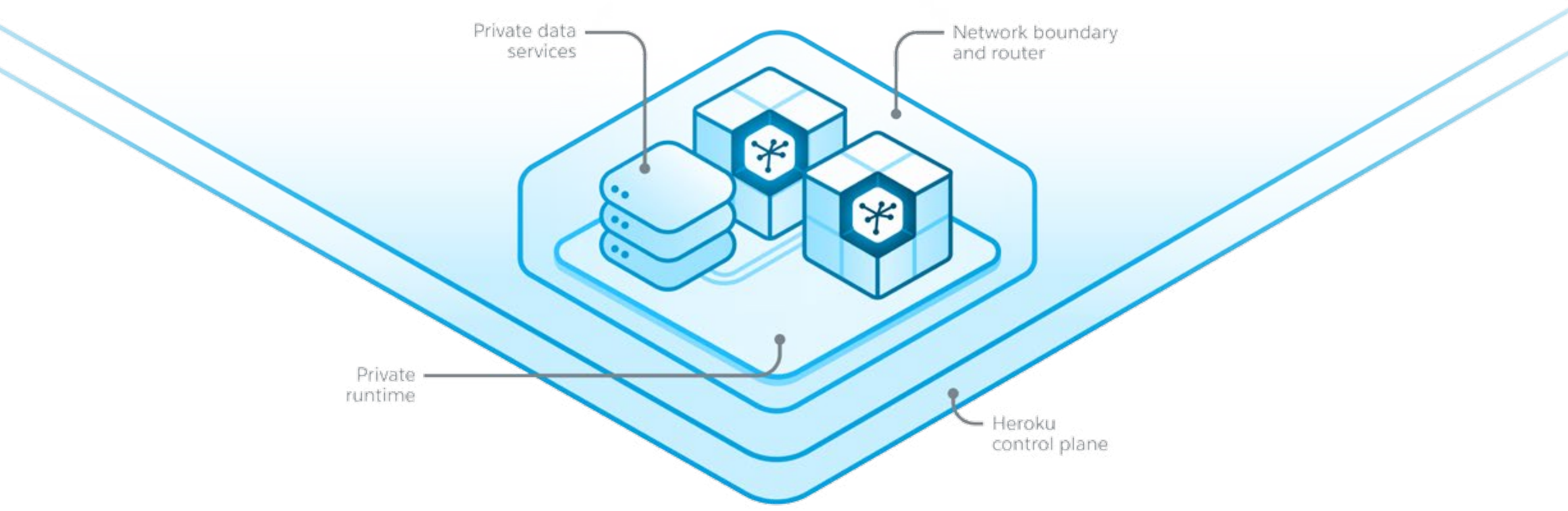
Private Spaces

For customers who require additional layers of network security, Heroku Private Spaces are delivered as a network-isolated group of apps and data services with a dedicated runtime environment, provisioned to Heroku in a geographic region specified by the customer. With Private Spaces, customers can build modern apps with the powerful Heroku developer experience and get enterprise-grade secure network topologies, enabling customers to securely connect to on-premises systems and other cloud services.

Only organization administrators can perform management functions such as creating, destroying, and changing settings on Private Spaces. Each Private Space has a set of trusted IP ranges, and only clients originating from one of these trusted IP ranges can access web processes running in the Private Space. All outbound traffic from apps in a Private Space will originate from a small, stable list of IP addresses that are dedicated to the space. This allows customers to use IP whitelisting to secure services being accessed by apps in the space.

Heroku Data Services, including Heroku Postgres and Heroku Redis, are created inside Heroku Private Spaces. The data services are automatically created in the same region as the Heroku Private Space and peered to the Heroku Private Space.

Heroku Private Spaces



Find out more at
heroku.com/private-spaces

Data Security

Heroku Postgres

Customer data is stored in access-controlled physically-segregated databases. Each database requires a unique username and password that is only valid for that specific database and is unique to a single application. Customers with multiple applications and databases are assigned separate database instances and accounts to mitigate the risk of unauthorized access.

Customer connections to Postgres databases require SSL encryption to ensure a high level of security and privacy. When deploying applications, Heroku encourages customers to take advantage of encrypted database connections.

Stored data can be encrypted by customer applications in order to meet data security requirements. Customers can implement data storage, key management, and data retention requirements when developing their application.

Encryption

Heroku's Premium and Enterprise Postgres database plans encrypt data at rest by using AES-256, block-level storage encryption. Data encryption is implemented using the AWS EBS disk encryption feature. Encryption keys are fully managed by AWS and are not visible to Heroku or Heroku customers. Postgres access credentials are also encrypted at rest.

Customer Data Retention and Destruction

Heroku customers define the data that is stored by their applications, and it is the customer's responsibility to purge data from databases to comply with data-retention requirements. If a customer deprovisions an application and any associated database, Heroku maintains the database's storage volume for at least one week after which time it is automatically destroyed, rendering the data unrecoverable. Physical storage volumes are deprovisioned as discussed later in this document, in the "Physical Security" section.

Security Monitoring

Logging and Network Monitoring

Heroku security and engineering staff monitor various tools and log feeds to detect anomalous behavior. The teams review authentication events, sudo requests, data traffic patterns, and other data sources.

From a network perspective, Heroku collects access logs and continually reviews them for indications of abuse or inappropriate access. Heroku also receives abuse data from its infrastructure provider, AWS.

The Heroku logging stream is based on logplex, a logging router that shuttles logs around to their eventual destination (called a "drain"). Heroku hosts a single-tenant instance of a log aggregation and analysis tool as the drain of record and analytics engine for logs. Only authorized Heroku personnel are allowed to view security logs.

DDoS

The Heroku platform provides mitigation for network-layer DDoS attacks, including TCP sync cookies and connection-rate limiting. The Heroku platform maintains multiple backbone connections and internal bandwidth capacity that exceeds the internet carrier-supplied bandwidth. Heroku works closely with network providers to quickly respond to events and enable DDoS mitigation controls when needed.

Customers who also need protection against application-layer DDoS will need to take tenant-level actions to mitigate these kinds of attacks. Heroku recommends that customers concerned about DDoS attacks use one or both of these techniques:

- Use a Content Delivery Network (CDN) with anti-DDoS features in front of the customer application. CloudFlare is a common choice.
- Customers can use a plug-in for their application framework that provides application-layer rate limiting and blocking. The specific tool used depends on which language and frameworks are being used; some examples are Rack::Attack (Ruby), turnstile (Python), and node-rate-limiter (Node.js).

Man in the Middle and IP Spoofing

Managed firewalls prevent IP, MAC, and ARP spoofing on the network and between virtual hosts to ensure spoofing is not possible. Packet sniffing is prevented by infrastructure including the hypervisor, which will not deliver traffic to an interface that it is not addressed to. Heroku utilizes application isolation, operating system restrictions, and encrypted connections to further ensure risk is mitigated at all levels.

Patch Management

The Heroku platform itself is continually upgraded and improved, usually with no effect on a customer's running application. If downtime is required for noncritical patches, customers are given the choice of scheduling their maintenance window for upgrades. OS patches can be deployed rapidly and invisibly to the customer. The vulnerability management process is designed to remediate risks with minimal customer interaction or impact.

Vulnerability management is handled by the Heroku Security team. Threat information is collected by internal tools, the engineering and operations teams, security team research, public-source vulnerability sources, and external programs such as the Heroku bug bounty program. Each vulnerability is reviewed to determine if it is applicable to Heroku's environment, ranked based on risk, and assigned to the appropriate team for resolution. Patches are deployed within 7, 30, 60, and 90 days for critical-, high-, medium-, and low-risk vulnerabilities, respectively.

Patch Notification

Heroku patches continuously, and as needed. Typically Heroku does not notify customers about minor patches, but posts notifications to the Heroku changelog for more important releases. If patches have a customer-facing impact – for example, a database upgrade that might require downtime – Heroku notifies customers in advance and provides a maintenance window.

> These changes are also communicated on the Heroku status site:
> **status.heroku.com**

Case Study: glibc

Heroku behind the Curtain: Patching the glibc Security Hole

Recently Google Security and Red Hat discovered a high-severity bug in a fundamental system library – glibc. This library is in common usage across the internet. If a server with a vulnerable version of the library were to make a DNS request to a malicious resolver, the DNS server could potentially execute code on the system making the request.

What Do We Do When a Security Vulnerability Lands?

Heroku took the glibc issue very seriously. We've done a lot of work to make sure our dyno containers are secure and we do everything possible to keep our customers safe

Our first step in any security incident is an immediate assessment by our security team. They work with our engineering teams to determine how big of a risk any vulnerability is to us. In this case, the potential for remote code execution meant that we considered it a high-priority patch for any system running glibc and querying DNS. That's pretty much all of them.

We patched our entire runtime fleet for both our Common Runtime and Private Spaces platforms. We also patched our Cedar stack image, to ensure that all the code you're running in your dynos stays safe. Last and most complicated, we patched our Data platform (Postgres and Redis) while keeping your data safe and available.

Case Study: glibc (Cont.)

How Do We Do This with Minimum Downtime?

We have standard practices for rolling out changes such as upgrades, new features, or security patches. These practices vary depending on the platform we're applying the changes to.

For the Common Runtime and Private Spaces, this is built into our infrastructure. When we push a new software version, we build a new base image (which is what dynos live on top of) and cycle your dynos off to a fresh instance using the new image.

An automated continuous integration (CI) pipeline updates our base images every time we update one of our components, and includes the latest Ubuntu base image. This new base image is automatically used by automated tests and new runtimes in staging. We cut a new release based on it and trigger the upgrade – first to our existing staging fleet, then to a small subset of production, then to the entire production fleet. Tests run between each of these stages.

In this case, we ensured that the latest base image contained the patched version of glibc and manually triggered our normal staged update and testing process.

What about the dynos themselves? They are security-hardened Linux containers that are isolated from the base image they run on. Dynos are composed of a few pieces – your code, your language-specific buildpack, and what we call stack images. Stack images provide basic system resources like glibc, and we make sure they have the latest security patches. In this case, we updated our stack image at the same time we updated our base image, as soon as we had a patched version of glibc. As dynos cycle, they pick up the new stack image.

This dyno cycle takes 24 hours by default, and if we feel the need to move more quickly, we can force a faster refresh. In this case, we waited for our normal 24-hour cycle to prevent customers from noticing disruption as everything restarts in quick succession. Our assessment of the vulnerability was that the risk was not so high that it would be worth potentially affecting running apps.

Case Study: glibc (Cont.)

What about Data?

Postgres, Redis, and any data store are more complicated to update. Customers store irreplaceable information that we can't architect around like we do for our runtimes. We need to be both available and secure, patching servers while keeping your data flowing.

To solve this problem for Heroku Postgres, we have follower databases. These automatically get a copy of all the data sent to your main database. When we need to update quickly, we can create a new, updated follower first and then change the follower to the main database. This does cause a short period of downtime, which is why we allow you to set maintenance windows so you can anticipate interruptions. In this case, based on our assessment of the vulnerability, most customers received their updates in their expected maintenance window.

For Heroku Redis, we have a similar storym, leveraging maintenance windows. Again, most customers received their updates in their expected maintenance window.

What about Heroku Itself?

A lot of Heroku runs on our own platform. We use all the tools we have to keep our customers secure and stable for ourselves. We did schedule some maintenance to patch our own internal databases with the above follower-changeover process. This maintenance affected our API, deployment, and orchestration system for a few minutes on each database. It didn't affect our routers or runtimes and allowed end users to access the apps they needed, even during maintenance.

Case Study: glibc (Cont.)

Keep Calm, Carry On.

The need to patch occasional security holes is serious and unavoidable. Before this glibc issue, there was last year's GHOST issue, and Heartbleed before that. At Heroku, we believe these patches shouldn't disrupt your flow. We work very hard to handle security issues and other platform maintenance with minimum impact to you or your apps – so you can carry on with your work without distraction.

"*Today it's not enough to simply offer hosting or black-box PaaS. Developers need services that allow them to deploy private networks in the public cloud, making geographical choices about where to host data.*

Public cloud wins on convenience, but the market has made it very clear that data location and governance are not optional extras.

Heroku is now addressing these security and compliance requirements across its portfolio, building on its heritage of simplicity and convenience."

JAMES GOVERNOR
Founder
RedMonk

Business Continuity

Heroku Platform: High Availability and Disaster Recovery

All important data and systems are backed up, with a high level of redundancy. Sensitive data, including any customer data, is backed up in an encrypted format. The Heroku platform does not use tapes, disks, or other removable media for data backup. All backups are made on the same cloud infrastructure that hosts production applications. The platform allows for recovering databases to within seconds of the last known state, restoring system instances from standard templates and deploying customer applications and data.

The backup process is essentially tested continuously. Backup procedures are validated by continually deploying, deprovisioning, and restoring customer apps and databases as part of the normal operation of the platform.

In addition to standard backup practices, Heroku's infrastructure is designed to scale and be fault tolerant by automatically replacing failed instances and reducing the likelihood of needing to restore from backup.

Heroku's core architectural constructs allow it to implement a disaster-recovery strategy using its distributed backups in a way that carries many benefits compared to traditional cold, secondary site configurations. All Heroku dynos and databases are designed to be disposable and easily replicated, allowing an application to scale up and down quickly or be entirely replaced with a new instance, whether within the same data center or in a distinct data center that is not impacted by an outage.

Heroku relies on its data center provider, AWS, to perform data center recovery operations in the event of an outage at one of their data centers.

The Heroku disaster-recovery plan is updated and tested continually as Heroku creates and removes computing resources. The operations teams review capacity and availability metrics on a daily basis to ensure that the platform is highly resistant to any outages.

Customer Applications

Applications deployed to the Heroku platform are automatically backed up as part of the deployment process on secure, access-controlled, and redundant storage. Heroku uses these backups to deploy customer applications across the Heroku platform and to automatically bring an application back online in the event of an outage. Customers share in the responsibility in designing and deploying applications and databases that can take advantage of this highly available architecture. The Heroku platform encourages customers to design applications leveraging the 12-factor app principles, including critical elements such as:

- Explicitly declaring and isolating dependencies
- Storing configuration in the environment
- Executing applications as one or more stateless processes
- Supporting disposability and maximizing robustness with fast startup and graceful shutdown

Postgres Databases

Heroku Postgres leverages a continuous protection architecture to keep data safe. Every change to customer data is written to write-ahead logs, which are shipped to high-durability storage across multiple data centers. In the unlikely event of unrecoverable hardware failure, these logs can be automatically "replayed" to recover the database to within seconds of its last known state. Heroku also supports the ability for customers to back up their databases to meet backup and data retention requirements.

> For additional technical information see:
> **devcenter.heroku.com/articles/heroku-postgres-data-safety-and-continuous-protection**

All Premium and Enterprise tier Heroku Postgres plans come with a high-availability (HA) feature, which leverages a database cluster and management system designed to increase database availability in the face of hardware or software failure – which could otherwise lead to longer downtime. When a primary database with HA fails, it is automatically replaced with another replica database called a standby. The database instance that exhibited failure is consequently destroyed and the standby is reconstructed.

Customer Configuration and Meta-Information

Customer configuration and meta-information is backed up every minute to the same high-durability, redundant infrastructure used to store database information. These frequent backups allow the system to capture changes made to the running application configuration added after the initial deployment.

Service Resiliency and Availability

The Heroku platform is designed for stability and scaling. The platform inherently mitigates common issues that lead to outages while maintaining recovery capabilities. The platform maintains redundancy to prevent single points of failure and is able to automatically replace failed components.

The Heroku platform is deployed across multiple data centers, and in the case of an outage, the platform can be recovered using current system images and data stored in backups. After any service-impacting incident, the Heroku team determines root cause, impact to customers, and improvements to the platform and processes to prevent future incidents.

> In the event of an interruption of Heroku services, details on severity, impact, and duration are posted on the status page:
>
> **status.heroku.com**

Business Continuity and Emergency Preparedness

As a business unit of Salesforce, Heroku adheres to Salesforce's global Business Continuity Management program, which includes business continuity plans (BCPs) for critical departments and functions across the organization that are integrated and aligned with site-specific emergency response plans (ERPs) and crisis management plans (CMPs) at a regional and global level.

The Salesforce Emergency Preparedness Plan (EPP) provides additional emergency preparedness, response information, instructions, and guidelines to protect the safety and well-being of employees and guests when an emergency situation transcends a landlord's facility emergency plan, if one is available.

Incident Response

The Heroku cloud application platform is monitored 24/7 by comprehensive automated systems. In the event of any issue affecting the health and operation of Heroku's infrastructure, core systems, or tools, Heroku's dedicated operations team is notified and will respond immediately to diagnose and correct any issues.

In the event of any service-impacting issue, notice is posted on the Heroku platform status site at status.heroku.com to promptly communicate the impact and status of any such issue.

In the event of a data breach, the Heroku Security team will coordinate remediation efforts with the Salesforce Computer Security Incident Response team. Any affected customers will be notified as soon as possible by Heroku's Security team. During an incident, the Heroku Security team assigns an incident commander (IC), who is responsible for coordinating response, communications, and postmortem analysis.

Elements Marketplace

One of the benefits of the curated Heroku Elements Marketplace is that add-ons (fully managed services integrated for use with Heroku) and buttons (which let you one-click provision, configure, and deploy third-party components, libraries, and pattern apps) are easily deployable into the Heroku platform, leveraging Heroku best practices. For example, add-ons are designed to leverage SSO and common identity with Heroku and can support integrated logging. Note that there is no formal validation process to ensure security or compliance requirements.

App Permissions

App permissions enable fine-grained access controls for Heroku Organization accounts, including control over the ability to deploy add-ons. Permissions are independently assigned and any combination of permissions can be assigned to a user on an app. This allows administrators to control the usage of add-ons and enforce controls in the software development process.

Building Secure Applications with Add-Ons

By leveraging add-ons, customers can more easily design applications that follow security best practices. For example, the Elements Marketplace includes several third-party components that support application monitoring, logging, and testing; network services such as static IPs and DNS management; alerts and notifications; user management such as authentication and single sign-on; and security services such as website security scanning, SSL encryption, and key management.

Physical Security

Data Center Access

Physical access is strictly controlled both at the perimeter and at building ingress points by professional security staff utilizing video surveillance, state-of-the-art intrusion detection systems, and other electronic means. Authorized staff must pass two-factor authentication no fewer than three times to access data center floors. All visitors and contractors are required to present identification and are signed in and continually escorted by authorized staff.

Our data center provider only provides data center access and information to employees who have a legitimate business need for such privileges. When an employee no longer has a business need for these privileges, his or her access is immediately revoked. All physical and electronic access to data centers by employees is logged and audited routinely.

Enviromental Controls

Fire Detection and Suppression

Automatic fire detection and suppression equipment has been installed to reduce risk. The fire detection system utilizes smoke detection sensors in all data center environments, mechanical and electrical infrastructure spaces, chiller rooms, and generator equipment rooms. These areas are protected by either wet-pipe, double-interlocked pre-action, or gaseous sprinkler systems.

Power

The data center electrical power systems are designed to be fully redundant and maintainable without impact to operations, 24 hours a day, and seven days a week. Uninterruptible power supply (UPS) units provide backup power in the event of an electrical failure for critical and essential loads in the facility. Data centers use generators to provide backup power for the entire facility.

Climate and Temperature Control

Climate control is required to maintain a constant operating temperature for servers and other hardware, which prevents overheating and reduces the possibility of service outages. Data centers are conditioned to maintain atmospheric conditions at optimal levels. Monitoring systems and data center personnel ensure temperature and humidity are at the appropriate levels.

Management

Data center staff monitor electrical, mechanical, and life support systems and equipment so issues are immediately identified. Preventative maintenance is performed to maintain the continued operability of equipment.

Storage Device Decommissioning

When a storage device has reached the end of its useful life, data center procedures include a decommissioning process that is designed to prevent customer data from being exposed to unauthorized individuals. Our data center provider uses the techniques detailed in DoD 5220.22-M ("National Industrial Security Program Operating Manual") or NIST 800-88 ("Guidelines for Media Sanitization") to destroy data as part of the decommissioning process. All decommissioned magnetic storage devices are degaussed and physically destroyed in accordance with industry-standard practices.

"Heroku's operational flexibility allows us to offer a safe product that is compliant with the law. We can iterate and deploy updates quickly and easily, which is very, very valuable."

JAKE ROSENBERG
CTO
LendUp

Compliance and Audit

Heroku's physical infrastructure is hosted and managed within Amazon's secure data centers and utilizes Amazon Web Services (AWS) technology. Amazon continually manages risk and undergoes recurring assessments to ensure compliance with industry standards. Amazon's data center operations have been accredited under:

- *ISO 27001:2013*. A security management standard that specifies security management best practices and comprehensive security controls following the ISO 27002 best practice guidance.
- *ISO 27017:2015.* The newest code of practice released by the International Organization for Standardization (ISO). It provides implementation guidance on information security controls that specifically relate to cloud services.
- *ISO 9001:2008.* A global standard for managing the quality of products and service, outlining a quality management system based on eight principles defined by the International Organization for Standardization (ISO) Technical Committee for Quality Management and Quality Assurance.
- *SOC 1, SOC 2/SSAE 16/ISAE 3402 (Previously SAS 70 Type II), and SOC 3*. Independent third-party examination reports that demonstrate how AWS achieves key compliance controls and objectives.
- *PCI Level 1*. A payment card industry standard that specifies best practices and various security controls, requiring organizations to build and maintain a secure network; protect cardholder data; maintain a vulnerability-management program; implement strong security measures; regularly test and monitor networks; and maintain an information-security policy.
- *FISMA Moderate*. Compliance with the Federal Information Security Management Act (FISMA), as evaluated by independent assessors.

> To review current audit reports and certifications, please contact your Salesforce/Heroku sales representative, or visit:
> **aws.amazon.com/compliance**

In addition to the certifications that are available for the data centers running the underlying infrastructure on which Heroku runs, Salesforce is currently in the process of pursuing PCI DSS Level 1 Service Provider certification and HIPAA compliance that extends to the core Heroku services and includes prescriptive support for designing applications that maintain proper security controls.

> Prior to the Heroku PCI DSS Level 1 Service Provider validation, Heroku has published guidelines on how to build PCI-compliant applications on the Heroku platform by leveraging third-party payment processors:
>
> **devcenter.heroku.com/articles/pci-compatible-apps**

Summary

Heroku Enterprise is a born-in-the-cloud platform service, and as such, it has been designed from the ground up to implement security into every layer of the stack. From physical security and network security up through Heroku's microservices architecture and container-based deployment model, Heroku Enterprise is designed to allow customers to focus their efforts on building differentiated business applications while letting Salesforce handle the challenges of managing and securing the platform.

If you have more questions, please contact your Salesforce or Heroku sales representative at 1-844-463-8026, or visit us at:

salesforce.com/platform