GitLab, Inc.

Independent Service Auditor's SOC 3 Report

For the

GitLab.com SaaS Platform System

For the Period of November 1, 2021, to October 31, 2022

Attestation and Compliance Services

INDEPENDENT SERVICE AUDITOR'S REPORT

To GitLab, Inc.:

Scope

We have examined GitLab, Inc.'s ("GitLab") accompanying assertion titled "Assertion of GitLab, Inc. Service Organization Management" ("assertion") that the controls within GitLab's GitLab.com SaaS Platform system ("system") were effective throughout the period November 1, 2021, to October 31, 2022, to provide reasonable assurance that GitLab's service commitments and system requirements were achieved based on the trust services criteria relevant to security, availability, and confidentiality (applicable trust services criteria) set forth in TSP section 100, *Trust Services Criteria for Security, Availability, Processing Integrity, Confidentiality, and Privacy (AICPA, Trust Services Criteria)*.

GitLab uses a subservice organization for cloud hosting services. The description of the boundaries of the system indicates that complementary subservice organization controls that are suitably designed and operating effectively are necessary, along with controls at GitLab, to achieve GitLab's service commitments and system requirements based on the applicable trust services criteria. The description of the boundaries of the system does not disclose the actual controls at the subservice organization. Our examination did not include the services provided by the subservice organization, and we have not evaluated the suitability of the design or operating effectiveness of such complementary subservice organization controls.

The description of the boundaries of the system indicates that complementary user entity controls that are suitably designed and operating effectively are necessary, along with controls at GitLab, to achieve GitLab's service commitments and system requirements based on the applicable trust services criteria. Our examination did not include such complementary user entity controls and we have not evaluated the suitability of the design or operating effectiveness of such controls.

Service Organization's Responsibilities

GitLab is responsible for its service commitments and system requirements and for designing, implementing, and operating effective controls within the system to provide reasonable assurance that GitLab's service commitments and system requirements were achieved. GitLab has also provided the accompanying assertion about the effectiveness of controls within the system. When preparing its assertion, GitLab is responsible for selecting, and identifying in its assertion, the applicable trust services criteria and for having a reasonable basis for its assertion by performing an assessment of the effectiveness of the controls within the system.

Service Auditor's Responsibilities

Our responsibility is to express an opinion, based on our examination, on whether management's assertion that controls within the system were effective throughout the period to provide reasonable assurance that the service organization's service commitments and systems requirements were achieved based on the applicable trust services criteria. Our examination was conducted in accordance with attestation standards established by the American Institute of Certified Public Accountants. Those standards require that we plan and perform our examination to obtain reasonable assurance about whether management's assertion is fairly stated, in all material respects. We believe that the evidence we obtained is sufficient and appropriate to provide a reasonable basis for our opinion.

Our examination included:

- Obtaining an understanding of the system and the service organization's service commitments and system requirements;
- Assessing the risks that controls were not effective to achieve GitLab's service commitments and system requirements based on the applicable trust services criteria; and

- Performing procedures to obtain evidence about whether controls within the system were effective to achieve GitLab's service commitments and system requirements based on the applicable trust services criteria.

Our examination also included performing such other procedures as we considered necessary in the circumstances.

Inherent limitations

There are inherent limitations in the effectiveness of any system of internal control, including the possibility of human error and the circumvention of controls.

Because of their nature, controls may not always operate effectively to provide reasonable assurance that GitLab's service commitments and system requirements were achieved based on the applicable trust services criteria. Also, the projection to the future of any conclusions about the effectiveness of controls is subject to the risk that controls may become inadequate because of changes in conditions or that the degree of compliance with the policies or procedures may deteriorate.

Opinion

In our opinion, management's assertion that the controls within GitLab's GitLab.com SaaS Platform system were effective throughout the period November 1, 2021, to October 31, 2022, to provide reasonable assurance that GitLab's service commitments and system requirements were achieved based on the applicable trust services criteria is fairly stated, in all material respects.

Schellman & Company, LLC

Tampa, Florida
November 28, 2022

ASSERTION OF GITLAB SERVICE ORGANIZATION MANAGEMENT

We are responsible for designing, implementing, operating, and maintaining effective controls within GitLab, Inc.'s ("GitLab") GitLab.com SaaS Platform system ("system") throughout the period November 1, 2021, to October 31, 2022, to provide reasonable assurance that GitLab's service commitments and system requirements relevant to security, availability, and confidentiality were achieved. Our description of the boundaries of the system is presented below and identifies the aspects of the system covered by our assertion.

We have performed an evaluation of the effectiveness of the controls within the system throughout the period November 1, 2021, to October 31, 2022, to provide reasonable assurance that GitLab's service commitments and system requirements were achieved based on the security, availability, and confidentiality (applicable trust services criteria) set forth in TSP section 100, *Trust Services Criteria for Security, Availability, Processing Integrity, Confidentiality, and Privacy (AICPA, Trust Services Criteria)*. GitLab's objectives for the system in applying the applicable trust services criteria are embodied in its service commitments and systems requirements relevant to the applicable trust services criteria. The principal service commitments and system requirements related to the applicable trust services criteria are presented below.

There are inherent limitations in any system of internal control, including the possibility of human error and the circumvention of controls. Because of these inherent limitations, a service organization may achieve reasonable, but not absolute, assurance that its service commitments and system requirements are achieved.

We assert that the controls within the system were effective throughout the period November 1, 2021, to October 31, 2022, to provide reasonable assurance that GitLab's service commitments and systems requirements were achieved based on the applicable trust services criteria.

DESCRIPTION OF THE BOUNDARIES OF THE GITLAB.COM SAAS PLATFORM SYSTEM

Company Background

Founded in 2011, with over 2,000 team members in more than 60 countries, GitLab, Inc. is an all-remote company with no central headquarters and no company-owned offices.

GitLab, Inc. is an open core company which develops software for the software development lifecycle used by more than 100,000 organizations, 30 million estimated registered users, and has an active community of more than 3,000 contributors. GitLab, Inc. openly shares information and is public by default, meaning projects, strategy, direction, and metrics are discussed openly and can be found within https://about.gitlab.com. GitLab, Inc.'s core values are Collaboration, Results, Efficiency, Diversity, Inclusion & Belonging, Iteration, and Transparency (CREDIT) and these form GitLab, Inc. culture.

GitLab, Inc.'s mission is to ensure that everyone can contribute. When everyone can contribute, consumers become contributors, and the rate of innovation is greatly increased.

Description of Services Provided

The GitLab.com software as a services (SaaS) platform, also referred to as GitLab.com, is a complete open-source DevOps platform that provides a continuous integration (CI) and continuous deployment (CD) toolchain out-of-the-box. The GitLab.com SaaS platform is a single application for each stage of the DevOps lifecycle. Enabling product, development, quality assurance (QA), security, and operations teams to work concurrently on the same project. The GitLab.com SaaS platform provides teams a single data store, one user interface, and one permission model across the DevOps lifecycle. This allows teams to collaborate and work on a project from a single conversation, significantly reducing cycle time and allowing teams to focus exclusively on building great software quickly.

Using the GitLab.com SaaS platform teams can utilize a variety of products including CI/CD, source code management (SCM), out-of-the-box pipelines (Auto DevOps), security (DevSecOps), agile development, value stream management and infrastructure automation (GitOps). Each of these popular use cases enable customers to remove technical barriers and focus on building efficiencies within their digital platforms.

Built on open source, the GitLab.com SaaS platform leverages the community contributions of thousands of developers and millions of users to continuously deliver new DevOps innovations.

System Boundaries

A system is designed, implemented, and operated to achieve specific business objectives in accordance with management-specified requirements. The purpose of the system description is to delineate the boundaries of the system, which includes the services outlined above and the five components described below: infrastructure, software, people, procedures, and data.

Principal Service Commitments and System Requirements

Principal Service Commitments

GitLab, Inc. designs its processes and procedures related to the GitLab.com SaaS platform to meet its objectives for its GitLab.com SaaS platform. Those objectives are based on the service commitments that GitLab, Inc. makes to customers, applicable laws, and regulations, and the financial, operational, and compliance requirements that GitLab, Inc. has established for the services.

Security, availability, and confidentiality commitments to user entities are documented and communicated in GitLab, Inc.'s privacy policy and terms of service, as well as in the description of the service offering provided on GitLab,

Inc.'s website. The principal security, availability, and confidentiality commitments are standardized and include the following:

- Maintain commercially reasonable administrative, technical, and organizational measures that are designed to protect customer data processed;
- Establish escalation procedures with customers;
- Encryption of data at rest and in transit;
- Encrypt confidential data stored within the database utilizing advanced encryption standard (AES) encryption;
- Maintain security procedures that are consistent with applicable industry standards;
- Notify customers of new patches applied to production environments;
- Not use confidential information for any purpose other than the purposes related to the customer agreement;
- Not disclose, make public or authorize any disclosure or publication of confidential information, except as expressly agreed to in the confidentiality agreement or otherwise in writing by the disclosing party;
- Document and enforce confidentiality agreements with third parties prior to sharing any confidential data;
- Review documentation from third-party providers to help ensure that they are in compliance with security and confidentiality policies;
- Maintain a business continuity and disaster recovery program;
- Restrict system access to authorized personnel only;
- Regularly assess security programs and processes;
- Identification and remediation of security incidents / events;
- Regularly update systems;
- Maintain the security of the information system from unauthorized access, use, modification, disclosure, destruction, threats, or hazards;
- Develop, implement, and maintain an information security program designed to protect the security, integrity, and confidentiality of the system and its information;
- Enable timely, reliable, and continuous access to, and use of, information and systems to support operations;
- Provide customers with access to technical support engineers for assistance in the proper installation and use of the software, and help resolve software problems;
- Availability to the SaaS Software will be measured, monitored, and reported on; and
- Maintain reasonable resiliency for data, compute, and network services.

System Requirements

GitLab, Inc.'s operational requirements that support the achievement of service commitments are communicated in GitLab, Inc.'s policies and procedures and agreements with user entities. GitLab, Inc.'s policies and procedures define an organization-wide approach to how the system and data is protected. These include policies around how the service is designed and developed, how the system is operated, how the internal business systems and networks are managed, and how team members are hired and trained. In addition to these policies, standard operating procedures have been documented on how to carry out specific manual and automated processes required in the operation and development of the GitLab.com SaaS platform.

In accordance with GitLab's assertion, and the description criteria, the aforementioned service commitments and requirements are those principal service commitments and requirements common to the broad base of users of the system and may therefore not fully address the specific service commitments and requirements made to all system users, in each individual case.

Infrastructure and Software

The GitLab.com SaaS platform is based on a multi-tenant, multi-user SaaS architecture built on the Google Cloud Platform (GCP) and is designed to integrate with multiple DevOps toolchains. GitLab, Inc. does not own or maintain the hardware located in GCP data centers and operates under a shared security responsibility model. GCP is responsible for the security of the underlying cloud infrastructure (e.g., physical infrastructure, geographical regions, availability zones, edge locations), and GitLab, Inc. is responsible for securing the platform deployed in GCP (e.g., customer data, applications, identity access management, operating system and network firewall configuration, network traffic, server-side encryption).

At its core, the GitLab.com SaaS platform is implemented using a combination of Linux, NGINX, Redis, PostgreSQL, Chef, Terraform, and GitLab.com itself. GitLab, Inc. utilizes GCP firewall rules to allow or deny traffic to and from their virtual machine instances based on configurations specified by the GitLab.com SaaS platform. GCP firewall rules are created and managed through Chef. GitLab, Inc. utilizes Cloudflare for web application firewall services to filter HTTP specific traffic.

Secondary Infrastructure and Supporting Software:

The following secondary infrastructure and supporting software is utilized in support of the delivery of the GitLab.com SaaS platform:

- NGINX – routes requests to appropriate application architectural components.
- GitLab Workhorse – reverse proxy and handles large hypertext transfer protocol (HTTP) requests.
- GitLab Shell – shell service handles git over secure shell (SSH) sessions.
- Unicorn – routes requests to application components and terminates secure socket layer (SSL).
- Sidekiq – ruby background job processor that pulls jobs from the Redis queue and processes them. Background jobs allow GitLab to provide a faster request / response cycle by moving work into the background.
- ops.gitlab.net – repositories for managing GitLab.com infrastructure and operational tasks.
- Gitaly – removes the need for network file system (NFS) for Git storage in distributed deployments of GitLab; this service handles Git level access in GitLab.
- Kubernetes – container scaling for a subset of the production environment.
- Chef – configuration management solution utilized to provision, configure, and manage production infrastructure.
- Panther – security event monitoring for the production environment and provides automated alerts.
- Prometheus – monitoring of metrics on individual processes running GitLab.
- Pingdom – uptime monitoring of the production environment.
- Alertmanager – tool by Prometheus that processes alerts sent by client applications.
- PagerDuty – alerting system utilized to deliver alerts to on-call personnel when predefined events occur.
- Tenable.io – cloud-based platform utilized to conduct automated, weekly vulnerability scans of the production environment.
- Terraform – infrastructure as code to deploy, monitor, and maintain production infrastructure.
- Slack – communications platform utilized to facilitate daily conversations related to various aspects of the business. Conversations are grouped into project channels, which are organized by topic and/or department.
- Cloudflare – host of Domain Name Server (DNS) and Web Application Firewall (WAF).
- OSQuery – universal endpoint agent utilized for production server monitoring.

- JAMF – endpoint device management tooling.
- Ansible – IT automation engine supports CI/CD, deploys, maintenance, and database tooling.

People

GitLab, Inc. is an all-remote company with team members located in more than 60 countries around the world. This geographic diversity allows fault tolerance resistance to disruptions in business continuity.

The following GitLab, Inc. departments are in-scope for this report:

- Executive and management – responsible for the overall operation of the GitLab.com SaaS platform. Direct and oversee the operation, maintenance, implementation, development, and monitoring of the system.
- Security – responsible for the security of the GitLab.com SaaS platform, including the triaging and responding to security incidents.
- Infrastructure – responsible for the availability, reliability, performance, and scalability of the GitLab.com SaaS platform.
- Engineering – responsible for building new features for the GitLab application.
- Business Technology – responsible for creating new processes, workflows, systems, and documentation for various interdepartmental groups. Responsible for guiding systems, workflows, and processes as well as being a singular reference point for corporate operational management.
- People Group – responsible for team-member satisfaction, benefits, code of conduct, and the overall hiring processes and procedures, including onboarding, offboarding, and role changes.
- Security Assurance – responsible for the assessment of the effectiveness of security risk management, control, and governance processes, and to provide insight and recommendations that can enhance these processes. Responsible for GitLab, Inc's operational risk management program, including risk identification, escalation, and treatment plans.
- Legal – responsible for ensuring that GitLab, Inc. remains in compliance with regulatory requirements, including requirements related to confidentiality and privacy of customer data.

Procedures

Access, Authentication, and Authorization

Documented information security policies and procedures defining key roles and responsibilities, risk management governing principles, and design principles to protect systems and data are in place and enforced by GitLab, Inc.

Access to production resources is controlled using permissions associated with GitLab, Inc. staff and strictly controlled system accounts. Chef is utilized as the infrastructure management and configuration management tool to help ensure security hardening and baseline configuration standards have been established on production servers. User accounts are propagated to the production servers, based on role, using Chef. Furthermore, the Chef client running on production servers are configured to check in with the Chef server every 30-minutes to help ensure that server configurations are up to date with the latest approved Chef cookbooks and recipes.

In order to access the production environment from the back end, users are required to authenticate into one of the bastion hosts, which are load-balanced for redundancy, using either two factor authentication or an SSH connection. Once authenticated to a bastion host, users can initiate an SSH session to the production servers as well as the production databases.

In order to access the production environment from the front-end via the GCP console, users must authenticate to the Okta console utilizing their Okta-credentials as well as entering a one-time generated security code or WebAuthn / FIDO2 biometric.

Administrator access to the production systems is granted based on job roles and responsibilities and limited to authorized personnel. GitLab, Inc. performs account and access reviews on a quarterly basis to help ensure that only authorized personnel maintain access to the production environment.

Access Requests and Access Revocation

GitLab, Inc. operates its access management under the principle of least privilege, wherein a team member should only be granted the minimum necessary access to perform their function. Baseline role-based entitlement access runbooks and issue templates are utilized for access management. For certain roles, role-based entitlement templates have been established and are used during the onboarding process. If users require access to systems that have not been pre-approved, an access request issue is submitted, reviewed by the system owner, and approved by management before access is granted.

In the event of a termination or a job transfer of an information system user, the people operation management system sends a notification to relevant personnel, or systems. As a component of the team member termination process, production system access is revoked for terminated team members as an element of the offboarding task checklist.

Change Management

GitLab, Inc. utilizes a continuous delivery model for software development and documented policies and procedures are in place to guide personnel in change control practices. The standard change management process is documented in a change control workflow. Prior to introducing changes into the production environment, approval from authorized personnel is required based on the change description, impact of change and test results. Emergency changes to the production environment follow the same change control workflow as standard changes. Approvals can be retroactively applied depending on the urgency of the change.

A production issue dashboard is available in the GitLab.com SaaS platform that outlines project details for the production environment. Issues are categorized based on current state: open, unscheduled, scheduled, etc.

GitLab version control software is utilized to manage production source code and provide roll back capabilities. GitLab, Inc. systematically prevents users from both developing and implementing code to the production environment by configuring merge request approval settings on GitLab projects that house source code to require approval from a peer prior to merging code. Access to modify source code is restricted to authorized personnel.

GitLab's issue tracking is in place to centrally maintain, manage, and monitor application and infrastructure changes from development through implementation. Upon initiation of a merge request, the GitLab.com SaaS platform is configured to require code review and approval from one individual independent of the individual who initiated the merge request. Various QA / smoke testing and source code security checks, such as dynamic application security testing (DAST) and static application security testing (SAST), are performed by GitLab's continuous integration tool prior to merging the code commits within the merge request. Prior to introducing changes into the production environment, approval from authorized personnel is required based on the change description, impact of change, and test results. The aforementioned process is repeated through the staging and canary environments prior to introducing changes into the production environment. The ability to implement application changes via the deployment tool is restricted to authorized personnel.

Separate environments exist for development, staging, and production. Development and testing activities are performed in distinct environments that are logically separate from production to help ensure that changes made within the development and staging environments do not affect the production environment.

Major software releases are developed using the DevOps Lifecycle (plan, create, verify, package secure, and release). Each phase is followed prior to introducing changes into the production environment. Release notes are available and communicated to internal and external system users through the GitLab handbook.

Incident Response

Security incidents are defined as any violation, or threat of violation, of GitLab, Inc. security, acceptable use, or other relevant policies. Infrastructure incidents are anomalous conditions that result in, or may lead to, service derogations or outages. These events often require human intervention to avert disruptions or restore service to operational status.

Documented incident response and management procedures for reporting security events are provided to team members to guide users in identifying and reporting system failures, incidents, concerns, and other complaints. The procedures define the types of incidents that need to be managed, tracked, and reported, including:

- Procedures for the identification and management of incidents;
- Procedures for the resolution of confirmed incidents;
- Key incident response systems;
- Incident coordination and communication strategy;
- Contact method for internal and external parties to report incidents;
- Support team contact information;
- Notification to relevant management in the event of a security breach;
- Provisions for updating and communicating the plan;
- Provisions for training of support team;
- Preservation of incident information;
- Timeline of incident;
- Action items;
- Lessons learned; and
- Management review and approval, annually, or when major changes to the organization occur.

Additionally, GitLab, Inc. provides a contact method for external parties on the about.gitlab.com/security site to submit complaints, inquiries, and incidents.

Confirmed incidents are assigned a priority level and managed to resolution following the incident management process. If applicable, GitLab, Inc. coordinates the incident response with business contingency activities. Security personnel complete incident summary issues for confirmed severity 1 security incidents that include the incident timeline, action items, and lessons learned. Corrective measures or changes that occur as a result of incidents and identified deficiencies follow the incident management process phases. If a production change is required, the standard change management process is followed.

The GitLab, Inc. security department leadership conducts a monthly staff meeting to communicate and align on P1 security threats, program performance, and resource prioritization. These meetings are also an opportunity to discuss mishandled incidents and process improvements.

System Monitoring

Established infrastructure management and configuration management tools are used for security hardening and baseline configuration standards for production servers. Production systems are monitored for deviations from baseline configurations in production environments. GitLab, Inc. performs independent third-party penetration testing for in scope production systems at least annually. Results from the aforementioned are evaluated and remediated according to risk rating.

In-scope production systems are monitored in accordance with predefined security criteria and alerts are sent to the Security Incident Response Team (SIRT). Confirmed security incidents are tracked to resolution. Enterprise monitoring applications are configured to monitor the in-scope systems capacity levels and alert operations personnel when predefined thresholds have been met. Infrastructure meetings are held on a monthly basis to review availability trends and forecasts as compared to production system commitments.

Data

Data within the GitLab.com SaaS platform is generated and uploaded by GitLab, Inc. customers who submit information through the GitLab.com SaaS platform. The transmission of confidential data is secured via an Internet connection encrypted with transport layer security (TLS) protocol.

Customers have the ability to retrieve reports related to their respective environments through the GitLab.com SaaS platform. If an error is identified, customers contact customer support and provide feedback to correct and resolve the issues. Significant events and conditions are captured in the system and application logs.

GitLab.com SaaS platform data is categorized according to the data classification standard and is protected according to its classification. The policy has four categories:

- Red – restricted and must remain confidential.
- Orange – data subject to laws and regulation that should not be made generally available.
- Yellow – data and information that should not be made publicly available that is created and used in the normal course of business.
- Green – data that is publicly shareable and does not expose GitLab or its customers to any harm or material impact.

The following table describes the information used and supported by the system:

Data Used and Supported by the System		
Data Description	**Data Reporting**	**Classification**
GitLab customer data	Available through the GitLab.com SaaS platform	Non-public data: Red
Metrics for NGINX, Gitaly, Postgres, Rails-app, Redis, Registry, and service platform	Publicly available through our public monitoring dashboards	Public data: Green
GitLab corporate data	Available through the GitLab.com SaaS platform	Non-public data: Orange and Yellow Public data: Green

Subservice Organizations

The cloud hosting services provided by GCP were not included within the scope of this examination. The following table presents the applicable Trust Services criteria that are intended to be met by controls at GCP, alone or in combination with controls at GitLab, and the types of controls expected to be implemented at GCP to meet those criteria.

Control Activities Expected to be Implemented by GCP	Applicable Trust Services Criteria
GCP is responsible for implementing controls to manage logical access to the underlying network and virtualization management software for its cloud hosting services where GitLab systems reside.	CC6.1 – CC6.3, CC6.6
GCP is responsible for restricting physical access to data center facilities, backup media, and other system components including firewalls, routers, and servers.	CC6.4 – CC6.5
GCP is responsible for ensuring data within GCP is stored in an encrypted at rest format.	CC6.7

Control Activities Expected to be Implemented by GCP	Applicable Trust Services Criteria
GCP is responsible for ensuring access to Cloud Storage server-side encryption keys is restricted to authorized personnel.	
GCP is responsible for implementing controls for the transmission, movement, and removal of the underlying storage devices for its cloud hosting services.	
GCP is responsible for managing logical access to the underlying network, virtualization management, and storage devices for its cloud hosting services where the system resides.	CC7.2
GCP is responsible for ensuring capacity demand controls are in place to meet GitLab's availability commitments and requirements.	A1.1
GCP is responsible for ensuring environmental protection controls are in place to meet GitLab's availability commitments and requirements.	A1.2

Complementary Controls at User Entities

GitLab's services are designed with the assumption that certain controls will be implemented by user entities. Such controls are called complementary user entity controls. It is not feasible for all of the applicable trust services criteria related to GitLab, Inc.'s services to be solely achieved by GitLab's control procedures. Accordingly, user entities, in conjunction with the GitLab.com SaaS platform system and related services, should establish their own internal controls or procedures to complement those of GitLab, Inc.

The following complementary user entities controls should be implemented by user entities to provide additional assurance that the applicable trust services criteria described within this report are met. As these items represent only a part of the control considerations that might be pertinent at the user entities' locations, user entities' auditors should exercise judgment in selecting and reviewing these complementary user entity controls:

#	Complementary User Entity Control	Related Applicable Trust Criteria
1.	User entities are responsible for adherence to their contractual security and confidentiality commitments.	CC1.1, CC2.3
2.	User entities are responsible for configuring authentication settings for end-users for their GitLab.com SaaS platform subscription.	CC6.1
3.	User entities are responsible for controlling end-user access management for their GitLab.com SaaS platform subscription.	CC6.1 – CC6.3, CC6.6
4.	User entities are responsible for understanding and implementing the GitLab.com SaaS platform security hardening recommendations according to their internal risk tolerance.	CC6.1 – CC6.2, CC6.6
5.	User entities are responsible for protecting and controlling access to tokens and other secrets they create within the GitLab.com SaaS platform.	CC6.2
6.	User entities are responsible for notification of actual or suspected information security breaches affecting the GitLab.com SaaS platform to GitLab, Inc.	CC2.2 – CC2.3, CC7.3
7.	User entities are responsible for monitoring activity and consuming audit events within their GitLab SaaS instance to detect potential security events.	CC7.2
8.	User entities are responsible for notification of system failures or upload failures affecting the GitLab.com SaaS platform to GitLab, Inc.	CC7.3 – CC7.4

Trust Services Criteria Not Applicable to the In-Scope System

All criteria within the security, availability, and confidentiality categories are applicable to the GitLab.com SaaS platform system.