Platform Security

Last Modified on 01/03/2024 3:45 pm AEDT

Overview

ReadiNow employs multiple layers of industry standard practices, protocols and techniques to mitigate the risk of unauthorised access or modification to data or systems.

This document covers ReadiNow network and protocol security mechanisms. For discussion of how co-hosted customers are isolated refer to the "ReadiNow Tenant Isolation Model" whitepaper. Customers may also optionally be hosted on dedicated servers. For discussion of configuring application-level record access, refer to the "ReadiNow Access Control Security" whitepaper.

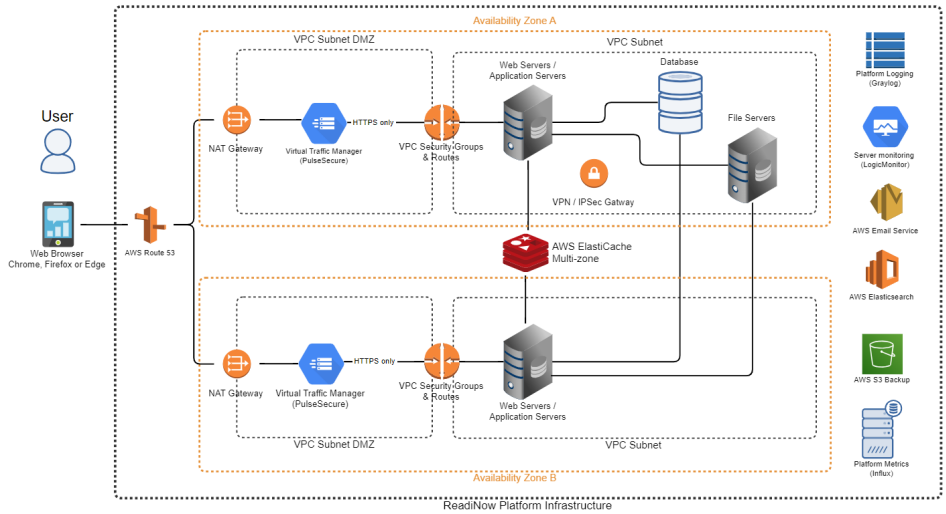
Network/Hosting

ReadiNow servers are hosted in a secure AWS virtual environment.

The virtual traffic manager (vTM) receives connections and forwards valid request onto web servers on the client's behalf. Only **encrypted HTTPS (TLS 1.2)** connections are admitted. The vTM also provides **load balancing** and **failover** services.

The web, file, database and cache servers all reside in the non-public zone such that direct connection to them is not possible.

All servers run **anti-virus** software.



Infrastructure Security

Data at rest in our databases and file repositories is encrypted using Amazon EBS encryption. This encrypts data on the volume, and in transit between the volume and operating system. EBS uses the AES-256 encryption standard

Data in transit is encrypted. All web requests to web servers is over encrypted secure HTTPS connections. This

includes login, data and file exchanges. This includes internal communications between the vTM and the web servers.

Communication between web servers and database servers is encrypted using SSL. Communication between web servers and cache servers is encrypted using SSL. Communication between web servers and the document repository is encrypted using SMB 3 AEC-CCM.

ReadiNow only admits TLS requests that are of version 1.2 and higher. The version of TLS (https) connections initiated by the end user's browser to our infrastructure is necessarily determined in part by their browser, operating system, and network infrastructure. However, attempts to connect using older TLS-1.0 and TLS-1.1 connections are denied.

ReadiNow platform backups are protected from accidental or malicious modification or erasure using the Amazon S3 Object Lock technology. Once a backup is generated, it is not possible for it to be modified or deleted by any means, for a minimum of 90 days - even by ReadiNow infrastructure administrators.

Database backups are encrypted when generated. Additionally, the S3 buckets used to store all backups are encrypted.

Web Request Security

During login, an authentication token is generated and encrypted using the **AES** (Rijndael) algorithm and returned to the browser. This token is attached to all subsequent web requests. The ReadiNow servers check the authentication token on every request as part of the request processing pipeline.

To mitigate against a malicious script from capturing the authentication token on the client side, authentication token cookies are tagged with the **HttpOnly** and **Secure** attributes which instruct the browser to not make the cookie available to scripts, and to only pass the cookies back over encrypted HTTPS/TLS-1.2 connections.

ReadiNow does not store the passwords used to log into the ReadiNow console and instead just stores the hash of these password to enable password validation at login. This hash is generated using an industry standard non-reversible salted hash protocol based on the **HMAC SHA1** algorithm.

To mitigate against Cross Site Scripting (XSS) attacks, HTTPS responses include **Content-Security-Policy** and **X-WebKit-CSP** headers instructing browsers to not run unsafe Javascript operations; as well as the **X-XSS-Protection** header that instructs browsers to sanitise potential attacks.

To mitigate against Cross Site Request Forgery (XSRF), an XSRF token is generated at login, and attached to the URL of all internal data requests. The server verifies the presence and validity of this token to further ensure that all requests are coming from a properly logged in session.

Additionally, Refresh Tokens are used to protect again session hijacking attempts.

The ReadiNow platform uses a combination of the React and Google AngularJS frameworks to generate HTML, which prevents unsanitised content from being directly rendered to the browser Document Object Model (DOM).

To mitigate against ReadiNow content being maliciously hosted inside an IFRAME, all responses include the **X-Frame-Options** header to instruct browsers to disallow this.

To mitigate against invalid or malicious MIMETYPE interpretation, all requests include the **X-Content-Type:nosniff** header.

To mitigate against protocol downgrade attacks and cookie hijacking attacks, all responses include the**Strict-Transport-Security** header, as per RFC 6797, to instruct browsers that cookies and authentication tokens issued over encrypted HTTPS are not to be available to any (i.e. malicious) content attempting to run in an unsecured HTTP context.

Secure Development and Security Penetration Testing

All source code changes are submitted for peer review, and security-sensitive code changes in particular must be reviewed by ReadiNow platform architects.

The ReadiNow platform undergoes an annual third-party penetration test to independently scan and test for security vulnerabilities in the application.

In addition, ReadiNow performs automated penetration tests as part of our continuous build process. We use the OWASP Zed Attack Proxy utility. It first captures a baseline sample of traffic generated by our browser automation testing, which it then uses to automatically generate and attempt malicious attacks against the platform. Any anomalous results are automatically raised as build errors. This is scheduled to automatically run for all release builds, and daily for development builds.