Ready Room Security

Infrastructure

Ready Room runs in the Google Cloud Platform (GCP), thereby inheriting Google's infrastructure security, including physical access, network management, server provisioning, VM isolation, and more. See here for detailed information: https://cloud.google.com/security/infrastructure/design

Database

Ready Room uses Google Cloud SQL, a fully managed instance of the PostgreSQL relational database (https://cloud.google.com/sql/). Data that is housed in Google Cloud SQL is always encrypted, both in the datastore and in backups. In addition, communication between the application servers and the database is also encrypted. That is, all data are encrypted at rest and in transit.

Database access credentials are managed by Ready Room's PaaS provider, Gigalixir, and the password is a fully random UUID.

Command line access to the database is available, but authentication is via private key authentication, a key that is stored securely and available to only one person.

The database is backed up automatically every 24 hours.

File Storage

All customer uploaded files are stored in Google Cloud Storage (GCS), a fully managed, secure, and durable object store. All communication between clients and GCS is encrypted. Files are stored in "buckets" configured with a uniform access control setting of "Not Public," accessible to a small number of authenticated users.

In order to read and write files to GCS, clients use a combination of Cross Origin Resource Sharing (CORS) and cryptographically signed URLs. These URLs are signed on the Ready Room servers using a securely managed private key that pairs with a public key retained by Google. Signed URLs have a lifetime of only 15 minutes.

Application Servers

Ready Room is developed using the Elixir programming language and the Phoenix Application Server. Phoenix is a "secure by default" environment that ensures adherence to OWASP web application security best practices. Thus, Ready Room has built in protection against cross site scripting and cross site request forgery attacks. Phoenix uses the Ecto database access library, which has built-in mitigation against SQL injection attacks.

Authentication

Ready Room currently manages its own user store. (In the future, we will default to single-sign on). When a user is invited into the system, we first store only the user's email address. Should the user accept the invitation, they are in charge of creating their password; thus, a user's password is known only to them. System invitations use randomly generated tokens. Only people in possession of the token can accept an invitation. The tokens are hashed (SHA256) before storing in the database and expire after seven days.

Ready Room passwords must be at least 12 characters long, but enforce no other patterns.

Passwords are hashed via the Bcrypt standard before being stored in the database. This ensures robustness over time (in the face of increased processing power) and mitigates against timing attacks during authentication. At no time is anyone, even the database administrator, able to see the plaintext password. Users can reset their own passwords. If a user has forgotten their password, they can start a password-reset flow that also uses tokens. Password reset tokens expire after 24 hours.

Authentication is maintained across requests using a single signed cookie. That cookie is removed when the browser is exited. In addition, the user can select "remember me," and Ready Room will keep that user authenticated for 60 days.

Authorization

Ready Room is a multi-tenant system. Synclinical developers have worked carefully to ensure that a tenant's data is not leaked across boundaries. Within a tenant, we have very few roles (authorization contexts). These are as follows:

- **Super User:** Can create an account. Currently only the two Synclinical founders have superuser privileges. There is no code path that can be exploited to grant superuser privileges.
- **Account Administrator:** Can create and manage users, create inspections, and access all inspections.
- **Team Members:** Can participate in inspections, manipulating tasks and uploading documents.
- **Inspectors:** Can view inspection requests that have been "released."

Access to an inspection is only available to admins and users who have been explicitly added to the inspection.

Inspector accounts are special. The username and password are randomly generated, and inspectors do not have control over their accounts. Access is limited to viewing only the tasks that have been released. In addition, system access for inspectors (and all other users) can be disabled as desired.

Super users have *no* special access or visibility into customer accounts unless they are invited into the account as an Account Administrator or Team Member.

Network Access

Ready Room uses TLS (SSL) communication only. If a user attempts to access Ready Room over HTTP (unencrypted), they will be automatically redirected to the HTTPS endpoint. In addition, Ready Room uses Strict Transport Security (https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security) such that the browser will never again allow an HTTP request to Ready Room, thus mitigating the "coffee shop attack" (credential sniffing).

System Integrity

Ready Room does not use GCP directly. Instead, it leverages a Platform as a Service (Paas) called Gigalixir (similar to Heroku). Gigalixir rebuilds docker containers from scratch with each deploy. The containers are running Heroku's latest Ubuntu stack, Heroku-18 (https://devcenter.heroku.com/articles/heroku-18-stack), which is widely used and derived from the latest Ubuntu Linux release with minimal packages installed (https://devcenter.heroku.com/articles/heroku-18-stack). In addition, Ready Room developers actively leverage current Elixir and JavaScript libraries. Together, these practices mitigate against CVEs and zero-day vulnerabilities that may be lingering in old libraries.

Gigalixir itself uses GCP's support for Kubernetes and Docker for OS and application isolation, while layering on additional security of its own: https://gigalixir.readthedocs.io/en/latest/main.html#how-secure-is-gigalixir

Finally, Ready Room developers run a static analysis tool against the code before each deployment. This tool (https://github.com/nccgroup/sobelow) looks specifically for security flaws. Currently, the only finding is that Ready Room does not set Content-Security Policy headers in the HTTP request. This is correct and will be addressed in the future. In the meantime, we deem the lack of these settings to be of low risk.