REDCap Technical Overview

Introduction

REDCap is a web application for building and managing online surveys and databases. This document delineates many of the broader technical aspects of REDCap, such as the infrastructure and third-party software required to host REDCap, details of its data storage model, user privileges, authentication options, logging and audit trails, data interoperability options with other systems, protective security measures, and best practices for hosting REDCap at your local institution.

REDCap Infrastructure: Best Practices and Dependencies

It must be pointed out that much of the security surrounding REDCap has nothing to do with the REDCap software itself but rather is dependent upon the IT infrastructure and environment in which REDCap has been installed. This includes the web server and database server, the communication between those two servers, and the communication of the web server with the REDCap end-user. Typical best practices are that the web server and database server be two separate servers and that the database server be located securely behind a firewall. The web server may be located either behind the firewall or in the DMZ. Many institutions host REDCap with their web server behind a firewall, but this is often done because it is required by institutional policy. Many institutions (including Vanderbilt University) host their web server in the DMZ so that it is available outside the firewall to the entire internet. SSL is required on the web server in order to maintain secure communication with the end-user, so the web server must be set up with an SSL certificate. With regard to performing data backups for REDCap, most institutions that host REDCap perform a daily (or twice daily) backup of their REDCap database tables, often using mysqldump or other similar software.

REDCap has no hard requirements with regard to server processing power, memory, or hard drive space since it is very light weight in most respects and requires very little initial drive space by either the web server or database server. It is typically recommended that 10GB be allotted to the web server and database server each in the beginning. That amount of storage space is typically sufficient for most institutions running REDCap for the first year (or longer), even under heavy usage. It is not always predictable how much drive space will be consumed by REDCap over time since all institutions and users are different, but after several months of solid utilization by day-to-day REDCap users, the amount of space used over time becomes fairly predictable.

If the web server is accessible to the web (i.e. in the DMZ), then it is recommended that documents in REDCap not be stored locally on the web server but instead on another server behind a firewall (similar to the database server). Whether you are storing REDCap documents on a typical file server, Network Attached Storage (NAS), or Network File System (NFS), the safest place is behind a firewall using secure communication to REDCap using WebDAV protocol (SSL supported), in which there is an option to enable the WebDAV option in the REDCap Control Center. If the web server is located behind a firewall and not accessible to the web (i.e. not in the DMZ), then it may be perfectly fine to store the REDCap documents on the local web server or on a file system mounted to the web server. Regardless of your setup, you should first consult policies at your institution to see if there are any institutional regulations or mandates with regard to file storage (especially when involving files containing identifying information - i.e. PHI) before finalizing your production REDCap environment.

There is a fairly short list of infrastructure requirements and dependencies for installing REDCap. REDCap can run on a number of different operating systems (Linux, Unix, Windows, Mac). The only requirements that REDCap has are that the hardware and software listed below be installed. (Note: All required software is open source.)

REDCap Infrastructure Requirements and Dependencies

1) **Web server** (e.g. Microsoft IIS or Apache) with PHP 8.0.0 or higher.
2) **Database server** with MySQL 5.5.5+ or MariaDB 5.5.5+
    a. MySQL client – required for installation/upgrades (e.g. phpMyAdmin, MySQL Workbench)
3) **SMTP email server** – In order to send emails from REDCap, an SMTP server must be configured with PHP on your web server. It can be installed on the same web server or on a separate server (preferred), such as an existing institutional SMTP server, if available.
4) **File server (optional)** – Depending on your infrastructure and setup, you may wish to employ a separate server solely for files uploaded/stored in REDCap. If your web server is accessible to the web (i.e. in the DMZ), it is highly recommended to have a separate file server located behind a firewall that communicates securely to REDCap using WebDAV protocol (SSL supported). Consult your local policy first in case your institution has regulations or mandates regarding this.

More info on server requirements can be found at https://redcap.link/performance.

REDCap User Privileges

To ensure that REDCap users have access only to data and information that they are supposed to have access to within the application, user privileges are utilized within the software. Each user has their own account, and their user account will only have access to REDCap projects that they themselves have created or to projects to which other users have granted them access. Some of the general user privileges in the application are dictated by the customizable settings that each institution can adjust, such as whether or not users can create their own projects or if an administrator must create it for them, among other settings.

User privileges are also granular on the project level and can be modified within any given project by someone with proper privileges accessing the project's User Rights page. The creator of a project will automatically be given full rights to everything within the project, after which they may grant other users access to the project and limit those users' privileges as desired. Within each project, there are user controls to limit access to various functionality and modules, such as being able to export data, to enter data, to add or modify database fields or survey questions, to build or run reports, to modify user privileges, to view the logging records, and so on. Another feature called Data Access Groups can be implemented to help segregate users and the data they enter by placing users into data access groups, after which they will only be able to access records created by someone in their group. This particular feature is entirely optional but is especially helpful in certain situations, such as for multi-institutional projects where the data entered by one institution should not be accessible or viewable by other institutions with access to that same project.

REDCap Authentication

REDCap implements authentication to validate the identity of end-users that log in to the system. Several authentication methods are available for use in REDCap: LDAP, Shibboleth, OpenID, Google OAuth2, an internal table-based authentication method, as well as a combination of LDAP and table-based together. Institutions running REDCap can choose which authentication method works best for them. The table-based authentication, which utilizes the storage of username/password pairs in a database table, is often the easiest to set up because it is built-in and requires no setup and no configuration with external services in order to operate. For security reasons, the password in the database table is not stored as plain text, but it is first salted and then hashed using a SHA-512 cryptographic hash function before being stored in the database table. Also notable is that each user account has its own unique salt value.

REDCap contains an auto-logout setting, which is customizable (default auto-logout time is 30 minutes), and will automatically log a user out of the system if they have not had any activity (e.g. clicking, typing, moving the mouse) on their current web page for the set amount of time. This prevents someone else from accessing their account and their project data if they leave a workstation without properly logging out or closing their browser window. There exist some customizable settings that govern login activity, such as being able to manually set the number of failed login attempts before a user is locked out of the system for a specified amount of time. Also available is a user suspension status, which can be set for any given user. Suspending a user allows them to remain a user in the system but denying them access to the entire REDCap application until their suspended status has been revoked. For various reasons, suspending a user is preferable to deleting the user permanently from the system.

There also exist some security settings that are specific to table-based authentication. These include allowing users to set their own password and, if desired, prevent them from re-using a recent password. Users can also be automatically forced to change their password after a specified number of days. If using LDAP or Shibboleth authentication, the system can be set to allow any and all users to be able to automatically create their own REDCap account, or conversely it can be set for those users to only be able to access REDCap when an administrator has first added them to a User Whitelist. In this way, the local REDCap administrators get to choose whether they want to be more or less restrictive with regard to how new users gain access to REDCap.

Security in relation to authentication can be improved by enabling two-factor authentication in REDCap. Two-factor authentication (sometimes referred to as two-step login) is an optional REDCap setting that can be enabled in the Control Center. Once enabled, two-factor authentication requires the user logging in to perform an additional step in the login process. There are several options available that can be enabled, such as the user having to enter a 6-digit code obtained via email, via SMS text message, or via Google Authenticator app on a mobile device. Another option is to use the Duo app if Duo two-factor authentication is already utilized at the institution.

REDCap Logging and Audit Trail

REDCap has a built-in audit trail that automatically logs all user activity and logs all pages viewed by every user, including contextual information (e.g. the project or record being accessed). Whether the activity be entering data, exporting data, modifying a field, running a report, or add/modifying a user, among a plethora of other activities, REDCap logs all actions. The logging record can itself be viewed within a project by users that have been given privileges to view the Logging page. The Logging page allows such users to view or export the entire audit trail for that project, and also to filter the audit trail in various ways based upon the type of activity and/or user. The built-in audit trail in REDCap allows administrators to be able to determine all the activity and all the data viewed or modified by any given user.

REDCap Data Export and De-Identification

REDCap allows users to export any and all data from their REDCap projects, supposing they have been given full data export privileges. Data may be exported in various ways and in various formats. Data may be exported as a CSV (comma-delimited) file from a report, and also in the form of a PDF file from the data entry page when viewing a particular record. The Data Export Tool page allows users to export a project’s collected data to a CSV file, which can be opened in Excel or several popular statistical analysis packages, such as SAS, SPSS, Stata, and R. Data may also be exported from a project using the REDCap API, which can export the data in several formats (CSV, JSON, or XML).

The Data Export Tool has advanced export features that allow one to implement data de-identification methods, such as being able to automatically remove free-form text fields, remove dates, perform date shifting, and remove fields tagged as identifiers (e.g. PHI) from the data file being exported by the user. User privileges can also be set so that some users may be allowed to export data from the project but will have the data de-identification methods imposed as a means of preventing them from exporting sensitive data, either mistakenly or intentionally.

REDCap Data Storage

REDCap stores its data and all system and project information in various relational database tables (i.e. utilizing foreign keys and indexes) within a single MySQL database, which is an open source RDBMS (relational database management system). The front end of REDCap is written in PHP, which is a widely used, robust, open source scripting language for web applications. Setting up the web server and database server and securing the communication of the servers to each other and to the end-user are the responsibilities of the partner institution that is installing REDCap, and thus they must be completed prior to installing REDCap. The institution installing REDCap will store all data captured in REDCap on its own servers. Therefore, all project data is stored and hosted there at the local institution, and no project data is ever transmitted at any time by REDCap from that institution to another institution or organization.

REDCap's native webpage encoding and database storage collation is UTF-8, which allows for non-English languages to be utilized in user-defined text that gets stored in REDCap. This includes data entered for a project or the text defined for a survey question or database field label, among many other types of user-defined text. REDCap's database tables implement MySQL's Innodb storage engine, which allows for the use of foreign keys for referential integrity, transactions, and row-level locking (as opposed to table-level locking), all of which are needed in REDCap for consistency, performance, and scalability.

REDCap does not employ any kind of encryption of data (i.e. encryption "at rest") on its database server. (This is not to be confused with encryption of data "in transit" (i.e. via SSL) to the database, which should always be done and must be set up by the partner institution.) The encryption of database data is not necessary if the database server is properly secured. However, some institutions or compliance offices impose requirements such that encryption is required. In those cases, partner institutions are encouraged to seek either filesystem-level encryption solutions or database-level encryption solutions. Currently, there are two options for data at rest encryption at the database level:

- MariaDB 10.1.3+ support encryption (using Google patch)
- MySQL 5.7.11+ (and Percona Server 5.7.11) has InnoDB tablespace level encryption

All documents that are uploaded and/or stored in REDCap will be stored on the local REDCap web server, on a file server locally mounted to the web server, or on a separate server using secure communication using WebDAV protocol (SSL supported). In older versions of REDCap, some types of files were stored in a database table, but this is no longer done. The files stored in REDCap are not stored in an encrypted format but are stored "as is" in their original form (although under a different file name). If the partner institution chooses to store the REDCap files on the local web server, it is highly recommended that the files be stored in a secure directory that is not accessible to the web (i.e. not under the web root).

REDCap Security

To help protect and secure the data stored in REDCap's back end database, the software application employs various methods to protect against malicious users who may attempt to identify and exploit any security vulnerabilities in the system. Such methods will be described here in technical detail.

In REDCap, all incoming data gets intentionally filtered, sanitized, and escaped. This includes all data submitted in an HTTP Post request and all query string data found in every URL when users access REDCap, among other modes through which user-defined data gets submitted in the application. Server environment variables that are vulnerable to forgery by end-users are also checked and sanitized. All user-submitted data is properly filtered for any possibly harmful markup tags (e.g. <script>) and is then escaped before ever being displayed on a web page within the application. SQL queries sent to the database server from REDCap are all properly escaped before being sent. If any values used in an SQL query originated from user-defined values, they would also have already been sanitized beforehand, as described above. User-defined data used within SQL queries have their data type checked to prevent any mismatching of data types (e.g. making sure a number is really a number). These processes of sanitization, filtering, data type checking, and escaping all help to protect against methods of attack, such as Cross-Site Scripting (XSS) and SQL Injection. To specifically protect against Cross-Site Request Forgery (CSRF), which is another method of attack, REDCap utilizes a "nonce" (a secret, user-specific token) on every web form used in the application. The nonce is generated as a unique value for each new HTTP request in every REDCap session.

Additionally, REDCap employs "rate limiting" on its web pages, in which there is a set maximum number of web requests per minute that are allowed from a single IP address, and after that maximum is hit, the IP address of that user is permanently banned from REDCap. The rate limiting value of requests per minute per IP is customizable and can be modified within REDCap's Control Center, if needed. Rate limiting prevents denial of service attacks by bots as well as preventing other types of hacker attacks that require making many requests to the server in a short amount of time, such as with a BREACH attack. Regarding the prevention of BREACH attacks specifically, in addition to using rate limiting, REDCap always outputs an invisible string of random text of random length on every web page (to conceal the page's true length) as an effective technique for mitigating such an attack. REDCap's use of a unique nonce token on every web form also greatly diminishes the possibility of a BREACH attack.

Many institutions that have installed REDCap have made use of enterprise-level web application security scanners to scan and test REDCap's security and its ability to withstand various methods of attack. REDCap has performed very well in such instances. Any partner institution that wishes to scan REDCap using security scanning software is free to do so without the consent of Vanderbilt University or of REDCap developers so long as the particular instance of REDCap being scanned is their own and is not hosted by another institution/organization. If an institution decides to perform a security scan of REDCap and finds any medium- to high-risk security issues that are directly REDCap-related, they are encouraged to contact the REDCap developers at Vanderbilt so that such issues can be immediately addressed.

With regard to the security of cookies used by REDCap, all session cookies and other cookies related to authentication that are created by REDCap will automatically have the "HttpOnly" attribute set to TRUE. By default, the "Secure" cookie flag will be set to FALSE. while this is slightly more insecure, setting its value as TRUE can sometimes cause login issues with certain server configurations, such as reverse proxies. So for compatibility reasons, it is set as FALSE by default. However, any institution can enable the "Secure" flag of session cookies to improve security by setting session.cookie_secure=On in the REDCap web server's PHP.INI configuration file.

Data Interoperability

REDCap has modules for easily exporting and importing data, which are very useful for getting data in and out of REDCap manually through the web interface. But for various reasons, there sometimes exists the need to be able to migrate data from system to system, often in an automated or programmatic fashion. REDCap has the capability to move data in and out of individual REDCap projects using a couple different methods that do not require a web interface.

The REDCap API is an interface that allows external applications to connect to REDCap remotely, and is used for programmatically retrieving or modifying data or settings within REDCap. This includes performing automated data imports/exports from a specified REDCap project, importing/exporting a project's metadata (i.e. data dictionary), events. This is even an API method for creating whole new projects. The API is a built-in feature of REDCap, so no installation is required. The REDCap API implements the use of tokens as a means of authenticating and validating all API requests that are received. Similar to the Data Import Tool in REDCap's web interface, the API also implements data validation when the API is used for data import purposes in order to ensure that only valid data gets stored. The API provides a very efficient way to move data either to or from another system easily.

Another data interoperability service that REDCap can utilize is the Dynamic Data Pull (DDP) module. DDP is a special feature for importing data into REDCap from an external source system. It provides an adjudication process whereby REDCap users can approve all incoming data from the source system before it is officially saved in their REDCap project. While the main DDP module comes preinstalled in REDCap, two web services (minimally) will need to be created by the host institution to function as middleware so that REDCap can use the services to communicate with the data source system and vice versa. DDP supports only data coming into REDCap from a source system (whereas the API supports both data import and export). Because DDP assumes that all incoming data from the source system may not be trusted as valid or that only a subset of the data coming from the source system needs to be imported, DDP utilizes an adjudication web page inside REDCap's web interface where end-users can review the data obtained from the source system before confirming that it be imported into their REDCap project. DDP can fetch data from the source system immediately in real time as soon as the record identifier (e.g., medical record number) is entered. It also has an auxiliary cron job that runs daily that fetches any potential new data that might be entered into the source system.