9magnets LLC Application Security Guide

Cameron Banga
cbanga@9magnets.com
9magnets, LLC
Last Revision – 2014.12.11

About

The following document describes the various security measures used by the development team at 9magnets, LLC, for protecting information contained inside of our client mobile or web applications. This document is a work in progress, and will be updated if any security measures were to change over the course of development.

General Security Practices

General Web Service Credential Exchange

The retainment and exchange of account credentials are managed using 1Password. A shared team vault is stored using a private Dropbox Pro account, which is protected on a per-user basis using a secure, complex, service unique password. The vault is encrypted using tamper-proof authenticated AES-256 encryption, which is also resistent to GPU cracking through use of PBKDF2-HMAC-SHA512.

When using 1Password, authenticated and decrypted user sessions are not stored to disk, and an authenticated vault is then locked after several minutes of inactivity on desktop, or after leaving the 1Password applciation on iPhone or Android.

In creation of credentials, our team policy is to create passwords or phrases that are as unique, and as long/complex as allowed by the accompanying web service. For applications and services that support multi-factor authentication, we strive to enable such security measures as well.

In the rare occassion that a security credential can not be shared between team members using

1Password, asymmetric public-key cryptographic cypher-text is used to exchange credentials over inherently insecure methods such as email or chat client. This encryption is managed on the UNIX command line, using GPG, a GPL licensed standard that is compliant with the current OpenPGP standard.

Team Communication

Email servers are hosted and managed using MediaTemple, and for some users, then sent and managed using Google's GMail web server. IMAP protocol is used to exchange messages between MediaTemple and/or Gmail, and host device mail clients on iPhone or Mac OS X. Email accounts are secure using complex and unique passwords, along with multi-factor authentication where applicable.

Day-to-day team communications internally are managed via Slack. Slack's privacy policy and security can be found here, but in summary, data exchanged in Slack's team chat client is encrypted end-to-end using 256-bit AES, using the Diffie-Hellman elliptic curve key exchange. Files exchanged are stored on the AWS web stack, and are subject to Amazon's security compliance policies. Like any other web service, we use long, unique, complex passwords to authenticate our Slack accounts.

Data Rentention and Backup

The 9magnets team performs day-to-day work using OS X. Standard practice within our company is to format drives using the Mac OS Extended, Journaled, and Encrypted disk format. Disk encryption passphrases are stored in 1Password using complex, unique keys on a drive-by-drive basis. User login accounts are created using passwords that are as long and complex as reasonable, given frequent use, and devices are typically set to require user authentication after turning off or after falling asleep. Apple's iCloud helps with physical device security, and in the case of a lost laptop, phone, or tablet, the device can remotely be wiped immediately upon acknowledgement of loss.

Each team member uses a triple-redundant backup plan to insure data redundance and protect against loss. External, local hard drives are used at each team member's home/apartment to provide an offsite yet local backup using Apple's Time Machine. A Mac Mini is located on site at our company office, which runs OS X Server, which similarly runs a Time Machine backup for each team member on-site at office. Finally, Arq is used in coordination with Amazon's AWS Glacier service to provide a redundant, encrypted, automatic, and incremental off-site backup, with data stored on Amazon's East Coast web servers, using an AWS account owned and managed by the 9magnets team. Each Arq backup is also encrypted against a strong, unique, complex password so that encrypted data stored server side is inaccessible if AWS credentials were to be compromised.

Client Project Data Storage

When applicable, for any web service that may control or manage end user or client data, we work with our clients as best as possible to ensure that our client has full access and ownership of account rights, with our usage of the service facilitaed through either a partner account, or through authorized access of the administrator account from our client. We respect your project or customer data, so that our clients have full data portability to move to an in-house development team or another developer and away from 9magnets as a provider, if so desired.

For managing of source code and software, our team uses Git as our revision control system of choice. Our online host of choice is Bitbucket. More information on their privacy policy can be found at their website. Team member user accounts are managed using complex, unique passwords, and source code contributions are managed and authenticated using a computer-specific public-private SSH keypair. In the case of a new computer purchase, computer wipe, or restore, new keys are generated and old keys are removed from the user Bitbucket profile. All source code is retained and managed using private repository accounts.

Occasionally, Bitbucket will be used as a wiki host and defect issue reporting tool, which could contain additional project information or communication between 9magnets and our clients.

For Photoshop PSD files, PNG images, internal outlines, specifications, etc, we use Dropbox Pro accounts to share and store documents between team members. Dropbox's security information can be found on their website. In general, all data transit is encrypted using 128-bit or higher AES encryption.

On some projects, Basecamp is used internally by our team to manage internal project discussion, project documents, calendar events, and more. For detailed information on Basecamp's security practices, see their website. In summary, Basecamp uses HTTPS for all data transit, their backups are fully encrypted using GPG, and servers are controlled to limit physical access to only authorized Basecamp employees.

Mobile Device Security

For day-to-day use, each 9magnets team member uses the iPhone 6 or 6+. Client data is often accessed or viewed on these devices using Slack, Dropbox, or Bitbucket, etc services. Each device is secured using Apple's Touch ID, to prevent data compromise. Apple's work in encryption and security is extensive and well recorded, for more information, see Apple's iOS Security Document.

For mobile application testing, we rely on a variety of iOS and Android devices to test against. As general policy, we typically tie these devices to dedicated test accounts. We have a single test Apple ID account for iOS devices, and likewise, a single Google account for Android devices. In some occasions, we may use our personal accounts to tie to these devices for frequent testing. Devices are purchased on a regular basis, and at any given time, we will have between 5 and 20 test devices on site in rotation for a specific project. With these devices, we typically use standard security practices to protect against theft/data loss.

Unless required for a project, we tend to refrain from attaching services such as Dropbox, personal email, or other services that could contain private client data. Such test devices typically only contain test application binaries. These devices are imaged frequently to install new operating system versions, etc.

Software Testing

In general, unless another method is requested by client or approved by client, all software testing occurs internally using 9magnets employees. Software testing is typically done using both automated and manual testing practices. Currently, our team uses HockeyApp to distribute pre-release software, as well as for collecting bug reports and crash information.

For information on HockeyApp's security practices, please see this knowlegdebase article. As a note, HockeyApp was recently purchased by Microsoft, and it's security/privacy polices could be in flux. Our team will continue to analyze potential impacts on our clients from this acquisition, and we will adjust our testing strategies as we see necessary.

Data Retention

As a general policy, we encourage clients to discuss data retention needs on a case-by-case basis. Client data will, unless requested otherwise, be retained within team member backup files, and could be retained indefinitely. As noted in the Data Rentention and Backup section, such data stored in backup will be encrypted using unique, complex passwords.

We will occasionally prune and remove old project documents, files, and passwords, but in most situations, project files will be retained for a minimum of one year. We strive to be transparent and provide easy access to source code, project art files, required login or password files, etc, and encourage our clients to verify that they have access and personal copies of any project files deemed important and necessary for retaining. Although in many projects, data will be retained indefinitely, 9magnets does not guarantee retention of client project files past 12-16 months of completion of a project.

Physical Security

In general practice, our team works with laptops that are most typically taken home from our physical office location every evening. In the case of a break-in to either home or office, our team should be aware of any lost devices within 12 hours of theft, and in most situations, much sooner. Password authentication for devices along with full-device encryption on hard drives, phones, and tablets will make it much more difficult for any attacker to bypass and gain data before the device can be wiped remotely.

Only 9magnets partners and our landlord have keys with physical access to our building and mailbox.

Physical key access is revoked from any partner upon leaving the company.

It is rare for us to have physical paper records on a project, as most of our work is digital. In the case of reciepts, invoices, payments, etc, we try to digitally scan for recordkeeping, and then retain physical copies as long as reasonable or required by law. For any project documentation in paper format deemed sensitive, we would retain this record as long as necessary to complete the project, and then would shred the document using a commercial-grade shredder.

Failure and Redundancy

As our mobile applications are written in native code, we typically have no dependency on consistent web connectivity, which can frequently be spotty or unavailable on mobile devices. We work our best to ensure applications are designed to run independent of web connection, allowing the user a seemless experience even if they can not connect to the server due to a lack of internet availability.

Because web connected apps' content can change so dynamically, with up-to-date data being essential, we have choose service providers to drive the systems powering content into the applications while ensuring exceptional up-time, making the likelihood of a downtime window exceedingly small. Although we do realize that web systems are not fail-proof, so we work to built our systems so that such a failure would not be a catastrophy.

All of our developed web services are built on top of free and widely-available open source systems, allowing us to quickly and easily re-deploy the entire production system to another hosting provider in a manner of hours. When feasible and desired, we aim to keep a slave-server instance readily available on a secondary service provider, in case of any potential primary service provider failures.

Breach Disclosure

At the time of writing this document, we have never had a known breach of client private project data, passwords, signing keys, etc. We follow industry best practices in searching for possible breaches using logs/tracking, etc, and keep abreast as best as possible with news about applicable security issues for our projects and accounts. We strive to keep our systems as up-to-date as possible to prevent against known vulnerabilities, and apply security patches as soon as possible.

If a situation were to arise where we became aware or suspicious to a scenario where client data could have been compromised, we would immediately work to isolate and shutdown any affected services, data, or servers in order to mitigate potential ongoing or future attackes. We would then do our best to immediately understand how the attack occured, instutute a plan to correct and prevent it, and deploy the new system. All while simulatenously communicating with affected clients to understand and discuss data potentially lost and it's effect on a project or the client.

Web Security Practices

Hosting

In general, unless detailed elsewise, we tend to host projects either at Heroku or Digital Ocean. Heroku is used in situations where scale and size are either unknown before the kickoff of the project or unpredictable. Heroku accounts used by our teams have unique, complex passwords, and we try whenever possible to use multi-factor authentication to additionally protect our accounts. For full detailed security information, see Heroku's website. In summary, Heroku uses HTTPS and SSL certifications to ensure encrypted end-to-end communication. Projects are deployed using Heroku's toolbelt, in a method similar to Git pushing or pulling. Additionally, databases and project files are stored by Heroku on Amazon's Web Services, encrypted and stored using industry standard practices. Heroku offers more information on their architecture at this website, and is used by some of the biggest projects on the internet due to their strong security and ability to handle scale. Heroku is owned by SalesForce.

Digital Ocean is a virtual private server provider renowned for their speed, uptime, and great performance. When a project has a known and predictable size and scale, Digital Ocean is often used to help reduce cost and offer increased capability. When using Digital Ocean, we typically deploy either Linux distributions Ubuntu (64 bit 14.04 most frequently at this time of writing), or CentOS (64 bit 6.5 or 7.0 most frequently at this time of writing).

When working in a UNIX server environment, we constantly strive to follow best security practices to keep the server secure, including but not limited to practices such as using SSH keys for login, limiting root user access, the use of unique users and passwords for essential processes, IP address access isolation when applicable, and more. For more information about your project and it's security, please feel free to ask!

In programming web services, we typically code in Python on top of the Django web framework. Django is an open source framework renowned for it's frequent contributions, flexibility, scalability, and security. It's currently used by projects such as Instagram, Pinterest, and Rdio.

When building content management services to help drive mobile application projects, we work to create strong administrator accounts to help prevent service or data compromise. When applicable, our root administrative user for content management systems will a complex and unique 30-50 character string used for both user name and passwords, to make brute force access mathmatically difficult. User access accounts are limited as best as feasible on a per-project basis, and we try to create complex passwords that work as best as possible within our client's needs on a project.

When applicable or requested by a client, we have also installed and managed SSL certificates to provide end-to-end encryption between our web services or API to the end user or mobile application.

Mobile App Security Practices

APIs and the App Stores

We work to use industry standard practices when coding our applications for iOS and Android, and work tirelessly to remain current in our knowledge of the operating systems, device requirements, system APIs, programming language best practices, etc.

When working at the system level API level, we work to the best of our ability to use manufacturer approved APIs, obey manufacturer application approval rules to keep app submission from app store rejections, etc. We follow developer newsletters and blogs to keep up with the most up-to-date information on general app store rules and regulations for Apple and Google's platforms, to best ensure that we meet any rules and regulations placed upon us by the platform providers.

Connecting to Web Services

When working with an API connected to a content management system produced by our team, we work to isolate API access through HTTP Basic Authentication, using a read-only user account when applicable.

In server/API instances where applicable, we work to implement advanced authentication techniques such as OAuth, in order to protect our clients and their customers.

We have extensive experience in connecting with some of the most popular web services that are prevalent on the internet today, including SalesForce, Instagram, Facebook, Twitter and more.

Working with OS Versions

Our general philosophy in developing mobile software is to tailor our work for the largest audience possible without jepordizing usability and security for our clients and the end user. Each project is unique, so no operating system target can be universal, but we work extensively to help provide the best balance for our clients in order to best meet their project needs.

We most commonly target official operating system releases from Google and Apple released in the past 18 months, when starting a project. Exceptions to this would be in scenarios where requested features are impossible in a previous OS version, a security vulnerability makes a previous OS version untenable, etc. With each new OS release by Apple or Google, we work to best understand the functionality and security ramifications for our clients, in order to best advise how to improve or modify each application best.

Push and Analytics

While not a universal addition to all projects, most mobile apps tend to include both push notifications and advanced user analytics, either by necessity or by client demand. Often, we can be flexible with our clients with respect to push and analytic service providers, but most commonly we use Parse for push notifications and Google Analytics for user analytics.

In most apps, Parse is only used as a service provider for push notifications, however it can also be used to store simple databases or user data as well, and may be used for these purposes on a case-by-case basis. For best understanding, we advise that you visit the Parse products page, or talk with us about how we use it for your needs. Parse is owned by Facebook, and their team works to employ industry standard security practices throughout all of their services. For more information, see their security page.

For general analytics, Google Analytics is most commonly used by our team. This service allows for us to see generalized, anonymoized information about general end user application use, so that we can report this data back to our clients and improve application use or performance based on the results. General data collected by this service could include data such as how many users used an application, how many times each user launched the application, how long the application was launched for, information on how frequently each feature inside of the application was used, etc. For further information on the data privacy concerns with respect to a specific application, please feel free to ask our team and read Google's privacy policy for analytics.

Mobile App Sandboxing

To protect data and documents once stored locally on an end user's iPhone, iPad, or Android device, both operating systems employ a security technique often defined as "App Sandboxing". As a general description of the technique, applications inside of a phone are limited and unable to see or edit documents and data from other applications stored on the device. This helps keep user private data safe and secure from malcious actors or programs.

9magnets works to follow and employ the best application sandboxing methods for iPhone and Android. For more information on app sandboxing, please read Apple's developer documentation on the topic.