OpenlyOperated.org

Audit Report - Website & Backend

Snapshot On May 31st, 2019 - Revision 1
Openly Operated Certification v1.0

https://openlyoperated.org/report/openlyoperated

Audit Date: 6/3/2019

Auditor Information

Name: Ryan Koven

Email: rpkoven@gmail.com

LinkedIn: https://www.linkedin.com/in/ryankoven/

Qualifications: Security Consultant - Stratum Security
3 years Application Security Engineer
5 years Software Engineer
Open source developer

The auditor certifies that they have performed the requested verifications in the attached Audit Kit. The auditor provides the following executive summary of their findings:

The review used the access provided to the read-only AWS user, along with the code hosted in GitHub, to test the claims made in this template, and to evaluate the app based on the Openly Operated criteria. The review confirmed directly (or found substantial evidence to support) each of the claims made in the audit template. The verifications in this template provide substantial support for the claim that openlyoperated.org is an Openly Operated app.

Auditor Signature

Audit Kit

This Audit Kit contains end-to-end, verifiable proof of how OpenlyOperated.org works. It lets auditors and anyone to fully audit or personally verify the privacy and security of OpenlyOperated.org. Additionally, everyone is invited to report issues, questions, or concerns via the contact methods under the Identify Operation section.

Read-Only Account

This read-only account allows access to viewing the operator's AWS Console for OpenlyOperated.org — it allows you to see a limited version of what the operators and employees can see. The read-only account helps users feel more in control of what is happening behind the scenes with their data, but for a comprehensive look, users should read through the proof-backed points, references, and Audit Reports in the rest of this Audit Kit.

DANGER: Your IP will be logged and saved in the publicly viewable and permanent Infrastructure Audit Trail if you log in to the console with the read-only account below. It cannot be deleted, so if you're concerned about this, use a proxy.

Account ID: `973242705982`
Username: `AuditUser`
Password: `QWERTY123!@#`
Region: `us-east-1` aka `Northern Virginia`
Console Login

Highlighted Services

- CodePipeline: Deploys code to the `Main` and `Admin` servers. Check that the code in the CodeCommit first step matches the code referenced in the Open Source section, and that the code is successfully deployed to the servers.
- EC2: Controller and status panel for the `Main` and `Admin` servers. Check that none of the servers (or "instances") were launched with SSH keys — there should be nothing under the "Key Name" column.
- CloudFormation: The infrastructure dashboard. Check that the templates of the stacks match up with the infrastructure files in the CloudFormation public repository.
- EC2 Security Groups: Firewall rules for both internal and external access. Check that for EC2 instances, the attached security group follows the Principle of Least Privilege.

Auditor Verifications

Describe Verification	Result	Comments/Issues/Questions
Accessed AWS read-only user account successfully. Viewed EC2, CloudFormation, Security Groups, CodePipeline, Database, Storage settings and configs. Verified read-only permissions.	Passed	
OOPROD-Main, OOPROD-Admin commit hashes from CodePipeline, Github match and are deployed	Passed	
Verified 3 EC2 instances launched without keys for SSH access.	Passed	
Verified with diffs that YAML config files in AWS match config files in Org-CloudFormation	Passed	
Security groups manage inbound/outbound traffic, database access, codebuild appropriately	Passed	

	Describe Verification	Result	Comments/Issues/Questions

Claims With Proof

The following are selected claims made by OpenlyOperated.org regarding user privacy and security. These claims are backed up by specific proof from source code, infrastructure, and other references. Learn about Claims With Proof, or for more technical details, view Requirements.

Email Address Privacy

The Claim The email addresses collected for the Openly Operated newsletter are not viewable by anyone, including the operators and employees of OpenlyOperated.org.

The Proof

- **Encrypted/Hashed Email Addresses -** User email addresses collected for the newsletter are encrypted[1][2] using AES-256[3], the same encryption the United States government uses for top secret information. The emails are also salted and hashed[4][5] using SHA-512[6] for fast lookup.
- **No Access To Encryption Keys, Hash Salt -** The encryption key and hash salt are randomly generated[7][8] on initialization and encrypted as a Secure String[9], so operators never see it and can't decrypt the email addresses.
- **No Unaudited Access To Database -** The database password is randomly generated[10] on initialization and encrypted as a Secure String[9], so operators never see it and can't access the database. Additionally, firewall rules[11][12] prevent the database from being accessed directly by operators or anyone external.

References

1. OpenlyOperated.org "Shared" source code repository, Newsletter Subscriber Model, Line 150
2. OpenlyOperated.org "Shared" source code repository, Security Utility, Lines 53-61
3. Wikipedia, "Advanced Encryption Standard", April 2019
4. OpenlyOperated.org "Shared" source code repository, Security Utility, Lines 12-16
5. OpenlyOperated.org "Shared" source code repository, Newsletter Subscriber Model, Line 149
6. Wikipedia, "Secure Hashing Algorithm 2", April 2019
7. OpenlyOperated.org infrastructure repository, "Shared" template, Lines 521-608
8. OpenlyOperated.org infrastructure repository, "Shared" template, Lines 950-960, 973-981
9. AWS Documentation, Systems Manager Parameter Store - Secure String
10. OpenlyOperated.org infrastructure repository, "Shared" template, Lines 1025-1033
11. OpenlyOperated.org infrastructure repository, "Shared" template, Lines 1428-1437
12. AWS Documentation, Security Groups For Linux Instances, April 2019

Confirmed three suggested proofs through AWS, source inspection.	Passed		
Confirmed encryption settings at rest and in transit. Confirmed salted hasies.	Passed		
Confirmed unable to access keys; confirmed properly salted hashes	Passed		
Confirmed on multiple PostgreSQL instances deployed in AWS	Passed		

No User Tracking

The Claim OpenlyOperated.org does not do user tracking or use any third party analytics tools.

The Proof

- **No Logging IPs Or Personally Identifiable Info -** Only errors are logged for debugging issues[1], and even in those cases, no IP or Personally Identifiable Information is logged[2]. Normal requests and successful requests, such as visits to the OpenlyOperated.org site, are not logged.[1]
- **No Third Party Analytics -** Third party analytics tools are a privacy risk. Some tools are hosted by ad companies (like Google Analytics), and other tools save obscene amounts of user activity onto third party servers, like full screen recording and tracking[3]. OpenlyOperated.org uses no third party analytics[4], minimizing user privacy risk.
- **No Access To User Email Address -** User tracking through the email newsletter is not possible as operators do not have direct access to the user email addresses — See the "Email Address Privacy" claim and proof above.

References

1. OpenlyOperated.org "Main" source code repository, App Logging, Lines 26-29
2. OpenlyOperated.org "Main" source code repository, App Logging, Line 23
3. "Investigation into at least 11 iOS apps sending sensitive data to Facebook, inc sexual activity", 9to5Mac
4. OpenlyOperated.org "Main" source code repository, Website Header Template

Confirmed three suggested proofs through AWS, source inspection.	Passed		
Confirmed in AWS, source.	Passed		
Confirmed in AWS settings, source.	Passed		
Confirmed in "Email Address Privacy" section.	Passed		

Requirements

The following are requirements for being Openly Operated, and they exist to ensure a product's full, verifiable transparency. Without them, claims about security or privacy could not be made because there could be gaps, loopholes, or backdoors in these claims. Learn about Requirements.

Identify Operation

Product: OpenlyOperated.org
Informative Website
OpenlyOperated.org is the website for Openly Operated, the certification for transparency. Its purpose is to inform and explain the certification, why it's necessary, and how to get certified.

website | contact

Location: United States
Delaware, California, and Florida
OpenlyOperated.org is operated by Confirmed Inc, which is incorporated in Delaware with filing number 6543601.

Corporate: 251 Little Falls Drive, Wilmington, DE 19808

San Francisco: 535 Mission St, 14th Floor, San Francisco, CA 94105

email | telegram | reddit | twitter

Co-Creator: Johnny Lin
San Francisco, CA
Johnny is responsible for technical decisions and building out Openly Operated's examples, specifications, and documentation. He was an engineer at Apple iCloud and has built several apps featured by the App Store. Johnny is an alumni of Brown University.

email | medium | telegram | linkedin | twitter

Co-Creator: Rahul Dewan
Miami, FL
Rahul builds tools for Openly Operated, and also handles legal, management, and directional decisions. He also created Duet Display, a second screen for your iPad, and has previously worked at Apple. Rahul is an alumni of Stanford University and Georgia Tech.

email | linkedin

Describe Verification	Result	Comments/Issues/Questions
Verified ICANN WHOIS registration. Verified Twitter, Telegram, Reddit presence.	Passed	
Verified Confirmed Inc entity information with Delaware Department of State:Division of Corporations website.	Passed	
Communicated with Co-Creator by phone and email. Verified Twitter, LinkedIn, Medium presence.	Passed	
LinkedIn profile is consistent with description provided.	Passed	

Open Source

OpenlyOperated.org is 100% open source. Architecture and code are detailed below.

Architecture

The majority of OpenlyOperated is written in Javascript, using Node.js on the Express framework. It currently has three code repositories: The `Main` and `Admin` repositories depend on the `Shared` repository. It is hosted on Amazon Web Services, with infrastructure brought up via CloudFormation. See Open Infrastructure for details.

Repository: `Shared`

Shared Node.js Modules, Models, Utilities | GitHub | `commit 5bfc4daab2981cd2596519510c4ed656be8cdec3`

This exists to reduce code duplication between the `Main` and `Admin` webapps, and contains common shared code like logging, email, database access, models, security, and other utilities. Updating `Shared` requires a redeployment of `Main` and `Admin` for changes to take effect.

Repository: `Main`

Node.js on Express | GitHub | `commit 5abf5ac4f2e5b1f1a1a587f051bf8b147cfd1048`

This is the public-facing OpenlyOperated.org website, which uses Handlebars.js templating to render pages generated by the Content Management System on `Admin` . It also has an API for the email newsletter signup.

Repository: `Admin`

Node.js on Express | GitHub | `commit 2939221e9a86651f0f8bc4cd45e287ff4d1cf839`

This is a private administrative dashboard for OpenlyOperated.org only accessible to whitelisted IPs. It uses Handlebars.js templating to render pages and has numerous documented APIs that make up its Content Management System and email newsletter features.

Security & Encryption

- **User Data Secured** - Sensitive user data should always be secured with strong encryption or strong hashes. This ensures that internal operators can't read user data, and also make it extremely difficult for hackers to read user data in the unlikely case of a database compromise.
  - **Strong Encryption** - User emails collected for the newsletter are encrypted using AES-256, which is the same encryption the United States government uses for top secret information. View the OpenlyOperated.org code using this encryption here and here.
  - **Strong Hash** - User emails collected for the newsletter are salted and hashed using SHA-512 for faster lookup. View the code using this hash here and here.
- **Brute Force Prevention** - Each client is limited to a certain number of requests per time window, preventing malicious users from spamming the website and API. See OpenlyOperated.org code for this feature here.

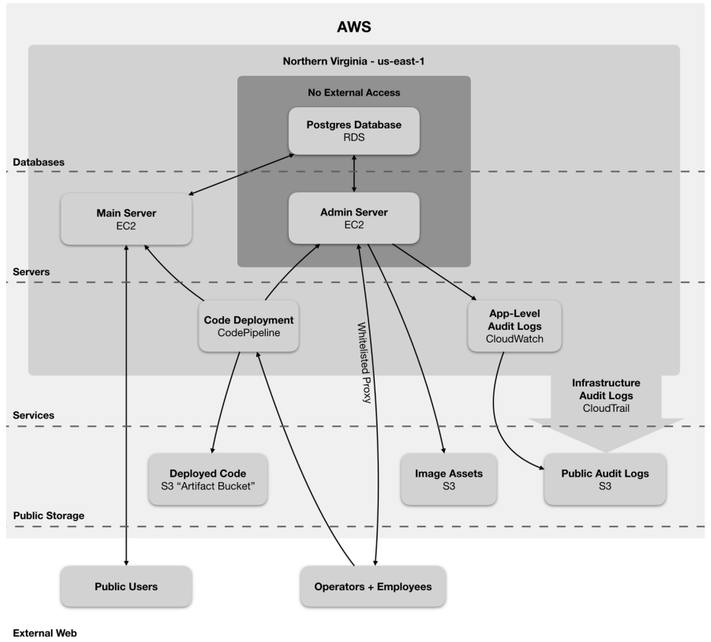
Describe Verification	Result	Comments/Issues/Questions
Viewed on GitHub, CodeCommit. Viewed dependencies, supporting libraries. Confirmed basic structure in description. Dependencies in package.json files are up-to-date, secure, commonly used libraries for session management, logging, etc.	Passed	
Reviewed code on GitHub. Basic code review confirms the functionality described.	Passed	
Reviewed code on GitHub. Basic code review confirms the functionality described. Confirmed supporting libraries.	Passed	
Reviewed code on GitHub. Basic code review confirms the functionality described. Confirmed supporting libraries.	Passed	
Confirmed in “ Claims with Proof" section. Confirmed TLS settings with SSLyze tool.	Passed	TLS server supports only TLS 1.2, which is a strong setting.
Confirmed with code review.	Passed	
Confirmed with code review.	Passed	

	Describe Verification	Result	Comments/Issues/Questions
	Inpsected front-end templates and package.json dependencies. Confirmed no analytics/tracking APIs.	Passed	
	Confirmed with code review.	Passed	Cookies are set with "secure" and " HttpOnly" flags. An effective session timeout of five minutes is a secure default setting.
	Confirmed with code review.	Passed	
	Confirmed with code review and CloudFormation settings.	Passed	
	Confirmed with code review.	Passed	App gets Admin setting from session, not cookie, which is a secure pattern.
	Confirmed with code review.	Passed	
	Confirmed with code review.	Passed	

- **No User Tracking** - Apps and services should have as little user tracking as possible, for both privacy and security reasons. OpenlyOperated.org doesn't track IPs or any other personally identifiable information, and only stores a temporary cookie that is only capable of displaying error messages.
  - **No Third Party Analytics** - Using third party analytics tools may be convenient, but they're also a privacy risk. Some tools are hosted by advertisement companies (like Google Analytics), and other tools save obscene amounts of user activity onto third party servers, like full screen recording and tracking. OpenlyOperated.org uses no third party analytics APIs, which minimizes the risk of user privacy violations.
  - **Temporary Cookie For Errors** - OpenlyOperated.org temporarily stores a secure cookie that's only capable of displaying one-time error messages, which is invalidated and expired after five minutes. The cookie does not store IP addresses or any personally-identifiable information. Additionally, since there is no direct access to the database (see above), operators can't abuse user cookies/sessions.
  - **Minimal Logging With No IPs/PII** - Only errors are logged for debugging purposes, and even in those cases, no IP or Personally Identifiable Information is logged. Normal requests and successful requests, such as visits to the OpenlyOperated.org site, are not logged.
- **Admin Dashboard Security** - The Admin server is able to send email newsletters to users, modify posts, and send database queries, so it's essential that this server has adequate protection.
  - **Firewall Whitelist** - The firewall (via Security Groups) is configured to only allow a single whitelisted IP is to connect to the Admin server. To all other IPs, the Admin server is not reachable and will time out.
  - **Login Required** - The Admin Dashboard requires registration and email confirmation of an account with an address ending in @openlyoperated.org, preventing anyone else from accessing the dashboard.
  - **Email Alerts** - Email alerts are sent to the administrator whenever a successful or failed login attempt occurs. All other errors and potentially suspicious activity also dispatch alerts to the administrator.
- **Updated Security Patches** - The latest NodeJS Long Term Support ("LTS") version is installed on all servers, and operators are alerted when a new update is available. The updates are manually reviewed for compatibility, then installed via a simple deployment.

Open Infrastructure

OpenlyOperated.org is 100% open infrastructure, with one public infrastructure repository. All configuration and usages of third-party APIs are described below.



Infrastructure

AWS CloudFormation | GitHub | `commit 54dd126a34d1f94678c888293a145f2928a82718`

OpenlyOperated.org runs on Amazon Web Services, using CloudFormation to bring up almost every resource, from servers (which run Ubuntu) to DNS configurations to deployment pipelines. All the CloudFormation templates are public on Github. OpenlyOperated.org is in the `us-east-1` Northern Virginia AWS region.

Security & Encryption

- **No Direct Server Access** - Direct, unaudited server access is a security and privacy risk because it allows changing code, installation of unauthorized/malicious applications, monitoring traffic, and more. OpenlyOperated.org disables all direct server access for everyone, including employees.
  - **Servers have no SSH key** - SSH is the most common way of accessing servers directly, but it requires an SSH key to be preloaded on the server. The KeyName property of the instances' launch configuration in both 2-Admin.yml and 3-Main.yml are not defined, so instances launch with no SSH key, making direct SSH access impossible.
  - **Firewall blocks SSH port** - AWS Security Group rules are configured to not allow traffic on SSH's port 22 for both Main and Admin, so even if an SSH key managed to get onto a server somehow, it still couldn't be used to directly access the server.

Describe Verification	Result	Comments/Issues/Questions
Confirmed AWS infrastructure, viewed CloudFormation configs, templates, matched CloudFormation configs in AWS with configs in Github.	Passed	
Confirmed with AWS read-only account, by inspection.	Passed	
Confirmed with Nmap scan and inspection of AWS security group settings.	Passed	Only port 80 is open for Admin, Main.

Describe Verification	Result	Comments/Issues/Questions
Confirmed by inspection of config file.	Passed	
Confirmed in AWS settings.	Passed	
Confirmed by inspecting config file.	Passed	
Confirmed auditing/logging on AWS.	Passed	
Confirmed by inspection using AWS read-only account and review of config files.	Passed	
Confirmed with code review and the SSLyze tool.	Passed	Only TLS 1.2 is supported, which is a secure setting.
Confirmed by inspection of AWS security groups and code review.	Passed	
Confirmed by review of CloudFormation config files.	Passed	

    - **Session Manager is disabled** - AWS's Session Manager is Amazon's brand of direct server access. This feature is blocked in 1-Shared.yml, and in the unlikely case that it were somehow activated, its root access is also disabled to minimize potential harm.

- **No Unaudited Database Access** - Direct, unaudited database access allows operators to export and modify user data, and also allows hijacking of user sessions. OpenlyOperated.org's external database access is not allowed, and operators do not have database credentials. Any direct database access is clearly logged and audited.
    - **Firewall blocks access** - AWS Security Group rules are configured to only allow access from `Main` and `Admin` servers, so even if the database credentials were stolen by someone, they still couldn't access the database.
    - **No Access To Password** - The database password is randomly generated on initialization, and encrypted as a Secure String, so even the operators never see it. Any access or decryption of the database password triggers a tamper-proof and publicly viewable CloudTrail event, which are reported as a violation by Open Watch.
    - **Only Audited Access** - Since databases sometimes require schema migration and other mandatory manual maintenance, non-direct queries to the database are allowed, but these queries are logged for auditing purposes, and can only be executed only through the Admin dashboard. This prevents potential internal and external abuse of this database access — see the Audit Trail requirement for complete explanation.
- **No Access To Keys Or Salts** - In addition to using strong encryption/hashes, operators have no access to the encryption keys or salts, because they're randomly generated on initialization, and encrypted as a Secure String, so even the operators never see it. Any access or decryption of the database password triggers a tamper-proof and publicly viewable CloudTrail event, which are reported as a violation by Open Watch.
- **HTTPS Required** - HTTPS is a basic necessity for encrypting internet connections, reducing the chances of eavesdropping and preventing certain attacks. OpenlyOperated.org requires HTTPS by redirecting all HTTP requests to HTTPS. Additionally, OpenlyOperated.org requires TLS 1.2, a modern type of HTTPS encryption.
- **Principle Of Least Privilege** - At all points in the infrastructure, access to data and resources abide by the Principle Of Least Privilege, so in case of any compromise, damage is minimized. Roles for server instances limit access to only what is necessary — for instance, the `Main` server role cannot access the `Admin` server's database password. Database roles for each server type are also restricted to what is necessary for the role. Firewall rules via AWS Security Groups limit server communication to only the ports and the target groups that are necessary.
- **Updated Security Patches** - All servers run the latest version of Ubuntu, and security patches are automatically installed as they're released. Operators are alerted when patches require a system restart, and the restarts or reprovisioning of servers is done at the earliest possible time, and leave audit logs in CloudTrail.

Third Party APIs

Since other companies that run third party APIs (such as analytics tools) are likely not Openly Operated nor provably transparent, usage of third party APIs should be minimized in order to reduce risks to user data. OpenlyOperated.org uses no third party APIs and and sends no user data to third parties. While it would have been faster and easier to use existing third party APIs for features like the newsletter, OpenlyOperated.org built those features from the ground up so that we can provide end-to-end proof of the privacy and security of user data.

Code Deployment

Code is deployed using AWS CodePipeline, which consists of CodeCommit (a hosting service for code repositories), CodeBuild (building and testing code), and CodeDeploy (deploying code to servers) — see the relevant infrastructure code here.

To deploy code, code is `git push` ed to the CodeCommit repository, where it is automatically built by CodeBuild and uploaded into an S3 "Artifact Bucket" (aka "Deployed Code" on the Infrastructure diagram). CodeDeploy then deploys the code in the artifact bucket to the servers. So, the code in the S3 "Artifact Bucket" is the same as the code in production.

Audit Trail

Audit trails show all operator actions since the creation of the Openly Operated product. Their purpose is to provide an unbiased, verifiable source of truth. For OpenlyOperated.org, since its Infrastructure Audit Trail does not record the operator's manual database actions, there is an additional audit trail called the Database Audit Trail.

Infrastructure Audit Trail

The infrastructure audit trail provides a log of all major operator actions in bringing up and maintaining the infrastructure. They are usually automatically generated by the platform that the product is built on, and should have a way to verify integrity. OpenlyOperated.org uses AWS's CloudTrail service for its infrastructure audit trail.

- **Impartial** - This audit trail is generated automatically by AWS CloudTrail. AWS CloudTrail logs all operator actions on AWS, and operators are not able to alter or delete any CloudTrail logs.
- **Authentic** - OpenlyOperated.org's CloudTrail logs have [Log File Integrity Validation] enabled, which generates an unforgeable checksum alongside every log file, preventing modification without detection. Even deletion of logs without detection is impossible, for two reasons: One, the checksum of each log is based partially on the previous file's checksum, and two, even if there is no operator activity, CloudTrail periodically generates a "no-op" checksummed log file, preventing the operator from claiming there are no logs because nothing has occurred. Verification of authenticity is discussed in the section below.
- **Comprehensive** - CloudTrail is enabled first, before bringing up any infrastructure, and it's never disabled, so logs cover the entire lifetime of the product. Any gaps would be detected by the verification mechanism below. For finding potentially dangerous operator actions, see the "Detect Dangerous Operator Actions" section below.

Describe Verification	Result	Comments/Issues/Questions
Confirmed in “ No user tracking" section. Reviewed code, config files.	Passed	
Confired with AWS read-only user. Commit hashes on Github match CodeCommit hashes.	Passed	
Confirmed with AWS user and with CloudTrail docs.	Passed	
Verified via AWS CloudTrail documentation, inspection of AWS settings.	Passed	
Verified by inspection using AWS read-only account.	Passed	

Log File Integrity Verification

AWS provides a manual way to check the integrity of CloudTrail logs against their checksums, but since there are thousands of logs, Openly Operated built a free, open source Mac tool that automatically does this integrity check for CloudTrail logs in an AWS account, called Open Watch. Here's how to use it

Download the open source Open Watch tool and either use the precompiled build release, or follow the instructions to build and run it. Then, use the following settings:

- Rules: Check all checkboxes
- Start: 05/02/2019 (The date OpenlyOperated.org was initialized)
- End: Today's Date
- AWS Key: `AKIA6FGOJFA7P3KC5ZVS`
- AWS Secret: `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`

Click "Audit" to begin. This may take up to 30 minutes or an hour, since Open Watch is downloading tens of thousands of audit logs and verifying each of them. If any integrity issues are found, they'll be displayed in the main whitespace area after verification is complete. Please note that it's better to keep the app in the foreground during auditing, because macOS may heavily throttle applications that are in the background and cause the process fail.

Detect Dangerous Operator Actions

To fulfill the Comprehensive section of the Audit Trail requirement, it can be difficult to sift through the thousands of CloudTrail logs to find potentially dangerous operator actions. Fortunately, the Open Watch tool above also performs detection of several common types of dangerous operator actions, displayed as "Rules" in the Open Watch interface, with explanations of each rule.

Manual Analysis

To manually analyze the infrastructure audit trail, download it below.

OpenlyOperated.org Infrastructure Audit Logs - May 3rd, 2019 - Download

Database Audit Trail

OpenlyOperated.org has a custom-built Administrator dashboard (the `Admin` repository) that allows the operator to send manual database queries, in order to perform the necessary administrative task of database migrations. However, because these queries are arbitrary, that could be abused by the operator to retrieve or modify user records, putting user privacy at risk. These queries are not tracked or logged by the Infrastructure Audit Trail (CloudTrail), so OpenlyOperated.org tracks this with the Database Audit Trail.

It works like this: Right before a database query is executed, the `Admin` server logs the query to two places: 1) A log viewable to the read-only account called `AdminAudit` [1][2][3][4], accessible here and 2) A log file in a public S3 bucket[5][6][7], accessible here. This way, the public and auditors get to see all database queries that the operator executes (but not the responses), keeping the operator honest.

- **Impartial** - The Database Audit Trail is generated by custom code in the `Admin` server, referenced above. The Infrastructure Audit Trail, along with the Establish Provenance section below, ensures that the referenced code is indeed the same code that's deployed in production.

Describe Verification	Result	Comments/Issues/Questions
Confirmed with Open Watch 6/3/2019.	Passed	
Inspected logs directly and with Kibana log viewer/query tool.	Passed	
Verified logs on AdminAudit AWS account and in S3 bucket. Audit logs have logged the same queries.	Passed	Viewed only one query in the logs (assuming not many queries have been made since this product was just deployed).
Confirmed audit trail and "secure code provenance" section.	Passed	

- **Authentic** - Modification of Database Audit Trail log lines is not possible because CloudWatch Logs is not capable of modifying log lines. However, it is possible to delete the entire log stream or log group and re-create it, creating the illusion that there never were any log entries. This tampering can be detected by Open Watch, which searches the Infrastructure Audit Logs for the DeleteLogGroup and DeleteLogStream API calls.
- **Comprehensive** - The same verification for Authenticity can be used to ensure comprehensiveness. As long as the Database Audit Trail log stream and group haven't been deleted, they should contain all database queries issued since the creation of OpenlyOperated.org.

References
1. OpenlyOperated.org infrastructure code repository, Global file, Lines 407-408
2. OpenlyOperated.org infrastructure code repository, Shared file, Lines 764-775
3. OpenlyOperated.org "Admin" code repository, Database Query Controller, Line 50
4. OpenlyOperated.org "Shared" code repository, Audit Utility, Lines 15-39
5. OpenlyOperated.org infrastructure code repository, Shared file, Lines 195-211
6. OpenlyOperated.org "Admin" code repository, Database Query Controller, Line 52
7. OpenlyOperated.org "Shared" code repository, Audit Utility, Lines 41-51

Detect Dangerous Operator Actions

Use the read-only account to view the CloudWatch log stream `PostgresQueries` in the `AdminAudit` log group to see a complete list of all database queries manually executed by the operator. Alternatively, check the public S3 bucket `AdminAuditBucket` for log files prefixed with `PostgresQueries`. The queries listed, if any, should be for database migrations, not queries for extracting rows from the database. For example, a `CREATE TABLE` query is fine, but a `SELECT * FROM USERS` is not.

Establish Provenance

Provenance is proof that both the code and infrastructure presented match the code and infrastructure in the actual users' production environment.

Infrastructure Provenance

The infrastructure presented above matches the infrastructure in production because it's brought up by CloudFormation, which is AWS's "code-as-infrastructure" service.

To verify this, use the Read-Only Account to log into the console and check the CloudFormation service in the `us-east-1` region. Under each stack, check the template tab and compare it with the infrastructure files in the CloudFormation public repository. Then, any differences between the template and the current infrastructure can be detected with the Detect Drift functionality.

Auditors should also verify that the domain that users are connecting to is indeed the one auditors are examining, by checking the Route 53, which is AWS's DNS service. Verify that the one of the nameservers in the openlyoperated.org Hosted Zone matches the nameserver returned by a host nameserver lookup from your local machine. On macOS, the command is `host -v openlyoperated.org`.

For further proof of infrastructure provenance, download OpenlyOperated.org's full infrastructure audit logs above, under the "Infrastructure Audit Trail - Manual Analysis" section.

Describe Verification	Result	Comments/Issues/Questions
Verified that deleting trails will trigger events that will be logged. No such events detected in logs.	Passed	
Verified by inspection.	Passed	
Viewed logs on AWS log viewer and in S3 bucket.	Passed	Only one (benign) query in logs: SELECT count(*) FROM posts;
Verified using AWS read-only account by inspecting diffs between CloudFormation files in AWS and files on GitHub.	Passed	
Confirmed with host command.	Passed	

Source Code Provenance

The source code presented above matches the source code in production because they're both `git remote` s of the same repository, which is built and deployed without modification to production servers via CodePipeline.

Manual modification of source code in production is not possible, because direct access to the servers is disabled and blocked — see the "Open Infrastructure - Security & Encryption - No Direct Server Access" section above.

To verify provenance, use the Read-Only Account and examine the CodePipeline service. Check that the code in the CodeCommit matches the code referenced in the repository, and that the code is successfully deployed to the servers. CodePipeline and CodeDeploy also displays the history of all the previous code deployments, allowing verification of any previous version of source code in production.

Describe Verification	Result	Comments/Issues/Questions
Verified no direct server access (no SSH). Confirmed in CodeCommit and CodeDeploy that the commit hashes for what's deployed match the head of the code in GitHub.	Passed	

Revision History

Revision 1 - May 31st, 2019

- Initial revision.

Issues, Comments, Questions Not Addressed In Audit Kit

It may be helpful for the audit template to have an easy way for the auditor to attest that they did not work on and have no financial stake in the app they're auditing. I note here that I did not work on this app in any way and I am not an investor in or an employee of Confirmed Inc.