Security Accreditation of **Year of Engineering**

Client: 23red
Client contact: Paul Blundell
Authors: Thomas Colliers & Ben Richebois
Version 1 (Dec. 27th, 2017)

Requirements

IL1 - This guidance is for services holding information that's classified by the government as 'official'.

IL2 or 3 - not "secret" or "top secret". These accreditations require Government Security Clearance. An audit is required, similarly to the process for the ISO 27001. In order to complete IL1, according to the GDS guideline, we must identify the risks posed by technology choices, processes, staffing and data aggregation we're using.

> To pass the live assessment for point 7 you usually need to:
> 1. Describe your team's approach to security and risk management
> 2. Describe your ongoing interactions with the business and information risk teams, eg SIRO, IAO and data guardians
> 3. Describe any outstanding legal concerns, eg data protection or data sharing
> 4. Explain how you're keeping your understanding of the threats to your service up to date, and explain how the threats have changed during beta
> 5. Explain how you're keeping your cookie policy and privacy policy up to date

Table of Content

Security approach

For Year of Engineering, we target the **IL1** accreditation - we only manage "information that's classified by the government as 'official' ".

Prerequisite

Our server infrastructure and development teams follow a security policy based on the ISO/IEC 17799 and ISO/IEC 27002 security standards as well as the OWASP Top Ten Project guideline documents. Security responsible team members have an OWASP membership.

The technology choices made for Year of Engineering are secure by default by offering strong protection against all common web security threats such as injection, XSS, CSRF, broken auth systems and data exposure.

We take the principle of least privilege seriously. We use separate virtual instances for the different system components to limit the impact of a compromised component.

Environment

We are using industry standards for:

- **Stack and hosting**: AWS (Amazon), GitHub, Travis CI, React (Facebook front-end framework) and Wagtail (Made by Torchbox, based on *Django*, a strong and secure Python framework. It is already used for other UK projects.)
- **Security protocols**: SSL certificates (HTTPS over TLS protocol)

We are also using high secure standards to maintain the code on the repository, with SSH connections. The repository is private for now, until we'll make the repository open after releasing the final deliverable. In this way, it prevents unfortunate leak during the development or the configuration of the servers.

The back-end and front-end are logically separated to provide an additional layer of protection.

The application comprises 4 different environments, linked to 2 different CMS (and database).

- **Local** - used for development. Testing back-end
- **Staging** - for internal review and tests on real server. Testing back-end
- **Testing** - shared with client. Production back-end
- **Live** - the official website that points at yearofengineering.gov.uk. Production back-end

That's also mean a testing CMS is setup, stored on a different EC2 container - assets (covers, thumbnails, PDF, photos) are also stored in a separate S3 bucket.

Most of the site content and all the user submitted content are not stored on the user facing webserver. They are managed by the CMS system and can only be accessed through an API or through the CMS.

Additional notes

Here are all the environment that we'll describe in the following document.

	Web site	CMS
Local	Localhost (development)	yoecms.bliss.build/admin
Staging	yoe.bliss.build	
Testing	yoengineering-website.eu-west-2.elasticbeanstalk.com	yoengineering-content.eu-west-2.elasticbeanstalk.com/
Live	yearofengineering.gov.uk	

Moreover, we discern 2 types of user: **end user** and **admin**.
Without considering Google Analytics (cf. dedicated section), the only informations we get from the end users are the forms that they decide to fill in ("Sign Up to Our Newsletter" and "Register to Be a Partner" pages).

The admin can connect to the CMS, and add content on the following sections: News, Events, Videos, Meet the engineer, Lesson ideas, Continuing Professional Development, Inspiring videos and Share your Experience.

Requirements

Confidentiality

Information should only be seen by people who are authorised to access it

Content is the same for everyone, there is no dedicated content or information, no member access. There is however user submitted information that should be protected from unauthorized access. The API only exposes methods to submit the user information, not to retrieve it. Retrieval is only possible through the CMS or through Mailchimp in the case of newsletter subscriptions.

Integrity

Information should only be modified by people who are authorised to do so

The API formalizes what public users are able to do with the data. Other actions than which are documented are not possible. The CMS is access protected with user accounts.

Availability

Information should be available when needed (problems or attacks shouldn't stop you getting information from the system)

We are working on an automatically scaling Elastic Beanstalk configuration that allows for high availability under varying amounts of load. AWS deploy new instances and route them in case of failures or DDoS. See detail on their white paper.

Non-repudiation

Nothing should happen in a system that can't be traced back to a responsible person

Changes in the CMS and in the source code itself are tracked through a revision control system. It is not possible to delete user submitted content through the CMS, only to retrieve it.

Technology

Code

Common attacks

The backend is built on Django, a secure framework that natively provide some tools such as CSRF protection, Form tampering protection, SQL injection prevention, and XSS prevention.

For the front end, React JS brings a reasonable level of security, made with ES6 (the latest JavaScript update), and is highly maintained by Facebook (who is the creator of this framework) and its community.

External packages

The source code is composed with many modules - a module could generate a simple layout element or build more complex components such as a dynamic map or a video player. These modules are made by third parties, and we are perfectly aware that they can contain some security risks. However, we have only selected well-known packages that are already used for a large amount of web app. And all these modules are open-source themselves, so everyone can review the source code to improve it or to fix security issues. We ensure to always get the latest versions of each module (according to their dependencies).

Server

AWS

"AWS: Overview of Security Processes" - AWS Security Responsibilities (pp. 6)

> Amazon Web Services is responsible for protecting the global infrastructure that runs all of the services offered in the AWS cloud. This infrastructure is comprised of the hardware, software, networking, and facilities that run AWS services.

The IT infrastructure that AWS provides to its cus
tomers is
designed and managed in alignment with security best practices and a variety of IT security standards, including:

- SOC 1/SSAE 16/ISAE 3402 (formerly SAS70)
- ISO 9001 / ISO 27001
- ITAR
- See the all list at https://do.awsstatic.com/whitepapers/aws-security-whitepaper.pdf

In addition, the flexibility and control that the AWS platform provides allows customers to deploy solutions that meet several industry -specific standards, including:

- Criminal Justice Information Services (CJIS)
- Cloud Security Alliance (CSA)

"AWS: Overview of Security Processes" - Physical and Environmental Security (pp. 8)

> AWS data centers are housed in nondescript facilities. Physical access is strictly controlled both at the perimeter and at building ingress points by professional security staff utilizing video surveillance, intrusion detection systems, and other electronic means. Authorized staff must pass two-factor authentication a minimum of two times to access data center floors. All visitors and contractors are required to present identification and are signed in and continually escorted by authorized staff.

"AWS: Overview of Security Processes" - Network Monitoring and Protection (pp. 13)

> The AWS network provides significant protection against traditional network security issues, and you can implement further protection. [...]
> Distributed Denial Of Service Attacks, Man in the Middle Attacks, IP spoofing and Port Scanning, Packet sniffing by other tenants

Docker Container - EC2 (AWS)

"AWS: Overview of Security Processes" - Multiple Levels of Security (pp. 20)

> Security within Amazon EC2 is provided on multiple levels: the operating system (OS) of the host platform, the virtual instance OS or guest OS, a firewall, and signed API calls. Each of these items builds on the capabilities of the others. The goal is to prevent data contained within Amazon EC2 from being intercepted by unauthorized systems or users

Because application code is deployed through a Docker container, the attack surface is limited to the application code itself. The container is a minimal container.

Database - RDS

Data is stored in Amazon RDS with the appropriate firewall protection that prevent connections from outside of the Amazon network.

Static data storage - S3

Large binary files are stored on Amazon S3 with appropriate access control settings.

Connections

For the user, the connection is encrypted over HTTPS; informations he sends (partner registrations, forms and the like) are hidden.
One SSL certificates is set up for the live website and one another for the CMS platform - meaning login and password are encrypted too.

Process

Repository

Open-source code

We identified two major risks:

1. Sensitives variables directly visible (i.e. third parties login and passwords or API keys)
2. Flaws in the code

We took the steps that had to be taken to prevent these risks. We are using Environmental variables (never sent on the repository). And we used other open source modules and snippets - it's a guarantee that they were reviewed by many different persons.

Deployment

Automated workflow

We set up an automatic deployment. When we release a new feature or when we fix an issue, we push the new code on the main branch of the repository. It triggers a test phase on Travis CI, based on unit tests we wrote throughout the development. It prevents critical changes we could send while implementing a new function. It works likewise for the other environments (testing and live)

Deployment workflow

Moreover, the current workflow to push new updates is the following:
First, we develop in local. Then we push to staging environment, and 23red is able to test the latest release on this environment. When approved, we push this same version to testing, where the final client can review it. When confirmed, then we release on live version - after 2 different reviews. We only push code to the next step when it has been approved in the previous environment:

Local → Staging → Testing → Live

Maintenance

We are responsible for management of the guest OS (including updates and security patches), any application software or utilities we install on the instances, and the configuration of the AWS-provided firewall on each instance.

We warranty our code for 30 days after the final release approved by client (it doesn't concern content posted on the CMS). After this period, we can offer a maintenance contract.

Staffing

Involved stakeholders

For this project, many stakeholders are involved: the developers, the design team, the intermediate management on every levels and the final client
There are few risk related to this context. First, we share few passwords: the CMS access (one for Staging and one for Testing) and the AWS account (Elastic Beanstalk). Moreover, these passwords are visible in multi places: emails, Trello cards, Google Drive documents and Skype conversations.

To prevent any future leaks, we will create new dedicated accounts for the CMS (to be also able to monitor where the leak comes from) and set up new strong passwords. We can also enable Two-Factors Authentication for AWS login.

Data aggregation

User forms

User can intentionally filsl in 3 different forms:

1. Newsletter
2. Partner registration
3. Feedback

All of them are sent over HTTPS and are stored in our secured database (refer to the chapter Technology > Server > Database for more details).

Admin users can download the 3 forms on dedicated sections of the CMS. The forms are formatted in CSV, and included into encrypted archives. A password is required to unlock these archives.

Google Analytics

Users behaviors are monitored by Google Analytics. Data privacy and security is entirely handled by Google - about monitoring, sending data to their servers and storage.
Excerpt from their white paper:

"Safeguarding your data" - Data privacy and security

> In web-based computing, security of both data and applications is critical. Google dedicates significant resources towards securing applications and data handling to prevent unauthorized access to data.
>
> Data is stored in an encoded format optimized for performance, rather than stored in a traditional file system or database manner. Data is dispersed across a number of physical and logical volumes for redundancy and expedient access, thereby obfuscating it from tampering.
>
> Google applications run in a multi-tenant, distributed environment. Rather than segregating each customer's data onto a single machine or set of machines, data from all Google customers (consumers, business, and even Google's own data) is distributed among a shared infrastructure composed of Google's many homogeneous machines and located in Google's data centers.

Inform users

Cookie

The website displays a cookie banner on the top of the page to warn the users that *YearOfEngineering* collects some informations about them (legal requirement). There's a link to a dedicated section in *Privacy policies* with more details.

Term and Conditions

There are two specific sections about user's rights: Term & Conditions and Privacy policies.

Appendix

GOV.UK

GOV.UK - Securing your information	Link
Service Assessments	Link
Vulnerability and penetration testing	Link

Amazon Web Services

AWS - Security White paper	PDF

Google Analytics

Google Analytics - Data privacy and security	Link
Products certified ISO 27001	Link