**HACCS**

Highly Automated, Control-Centric Security

Small business users get a very low-touch, automated, control-centric security scan that covers a fair majority of real world attacks.  
  
The goal is to have a tool that covers as many key controls as possible while requiring minimal input. It should require NO technical/IT knowledge on behalf of the admin.

**Key Components:**

* **Web Scan:** Admin enters URL of website, and our server analyzes it. Checks technologies used to build the site, whether those technologies are up-to-date (or, often the case with new technologies, entirely cloud hosted). It will also attempt to discern business name, industry, and street address. If there appears to be a login page, it will attempt to analyze whether that page meets some minimum criteria (minimum password length, max number of login attempts, OR not-a-robot checking). Provide clean, explanatory output, as well as more extensive information in links to component scans.
* **Home Assessment** (Employee/Device Scan/Inventory): User devices/browsers. Admin shares link, or shares a list of emails for completion tracking. Automatically inventoried while users receive some very smart/fast security training and authorize future phishing training.
* **Report Generation** visualizing necessary remediations, prioritized.
* **Assessment Generation** auto-population (to whatever extent possible) of simple, controls-based assessment forms (like CyberEssentials).

The first two components are developed in more detail below. Notes on generation of report/remediations and auto-population of controls-based assessments are included specific to each section.

**Web Scan:**

Admin enters URL. We hit the page, Analyze it:

There are many existing website security scanners. The key insight here is to run an initial analysis of the site to check what technologies are used, and then run appropriate 3rd party scanners via API, and feed the results into a standardized framework for entry into control-based security compliance forms. Initial analysis will also scrape things like business name, industry, and address to autofill those fields on forms.

Code Needed:

**Technology Mapping:** Flags that indicate the use of web app technologies (ie wordpress or joomla) and then either mapping to appropriate third party scanners, or further analysis to determine whether the package being used is (at a minimum) a supported version or (better) correctly implemented and up-to-date

**Basic Info Scraping:** To save time on any later certification submission, as well at to build internal database/heuristics

**Store Data:** Associated with a given user/business, as appropriate.

**User Friendly Output** including:

* Technologies Used
* Positive security features (like CSP Header)
* Remediations necessary
* Comparison to other businesses in industry/locality (as appropriate)

**Home Assessment:**

A unique aspect of this system is the ‘home assessment’ which populates an inventory of employees, roles, and devices from a simple list of emails (or link sent to all employees). When employees click the link in the email, they are taken to a webpage which records their browser version and operating system from javascript, and asks them what their role is (“admin or normal”) and whether they ever/regularly access client information such as credit cards, addresses, etc from that device. It contains a brief paragraph or video on security training, highlights key organizational security policy, and asks for consent to engage in false phishing attacks for educational/research purposes.

Code needed:

**link generation** logic. specific to organization (hashed ID) and email address, and authentication code of those things hashed with a weekly pepper. Allow resend of email. But rate limit.

**Send emails:**  using ie sendgrid or mailjet. Trigger on admin hitting ‘enter’ on main page: Generate Organization ID (if not already present) cycle through list of email addresses, and for each, generate a link, generate the email containing that link, and send to that address.

**Web Page:** Lightweight, clean modern front end. Ask a friend for help with best way to do this in 2021.

**Store Data:** Store gathered data to low-maintenance scalable database, preferably something like firebase.

**Allow Data Access:**

Via admin page, with login flow, render prettified result of database entries for that organization ID, as well as graph breakdowns of operating system (google charts API) and flagging of outdated or unknown operating systems (by comparing each against an internal list)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Email | Role | OS | Data Access | Data Types | IP/network |
| Kelly | Admin | Mac OS 10.13 | Weekly | All |  |
| Joe | Support | Mac OS 10.12 | Daily | Credit Cards, IP |  |
| Miran | Marketing | Mac OS 10.12  iOS 7.8 | Daily | Emails |  |

**User Friendly Output** including:

* Pie Chart of operating systems
* Number of users at each role who have/haven’t replied
* List of browsers used.
* List of operating systems used.
* List of machines used if Possible.