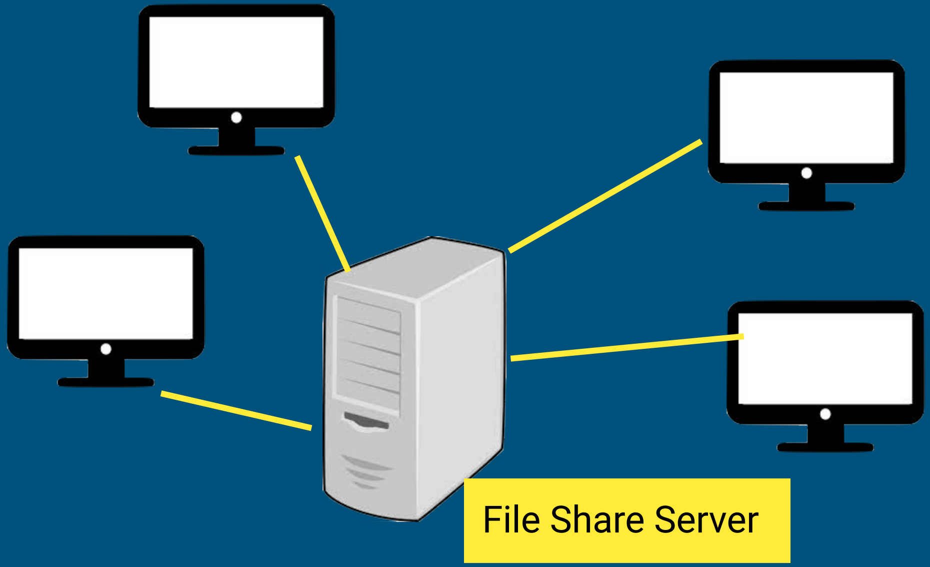
VulnTraq

Nathan Pavlovsky

How SMBs Use Excel For Vuln Tracking



Tracked Information:

- Vulnerability name
- Status [Ongoing/Closed/etc]
- Tracking & Patching Ticket(s)
- Dates Tickets Opened + Closed
- Other information

Issues With Such Usage of Excel

- Loading an Excel spreadsheet from the shared drive and saving the changes made to it can be slow
- There is no automated graph/report-generation mechanism built-in
- Organizations typically create a brand new Excel spreadsheet template when starting their vulnerability programs instead of having a versatile vulnerability-handling tracking solution at hand
- If multiple staff members have a share-file based spreadsheet open simultaneously, ensuring that their changes properly synchronize and versions do not clash is a challenge
    - Cloud tools like Google Drive + Microsoft OneDrive are available to ensure syncing.
    - Orgs. are understandably reluctant (90% are very/moderately concerned with public cloud security per Bitglass' 2015 *Cloud Security Report*) [5]
- Need something that is robust, straight out of box, and not cost-prohibitive for SMBs

Possible Solutions?

Some vulnerability handling tools are available:

- **Open Source:** Vulnerability scanners such as OpenVAS, nmap, Vulners, and vuls.io.
- **Commercial:** *Alienvault's USM Anywhere Vulnerability Management & SIEM*

Do not either have capabilities for ticket management + report analysis or are too expensive

Problem Statement

Description of Problem To Solve

There is a need for a vulnerability-patch-ticket tracker solution that addresses the challenges currently affecting vulnerability-handling programs using Excel-based tracking approach.

It must

- Be free & open source to meet SMBs budgetary constraints
- Avoid the pitfalls of the popularly-used Excel spreadsheet approach
- Generate reports
- Work “out-of-the-box” (i.e. no excessive man-hours used to implement custom functionalities)
- Work in conjunction with a free & open source ticketing system

Demo

Limitations

Limitations – UVDesk API

- The UVDesk updated API documentation on Github* is sparse in terms of writing
  - The official UVDesk website's detailed documentation **is outdated** and for older API versions
- Extra frontend-side processing and workarounds in the ticket-creation process due to insufficient built-in API capabilities.
- Custom API commands can be created. Downside is time constraints for development + not easily reusable, so not done

* https://github.com/uvdesk/api-bundle/wiki/Ticket-Related-APIs

Limitations – Evaluation

- Evaluating whether it improves efficiency and provides any and all desired functionalities involves field testing it in 1+ enterprises for an extended period of time
  - Before & after metrics collected
  - Implementing it + field testing it during the compressed Summer 2022 **not feasible**
- Instructor-agreed upon evaluation approach: present VulnTraq survey to potential users & survey
  - *South Carolina Department of Revenue's* security team members

Limitations – Development

- Time Constraints:
  - ~ 1.5 months of actual programming with goal of a viable prototype
    - May not be ideal in terms of algorithmic efficiency
  - Hasn’t gone through an extensive testing process to flag bugs with unit tests
  - Didn’t undergo software security pentesting

Future Work

Future Improvements for VulnTraq

- Changing the reports capabilities from a pop-up modal to a new page, adding new reports + graphs
- Capabilities for exporting presented table information and reports as .CSV/Excel files
- Integrating with open source vulnerability scanners to allow for vuln ticket patching verification
- Formal field testing it in at least one enterprise to verify operational capacity and gather suggestions for improvement
    - If not possible, presenting to larger information security professional audience for evaluation

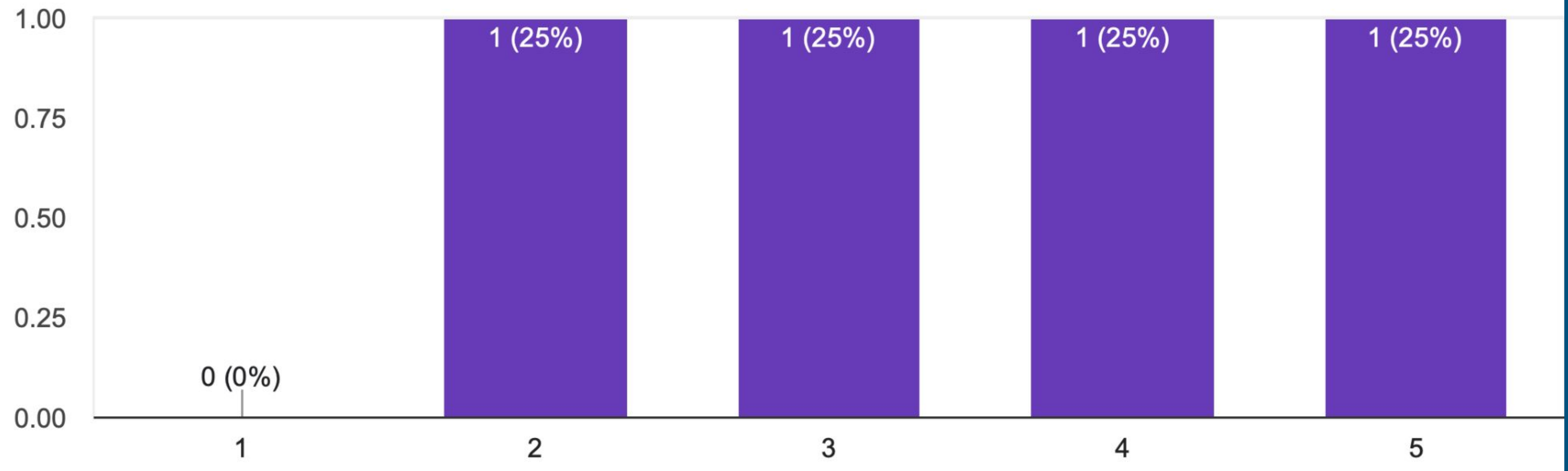
Evaluation

Evaluation Method

- Field testing not feasible
- Survey presented to *South Carolina Department of Revenue's* IT Security Team
    - Independent outside panel
    - Security Analysts and Manager
    - 4 respondents

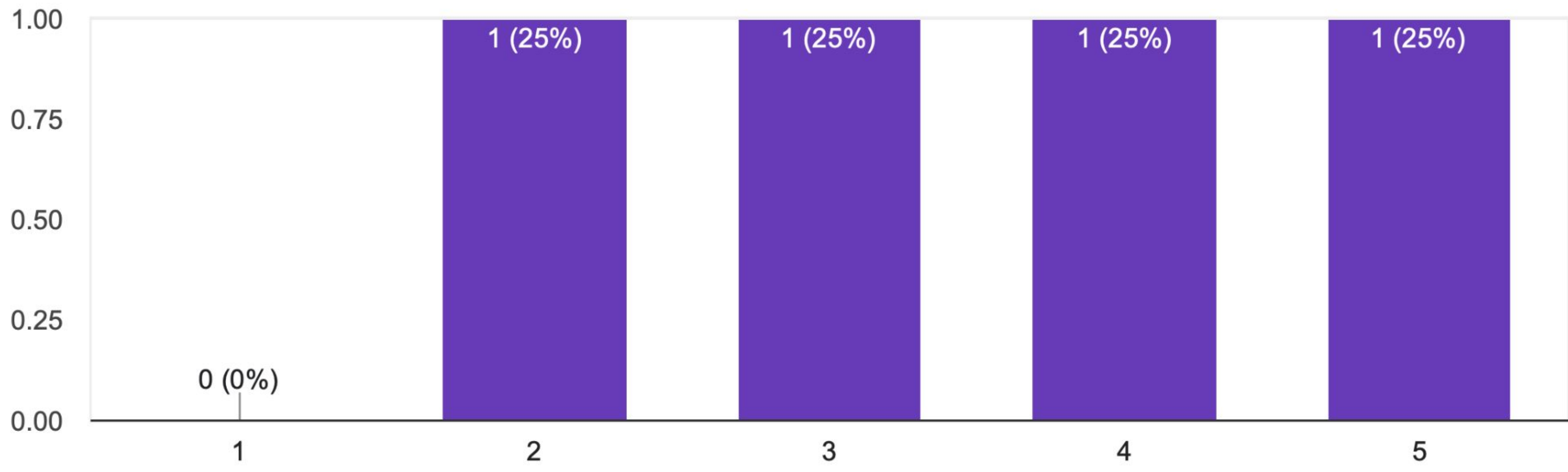
On a scale of 1 - 5 (Never to Always), how often do you work with vulnerability management on a daily basis?

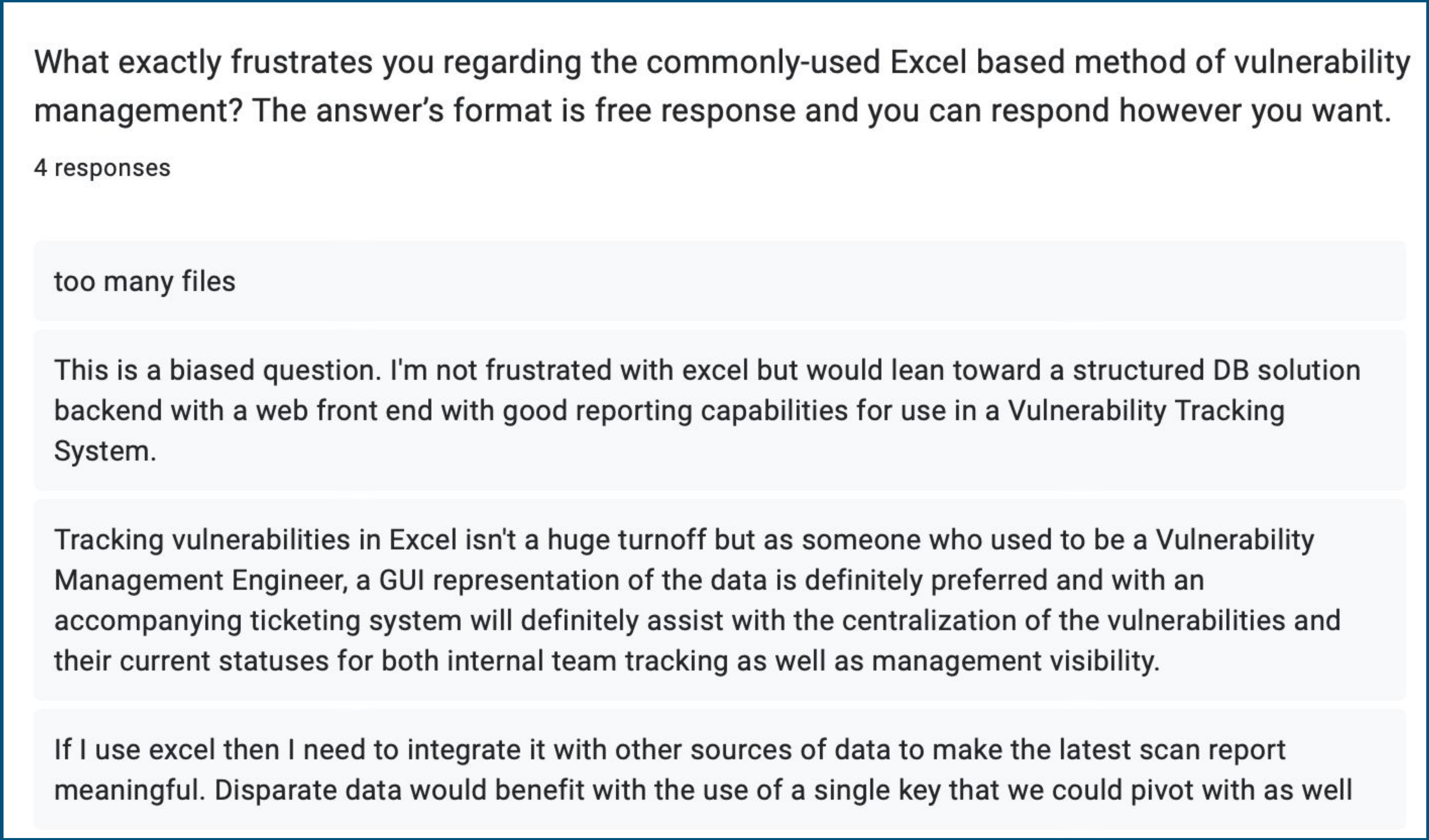
4 responses



On a scale of 1 - 5 (Never to Always), how often do you find yourself frustrated with the commonly-used Excel-based method of vulnerability management?

4 responses





What exactly frustrates you regarding the commonly-used Excel based method of vulnerability management? The answer's format is free response and you can respond however you want.

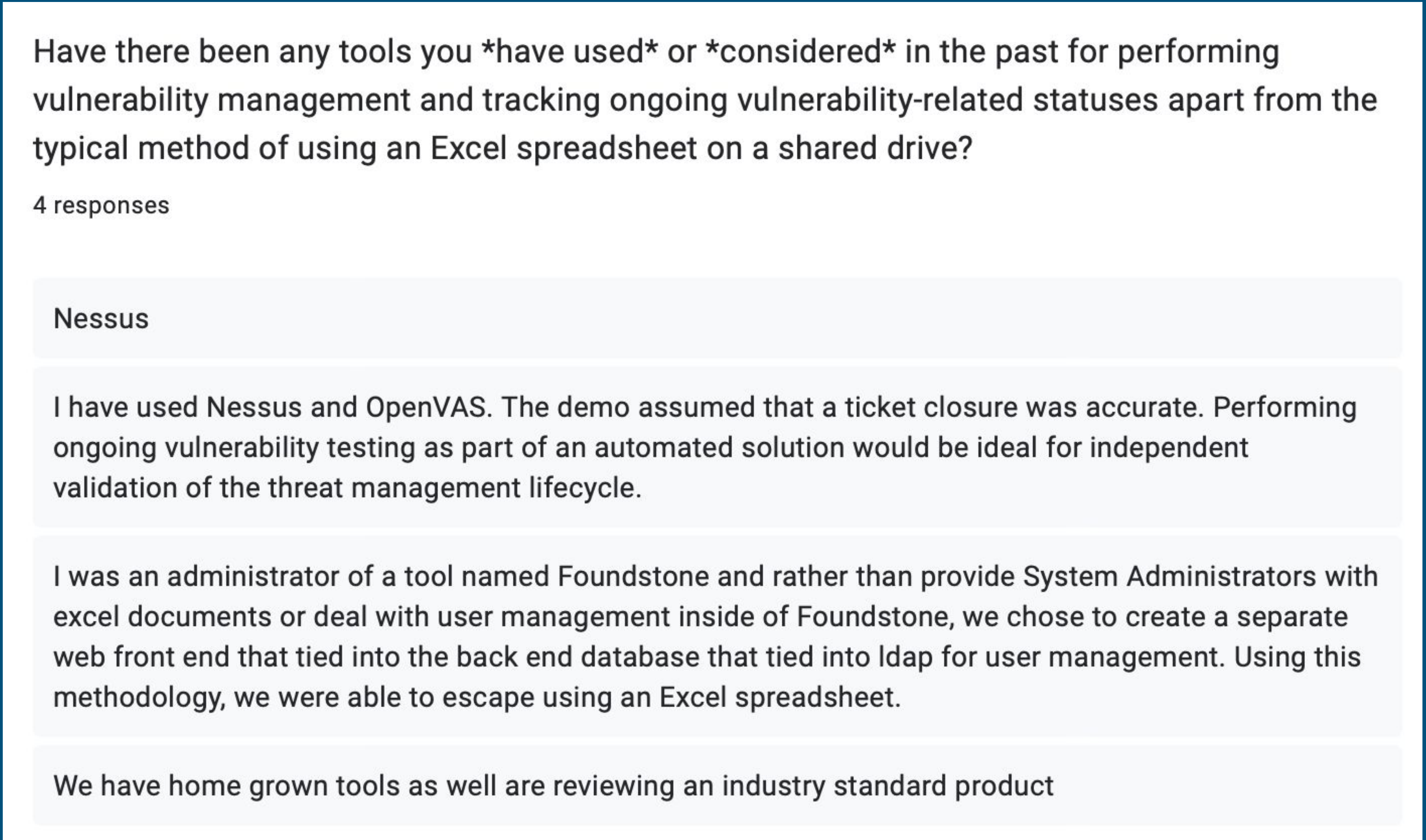
4 responses

too many files

This is a biased question. I'm not frustrated with excel but would lean toward a structured DB solution backend with a web front end with good reporting capabilities for use in a Vulnerability Tracking System.

Tracking vulnerabilities in Excel isn't a huge turnoff but as someone who used to be a Vulnerability Management Engineer, a GUI representation of the data is definitely preferred and with an accompanying ticketing system will definitely assist with the centralization of the vulnerabilities and their current statuses for both internal team tracking as well as management visibility.

If I use excel then I need to integrate it with other sources of data to make the latest scan report meaningful. Disparate data would benefit with the use of a single key that we could pivot with as well



Have there been any tools you *have used* or *considered* in the past for performing vulnerability management and tracking ongoing vulnerability-related statuses apart from the typical method of using an Excel spreadsheet on a shared drive?

4 responses

Nessus

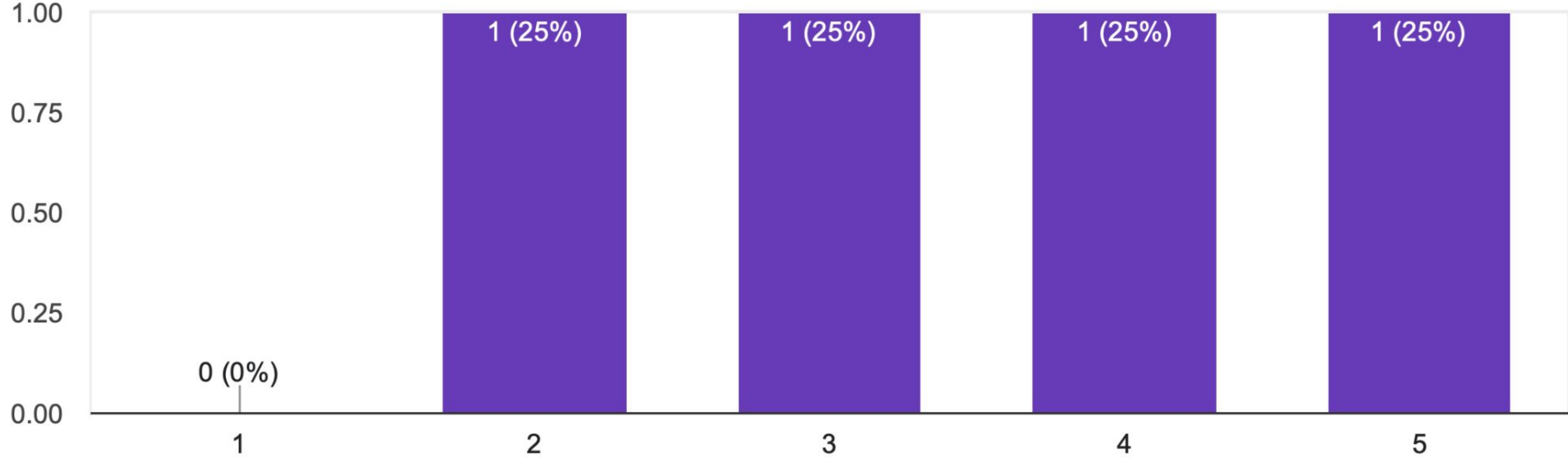
I have used Nessus and OpenVAS. The demo assumed that a ticket closure was accurate. Performing ongoing vulnerability testing as part of an automated solution would be ideal for independent validation of the threat management lifecycle.

I was an administrator of a tool named Foundstone and rather than provide System Administrators with excel documents or deal with user management inside of Foundstone, we chose to create a separate web front end that tied into the back end database that tied into ldap for user management. Using this methodology, we were able to escape using an Excel spreadsheet.

We have home grown tools as well are reviewing an industry standard product

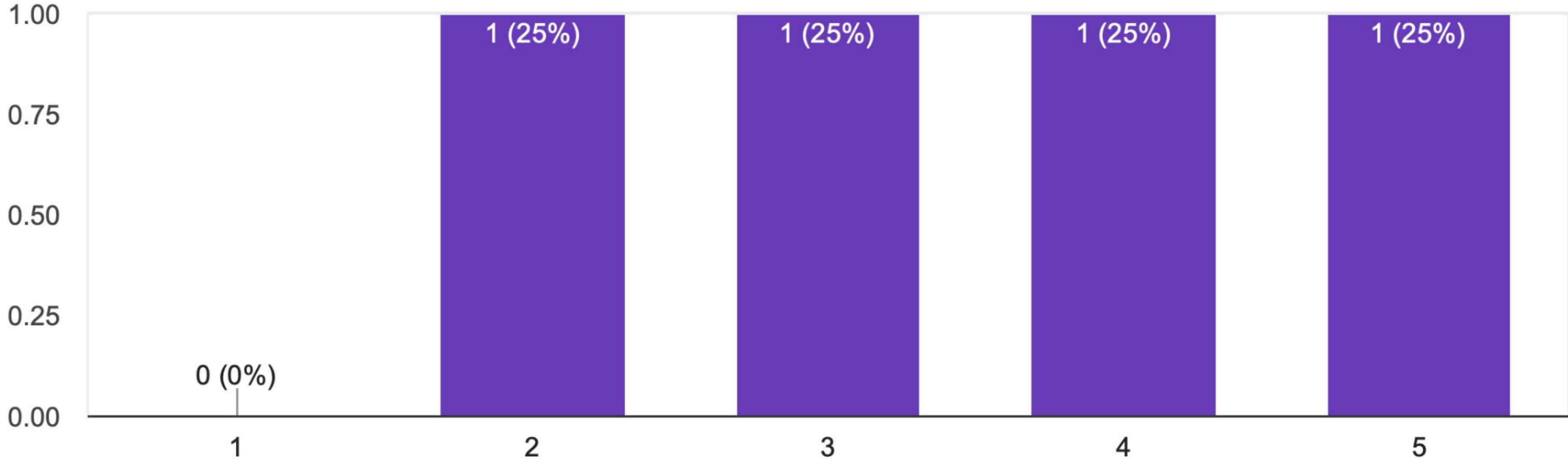
On a scale of 1 - 5, how often do you envision yourself using a tool like VulnTraq? Doesn't have to be VulnTraq specifically

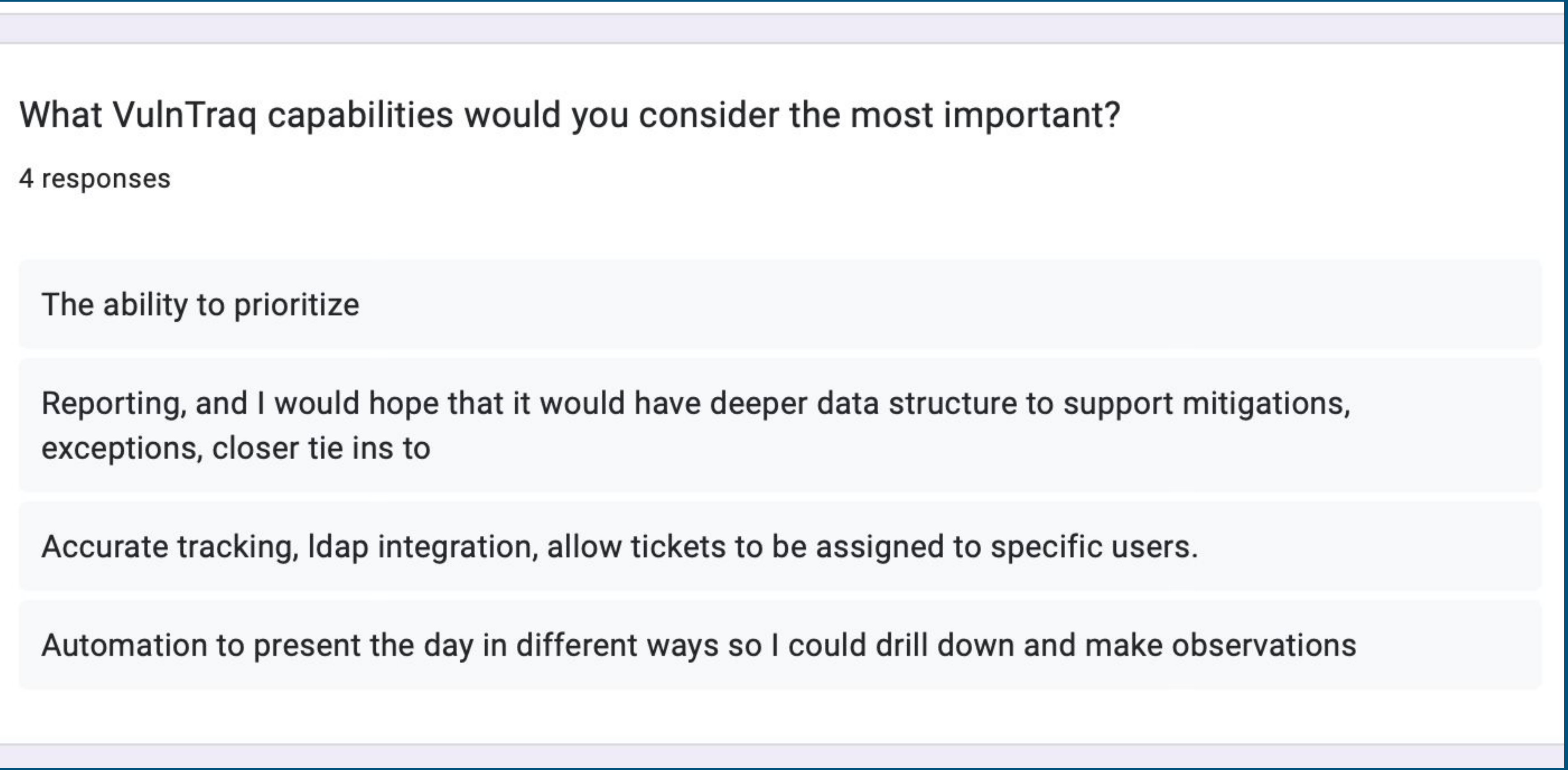
4 responses



How often would you find yourself using VulnTraq when handling vulnerabilities?

4 responses





What VulnTraq capabilities would you consider the most important?

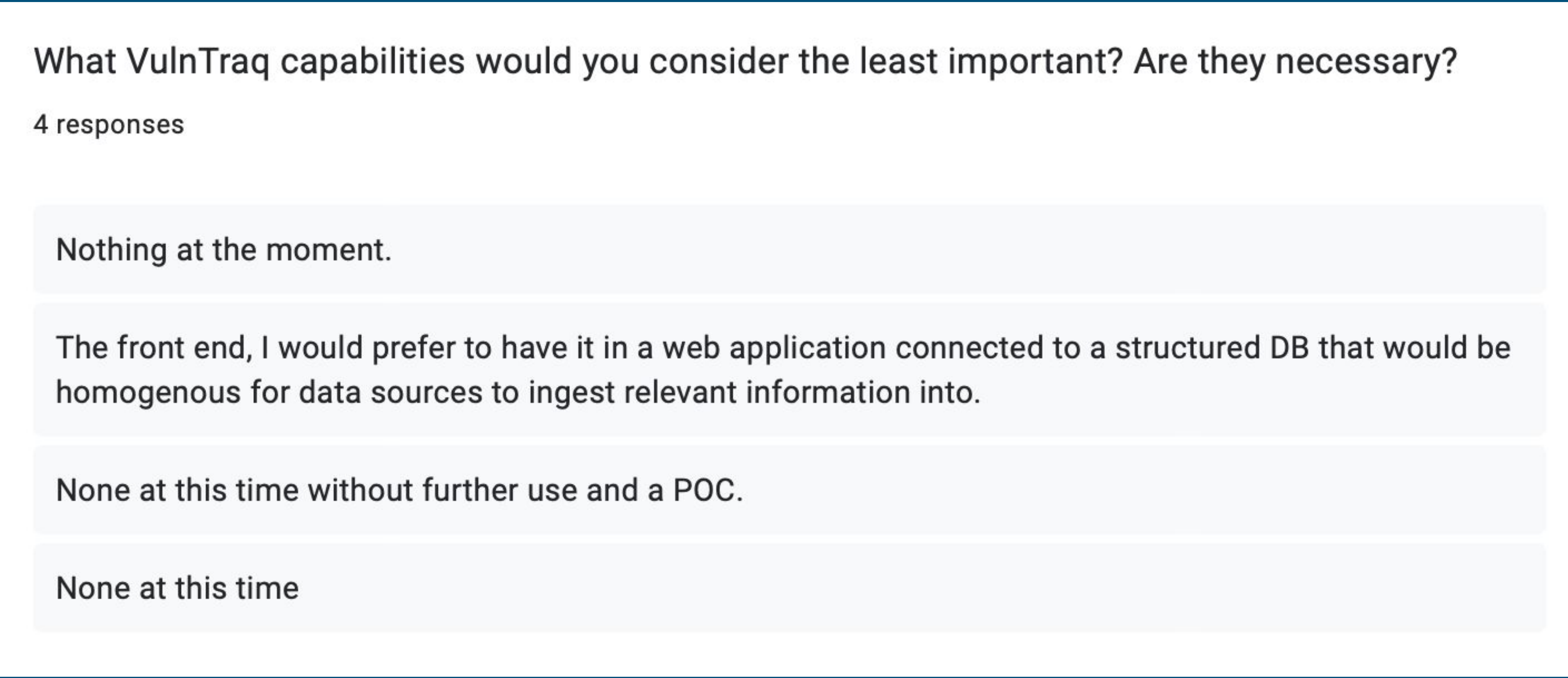
4 responses

The ability to prioritize

Reporting, and I would hope that it would have deeper data structure to support mitigations, exceptions, closer tie ins to

Accurate tracking, ldap integration, allow tickets to be assigned to specific users.

Automation to present the day in different ways so I could drill down and make observations



What VulnTraq capabilities would you consider the least important? Are they necessary?

4 responses

Nothing at the moment.

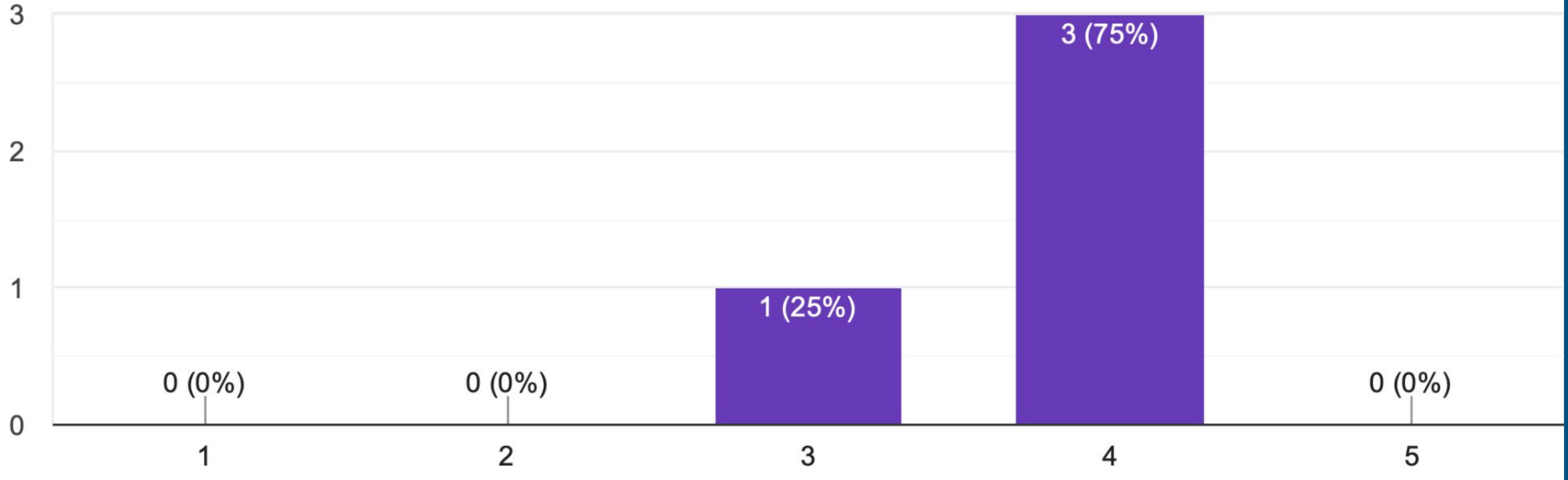
The front end, I would prefer to have it in a web application connected to a structured DB that would be homogenous for data sources to ingest relevant information into.

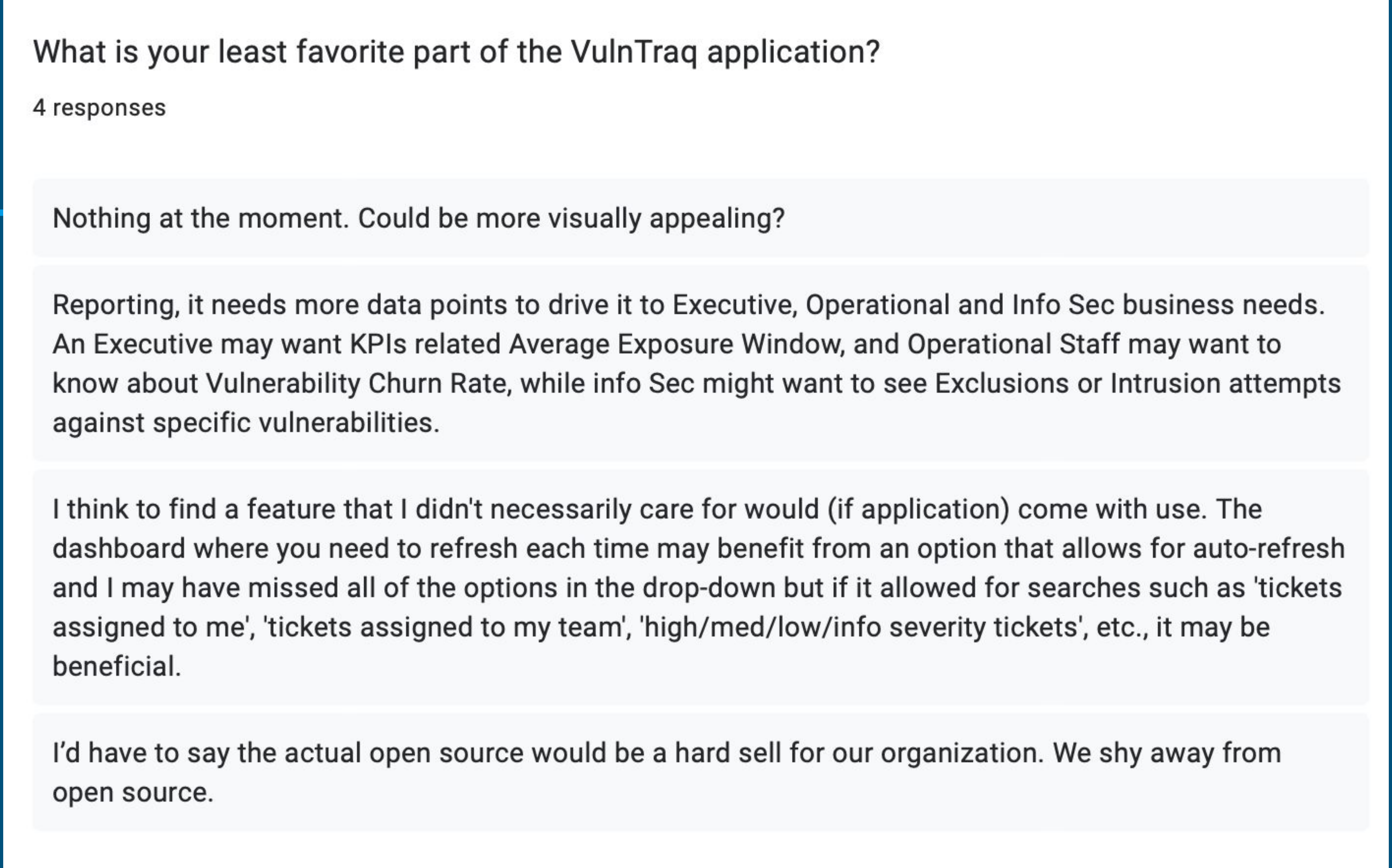
None at this time without further use and a POC.

None at this time

How easy do you find the application's user interface to use?

4 responses





What is your least favorite part of the VulnTraq application?

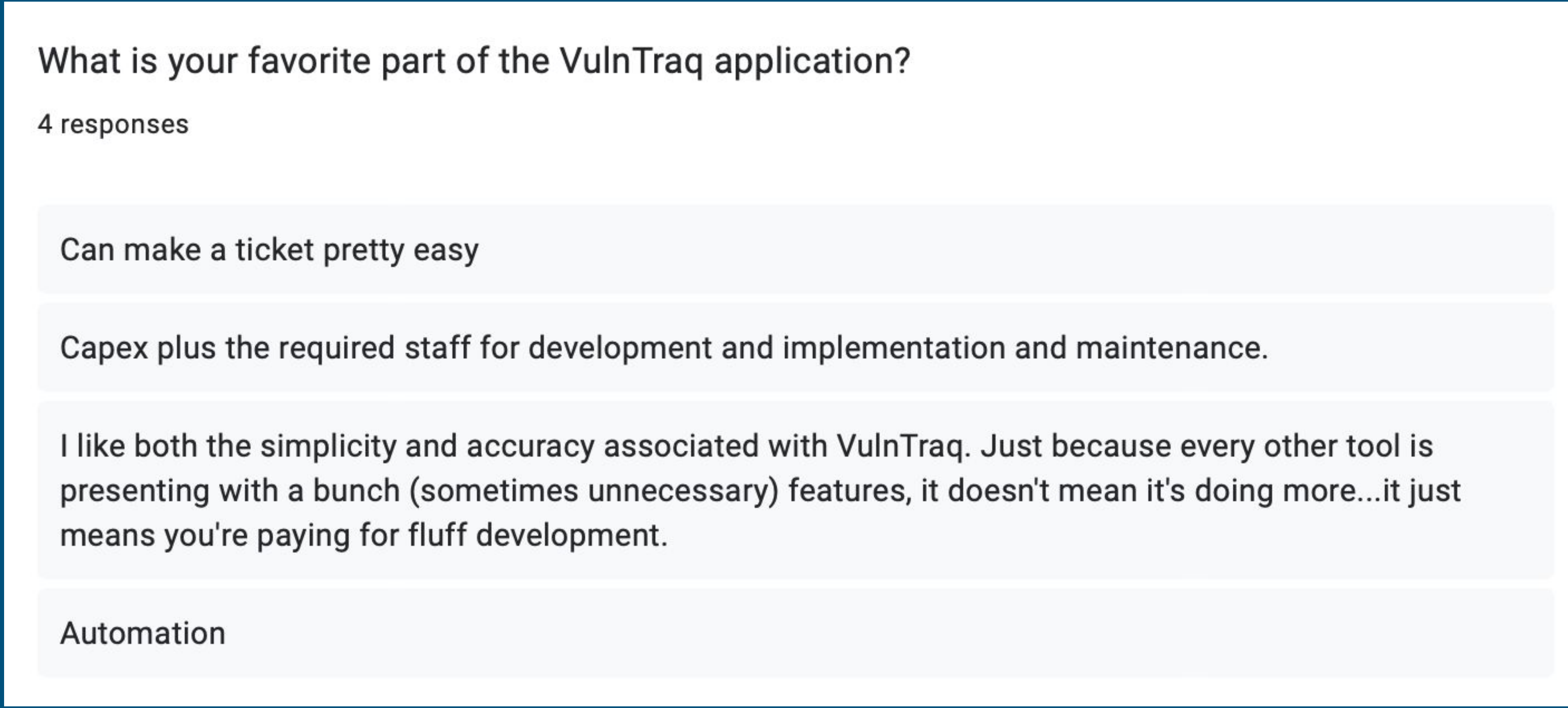
4 responses

Nothing at the moment. Could be more visually appealing?

Reporting, it needs more data points to drive it to Executive, Operational and Info Sec business needs. An Executive may want KPIs related Average Exposure Window, and Operational Staff may want to know about Vulnerability Churn Rate, while info Sec might want to see Exclusions or Intrusion attempts against specific vulnerabilities.

I think to find a feature that I didn't necessarily care for would (if application) come with use. The dashboard where you need to refresh each time may benefit from an option that allows for auto-refresh and I may have missed all of the options in the drop-down but if it allowed for searches such as 'tickets assigned to me', 'tickets assigned to my team', 'high/med/low/info severity tickets', etc., it may be beneficial.

I'd have to say the actual open source would be a hard sell for our organization. We shy away from open source.



What is your favorite part of the VulnTraq application?

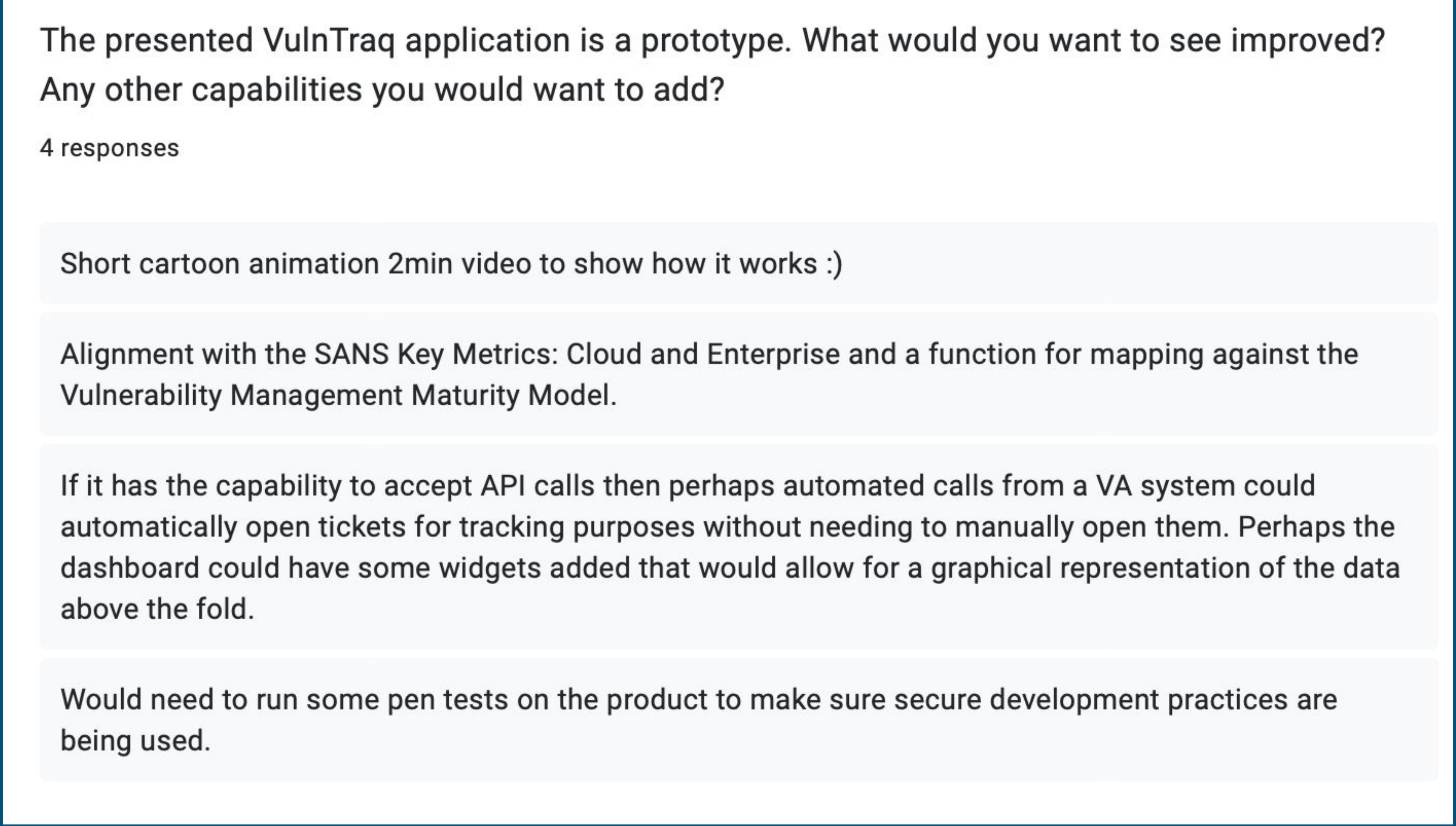
4 responses

Can make a ticket pretty easy

Capex plus the required staff for development and implementation and maintenance.

I like both the simplicity and accuracy associated with VulnTraq. Just because every other tool is presenting with a bunch (sometimes unnecessary) features, it doesn't mean it's doing more...it just means you're paying for fluff development.

Automation



The presented VulnTraq application is a prototype. What would you want to see improved? Any other capabilities you would want to add?

4 responses

Short cartoon animation 2min video to show how it works :)

Alignment with the SANS Key Metrics: Cloud and Enterprise and a function for mapping against the Vulnerability Management Maturity Model.

If it has the capability to accept API calls then perhaps automated calls from a VA system could automatically open tickets for tracking purposes without needing to manually open them. Perhaps the dashboard could have some widgets added that would allow for a graphical representation of the data above the fold.

Would need to run some pen tests on the product to make sure secure development practices are being used.