Windows OS and Top 20 Apps Vulnerability

Scanner Ethical Self-Assessment

User Privacy

It is pertinent that the user's privacy is both protected and respected by myself and my program. The ACM Code of Ethics states that "[t]he responsibility of respecting privacy applies… in a particularly profound way" and that "a computing professional should" be knowledgeable in the different definitions and forms of privacy, as well as understanding of the "rights and responsibilities" that are associated with the use and collection of personal information (Gotterbarn et al., Section 1.6). Similarly, the IEEE Code of Ethics asserts that we should "uphold the highest standards of integrity, responsible behavior, and ethical conduct in professional activities" by "[protecting] the privacy of others" (IEEE Code of Ethics, I.1).

The primary way that user privacy is addressed by my program is by way of the program only collecting, using, and sharing data that is required for the program to function. What this means practically is that:

- The only data collected directly from the user is the user's email address, as it is required in order to email the user the results of their scan. The data generated by the scan is kept until the scan is performed again.
- The user's email is only used by my program to send the user an email containing their vulnerability summary information. Additionally, the only way the program uses the information generated by the vulnerability summary is to attach it in the email to the user.
- The user's email is not shared with anyone other than my program, as the program needs to know the user's email in order to send them an email. Similarly, the user's vulnerability summary is not shared with any other parties or groups.

The manner in which the data is protected, while different from user privacy, is still a closely-related concept. More information on user data security can be found in the 'User Data Security' section below.

User Safety

The importance of user safety is another critical component to consider when designing and creating systems. The IEEE Code of Ethics boldly claims that we must "hold paramount the safety, health, and welfare of the public" in all of our conduct and activities (IEEE Code of Ethics, I.1). Likewise, the ACM Code of Ethics asserts that minimizing "negative consequences of computing, including threats to health [and] safety" should be an "essential aim of computing professionals" (Gotterbarn et al., Section 1.1). Therefore, the safety of the user, both physically and digitally, should be carefully considered when working as a computing professional.

The primary user safety concern for my program is preventing unauthorized access to the information collected, stored, and used by my program and the mitigation of any potential risks associated with the use of my program. The handling of this issue is linked very closely to the handling of privacy concerns discussed in the previous section of this self assessment. In addition to the handling of sensitive information like the user's email and vulnerability information, I needed to consider how the user would interact with the program. To elaborate, I needed to design my program in such a way that the user would not be able to divulge any information that could cause them harm that was not strictly necessary for my program to function. Additionally, I needed to design my program in a manner that reduced the risk of unauthorized outside intervention and provided software reliability.

I addressed these safety concerns by limiting the amount and kind of input that the user would need to provide in order for the program to function. My program consists of three buttons and one textbox. This limitation reduces the number of fields the user needs to provide information for, which results in a reduction of potential points where the user could compromise their safety. My program was designed in such a way that there is very little the user can do to affect the scanning process. This ensures that the program runs reliably, is unlikely to be interrupted, and will not encounter any unexpected errors.

User Accessibility

Accessibility, while not vital to the core functionality of the program, is nonetheless an important aspect of system design. It is imperative that computing professionals "treat all persons fairly and with respect" and do "not engage in discrimination based on characteristics" (IEEE Code of Ethics, II.7). Both technologies and computing practices should be "as inclusive and accessible as possible", and computing professionals should take steps to "avoid creating systems or technologies that disenfranchise or oppress" others (Gotterbarn et al., Section 1.4).

As previously stated, addressing the accessibility of my program was not my primary focus, though it was considered throughout the design and implementation processes. While I attempted to design my program in a manner that would allow as many people to use it as possible, there are some groups of people who may have more trouble than others when trying to use my program. During the design process, I determined that it would be difficult to account for some disabilities and still provide a functioning program. As such, I prioritized creating a working program and gave my best effort to improve my program's accessibility where I could.

I focused my efforts on designing a program that was easy for anyone to understand, whether they were technically-inclined or not. As such, I did my best to create a simple, easy to understand and use GUI, as well as clear and useful error messages. As far as traditional accessibility is concerned, I attempted to design the GUI in a way that would be usable for as many people as possible. I created buttons that were certain colors based on the function the button controlled, and added text on the buttons so that users could see what each button controlled. One improvement on my current design would be to change the colors used for the buttons, as two of the buttons are red and green respectively. The best way to address this change would be to swap out the red and green for colors that can be easily seen by

everyone, including those with any form of colorblindness. This change would be very easy to implement and would not take very long, but would make the program that much more accessible for the users.

User Data Security

Data security for the user is another essential component to consider when designing and creating systems. The issue of user data security is closely related to the privacy of the user, as sensitive information must be kept out of the hands of malicious actors. Any breach of security can cause harm to the user, meaning "robust security should be a primary consideration" when designing and creating a system (Gotterbarn et al., Section 2.9). User data security, then, should be considered heavily throughout the entire process.

The main areas of concern regarding my program and the data security of its users relate to the use and storage of their email addresses and the recording & storage of OS and application version information. Neither the user's email nor the version information is encrypted, meaning that information would be vulnerable if an attacker were to try and access it. While the version information is not particularly confidential, it could still be used in a malicious way, especially if the user's OS or applications were not up-to-date. The user's email, however, could be gathered by an attacker and used in multiple different attacks. This would most likely take the form of a phishing attack sent to the user's email after the attacker has discovered their email. It is worth noting that my program does ***not*** store or collect for any other data outside of the user's email and their vulnerability summary information.

One potential mitigation for these privacy concerns would be implementing a database component to my project. Storing the user's email and vulnerability summary information in a database would allow for the information to be encrypted, which would significantly enhance the privacy aspect form the user's point of view. As I have not yet implemented the database component, I decided to mitigate potential risks to privacy by having the program complete nearly all of its functions on the client's side. The only times the program does something outside of the user's system is when it accesses CVE Details to gather vulnerability information and when it sends the email containing the vulnerability summary to the user. In short, by having the program perform everything it possibly can on the user's system and only accessing anything outside of the user's system when absolutely necessary, I compensated for the lack of encryption on the user's email and vulnerability information. If I were to make only one more change to my program, I would implement and integrate database systems in order to drastically increase data security.

Conclusion

After conducting the self-assessment, it is apparent where improvements can be made. As previously mentioned, user data security is critical in order to protect users from harm and should be a major driving force behind all decisions. I personally believe the implementation of a database system into my program is the single best improvement I could make, in no small part due to the massive increase in

data security that would arise from storing all sensitive information in an encrypted database. It is worth noting, however, that the implementation of a database could lead to other potential attack vectors like SQL injection or vulnerabilities in the database itself. I do believe that the benefits of implementing the database solution would greatly outweigh any potential risks. My program would also benefit from being made more accessible, as discussed with the suggested GUI change. While a relatively minor and simple change, I firmly believe that changing the colors of the buttons to be more accessible is a worthy correction which would promote ethical conduct and compliance.

Other potential improvements include implementing a scheduled deletion of old vulnerability data. This would prevent potentially sensitive information from being stored, which could help prevent a potential attack. This solution would be most easily implemented once my program was working with a database. A potential improvement intended to increase user privacy could be storing the user's email as a salted hash. Doing so would allow the email to be "stored" while making it much harder for any potential attackers to discover what the email truly was. This would require changing the current email storing and retrieval implementation, as the current system simply recalls the user's email from a text file.

The proposed improvement might look something like:

- User's email is stored as a salted hash in the database.
- User starts program and runs scan.
- Scan completes and user is prompted to enter their email.
- User enters their email. If the hash of the user's email matches what is stored in the database, the program continues and emails the user their summary.
- Otherwise, the user is prompted to enter their email again until the salted hashes match.

While there are many other improvements that could be made, I believe these are the ones that would be most impactful from an ethical standpoint.

Works Cited

Board of Directors. “IEEE Code of Ethics.” *IEEE*, Institute of Electrical and Electronics Engineers, June 2020, www.ieee.org/about/corporate/governance/p7-8.html.

Gotterbarn, D., Brinkman, et. al., (2018, June 22). ACM Code of Ethics and Professional Conduct. Retrieved April 12, 2025, from https://www.acm.org/code-of-ethics