Software Engineering & Cloud Computing

Umeå University - Joint Curriculum

Sabine Houy

1 Introduction

My research project focuses on Android root exploits, which means that I am analyzing malicious apps that can gain root access, i.e., almost complete control, of the infected smartphone. My project aims to find ways to facilitate the analysis of such applications to improve detection capabilities. The analysis technique used is called reverse engineering, which is the process of understanding the behavior of programs and applications. This process is complicated and time-consuming. Therefore, I want to automate these steps and thus make the process more efficient. Attackers also use reverse engineering to develop new malware based on existing malicious and benign applications.

2 Software Engineering and my Research

As my research is not related to artificial intelligence and machine learning, I will separately discuss these part. I start with my understanding of the topic in general. The I discuss the opportunities in regards of my research and then try to think of opportunities regarding artificial intelligence and machine learning.

2.1 Automated Software Testing

In automated software testing, the manual process of testing software is automated by applying tools or scripts. These tools automatically perform some of the common steps or automatically execute a set of tests. The tests usually aim to identify unexpected behavior in software before deployment. Since performing software tests manually is quite time-consuming, the automation of it offers great opportunities to companies. They can easily run the tests alongside the development of software and thus save time and resources. However, there will always be a need for manual testing in specific cases and also for manual program analysis.

My research is highly related to security and program analysis. Program analysis can have various purposes, one of them being finding unexpected behavior and bugs in software or programs before deployment. Thus, one could argue that software testing is partly overlapping with the field of program analysis. Automated software testing tools can be used to identify bugs in a program, when combining that with additional information gained through program analysis (static and dynamic), it might be possible to develop an application that cannot just identify bugs but also automatically fix them.

Artificial intelligence and especially machine learning could be used to further improved the idea mentioned above. The features of bugs can be extracted and used to train a machine learning model. Based on these features it will be faster and more efficient to find bugs. These kinds of models already exist [3, 6, 12]. These models could be extended with data collected from static and dynamic program analysis and thus make it possible to fix the detected bugs on the fly.

2.2 Human-Computer Interaction

Human-Computer interaction (HCI) is the study of how humans can interact with user interfaces (computers). One of the first emerges of HCI was the graphical computer interface [10], our desktop which we still use everyday when turning on our computers. Since the first graphical interface, many years past by and the look has change significantly. However, the two main goals of HCI remain the same, providing the best possible functionality and usability [7]. Our systems got more complex and thus the interfaces as well. Nowadays, we can interact with software, chatbots, and in some cases even robots. All these interaction need to be made accessible in an easy-to-understand and easy-to-use manner. Companies often have a team of designers and software engineers, social researchers and business advocates only working on these user interfaces. The better the interface, the easier the interaction, the more like that people or other companies will by your product. Therefore, HCI research forms a curial field of research in industry.

In software engineering, developers have to face a trade-off between usability and security on a regular basis. Usually, the easier to use an interface is, the more insecure it is. It is, therefore, challenging to find the golden mean. From a security point of few the demand for high usability can be quite frustrating as it often seems to be more important to companies. However, the sense of security started to increase over the past years which makes it more curial to find an adequate solution bringing usability and security together. One way of achieving this, is to integrate security aspects into the design process of the software and user interfaces.

User interfaces have changed over time which also results from the change of software we are using. 20 years ago, it was not possible to easily interact with a chatrobot. Most of our parents' generation won't know what that is and will prefer to speak or interact with a real human. Software engineers work constantly on improving the appearance of the chatrobots or smart assistance to make them seem more *human*. In the future, it will become increasingly difficult to distinguish whether you are communicating with a real person or an artificial intelligence.

2.3 Security and Privacy

Software is gaining more and more importance in all aspects in our everyday-lives. In the more recent years, concerns regarding privacy and security in this context became more prominent. Data breaches such as [5] increase the demand of protection for user's (private) data. In order, to provide privacy for user data, a systems needs to be protected against *hackers* which aim to steal data by exploiting vulnerabilities in the system or software. Considering this, privacy and security go hand in hand. Developers need to consider privacy and security concern in the process of software engineering. Potential security and privacy risks need to be identified and fixed before thy can be exploited. Privacy considers usually user data while security is a broader term covering everything from data leakage to actively harming users such as ransomware or banking trojans. The former blocks the user's device until a specific amount of assets is paid (ransom). The latter steals money from users by hacking their online banking application.

There exist numerous research fields concerning privacy and security. My research is highly related to security and privacy, in particular it is in the research field of security and program analysis. I think that the field of privacy and security will keep growing and gain even more importance than it already has as software will become more and more important. Software is already everywhere and this will even increase in the future. By now, we already have a small powerful *computer* everywhere we go with us- our mobile phone. Moreover, more and more people get so-called smart homes, including devices such as smart assistance and smart fridges. Although all these new technologies make our lives more comfortable and reduce our workload, they also pose a significant threat to our privacy and security (e.g., [9]). Therefore, the demand for security and privacy in the field of software engineering will further increase over time.

Machine learning can be seen as a double-edged sword. On the one hand, can machine learning be used to improve the privacy and security in systems such as the approaches mentioned in 2.1. On the other hand, the models need to be trained with data which can cause privacy risks. However, while there is research conducted to make machine learning models more secure and privacy-preserving [2, 11, 1], the interest and benefits of applying it in our everyday lives seems to out-weight the concerns for most people.

3 Future Trends and Directions of Software Engineering

I start with discussing the future trends and directions of Artificial Intelligence and Machine Learning more generally in the context of software engineering. Afterwards, I discuss the future trends in the field of cybersecurity and more particular of program analysis.

In general, I do not expect machine learning to wipe out software engineering, as there will continue to be many areas where machine learning cannot be used, or at least not exclusively. From my perspective, machine learning and software engineering are two overlapping or related fields. machine learning can be integrated into software engineering and thus programs but will never "eat software". Machine learning is only a small part of software, which is growing but will never reach 100%. Machine learning can be used as one of many elements or building blocks in a program or project. These projects usually deal only with data processing. For example, to analyze the online behavior of users to display better or more personalized advertising. Once the data is analyzed, the realization still has to be done with the help of software engineering. The use of machine learning only makes sense in combination with data analysis, but software engineering deals with so much more than just data and its analysis. Every program we use on our computers, every app on our phones, and every ATM we withdraw money from was created through software engineering. This kind of software engineering usually has nothing to do with machine learning.

From my point of view, machine learning can be seen as a subcategory of software engineering or as a separate field with overlaps. Machine learning as a subcategory or area of its own can then be used as modules in programs, i.e., software engineering. As data becomes more important, the importance and use of machine learning will also increase. However, the same is true for software engineering as technology becomes more central to our lives and we can no longer live without it. I expect that both fields will continue to grow and have a certain connection, but software engineering will remain the stronger field, as it deals with much more than machine learning and has more applications that machine learning can never replace.

Considering the usage of machine learning in cybersecurity, it is usually used as some kind of anomaly detection and pattern recognition to identify malware. There exist multiple approaches using machine learning to detect malware such as [13], [4], and [14]. Even if most of the results are promising, there are some issues [8] that lead to the fact that most of the approaches are not really applicable in the real world. In my opinion, one of the main challenges is the ever-changing nature of malware. The Internet is flooded with malware of all kinds and new ones are emerging every day. Since technology has existed, there has also been the cat and mouse game between hackers and developers. Since the behavior of malware is constantly and rapidly changing, it is difficult to create a reliable training dataset and extract the right features. This challenge often leads to the fact that the presented machine learning models perform very well for a certain dataset but can hardly do anything in reality. If it is possible to solve this problem in the future, malware detection will take a giant leap forward. But even if this is possible, software engineering will always be a significant part and cannot be replaced by machine learning, such as developing new systems and closing their vulnerabilities.

References

[1] Ghulam Abbas, Amjad Mehmood, Maple Carsten, Gregory Epiphaniou, and Jaime Lloret. Safety, security and privacy in machine learning based internet of things. *Journal of Sensor and Actuator Networks*, 11(3):38, 2022.

[2] Mohammad Al-Rubaie and J Morris Chang. Privacy-preserving machine learning: Threats and solutions. *IEEE Security & Privacy*, 17(2):49–58, 2019.

[3] Saiqa Aleem, Luiz Fernando Capretz, and Faheem Ahmed. Benchmarking machine learning technologies for software defect detection. *arXiv preprint arXiv:1506.07563*, 2015.

[4] Dragoş Gavriluţ, Mihai Cimpoeşu, Dan Anton, and Liviu Ciortuz. Malware detection using machine learning. In *2009 International Multiconference on Computer Science and Information Technology*, pages 735–741. IEEE, 2009.

[5] Insider Aaron Holmes. Stolen Data of 533 Million Facebook Users Leaked Online, 2021.

[6] S Delphine Immaculate, M Farida Begam, and M Floramary. Software bug prediction using supervised machine learning algorithms. In *2019 International conference on data science and communication (IconDSC)*, pages 1–7. IEEE, 2019.

[7] Fakhreddine Karray, Milad Alemzadeh, Jamil Abou Saleh, and Mo Nours Arab. Human-computer interaction: Overview on state of the art. *International journal on smart sensing and intelligent systems*, 1(1):137, 2008.

[8] Kaijun Liu, Shengwei Xu, Guoai Xu, Miao Zhang, Dawei Sun, and Haifeng Liu. A review of android malware detection approaches based on machine learning. *IEEE Access*, 8:124579–124607, 2020.

[9] The New York Times | Wirecutter Grant Clauser. Amazon's Alexa Never Stops Listening to You | Wirecutter, 2019.

[10] Brad A Myers. A brief history of human-computer interaction technology. *interactions*, 5(2):44–54, 1998.

[11] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael Wellman. Towards the science of security and privacy in machine learning. *arXiv preprint arXiv:1611.03814*, 2016.

[12] Michael Pradel and Koushik Sen. Deepbugs: A learning approach to name-based bug detection. *Proceedings of the ACM on Programming Languages*, 2(OOPSLA):1–25, 2018.

[13] Justin Sahs and Latifur Khan. A machine learning approach to android malware detection. In *2012 European Intelligence and Security Informatics Conference*, pages 141–147. IEEE, 2012.

[14] Igor Santos, Jaime Devesa, Felix Brezo, Javier Nieves, and Pablo Garcia Bringas. Opem: A static-dynamic approach for machine-learning-based malware detection. In *International joint conference CISIS'12-ICEUTE´ 12-SOCO´ 12 special sessions*, pages 271–280. Springer, 2013.