Predictive Markets: A Software Security Application

Jonathan Oakley, Kushal Hebbar, Nathan Tusing, Carl Worley
Holcombe Department of Electrical and Computer Engineering
Clemson University
Clemson, USA
{joakley,khebbar,ntusing,cworley}g.clemson.edu

***Abstract*—Accountability is the biggest challenge facing software security. We propose a predictive market based on Augur's framework that will facilitate software security. This will introduce a financial disincentive for developers to produce secure code. By leveraging Augur's existing framework, we can automate the creation of software markets where users will decide whether or not they believe a specific software version is secure. This metric will allow users to easily evaluate software security, as well as affect a company's stock price.**

***Index Terms*—Predictive Markets, Augur, Software Security**

I. Introduction

Accountability is the biggest challenge facing software security. Financial accountability ensures there are fiscal repercussions for negligent security vulnerabilities. One of the most important factors affecting software security is how well the software is written. Well written software is clean, efficient, and uses techniques that reduce the overall complexity of the code–effectively reducing the possible attack surface. Unfortunately, most software companies lack appropriate incentives to produce well-written code. If a company produces software with security vulnerabilities, they simply release a new version later and before the software's vulnerability are publicly revealed and fixed. The companies often make enough money to offset the cost of the vulnerability, which actually provides a financial disincentive towards developing secure software.

II. Background

Security is a vital aspect of all aspects of technology, whether it be application security, software security, network security, or cyber security. Technology today has advanced immensely. With every emerging technology introduced, our life gets easier. However, along with these technological advancements come attacks such as identity thefts, frauds, Man-in-the-middle (MitM) attacks, and Denial-of-service (DoS) attacks. These technological advancements are highly dependent on software–every business relies on the internet and computer network for operations. Therefore, software security must be given top priority.

Software security is the concept of building software so it functions and produces accurate results under malicious attack. As software developers are becoming increasingly aware of software vulnerabilities, they are adapting and evolving a set of best practices to address attacks on security. Some of the best practices include building abuse cases, collecting security requirements from clients, performing risk analysis, external review, and carrying out static analysis [1].

Another aspect of software security that assists in protecting a system against attacks are software security standards such as OWASP (Open Web Application Security Project), which provides a structure for standards and best practices that are a vital aspect of software development [2]. OWASP has developed the SAMM (Software Assurance Maturity Model) [3] to assist such security measures. NIST (National Institute of Standards and Technology) [4] is another security standard that provides a common language and definition structure for software development. Another such standard is the CWE (Common Weakness Enumeration) which provides a common language to define threats [5]. There is a constant increase in security threats as the rate of software developments increase. These software security standards act as a pillar to develop secure and productive applications.

A. Predictive Markets

Prediction markets are markets where the payout depends on the outcome of a future event. The lottery is a simple example of a prediction market. The lottery ticket is a *share* in the market. If someone has the same numbers, then the payout is split two ways. The market closes when the numbers are read. In more complex markets, the relative prices of the various shares, in relation to the payout of those shares, represents the market's prediction of the probability of each outcome. Because the efficient market hypothesis predicts that a market will incorporate all available information, a prediction market will reflect the sum knowledge of its participants. Empirical studies have shown prediction markets to be at least as accurate as a panel of experts–for instance, prediction markets have often outperformed national polls regarding presidential elections [6].

III. Architecture

The predictive market security framework leverages three main pieces of technology: Ethereum, Augur, a frontend marketplace. Augur's trading platform is built on Ethereum's smart contract framework, and Augur's APIs can be integrated with a marketplace websites to provide users with information about current trends and updated information.

A. Ethereum

Ethereum is a cryptocurrency that extends traditional blockchain technology with smart contract applications [7]. Smart contracts are essentially programs that are run on the blockchain in a special environment. Smart contracts guarantee the output of code through the same consensus that secures the blockchain.

B. Augur and Smart Contract Templates

Augur [8] builds on this secure execution environment by creating libraries of Ethereum smart contracts that support token trading. Augur allows users to create *markets* for an *event*. When Alice creates a market on Augur, she puts Ether into two escrow accounts that will be returned if certain conditions are met. Alice also specifies the market duration and the types of tokens that will be traded, such as: event will happen, and event won't happen. When Bob believes the event will happen, he buys the initial *event will happen* tokens from the contract (the market created by Alice). Similarly, when other users buy the initial *event won't happen* tokens, they also buy them from the contract. This creates a fund held by the contract that will be paid out to the users whose tokens reflect reality. If the event doesn't happen, the fund is paid to all users who have *event will not happen* tokens.

Augur also supports trading tokens. So, Bob may not truly believe the event will happen, but he may think that others will believe the event will happen, and therefore, he may buy tokens and try to sell them for a higher price.

The final issue Augur addresses is handling the payout after a market closes. Any market that is defined so that the terms of resolution are not perfectly unambiguous can, in the reporting phase, be marked "INVALID". In this case, no outcome is marked as "true" and the value of the shares is distributed equally among all shareholders. Since a controversial reporting of outcome can cause the Augur platform to fork into competing universes, and since the worst case scenario involves REP tokens (a token that indicates the reliability of a reporter) being migrated into multiple child universes, so that each universe has positive value less than the value of the parent universe, the REP holders are strongly incentivized to avoid forks at all costs. This likely means that INVALID outcomes occur extremely easily, so as to avoid possible controversy. Any market that may possible be INVALID will be ignored by traders, who are as likely to lose money as gain it.

We will automate the creation of Augur markets around whether or not a particular version of the code are secure. These markets will stay open for a number of weeks before the resolution period. Market value of the tokens is expected to be affected by the available source code and past performance. After the resolution period, during which companies will examine the submitted vulnerabilities, responsible disclosure will permit users to release the vulnerabilities they found (if any), and will allow the designated reporter to make a final decision regarding the security of the code during that particular period. After the payout, a new market can be opened for the same version of the software, or a newer version if vulnerabilities were fixed.

C. Marketplace

A frontend marketplace will provide users with easy access to all available software markets. User and developer profiles will also be linked to software to prevent parasitic markets (such as developers betting against their own software). The frontend will also link the markets with all available source code, documentation, and resources relevant to the market. News listings will also be available to provide users with the latest information.

IV. Experimental Setup

We will set up an Ethereum testnet on GENI. This will allow us to have geographically isolated nodes as well as the full functionality of Ethereum. We will deploy Augur on this testnet and integrate our tools with this version of Augur. Our deliverables will consist of a tool set and a marketplace. The tool set will automate the market creation process, and the marketplace will provide users with the ability to speculate on the security of a specific software version. Time permitting, we will also develop naive trading bots that will attempt to emulate users trading on limited information. This will be purely for demonstration purposes, as these bots will simply drive the price up and down based on the information they're exposed to. Since the experiments will be conducted on the GENI testbed, no additional hardware is required, and all software is provided under public license.

V. Work Plan

Research on this project is currently being conducted for the Kaspersky Hackathon, so additional work is also being completed. The work conducted by the team members in CPSC 8580 is limited to the Augur tool set, and the major milestones are listed below. Both team members will be working on these tasks, so some duplication of effort is expected.

- 10/16: Working predictive market for software security
- 10/26: Automated predictive market
- 11/1: Trading bots

VI. Conclusion

We are proposing a predictive market framework that will facilitate software security. There is not financial disincentive for developers to produce secure code. Leveraging Augur's existing framework and Ethereum, we will automate the creation of software markets, where users can decide whether or not they believe a specific software version is secure. We expect this *hive mind* approach to security will provide users with an accurate metric when evaluating software security, as well as open another feedback channel that will affect stock prices and board meetings.

REFERENCES

[1] G. McGraw, "Software security," *IEEE Security & Privacy*, vol. 2, no. 2, pp. 80–83, 2004.

[2] "The OWASP Foundation," https://www.owasp.org/index.php/Main_Page.

[3] Defense Security Cooperation Agency, "Security Assistance Management Manual," http://www.samm.dsca.mil/.

[4] U.S. Department of Commerce, "National Institute of Standards and Technology," https://www.nist.gov/.

[5] "Security standards in software development," https://www.kiuwan.com/blog/security-standards-in-software-development/.

[6] R. Forsythe, F. Nelson, G. Neumann, and J. Wright, "The 1992 iowa political stock market: September forecasts," *The Political Methodologist*, vol. 5, no. 2, pp. 15–19, 1994.

[7] V. Buterin *et al.*, "Ethereum white paper, 2014," *URL https://github.com/ethereum/wiki/wiki/White-Paper*, 2013.

[8] C. J. Lee, "Whitepaper v2. 0 human consensus prediction."