Smart Contract Development in Practice: Trends, Issues, and Discussions on Stack Overflow

Afiya Ayman, Amna Aziz, Amin Alipour, Aron Laszka
Department of Computer Science
University of Houston

***Abstract*—Blockchain based platforms are emerging as a transformative technology that can provide reliability, integrity, and auditability without trusted entities. One of the key features of these platforms is the trustworthy decentralized execution of general-purpose computation in the form of *smart contracts*, which are envisioned to have a wide range of applications from finance to the Internet of Things. As a result, a rapidly growing and active community of smart contract developers has emerged in recent years. A number of research efforts have investigated the technological challenges that smart contract developers face. However, very little is known about the community itself, about the developers, and about the issues that they discuss and care about.**

To address this gap, we study the online community of smart contract developers on Stack Overflow. We provide insight into the topics that they discuss, their technological and demographic background, and their awareness of security issues and tools. Our results show that the community of smart contract developers is very active and growing rapidly, in comparison with the general user population. However, a large fraction of smart contract related questions remain unanswered, which can pose a real threat to the viability of a sustainable community and may indicate gaps in community knowledge. Further, we observe very limited discussion of security related topics, which is concerning since smart contracts in practice are plagued by security issues.

***Index Terms*—Smart Contract, Blockchain, Stack Overflow, Software Development, Data Analysis, Security,**

1. Introduction

The popularity and adoption of blockchain based platforms are growing rapidly both in academia and industry. This growth is driven by the unique features of blockchains: providing reliability, integrity, and auditability in a decentralized system. While earlier blockchains, such as Bitcoin, used these features to establish cryptocurrencies, more recent blockchains, such as Ethereum, also function as distributed computational platforms [1], [2]. These platforms enable developers to deploy general-purpose computational code in the form of smart contracts, which can then be executed by a decentralized but trustworthy system. Smart contracts are envisioned to have a range of innovative applications, such as privacy-preserving transactive energy systems [3], [4], asset tracking in the Internet of Things [5], and various financial applications [6]. Unfortunately, due to the peculiarities of smart contract platforms and languages, the development of smart contracts has proven to be a challenging and error-prone process [7]. These errors often manifest as security vulnerabilities [8], which have led to multiple notable security incidents, with losses in the range of hundreds of millions of dollars worth of cryptocurrencies [9], [10]. As a response, the research community has stepped forward and introduced a number of security tools [11], [12], [13], [8], frameworks [14], [15], [16], and even new languages [17] to help smart contract developers.

However, prior studies have primarily focused only on the effectiveness, usability, etc. of these tools and frameworks, but not on whether developers actually use them [18]. Some studies have investigated security issues and vulnerabilities in smart contracts that are deployed in practice, and found that many of these vulnerabilities could have been avoided or found using the appropriate, existing tools and techniques [12]. However, they have not considered whether developers are aware of security risks and whether they are familiar with existing security tools and techniques.

In light of this, there is a clear gap in research regarding developers' perception of smart contract issues and technologies. Further, very little is known about the developers and other stakeholders of the smart contract development community. Such information is crucial for enabling researchers to better understand the potential entry barriers for smart contract technology and for guiding researchers to address the developers' needs.

In this paper, we set out to provide insights about the adoption and perception of smart contracts by the software developer community through analysis of Stack Overflow discussions. Stack Overflow[1] is an online forum where registered users discuss issues related to coding and software development. We chose Stack Overflow because it has been ranked the best active forum for programmers and software developers by surveys [19], [20]. The unique popularity of Stack Overflow in the software developer community makes it an ideal candidate for evaluating and analyzing the overall trends, issues, and adoption of smart contract technologies among developers. We seek to provide an analysis

1. www.stackoverflow.com

of smart contract discussions, characteristics of developers who participate in these discussions, and issues that these developers are most concerned about. Our ultimate goal is to better understand the needs and concerns of existing and potential developers.

Research Questions: In this study, we aim to answer the following research questions.

- *RQ1:* What are the main trends in smart contract related discussions on Stack Overflow?
- *RQ2:* What are the characteristics of typical smart contract developers?
- *RQ3:* Do developers discuss smart contract security issues and tools on Stack Overflow?

The results of our study suggest an upward trend in the number of new posts on smart contract related issues, which confirms the increasing popularity of this technology. However, a large portion (more than a quarter) of smart contract posts are left unanswered, which is much higher than for more mature technologies, such as Python. This may pose a serious challenge for the growth and sustainability of the community of smart contract developers.

Our analysis also finds that smart contract developers tend to have either very recent or very old Stack Overflow profiles, which suggests that smart contract discussions attract both experienced users as well as new ones. This is good news for the smart contract community since it seems to be able grow while also engaging experienced developers. Further, smart contract developers tend to have higher reputation scores than other Stack Overflow users, which means that the community is particularly active.

Finally, our analysis of topics in smart contract related posts shows that questions cover a wide spectrum, from languages and frameworks to deployment and maintenance. The most discussed technologies are Ethereum, Solidity, web3.js, and Truffle. However, we find very few mentions of security issues and tools.

Contributions: We make the following contributions:

- We study smart contract related questions and answers on Stack Overflow, identifying trends and providing topic analysis results.
- We study the demographics and background of developers who participate in smart contract related discussions, and we compare them with other users on Stack Overflow.
- We study whether developers discuss smart contract security issues and tools.
- We make the data and source code publicly available to encourage future research.[2]

The remainder of this paper is organized as follows. In Section 2, we describe our dataset and methodology. In Section 3, we present and discuss our findings. In Section 4, we give a brief overview of related work. In Section 5, we discuss and address threats to the validity of our results. Finally, in Section 6, we provide concluding remarks.

2. We omit the link to the data and source code for anonymity. We will add the link at the time of publication.

2. Study Design

In this section, we explain the design of our study in detail. First, we describe our research data and collection method. Then, we present our research methodology and the main metrics that we use to answer our research questions.

2.1. Data Collection

Stack Overflow is an online forum that allows users to ask and answer technology-related questions. A Stack Overflow post includes a question, a set of associated tags, and a set of answers. Every Stack Overflow user can post new questions or answer existing ones. To facilitate searching and categorizing posts, Stack Overflow requires users to associate one or more tags with their questions. These unstructured tags are defined and chosen by the users, and they include a wide range of terms (e.g., *Python*, *linked-list*, *email*). Stack Overflow also has a reputation based reward system, which encourages users to participate in discussions and contribute to the community by rewarding them for positive actions, such as answering a question.

We collect all posts and users from the quarterly archive of Stack Overflow data hosted on the Internet Archive [21], which we complement with data from the Stack Exchange Data Explorer [22] to acquire the most recent posts and users. To find posts that are related to smart contracts, we use a snowballing methodology. Initially, we collect all posts with the tag *smartcontract*. We then extract other tags from the collected posts, extend our search by including the most frequent tags that are smart contract related, and collect an extended set of posts. We continue this process until we find no new tags that are strictly related to smart contracts. Using this snowballing method, we collect all posts whose tags contain the following strings (except for *ether*, which needs to be an exact match to avoid matching *ethernet* and other similar terms): *smartcontract*, *solidity*, *ether*, *ethereum*, *truffle*, *web3*, *etherscan*. [3]

2.2. Research Methodology

Figure 1 depicts the workflow of our study. First, we perform standard statistical analysis on the posts that we collected from Stack Overflow. We consider the tag distribution of these posts to find the tags that are most frequently mentioned in smart contract posts. These tags help us to identify the most popular topics (frameworks, IDE, libraries, tools, etc.) among smart contract developers. However, tags do not capture the topic of the posts' text thoroughly. Thus, it is necessary to look at the actual textual content of the posts to discover the main discussion topics.

To discover the main topics of discussion in smart contract posts, we use topic modeling. Topic modeling is an information retrieval technique that uses word frequencies

3. We manually check a random sample of these posts to confirm that they are all related to smart contracts.

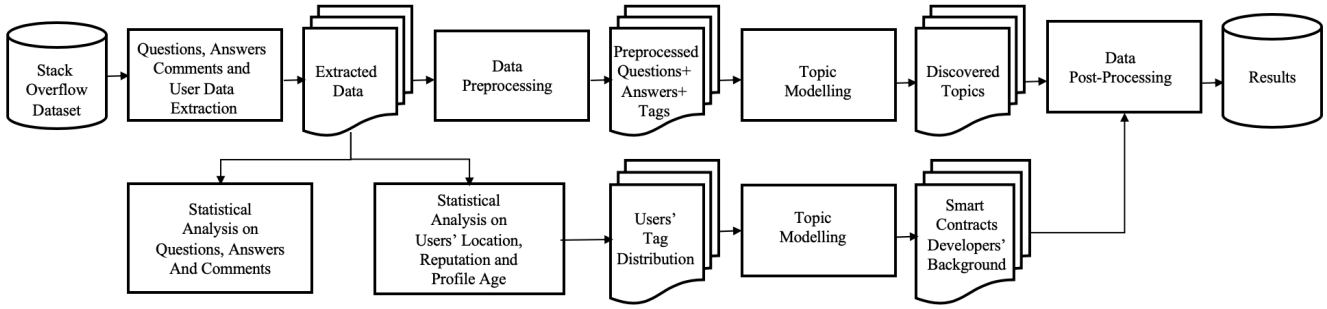

Figure 1. Workflow of this study

and co-occurrence frequencies in documents to identify patterns of related words. In our study, we use Latent Dirichlet Allocation [23] (LDA) for topic modeling.

2.2.1. Statistical Analysis. The first step of our workflow studies statistics of smart contract discussions from Stack Overflow to understand the rate of growth in the popularity of smart contracts. We consider the number of posts related to smart contracts found on Stack Overflow and the ratio of answered and unanswered questions. We also apply demographic analysis on the users' information, such as their reputation scores and the creation dates of their Stack Overflow profiles.

2.2.2. Data Preprocessing. The second step consists of data preprocessing to prepare the posts for topic modeling. Preprocessing involves three steps:

1) cleaning HTML tags and code snippets from the text body of posts;
2) applying stemming algorithm to reduce words to their base forms;
3) removing common English language stop words from the list of stems.

Title: Creating Ethereum Contracts (go ethereum)

Trying to follow the wiki example for go ethereum to create a basic contract: https://github.com/ethereum/go-ethereum/wiki/Contracts-and-Transactions

Everything seems to work until I get down until the last line:

<pre><code>source = "contract test function multiply(uint a) returns(uint d) return a * 7; " contract = eth.compile.solidity(source).test primaryAddress = eth.accounts[0]

**Problems start here ** MyContract = eth.contract(abi); contact = MyContract.new(arg1, arg2, ...,from: primaryAddress, data: evmCode) </code></pre>

What is the "abi" argument for the eth.contract method? Also, what would I put in the "evmCode" argument? In this particular example, seems like I would put in an integer for "arg1" but not sure what the full example should look like.

Figure 2. An example post from Stack Overflow, which discusses smart contract development.

Most Stack Overflow posts contain code snippets, HTML tags, and common stop words. Figure 2 shows an example post from Stack Overflow before preprocessing. Before applying topic modeling to the textual content of the posts, it is essential to first remove the HTML tags and code snippets. After removing the HTML tags and code snippets, we join the title and body of each post together. The reason for considering both the title and body is that for many posts, key information is included only in either the title or in the body. Figure 3 shows the example post after cleaning the HTML tags and code snippets.

Creating Ethereum Contracts (go ethereum) Trying to follow the wiki example for go ethereum to create a basic contract: https://github.com/ethereum/go-ethereum/wiki/Contracts-and-Transactions Everything seems to work until I get down until the last line: What is the "abi" argument for the eth.contract method? Also, what would I put in the "evmCode" argument? In this particular example, seems like I would put in an integer for "arg1" but not sure what the full example should look like.

Figure 3. Example post after cleaning HTML tags and code snippets, and joining the title and text body.

Next, we apply a stemming algorithm. Stemming is the process of reducing derived words to their base forms. The process does not always produce complete words. For example, the Porter stemming algorithm reduces the words 'argue', 'argued', 'argues', 'arguing', and 'argus' to the stem 'argu'. It does not check if a word has a meaning before or after stemming. We try three different stemming algorithms, Porter stemming, Snowball stemming, and Lancaster stemming, and observe the differences between their results. The first two algorithms deliver almost identical results. For example, the word *'language'* reduces to *'languag'* with Porter stemming and Snowball stemming algorithms, and to *'langu'* with Lancaster stemming. Since Porter stemming has been widely adopted as the standard approach for word conflation in information retrieval, we use the results of Porter stemming. Figure 4 shows the example post after applying the Porter stemming algorithm.

Finally, we remove the common stop words of the English language, such as articles, auxiliary verbs, and pronouns, which do not contribute to creating significant topics. We use the standard set of stop words from the Natural Language Toolkit (NLTK) library for this purpose. Figure 5 shows the example post after cleaning the stop words and punctuation marks from the list of stems.

creat ethereum contract (go ethereum) tri to follow the wiki exampl for go ethereum to creat a basic contract : http : //github.com/ethereum/go-ethereum/wiki/contracts-and-transact everyth seem to work until I get down until the last line : what is the " abi " argument for the eth.contract method ? also , what would I put in the " evmcod " argument ? In thi particular exampl , seem like I would put in an integ for " arg1 " but not sure what the full exampl should look like

Figure 4. Example post after applying the Porter stemming algorithm to each word.

creat ethereum contract go ethereum tri follow wiki exampl go ethereum creat basic contract http everyth seem work get last line abi argument method also would put evmcod argument particular exampl seem like would put integ sure full exampl look like

Figure 5. Example post after removing stop words from the list of stems.

2.2.3. Topic Analysis. Topic modeling is an approach for making sense of large volumes of text, which is a commonly used in text mining to identify hidden semantic structures in textual content. In this paper, we use the Latent Dirichlet Allocation (LDA) algorithm for topic analysis.

LDA is a form of unsupervised learning, which considers each document to be a mixture of different topics. LDA represents topics as probability distributions over the words of a text corpus. When LDA discovers a set of words that are frequently used together by documents, it creates a topic. Usually, the words of a topic are semantically related, which results in a meaningful topic. For example, Figure 6 shows a topic found by applying topic analysis to words used in smart contracts posts, treating the posts as LDA documents.

Topic: Words: 0.321*"solidity" + 0.145*"smartcontracts" + 0.124*"web3" + 0.093*"truffle" + 0.086*"web3js" + 0.026*"metamask" + 0.024*"remix" + 0.022*"erc2" + 0.014*"token" + 0.012*"contract"

Figure 6. An example topic discovered by LDA, with the presence probabilities of the most related tags.

2.3. Metrics

In our study, we use a number of quantitative metrics as well as qualitative ones.

2.3.1. Statistical Metrics for Posts. The statistical metrics are mostly quantitative. After aggregating data from Stack Overflow, we use a few standard metrics to gain insight into the community's understanding of the topic. We consider metrics such as

- number of smart contract related posts over time,
- distribution of the number of answers to each post,
- frequencies of common tags associated with the posts.

2.3.2. Statistical Metrics for Users. For user demographics, we use a set of quantitative metrics, such as

- reputation distribution of smart contract developers,
- age distribution of their Stack Overflow profiles,
- ratio of smart contract developers from each country.

We compare the smart contract developers' demographics with general Stack Overflow users.

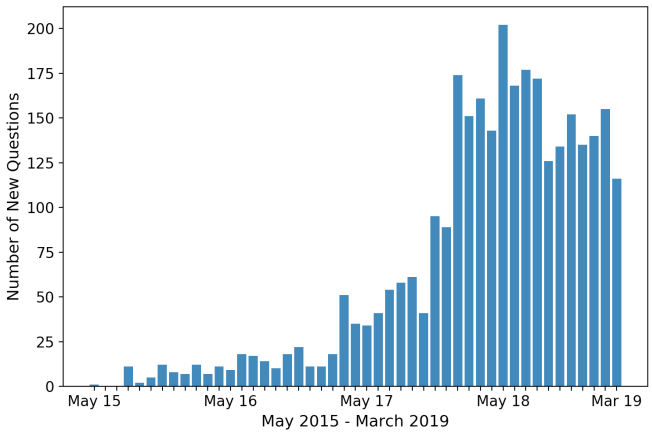

Figure 7. Number of new smart contract posts per month on Stack Overflow.

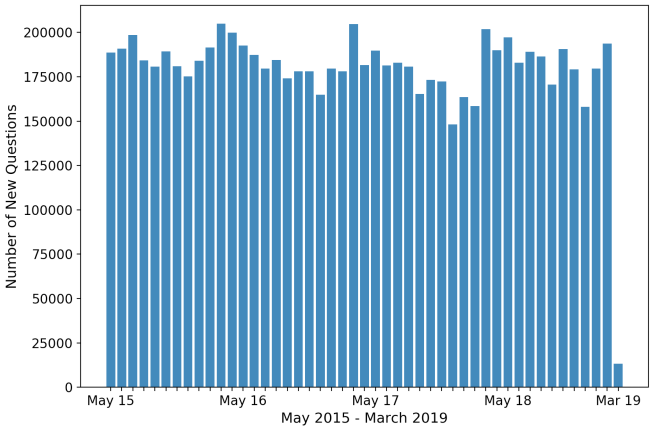

Figure 8. Number of new posts per month on Stack Overflow.

2.3.3. Metrics for Topic Analysis. LDA topic modeling discovers a number of topics and represents each topic as a collection of words. Each word is assigned a probability that indicates its contribution to forming the topic. The metrics for LDA are generally qualitative.

3. Results and Discussion

This section summarizes the results of our study. We first discuss trends in Stack Overflow posts related to smart contracts. We then discuss the results of topic modeling. Next, we discuss the the demographics and background of smart contract developers. Finally, we discuss the prevalence of security related topics in smart contract posts.

3.1. Trends in Smart Contract Related Posts

Our first question is: *What are the main trends in smart contract discussions?* To answer this question, we study multiple metrics of smart contract related posts.

TABLE 1. MOST FREQUENT TAGS IN SMART CONTRACT POSTS

Tag	Occurrences	Definition
Ethereum	2224	Open-source, blockchain-based distributed computing platform.
Solidity	1200	Contract-oriented, high-level language.
Blockchain	766	Digital, public ledger that records online transactions.
Smartcontracts	562	Self-executing contracts with the terms of the agreement being directly written into lines of code.
Web3/Web3js	378	JavaScript framework for interacting with Ethereum blockchain.
Truffle	381	Development environment, testing framework and asset pipeline for Ethereum.
JavaScript	320	Prototype-based, multi-paradigm, dynamic language.
Node.js	247	Open-source, cross-platform JavaScript run-time environment.
Go-Ethereum	211	Go (programming language) library supported by Ethereum.
Geth	148	Command line interface for running a full Ethereum node implemented in Go.
Remix	107	Powerful, open source tool that helps writing Solidity contracts from the browser.
Metamask	86	Extension for accessing Ethereum enabled distributed applications from the browser.
ReactJS	83	A declarative, efficient, and flexible JavaScript library.
Python	82	An interpreted, high-level, general-purpose programming language.
ERC20	76	Technical standard used for smart contracts on the Ethereum blockchain for implementing tokens.

3.1.1. Number of Posts over Time. As of March 2019, there are a total of 3089 posts related to smart contracts. In particular, there are 39, 156, 588, and 1895 posts in years 2015, 2016, 2017, and 2018, respectively. There is no mention of smart contracts in Stack Overflow posts prior to May 2015. Figure 7 provides a fine-grained view of the number of new smart contract posts, showing the number of new posts in each month from May 2015. There is a clear upward trend in the frequency of posts from 2015 to 2018, with the vast majority of posts created in 2018. However, for 2019, we measure an average of only 137 posts per month (based on our dataset, which contains posts up to March 22). Extrapolating, we expect the total number of smart contract post in 2019 to be around 1644, which is lower than in 2018.

Next, we compare the trend of smart contract related posts with the overall trend of posts on Stack Overflow in the same time period, from May 2015 to March 2019. Figure 8 shows the number of new posts on Stack Overflow per month. We see a mostly stable but slightly decreasing trend, which is in strong contrast with the upward trend in the number of smart contract related posts.

Observation 1: While the overall rate of new posts on Stack Overflow is mostly stable, the rate of new smart contract posts has increased rapidly from 2015 to 2018.

3.1.2. Number of Answers per Post. We also analyze the number of answers per smart contract post. We observe that there are 2884 answers for 3089 posts, which means an average of 0.93 answers per post. Figure 9 shows the distribution of the number of answers for each smart contract post. A significant fraction of smart contract questions (28%) have no answers at all. Further, 55.4% majority of the questions have only one answer. Three or more answers have been posted for only 3.9% of the smart contract questions.

The fraction of posts that contain questions without answers seems unusually high, and it suggests that the user base still has a lot to learn. The average number of responses per post indicates that discussion is prevalent in the community, but the clustering of 3 or more answers to a minority of posts suggest that the average posts do not induce much discussion.

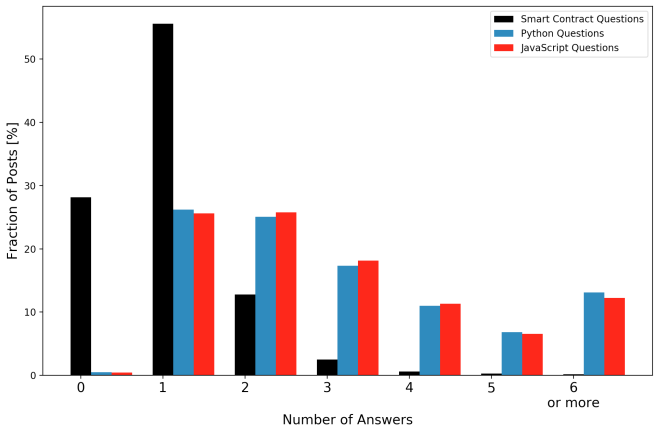

Figure 9. Number of answers posted to smart contract, Python, and JavaScript related questions.

We compare the average number of answers in smart contracts posts to two popular topics, *Python* and *JavaScript*. Python has an average of 3.2 answers per post with the highest number of answers to a single post being 191. The highest number of answers to a single JavaScript post is 99 with an average of 3.3 answers per post.

Observation 2: A significant percentage of smart contract related questions (28%) have no answers at all.

3.1.3. Tag Frequencies from Posts. Next, we look at the tags that developers most frequently used in their smart contracts posts. Although tags do not necessarily represent the main topics of discussion in these posts, they provide important insights about the scope and the context of the questions. For example, when we see the *Ethereum* tag in a post, we know that the issue is related to the Ethereum platform. We find that a total of 727 distinct tags have been used in smart contract related posts. Table 1 shows the 15

TABLE 2. 10 SMART CONTRACT TOPICS FOUND BY LDA

Topic Name	Top Words in Topic
Solidity events	event, function, return, promise, callback, value, call, async, use, get
Contract errors	error, contract, code, using, truffle, file, web3, im, version, get
Transactions	contract, transaction, token, address, smart, gas, function, send, ether, code
Blockchain basics	blockchain, block, ethereum, data, would, use, like, transaction, user, node
Hyperledger	hyperledger, asset, fabric, wave, script, channel, smart, participant, quorum, platform
Installing with Node.js	err, npm, node, gyp, block, poa, step, rebuild, aws
Solidity functions	function, solidity, array, contract, string, variable, return, value, im, like
Web-based DApps	proxy, heroku, source, nginx, nethereum, bit, document, atom, program, aptget
Private blockchain networks	private, account, ethereum, node, network, geth, key, using, transaction, wallet
Platforms and environments	docker, visual, studio, container, extension, port, image, remixide, bloc, ec2

most frequent tags with the number of times that they occur in smart contract posts. We also provide a brief description of the meaning of these tags. The tags include a wide range of core blockchain platforms, languages, tools, and frameworks, such *Solidity*, *geth*, and *Truffle*. Interestingly, we also see tags that are not specific to smart contracts, such as *Python* and *Node.js*. These are languages and environments that developers often use to integrate with smart contracts.

Observation 3: The most discussed technologies include *Ethereum*, *Solidity*, *web3.js*, and *Truffle*.

3.2. Topic Analysis on Smart Contract Posts

Next, we study what common issues and difficulties programmers face while developing smart contracts. To this end, we perform topic analysis on the smart contract related Stack Overflow posts. We represent each post as a concatenation of the title, the question, all the answers, and all the tags assigned to the post. Then, we apply the Latent Dirichlet Allocation (LDA) to the set of posts.

For the number of topics in LDA, we experiment with integer values from 5 to 15, and we find that 10 yields the best result. Table 2 summarizes the results of the topic analysis, showing the most relevant words for each topic. Topics range from smart contract concepts, such as Solidity events, to testing and deployment related topics, such as setting up a private blockchain network.

Observation 4: Discussions span a broad range of topics from development to testing and deployment.

3.3. Smart Contract Developer Demographics

Our second research question asks: *Who are the typical smart contract developers?* To answer this, we collect all the Stack Overflow users who posted questions or answers related to smart contracts, and we study the characteristics of these users and compare them with the general user base.

3.3.1. Age of User Profile. Stack Overflow started its journey in 2008, and as of April 5th, 2019, it has 10,232,569 registered users [22]. Of these Stack Overflow users, there are a total of 2210 users who posted questions or answers related to smart contracts. First, we look at the creation dates

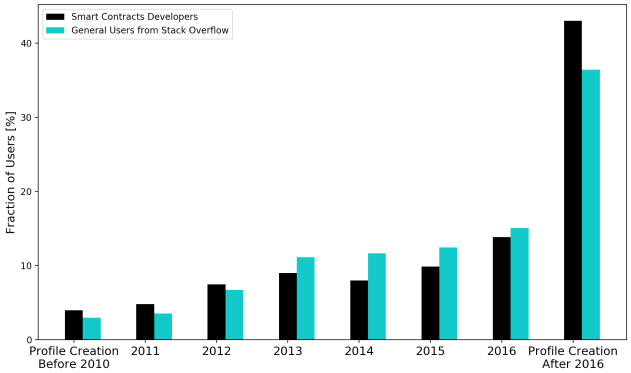

Figure 10. Profile creation dates of smart contract developers and other Stack Overflow users.

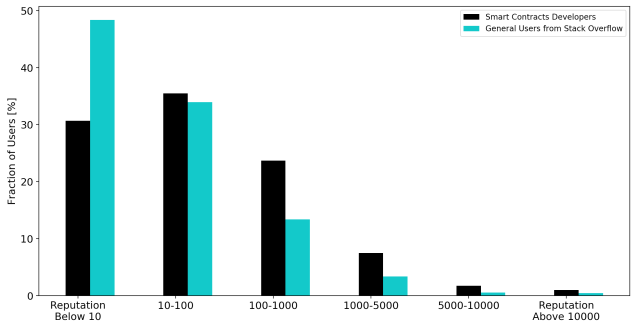

Figure 11. Reputation of smart contract developers and other Stack Overflow users.

of these smart contract developers' user profiles. Figure 10 shows the distribution of profile creation dates for smart contract developers, comparing them to other Stack Overflow users. We see that compared to other users, a larger fraction (43%) of the smart contract developers' profiles are new (i.e., created after the year 2016). However, we also see that compared to other users, smart contract developers also include an unusually high number of users with very old profiles.

3.3.2. Reputation of Users. Stack Overflow users receive reputation scores for various activities, such as answering a

TABLE 3. TOPIC ANALYSIS ON SMART CONTRACT DEVELOPERS' BACKGROUND

Background Topic	Top Tags for Topics
C/C++	c, c++, multithreading, linux, pointers, uml, arrays, sockets, algorithm, windows
Web development	html, php, jquery, css, ajax, json, html5, mysql, regex, arrays
Database	sql, google-bigquery, mysql, sql-server, database, sql-server-2008, oracle, postgresql, google-cloud-platform, lua
Unix development	bash, shell, docker, linux, awk, sed, unix, regex, docker-compose, grep
Python development	python, python-3.x, python-2.7, django, numpy, list, dictionary, pandas, tensorflow, pip
Google mobile platform	firebase, firebase-realtime-database, android, firebase-authentication, ios, google-cloud-firestore, swift, google-cloud-functions, firebase-cloud-messaging, angularjs
Android development	android, realm, rust, android-studio, android-layout, kotlin, actionscript-3, android-fragments, android-activity, eclipse
Web applications	ruby, ruby-on-rails, go, haskell, ruby-on-rails-3, scala, activerecord, ruby-on-rails-4, rspec, functional-programming
Decentralized applications	blockchain, google-app-engine, solidity,hyperledger-fabric, hyperledger, hyperledger-composer, corda, smartcontracts, web3, truffle
Version control and CI	git, github, eclipse, docker, version-control, go, gitlab, windows, ssh, jenkins
Cloud computing	azure, arrays, regex, powershell, erlang, algorithm, string, vba, json, azure-virtual-machine
Front-end development	reactjs, express, angularjs, angular, jquery, typescript, mongodb, promise, socket.io, react-native
iOS development	ios, multithreading, objective-c, swift, arrays, performance, iphone, string, xcode,jvm
.NET development	c#, asp.net, .net, asp.net-mvc, unit-testing, asp.net-web-api, asp.net-core, entity-framework, linq, vb.net
Other	scala, meteor, spring, maven, amazon-web-services, mongodb, spring-boot, grails, hibernate, eclipse

question. We study the users' reputation because it indicates how engaged and active smart contract developers are on Stack Overflow. Figure 11 compares the distribution of smart contract developers' reputation with that of other Stack Overflow users.[4] The figure clearly shows that smart contract developers tend to have higher reputation scores than other Stack Overflow users. This is surprising considering that many smart contract developers joined Stack Overflow recently (see Figure 10).

Observation 5: Smart contract developers are more active in discussions than other Stack Overflow users.

3.3.3. Location of Users. Users on Stack Overflow may also specify their real-world location. Among the 2210 smart contract developers, only 1026 had information related to their location. We used the HERE Maps API[5] to translate the locations into complete addresses, and then we categorized the users according to their countries (Figure 12). We found that India and the United States have the largest number of users with 229 and 171 users, respectively. The distribution is very similar to that of other Stack Overflow users, where again India and the United States have the largest number of users.

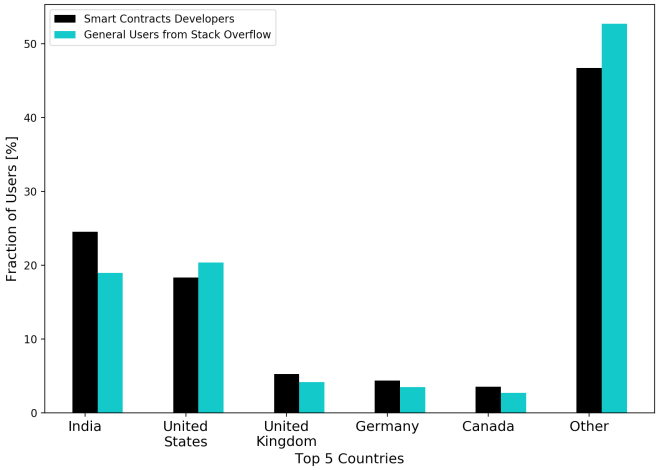

Figure 12. Locations of smart contract developers compared to other Stack Overflow users.

3.4. Background of Smart Contracts Developers

To discover the background and skills of smart contract developers, we study what other topics they usually discuss on Stack Overflow. First, we retrieve all the questions and answers posted by every one of these users, collecting all the tags mentioned in those posts for each user. Then, we run an LDA topic analysis for the smart contract users and their tags. Table 3 shows the topics found by the analysis. These encompass a wide spectrum of topics in software development, with a focus on web and Unix/Linux based platforms.

3.5. Vulnerabilities and Security Tools

Our third research question asks: *Are smart contract developers concerned about security?* To answer this question, we first study papers on security issues and security tools for smart contracts. Then, we search to see if Stack Overflow users mention these issues or tools when discussing smart contracts.

3.5.1. Vulnerability Issues Mentioned. Based on prior work on smart contract security [12], [7], we first compile a list of common vulnerability types: *re-entrancy*, *timestamp dependence*, *transaction-ordering dependence*, and *race condition*. Then, we search for mentions of these issues in the cleaned text body of smart contract posts. We consider

4. Note that for the sake of fair comparison, we only consider other users who have posted at least one question or answer on Stack Overflow. If we consider all other users, the difference compared smart contract developers is even more significant.

5. HERE Maps API: https://developer.here.com/products/maps

TABLE 4. MENTIONS OF COMMON SMART CONTRACT VULNERABILITY TYPES

Security Issue	Number of Posts
Timestamp Dependence	14
Re-Entrancy	2
Transaction-Ordering Dependence	1
Race Condition	0

all the answers along with all the tags, and we consider a wide range of variations and stems of the above terms (e.g., "dependence" and "dependency"). Nonetheless, we find very few posts in our dataset—only 17—that mention these common vulnerability types. Table 4 shows the breakdown of security-related posts.

3.5.2. Security Tools Mentioned. Similarly, we study whether developers discuss any security tools for smart contracts. We search for mentions of the following security tools and frameworks: `Oyente` [12], `Mythril` [24], `Securify` [13], `ContractLarva` [25], `EthIR` [26], `MAIAN` [8], `Vandal` [27], `Rattle` [28], `FSolidM` [14], `VeriSolid` [15]. We find only one mention of `Securify` and one mention of `MAIAN`. For most of these tools, the lack of discussions can be explained by the fact that they were published very recently. However, there are some older tools (e.g., `Oyente` was introduced in 2016), of which the developer community could be aware.

Observation 6: Very few posts on Stack Overflow discuss security-related topics in smart contracts.

4. Related Work

We discuss three areas of related work: surveys of development practices and discussions (Section 4.1); reviews of smart contract languages, tools, and security issues (Section 4.2); and studies of smart contract development challenges from an educational perspective (Section 4.3).

4.1. Smart Contract and Blockchain Related Discussions

Bartoletti et al. [29] examined smart contract platforms, applications, and design patterns. The study aggregated articles about smart contracts from coindesk.com and identified nine common design patterns used in some combination by most of the smart contracts found. It was the first quantitative investigation on the usage of design patterns and categories of contract. It provided a categorized and tabulated repository of data related to smart contracts similar to what this paper seeks to accomplish.

Atzei et al. [7] presented an analysis on security vulnerabilities based both on the growing academic literature on Ethereum smart contracts, participation to Internet blogs and discussion forums about Ethereum, and practical experience on programming smart contracts. The causes of vulnerabilities are organized in a taxonomy, whose purpose is to know and avoid common pitfalls and to foster the development of analysis and verification techniques for smart contracts.

Wohrer et al. [30] examined design patterns for smart contracts in Ethereum. They began by focusing on two questions; which design patterns appear in the Ethereum ecosystem and how do these design patterns map to Solidity coding practices. The authors employed a multivocal literature review which consisted of academic sources related to Ethereum and Solidity patterns, official Solidity development documentation and smart contract best practices; internet blogs and forums about Ethereum. They also included Ethereum conference talks, and GitHub repositories related to smart contract coding patterns related to Solidity. The analysis of the pool yielded five categories of design pattern based on their operational scope i.e. Action and Control, Authorization, Lifecycle, Maintenance, Security. A total of eighteen relevant design patterns were uncovered along with examples of published Solidity source code for each pattern from the Ethereum mainnet. The paper provided a robust repository of Ethereum design patterns for the benefit of future Ethereum developers. Each of these papers provided context and relevant techniques that were used in its own research methodology. However, the authors didn't include smart contracts in their study.

Jiang et al. [31] examined blockchain technology as interpreted by developers in their study. The paper structured its analysis about three questions that informed the metrics of the study; how frequently were blockchain questions asked on Stack Overflow; what were common developer problems with blockchains and what blockchain entities existed. The paper extracted posts from Stack Overflow, filtered out code snippets and hyperlink text, and examined blockchain posts for each month since they first appeared. The article answered its three questions as follows. Blockchain questions represent a growing minority of posts on Stack Overflow. The most common problems with blockchain are related to configuration, deployment, and discussion followed by ten less common categories. There are 45 different blockchain entities mentioned each of which belongs to three different categories (Hyperledger, Ethereum, Bitcoin). However, the analysis presented in that paper is very shallow and does not focus on smart contracts.

Barua et al. [32] covered general developer discussion on Stack Overflow in their study. The study separated gathered posts into Q/A and employed a latent Dirichlet allocation to isolate topics and applied a short descriptor to each topic for readability. Finally, it applied a membership percentage for each topic in each post to determine each topic's total share of posts, examine common topic pairing, and track topic relevance over time. The paper provided data on relationships between different topics and trends of topic discussion over time.

4.2. Smart Contract Development, Security, Tools

Parizi et al. [33], conducted an empirical analysis of smart contract programming languages based on usability and security from the new developers' point of view. They

considered three programming languages for their study, i.e. Solidity, Pact7, and Liquidity9. The study concluded that although Solidity is the most useful language to a new developer, it is also the most vulnerable to malicious attacks as new developers tend to leave behind security vulnerabilities which can leave the contracts insecure.

Most recently in another study, Parizi et al. [18] carried out an assessment of current static smart contracts security testing tools, for the most widely used blockchain, the Ethereum and its domain-specific programming language, Solidity, to provide the first body of knowledge for creating more secure blockchain-based software. The study tested tools on ten real-world smart contracts from both vulnerability effectiveness and accuracy of true detection viewpoints. The results showed that SmartCheck tool is statistically more effective than the other automated security testing tools. However, the study includes only the effectiveness, usability, etc. of the tools, but not whether developers use them.

Luu et al. [12] investigated the security of running smart contracts based on Ethereum in an open distributed network like those of cryptocurrencies. The study introduced several new security problems in which an adversary can manipulate smart contract execution to gain profit. These bugs suggest complex gaps in the perception of the distributed semantics of the underlying platform. The study also proposed ways to enhance the operational semantics of Ethereum to make contracts less vulnerable and provided a symbolic execution tool, Oyente which analyzes Ethereum smart contracts to detect bugs.

4.3. Smart Contract Development from Educational Perspective

Delmolino et al. [34] documented their experiences in teaching smart contract programming to undergraduate students. Their study exposed numerous common pitfalls in designing safe and secure smart contracts. They documented several typical classes of mistakes students made, suggested ways to fix/avoid them, and advocated best practices for programming smart contracts. even a simple self-construct contract (e.g., "Rock, Paper, Scissors") can contain several logic problems, such as contracts do not refund, lack of cryptography to achieve fairness i.e. malicious users can submit inputs biased in their favour and contracts do not incentivize users to follow intended behaviour.

Angelo et al. [35] described the importance to teach smart contract development to the graduate level computer science students. Since the development of smart contracts combines areas such as; distributed systems, security, data structures, software engineering, algorithms, etc., teaching this particular topic bears much significance. The paper discusses their approach to teaching the development of secure smart contracts on the Ethereum platform. Developing decentralized applications (DAPPs) is a demanding task and presents substantial complexities that result, e.g., from the concurrency, transparency, and immutability of transactions. Their experience also shows the difficulties of the task since the underlying technologies emerge rapidly and documentation lags behind. They also pointed out that the available tools for smart contracts construction are in different stages of development, and even the most mature ones posses challenges to the users.

5. Threats to Validity

Finally, we discuss threats to the validity of our findings. First, we included only Stack Overflow posts in this study. While Stack Overflow is the most popular forum for most development related discussions, it is possible that other online forums are more popular in the smart contract community. Nonetheless, given the sheer size of Stack Overflow, we believe that our study provides reliable and unbiased findings about smart contract developers and the issues that they face. Second, although in our snowball methodology, we tried to include as many relevant Stack Overflow posts in the analysis as possible, we may have missed some of the Stack Overflow posts. However, based on manual inspection, we believe that the number of missed posts is negligible compared to the number of included posts, and our primary goal was to avoid including unrelated posts.

6. Conclusions

This paper provided insights into the community of smart contract developers by analyzing trends, topics, and developers on Stack Overflow. We studied the topics that they discuss, their demographics and backgrounds, and their awareness of security issues and tools. Our results showed that the community of smart contract developers is very active and growing rapidly, in comparison with the general user population. However, we also found that a large fraction of smart contract related questions remain unanswered. This indicates that there may be a gap in community knowledge regarding a number of issues, which may pose a threat to the sustainability of an active community. Finally, we observed limited discussion of security related topics, which is very concerning given that many smart contracts suffer from security vulnerabilities in practice.

References

[1] S. Underwood, "Blockchain beyond Bitcoin," *Communications of the ACM*, vol. 59, no. 11, pp. 15–17, 2016.

[2] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," Ethereum Project – Yellow Paper, Tech. Rep. EIP-150, April 2014.

[3] A. Laszka, S. Eisele, A. Dubey, and G. Karsai, "TRANSAX: A blockchain-based decentralized forward-trading energy exchange for transactive microgrids," in *Proceedings of the 24th IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, December 2018.

[4] N. Wang, X. Zhou, X. Lu, Z. Guan, L. Wu, X. Du, and M. Guizani, "When energy trading meets blockchain in electrical power system: The state of the art," *Applied Sciences*, vol. 9, no. 8, 2019.

[5] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[6] A. Tapscott and D. Tapscott, "How blockchain is changing finance," *Harvard Business Review*, vol. 1, no. 9, 2017.

[7] N. Atzei, M. Bartoletti, and T. Cimoli, "A survey of attacks on Ethereum smart contracts (SoK)," in *Proceedings of the 6th International Conference on Principles of Security and Trust (POST)*. Springer, April 2017, pp. 164–186.

[8] I. Nikolić, A. Kolluri, I. Sergey, P. Saxena, and A. Hobor, "Finding the greedy, prodigal, and suicidal contracts at scale," in *Proceedings of the 34th Annual Computer Security Applications Conference (ACSAC)*, 2018, pp. 653–663.

[9] K. Finley, "A $50 million hack just showed that the DAO was all too human," Wired https://www.wired.com/2016/06/50-million-hack-just-showed-dao-human/, June 2016.

[10] L. H. Newman, "Security news this week: $280m worth of Ethereum is trapped thanks to a dumb bug," WIRED, https://www.wired.com/story/280m-worth-of-ethereum-is-trapped-for-a-pretty-dumb-reason/, November 2017.

[11] M. di Angelo and G. Salzer, "A survey of tools for analyzing ethereum smart contracts," 2019. [Online]. Available: https://pdfs.semanticscholar.org/5fcd/6089a4973d3ddd7ca831b7129046c87f33c6.pdf

[12] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proceedings of the 23rd ACM SIGSAC Conference on Computer and Communications Security (CCS)*, October 2016, pp. 254–269.

[13] P. Tsankov, A. Dan, D. D. Cohen, A. Gervais, F. Buenzli, and M. Vechev, "Securify: Practical security analysis of smart contracts," in *25th ACM Conference on Computer and Communications Security (CCS)*, 2018.

[14] A. Mavridou and A. Laszka, "Designing secure Ethereum smart contracts: A finite state machine based approach," in *Proceedings of the 22nd International Conference on Financial Cryptography and Data Security (FC)*, February 2018.

[15] A. Mavridou, A. Laszka, E. Stachtiari, and A. Dubey, "VeriSolid: Correct-by-design smart contracts for Ethereum," in *Proceedings of the 23rd International Conference on Financial Cryptography and Data Security (FC)*, February 2019.

[16] A. Mavridou and A. Laszka, "Tool demonstration: FSolidM for designing secure Ethereum smart contracts," in *Proceedings of the 7th International Conference on Principles of Security and Trust (POST)*, April 2018.

[17] R. O'Connor, "Simplicity: A new language for blockchains," in *Proceedings of the 2017 Workshop on Programming Languages and Analysis for Security*, ser. PLAS, New York, NY, USA, 2017, pp. 107–120. [Online]. Available: http://doi.acm.org/10.1145/3139337.3139340

[18] R. M. Parizi, A. Dehghantanha, K.-K. R. Choo, and A. Singh, "Empirical vulnerability analysis of automated smart contracts security testing on blockchains," in *Proceedings of the 28th Annual International Conference on Computer Science and Software Engineering (CASCON)*, 2018.

[19] D. Tryfanava, "25 Best Active Forums for Programmers," https://vironit.com/best-active-forums-for-programmers/, 2018, accessed: April 1st, 2019.

[20] Ashutosh KS, "Top 10 Sites to Ask All Your Programming Questions," https://www.hongkiat.com/blog/programming-questions-websites/, 2017, accessed: April 1st, 2019.

[21] "Stack Exchange Directory Listing-Internet Archive," https://archive.org/download/stackexchange, accessed: April 1st, 2019.

[22] "Stack Exchange Data Explore," https://data.stackexchange.com/, accessed: March 27, 2019.

[23] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[24] B. Mueller, "Smashing Ethereum smart contracts for fun and real profit," 9th Annual HITB Security Conference (HITBSecConf), 2018.

[25] J. Ellul and G. Pace, "Runtime verification of Ethereum smart contracts," in *Workshop on Blockchain Dependability (WBD), in conjunction with 14th European Dependable Computing Conference (EDCC)*, 2018.

[26] E. Albert, P. Gordillo, B. Livshits, A. Rubio, and I. Sergey, "EthIR: A framework for high-level analysis of Ethereum bytecode," in *16th International Symposium on Automated Technology for Verification and Analysis (ATVA)*, 2018.

[27] L. Brent, A. Jurisevic, M. Kong, E. Liu, F. Gauthier, V. Gramoli, R. Holz, and B. Scholz, "Vandal: A scalable security analysis framework for smart contracts," *arXiv preprint arXiv:1809.03981*, 2018.

[28] R. Stortz, "Rattle – an Ethereum EVM binary analysis framework," REcon Montreal, 2018.

[29] M. Bartoletti and L. Pompianu, "An empirical analysis of smart contracts: Platforms, applications, and design patterns," in *Proceedings of the 1st Workshop on Trusted Smart Contracts (WTSC)*, April 2017, pp. 494–509.

[30] M. Wöhrer and U. Zdun, "Design patterns for smart contracts in the ethereum ecosystem," in *Proceedings of the 2018 IEEE Conference on Blockchain*, 2018, pp. 1513–1520.

[31] H. Jiang, D. Liu, Z. Ren, and T. Zhang, "Blockchain in the eyes of developers," 2018.

[32] A. Barua, S. W. Thomas, and A. E. Hassan, "What are developers talking about? an analysis of topics and trends in stack overflow," *Empirical Software Engineering*, vol. 19, no. 3, pp. 619–654, Jun 2014.

[33] R. M. Parizi, Amritraj, and A. Dehghantanha, "Smart contract programming languages on blockchains: An empirical evaluation of usability and security," in *Proceedings of the 1st International Conference on Blockchain (ICBC)*, 2018, pp. 75–91.

[34] K. Delmolino, M. Arnett, A. Kosba, A. Miller, and E. Shi, "Step by step towards creating a safe smart contract: Lessons and insights from a cryptocurrency lab," in *Proceedings of the 3rd Workshop on Bitcoin and Blockchain Research (BITCOIN)*, 2016, pp. 79–94.

[35] M. di Angelo, C. Sack, and G. Salzer, "SoK: Development of secure smart contracts–lessons from a graduate course," in *Proceedings of the 3rd Workshop on Trusted Smart Contracts (WTSC)*, 2019.