AFRL-RI-RS-TR-2020-076

BLOCKCHAIN-BASED CYBER SECURITY MANAGEMENT (B2CSM): DESIGN, ANALYSIS, AND PROTOTYPE IMPLEMENTATION

UNIVERSITY OF TEXAS AT SAN ANTONIO

MAY 2020

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nations. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2020-076 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /
TODD N. CUSHMAN
Work Unit Manager

/ S /
JAMES S. PERRETTA
Deputy Chief, Information
Exploitation & Operations Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
MAY 2020	FINAL TECHNICAL REPORT	MAR 2019 – JAN 2020

4. TITLE AND SUBTITLE

BLOCKCHAIN-BASED CYBER SECURITY MANAGEMENT (B2CSM): DESIGN, ANALYSIS, AND PROTOTYPE IMPLEMENTATION

5a. CONTRACT NUMBER
N/A

5b. GRANT NUMBER
FA8750-19-1-0019

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

Shouhuai Xu

5d. PROJECT NUMBER
BC2B

5e. TASK NUMBER
2C

5f. WORK UNIT NUMBER
SM

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Texas at San Antonio
One UTSA Circle
San Antonio, TX 78249

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RIGA
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER
AFRL-RI-RS-TR-2020-076

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The research objective of the project is to investigate an innovative Cyber Security Management (CSM) system, which can automate the process for accomplishing Bob's mission based on, for example, (i) the history data that is relevant to the mission and (ii) characteristics of the APT received from the third party. The project addresses two research problems: (i) How can one assure the integrity of the history data, despite that the APT may have compromised a fraction of the storage system and attempted to manipulate the history data to hide its footprints and persist its presence in the network? (ii) How can one automate the inference process for determining whether or not the network has been penetrated into and assess the damage thereof (according to the history data and characteristics of the APT)?

15. SUBJECT TERMS

Cyber security, internet of things, blockchain

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			**TODD N. CUSHMAN**
U	U	U	UU	29	19b. TELEPHONE NUMBER (*Include area code*) **N/A**

TABLE OF CONTENTS

LIST OF FIGURES

1.0 SUMMARY

Suppose Bob is the defender of a network, such as the Air Force enterprise network. Suppose on March 1, 2020 Bob is informed with characteristics of a new Advanced Persistent Threat (APT) attack, which apparently has been in the wild for a while. In this scenario, Bob needs to investigate whether or not the network was penetrated into by the APT prior to March 1, 2020 (and if so, what the cascading damages are as of March 1, 2020). How can Bob accomplish his mission? The research objective of the project is to investigate an innovative Cyber Security Management (CSM) system, which can automate the process for accomplishing Bob's mission based on, for example, (i) the history data that is relevant to the mission and (ii) characteristics of the APT received from the third party. The project addresses two research problems: (i) How can one assure the integrity of the history data, despite that the APT may have compromised a fraction of the storage system and attempted to manipulate the history data to hide its footprints and persist its presence in the network? (ii) How can one automate the inference process for determining whether or not the network has been penetrated into and assess the damage thereof (according to the history data and characteristics of the APT)?

2.0 INTRODUCTION

Cyber Security Management (CSM) is intuitive and necessary for any networked systems, including enterprise networks, Internet-of-Things (IoT) networks, and Cyber-Physical System (CPS) networks. However, the problem of CSM has not been systematically formulated in a rigorous fashion. In this paper, we initiate the study of CSM related to cyber intelligence sharing, meaning that the participating defenders share cyber intelligence with each other and leverage the shared cyber intelligence for their routine CSM missions. In particular, we focus on how a defender can leverage such cyber intelligence to answer a range of cyber security questions. The kinds of cyber intelligence we consider include: newly detected cyber attackers, which may be leveraged to detect previously undetected victims; newly detected victims, which may be leveraged to detect previously undetected attackers; and new defense capabilities, which may be leveraged to detect previously undetected attacks.

In order to see the importance of CSM, let us consider the following example scenario. Suppose Bob is the defender of a network, such as enterprise network. Suppose on March 1, 2020 Bob is informed with characteristics of a new Advanced Persistent Threat (APT) attack, which has been active in the wild for a while. Given this piece of cyber intelligence, Bob would need to investigate whether or not his network (i.e., the network he manages) was penetrated into by the APT prior to March 1, 2020 and if so, what the damages are as of March 1, 2020. In current practice, this task (if present at all) would have been done manually by Bob, for example by going through some tedious procedure (e.g., manually analyzing the historic network data). Our objective is to automate the procedure and answer a range of questions, such as the preceding one, to accomplish Bob's mission.

A. Innovation Claims

In this paper, we formulate the CSM problem and propose a solution to tackling it. Specifically, we make the following innovation claims.

First, we formulate the CSM problem from three perspectives: Network-centric CSM (N-CSM), which leverages network-layer data for CSM purposes; Tools-centric CSM (T-CSM), which leverages data collected from cyber defense tools for CSM purposes; and Application-centric CSM (A-CSM), which leverages application-specific data for CSM purposes. For organizing and storing these kinds of cyber security data, we propose using the abstraction of Annotated Graph Time Series Representation (AGTSR). We give a detailed description on how AGTSR can be instantiated to represent data for N-CSM, T-CSM and A-TSM purposes.

Second, we propose and implement three specific N-CSM functions; the details of the algorithms are given in Reference [1].

Third, we propose and implement three specific T-CSM functions; the details of the algorithms are given in Reference [1].

Fourth, we propose and implement three specific A-CSM functions; the details of the algorithms are given in Reference [1].

Fifth, in order to make the resulting N-CSM, T-CSM and A-CSM functions themselves robust against advanced cyber attacks (e.g., APTs that may attempt to compromise the CSM functions), we leverage the blockchain technology to assure the integrity of the cyber security data mentioned above, leading to the notion of Blockchain-Based CSM (B2CSM). B2CSM can tolerate the compromise of a certain threshold fraction of the data storage systems (i.e., nodes in the blockchain network). We implement a prototype B2CSM system and report its performance, which shows that B2CSM is effective.

This project also led to a top-quality publication in Reference [2].

B. Related Work

To the best of our knowledge, we are the first to investigate the application of blockchain for the kinds of cyber security management functions discussed in the present paper, namely N-CSM, T-CSM and A-CSM. This is true despite that blockchain has been applied to some security problems in the context of PKI, data integrity, access control, and voting. Specifically, the blockchain technology has been leveraged for applications other than bitcoin and smart contracts (e.g., References [3,4]), such as: (i) managing public key certificate and revocation states (e.g., Reference [5]); (ii) enhancing the trustworthiness of cryptographic digital signatures in the presence of compromised private signing keys (e.g., References [6,7,8]); (iii) facilitating data integrity and resource sharing in IoT systems (e.g., Reference [9]); (iv) detecting violations of access control policies in cloud environments (e.g., Reference [10]); (v) securing cloud storage (e.g., Reference [11]); (vi) managing data provenance and accountability (e.g., Reference [12]); (vii) enabling data sharing (e.g., Reference [13]); (viii) enabling malware detection in mobile devices (e.g., Reference [14]); (ix) enabling secure voting systems (e.g., Reference [15]); and (x) achieving global naming and storage (e.g., Reference [16]). We foresee that CSM will become an integral part of Cybersecurity Data Analytics (e.g., References [17-48]), which is, together with First-Principle Modeling (e.g., References [49-60]) and Cybersecurity Metrics (e.g., References [61-69]), an integral part of the Cybersecurity Dynamics framework (e.g., References [70-72]). In this report we focus on three classes of CSM functions, which could be extended to accommodate other kinds of CSM functions (e.g., References [73-88]).

3.0 METHODS, ASSUMPTIONS, AND PROCEDURES

A. Terminology

A cyber defender, Bob, manages a set of entities, which are broadly defined to accommodate computers/devices or other kinds of objects of cybersecurity significance (e.g., browsers or email readers). As illustrated in Figure 1, we make the distinction between external entities (i.e., the ones that are not managed by Bob but are managed by another defender Cindy) and internal entities (i.e., the ones that are managed by Bob); it is clear that this external vs. internal distinction is from a specific defender's point of view (i.e., Bob in Figure 1). An entity, internal and external alike, can be in one of three states: normal, victim or attacker. A victim entity is one that has been compromised by an external or internal attacker entity; an attacker entity is one that exhibits malicious behavior; and a normal entity is one that is neither a victim or nor attacker entity. A normal entity can become a victim entity when it is attacked by an external or internal attacker entity; a victim entity can elevate to an attacker computer.

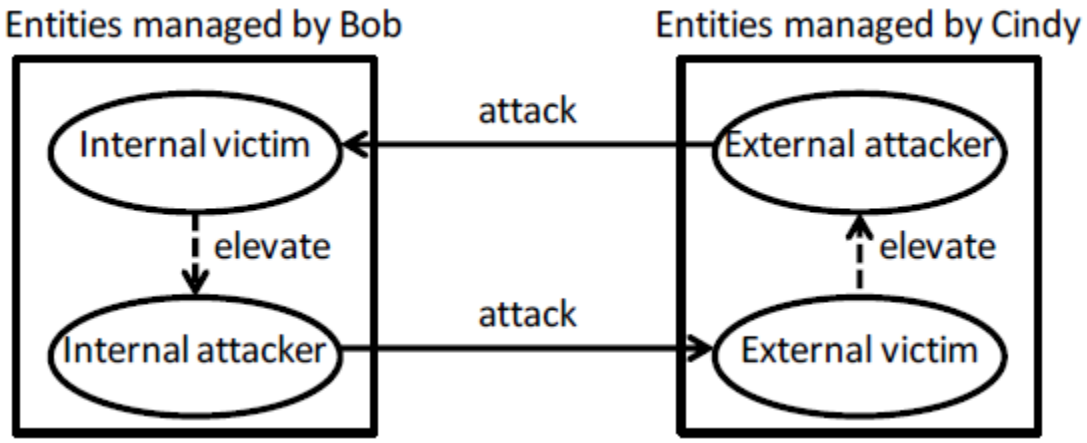


Figure 1. Illustration of external vs. internal attacker and external vs. internal victim from defender Bob's point of view.

It is worth mentioning that depending on the specificity of the input cyber intelligence, the victims and attackers identified by the CSM functions may or may not need to be further analyzed for confirming whether or not an entity is indeed an attacker/victim. In the former case, the value of the CSM functions is in automatically and substantially narrowing down the potential victims and attackers for further investigation.

B. CSM Model

In the CSM model, a defender Bob, or more precisely his CSM App (CSMA), leverages some input cyber intelligence to identify victim and attacker entities, where the input intelligence may be (i) shared by some third parties or another defender or (ii) discovered by some cyber defense tools used by Bob. In what follows, we describe five kinds of cyber intelligence, three classes of CSM function, and a general data structure design to facilitate those CSM functions.

Input Cyber Intelligence.
As illustrated in **Figure 2**, we consider the following five kinds of input cyber intelligence (prefixed by 'I-').

- (I-1A) This kind of intelligence pointing to some external attackers, possibly accompanied by the time window during which an external attacker is active.
- (I-1B) This kind of intelligence pointing to some internal attackers that have attacked some external victims that are detected by another defender, or some internal victims that are detected by some cyber defense tools used by Bob.
- (I-2A) This kind of intelligence pointing to some external victims, which have been attacked by some internal or external attackers.
- (I-2B) This kind of intelligence pointing to some internal victims, which have been attacked by some internal or external attackers. The intelligence may be identified, for example, by the leakage of data specific to the victim (e.g., social security numbers or passwords) or by a cyber defense tool (e.g., intrusion detection system or anti-malware tool).
- (I-3) This kind of intelligence points to some new defense capabilities, such as methods for detecting previously undetected attacks (e.g., 0-day attacks).

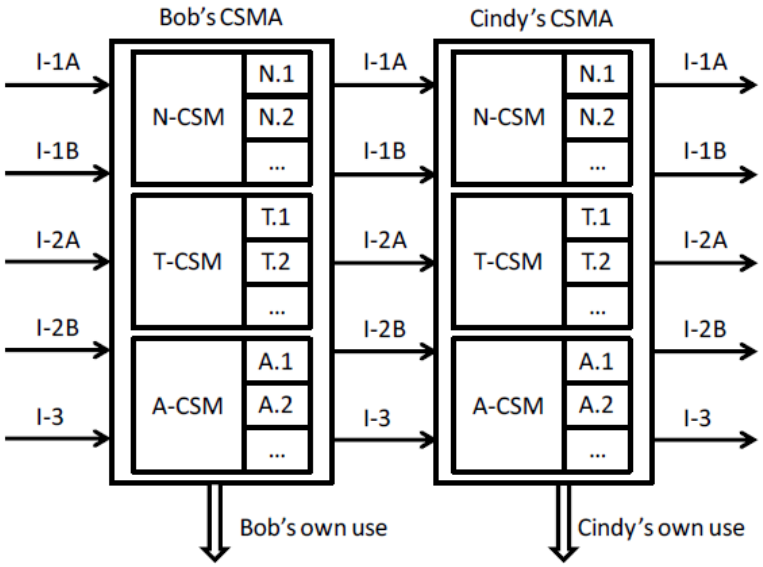


Figure 2. CSM model illustrated with five kinds of input cyber intelligence (prefixed by 'I-') and three classes of CSM functions.

As illustrated in Figure 2, Bob's CSMA takes as input some of these kinds of cyber intelligence and the relevant cyber data, uses the CSM functions (specified below) to identify the other internal attackers/victims and/or external attackers/victims, and possibly shares the resulting intelligence with another defender, say Cindy, about her internal attackers/victims and external attackers/victims (i.e., input cyber intelligence I-1A, I-1B, I-2A, I-2B from Cindy's point of view).

Overview of Three Classes of CSM Functions.
Having specified the kinds of cyber intelligence as input to CSMAs, now we specify the CSMA functions that leverage the intelligence for CSM purposes. Given the complexity of CSM, we divide it into three classes, as illustrated in Figure 2. The three classes are: (i) Network-centric CSM (N-CSM), which leverages network-layer data and cyber intelligence for CSM purposes; (ii) Tools-centric CSM (T-CSM), which leverages data collected from cyber defense tools and cyber intelligence for CSM purposes; and (iii) Application-centric CSM (ACSM), which leverages application-specific data and cyber intelligence for CSM purposes. Each class contains multiple CSM functions. The basics ideas of these functions are described below, and their respective algorithmic specifications are presented in Sections II-B, II-C, and II-D.

N-CSM functions are centered at examining the input cyber intelligence against network traffic data, which may be collected at a network gateway between the external network (e.g., the Internet) and the internal network (e.g., an enterprise network). In principle, network traffic data can be represented by IP packets and TCP/UDP flows, which incur different costs on storage. As examples, we define three T-CSM functions.

T-CSM functions are centered at cyber defense tools, such as Network-based Intrusion Detection Systems (NIDS) and Host-based Intrusion Detection Systems (HIDS) including anti-malware systems. These cyber defense tools may be based on known signatures, Artificial Intelligence or Machine Learning (AI/ML). These tools often output alerts as indicators of malicious or suspicious activities. As examples, we define three T-CSM functions.

A-CSM functions are centered at specific applications that are often exploited to wage attacks, such as web applications and email systems. Web applications have been widely abused to wage drive-by download attacks (i.e., a vulnerable browser gets compromised when visiting a malicious website or URL) and support attacker's command-and-control (e.g., botnet command-and-control). Emails have been abused to wage social engineering attacks, especially spear phishing, which often preludes devastating attacks (including Advanced Persistent Threats or APT). As examples, we consider three A-CSM functions.

Details of these functions can be found in Reference [1], which also includes the algorithms that implement these CSM functions as well as their accompanying data structures.

C. B2CSM Model and Architecture

A straightforward realization of the CSM model highlighted in Figure 2 would be that each defender secures its own cyber data. This realization is vulnerable to single-point-of-failure because the compromise of the cyber data storage undermines the CSM functions. In order to enhance the robustness of the CSM functions against such attacks, we propose leveraging the blockchain technology, leading to the notion of B2CSM (Blockchain-Based CSM).

Figure 3 highlights the B2CSM model, which extends the CSM model (Figure 2) by storing the cyber data in a B2CSM blockchain. The B2CSM model also has two other kinds of modules: B2CSM Apps and B2CSM Agents. At a high level, B2CSM Agents collect the relevant cyber data and write the data to the B2CSM blockchain; B2CSM Apps take input cyber intelligence and run their N-CSM, T-CSM and A-CSM functions to fetch the relevant cyber data from the blockchain, and possibly generate output cyber intelligence for other defenders to use.

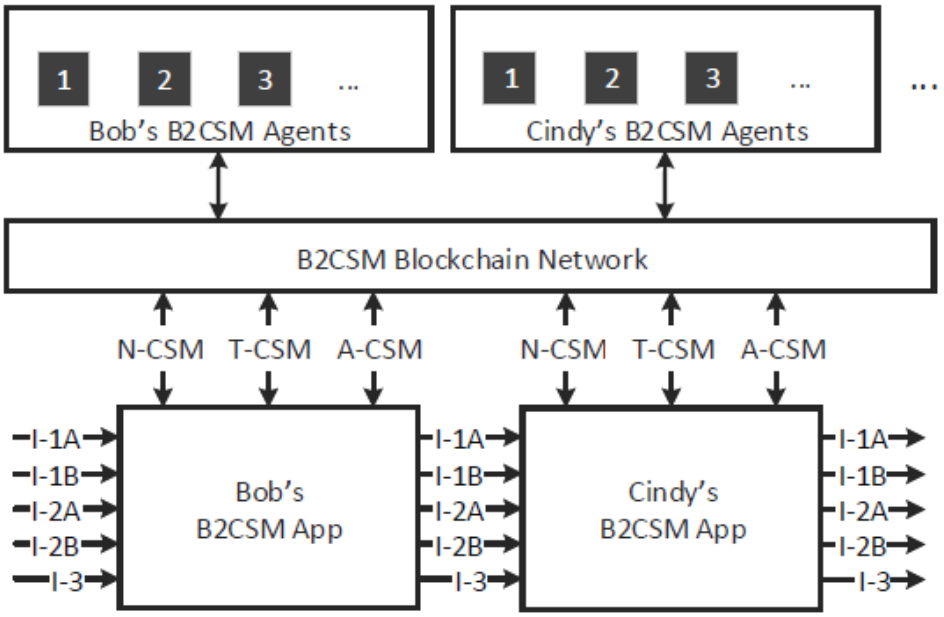


Figure 3. The B2CSM model extends the CSM model (Figure 2).

D. Instantiating B2CSM Architecture into B2CSM Systems

The B2CSM architecture described in Figure 4 can be instantiated into B2CSM systems in different ways.

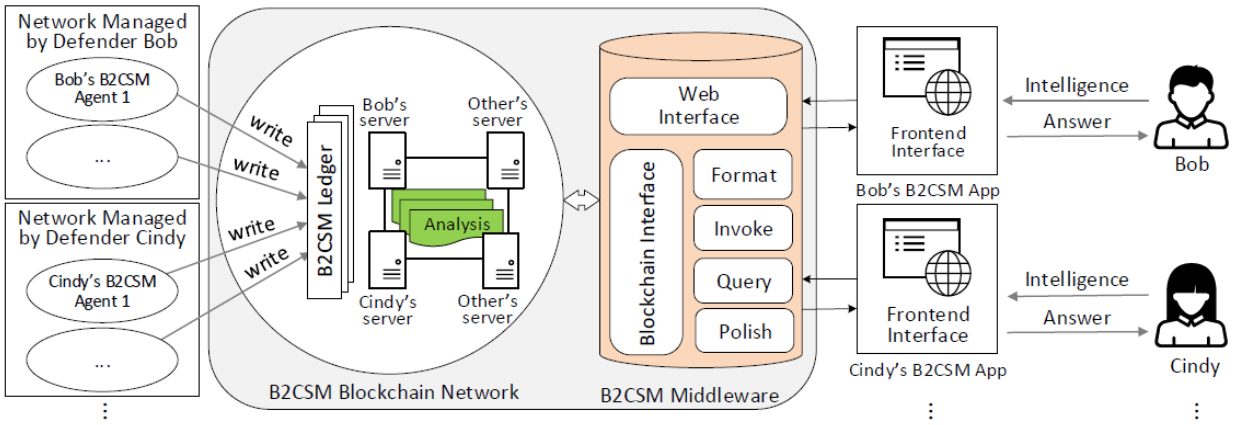


Figure 4. B2CSM architecture.

E. B2CSM System Security Analysis

Now we informally analyze the security of the B2CSM system as described above. The analysis is with respect to the security objectives, threat model, and assumptions mentioned above.

Security objectives.
We consider four security objectives:

- Integrity. The integrity of the data stored in the blockchain is assured, as long as the fraction of compromised nodes in the blockchain is bounded from above by a certain threshold.
- Availability. The availability of the data stored in the blockchain is assured, as long as the fraction of compromised nodes in the blockchain is bounded from above by a certain threshold.
- Consistency. The consistency of the data stored in the blockchain is assured, as long as the fraction of compromised nodes in the blockchain is bounded from above by a certain threshold.
- Accountability. The B2CSM Agents are held accountable for the data they write into the blockchain and the B2CSM Apps are held accountable for the CSM functions they run against the blockchain.

Threat model.
We consider an attacker with the following capabilities.

- Compromising B2CSM blockchain full nodes. The attacker can penetrate into a threshold fraction of the blockchain full nodes. The attacker has total control over these compromised nodes and can coordinate their activities in an arbitrary fashion (i.e., Byzantine).
- Interfering with message deliveries. The attacker can control the order of message deliveries in the blockchain network. The attacker can arbitrarily delay message deliveries to

each computer (but not forever, see assumption below), for example by waging Denial-of-Service (DoS) attacks during a finite period of time.

We make the following assumptions on what the attacker cannot do.

- **Assumption 1** (cryptographic assurance): Assumption related to cryptography. We make standard assumptions to assure the security of cryptographic schemes (e.g., hash functions and digital signatures) in the framework of modern cryptography. Informally speaking, these assumptions say that as long as cryptographic keys (if applicable) are not compromised, cryptographic schemes are secure. That is, in order for the attacker to compromise a cryptographic assurance, the attacker has to penetrate into a system in question to compromise the cryptographic key or cryptographic service (for attaining "oracle" access to a cryptographic function).
- **Assumption 2** (consensus assurance): Assumption related to consensus and/or communication network in terms of network synchrony. We consider the partially synchronous model, which assumes that although message delivery in the blockchain network can be interfered by the attacker, messages will be eventually delivered within a finite amount of time, which is however not known to the system designer.
- **Assumption 3** (attacker capability): Assumption related to the compromise of nodes maintaining the blockchain. For the full nodes that maintain the blockchain, we assume that no more than one-third of them are compromised simultaneously, which is inherent to the underlying Byzantine Fault-Tolerance (BFT) protocol we will adopt.
- **Assumption 4** (data and intelligence authenticity): Assumption related to the data for CSM purposes. We assume that the integrity of the data collected for N-CSM, T-CSM and A-CSM purposes, namely the G(t)'s mentioned above, is assured. We also assume that the cyber intelligence is authentic. Assuring that these two assumptions hold is an orthogonal research problem because the CSM functions are defined to operate on given inputs; if these inputs are not authenticate, the output of the CSM functions are not assured to be correct or useful.
- **Assumption 5** (B2CSM implementation security): The attacker cannot compromise a defender Bob, the B2CSM App, or the B2CSM Middleware because compromising these components of the B2CSM system can immediately render it to give arbitrary output as desired by the attacker.

4.0 RESULTS AND DISCUSSION

A. B2CSM Prototype System

The preceding design choices guide us to design a prototype system. Figure **5** illustrates the B2CSM prototype system for experimental evaluation. Detailed description can be found in Reference [1].

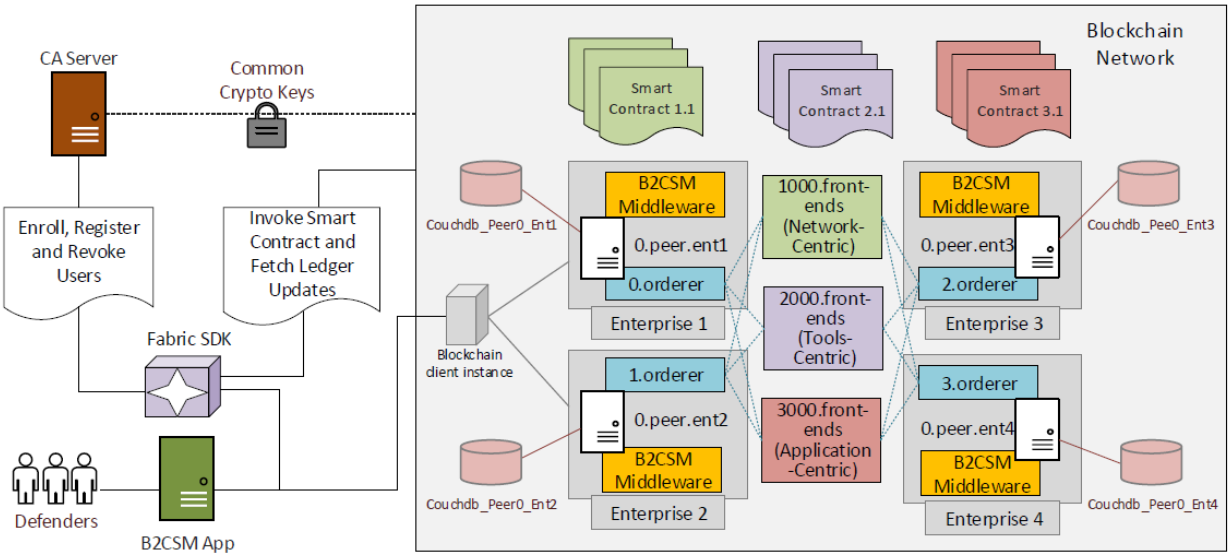


Figure 5. Illustration of the B2CSM prototype system with four peer blockchain nodes with four enterpries.

B. Security Analysis of the B2CSM Prototype System

The integrity objective, namely that the data stored in B2CSM blockchain cannot be maliciously manipulated, is assured by the immutability of the B2CSM ledger. This is because the data stored in the B2CSM blockchain is endorsed by multiple blockchain full nodes according to the endorsement policy. Under Assumptions 1 and 3, the attacker, despite being able to compromise no more than one-third of the full nodes, can neither mislead the full nodes to write the data collected by the B2CSM Agents4 into the B2CSM blockchain, nor mislead the full nodes to accept manipulated data returned by the B2CSM blockchain as valid.

The availability objective is assured because of the following. First, even if one-third of the B2CSM blockchain full nodes are compromised or not participating in executing the protocols that are incurred by the CSM functions, the B2CSM prototype system can still return the correct result because of Assumption 3. Second, by design, each full node of the B2CSM blockchain runs the duplicated middleware services. This means that even if the attacker can compromise one-third of the full nodes, therefore one-third of the middleware ser- vices, the B2CSM prototype system remains to be functioning.

The consistency objective requires that the data items stored on the full nodes of the B2CSM blockchain are same. This is assured because of the following. Recall that Fabric follows the

execute-order-validate discipline to reach the consensus when adding new blocks to the blockchain. Based on Assumptions 1 and 3, the honest peers always append the same block to their local copy of the blockchain. These data items include both the transaction data items corresponding to the data collected by the B2CSM Agents and the transaction data items corresponding to the execution transactions of the CSM functions.

The accountability objective assures that the B2CSM Agents and Apps can be held accountable for their activities. This is assured because (i) both the B2CSM Agents and B2CSM Apps are authenticated through their public keys?; (ii) their writing operations to the B2CSM blockchain are verified by the honest full nodes of the blockchain, despite that some of them may be compromised as assured by Assumptions 1 and 3; and (iii) the integrity of the blockchain data is assured, as discussed above.

C. B2CSM Performance Evaluation Metrics

A blockchain's performance is often measured by the read/write latency/throughput [9]. This suggests us to examine the B2CSM blockchain's performance without considering the CSM functions. Specifically, we consider the following three vanilla metrics:

- (i) B2CSM blockchain's vanilla read latency, which is defined as the time difference between when a data read request is issued to the blockchain and when the response is received from the blockchain;
- (ii) B2CSM blockchain's vanilla write throughput, which is defined as total number committed transactions _ transaction size / total amount of time on writing to blockchain, and the unit can be KBytes/second;
- (iii) B2CSM blockchain's vanilla write latency, which is defined as transaction confirmation time -transaction submission time. These metrics can be measured by using "dummy" data (i.e., with no application semantics) because they are geared towards the B2CSM blockchain rather than the CSM functions. These metrics are measured by taking the average of many independent experimental runs.

Since B2CSM is CSM-specific, we further propose two CSM-specific performance metrics:

- (iv) Application Query Latency (AQL) and
- (v) Data Replication Throughput (DRT).

We stress that (v) is important despite that we already use (ii); this is because (ii) is geared towards traditional transactions of small data volume, but B2CSM often encounters transactions of large data volume in G(t). The measurement of (iv) and (v) will be based on some real-world datasets (i.e., with application semantics). The AQL metric measures the time interval between when a defender invokes a CSM function and when the defender receives the response. Detailed definitions of metrics (iv) and (v) can be found in Reference [1].

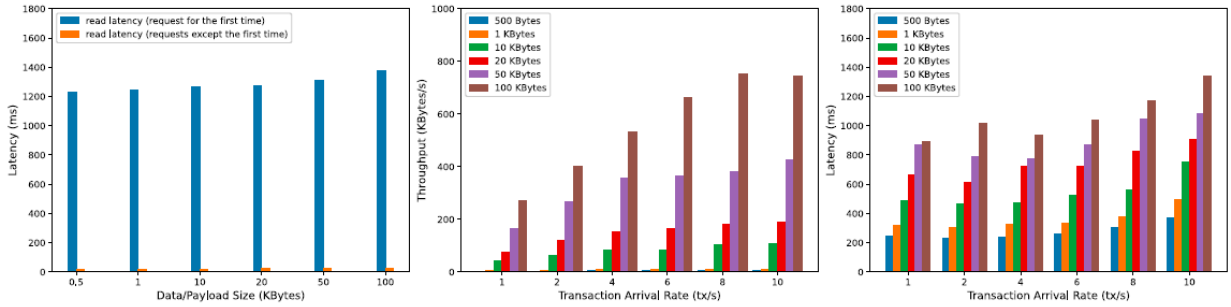
D. B2CSM Prototype Performance Evaluation Experiments

In order to evaluate the performance of the B2CSM prototype system, we conduct experiments using a small-scale cluster composed of four Virtual Machines (VMs) residing on two heterogeneous servers, representing four enterprises to formulate a consortium B2CSM blockchain. One

server is Dell PowerEdge R740, which is equipped with 2 processors in Intel(R) Xeon(R) CPU Silver 4114 (with 13.75 MB L3 cache and 20 cores of 2.2 GHz for each processor), 256 GB (16 slots x 16 GB/slot) 2400MHz DDR4 RDIMM memory, and 8 TB (8 slots x 1TB/slot) 2.5 inch SATA hard drive. The other server is Dell Precision Rack 7910, which is equipped with 2 processors in Intel(R) Xeon(R) CPU E5-2630 v3 (with 15MB cache and 6 cores of 2.4GHz for each processor), 16GB 2133MHz DDR4 RDIMM ECC memory, and 256GB 2.5 inch SATA solid state drive. All of the four VMs have the same configuration of 8 vCPUs, 24 GB memory and 800 GB hard drive and are connected via a Local Area Network (LAN). The operating system in each VM is Ubuntu 16.04 (64-bit) with kernel version 4.15. The Fabric version is 1.2, the Java version is 1.8.0 211, and the golang version is 1.11.10.

E. B2CSM Blockchain's Vanilla Performance

Now we report B2CSM blockchain's vanilla performance in read latency, write throughput and write latency. Figure 6a plots the B2CSM blockchain's vanilla read latency. Experimental results show that the read latency is small because its queries are geared towards the world state instead of the blocks. Besides, the latency slightly increases as the transaction size. It is worth mentioning that in the experiments, the VMs or full nodes locate in a single LAN, meaning that the read latency would be larger when the VMs or full nodes are deployed in a WAN (Wide Area Network).



(a) Vanilla read latency (b) Vanilla write throughput (c) Vanilla write latency

Figure 6. B2CSM's vanilla performance, averaged over five independent experimental runs

Figure 6b plots B2CSM blockchain's vanilla write throughput, from which we make the following observations. First, for a fixed transaction arrival rate, increasing the payload/data size leads to an increase in the throughput (e.g., from 500 bytes to 100 KBytes when the transaction arrival rate is 1 transaction per second). This suggests that the time cost has a smaller impact on the throughput. Second, for a fixed transaction arrival rate, when the data size is small, the size itself becomes the bottleneck of the throughput. At the same time, the data, correspondingly the transaction, size should be limited to a certain threshold; otherwise, the data cannot be written to the blockchain. Our experiment suggests that once the data size in a transaction exceeds 100 KBytes, timeout is often incurred and the transaction data cannot be successfully written to the blockchain.

Figure 6c plots the write latency corresponding to the throughput shown in Figure 10b. We make the following observations. First, for a fixed transaction arrival rate, the latency increases with the data size because a bigger volume of data data needs to be transfer. Second, for a fixed data

size, the latency varies slightly when the transaction arrival rate is relatively small. However, the delay gradually increases with the transaction arrival rate. This is caused by two factors. (i) During the process of data replication, each transaction submission is bound to a unique thread and each VM in the experiment is equipped with 8 vCPUs, indicating that handling 8 threads in parallel is the optima for a single VM. When a B2CSM App simultaneously submits multiple transactions, the other nodes need to simulate the transactions because the endorsement policy requires 2f + 1 (where f = 1 in the experiment) nodes to endorse. Therefore, if the number of submitted transactions exceeds 8, the endorsing nodes need more time to complete the endorsement procedure, resulting in a higher latency. (ii) The communication complexity of the BFT-SMaRt consensus protocol is $O(n^2)$, where n is the number of full nodes in a blockchain network and n = 4 in the experiment. A higher transaction arrival rate means that more data will need to be simultaneously transferred between the full nodes, incurring a higher latency. As a consequence, the throughput stays relatively stable, as shown in Figure 10b when the transaction arrival rate is 8 transactions per second and 10 transactions per second.

In summary, small transaction size and low transaction arrival rate lead to a lower throughput; large transaction size and high transaction arrival rate do not really improve the throughput and actually could congest the network (i.e., a longer waiting time in transferring message can trigger timeouts and fail the writing of data to the B2CSM blockchain). This highlights the importance in feeding the data (wrapped as transactions) with a proper transaction size and transaction arriving rate (while noting that for a given setting, such as N-CSM, the total volume of data is inherent to the network in question). Our experiment shows that 100 KBytes per data unit and 8 data units per second achieves a better throughput. In order to achieve such a better throughput in general, the CSMAs can split the large dataset into data units and maintain a buffer to periodically submit these data units as transactions to blockchain network. Besides, adjusting the endorsement policy (e.g., only one organization or full node, rather than all of them, is required to endorse the transactions) can contribute to improving the throughput. However, this gain in throughput demands a stronger trust assumption about the consortium peers; otherwise, B2CSM will achieve a lower degree of robustness against Byzantine faults.

Insight 1: Transaction arrival rate and transaction size are two parameters that need to be carefully selected because they collectively have a big impact on the B2CSM blockchain's throughput. These parameters need to be fine-tuned based on the compute resources that are available to the full nodes of the B2CSM blockchain.

F. B2CSM Performance based on Experiments with Real-World Datasets

Now we report B2CSM's performance based on experiments using some real-world datasets respectively for N-CSM, T-CSM and A-CSM.

- In our N-CSM experiment, we use a dataset collected from a honeypot. The dataset contains a /22 external subnet. Since the honeypot is "flat" network. The experiment is based the dataset corresponding to 7 days, and the time resolution is day (i.e., each day is a time interval).
- In our T-CSM experiment, we use a dataset collected by the USMA team from the 2017 CDX Competition, as if it were collected at production enterprise networks.

- In our A-CSM experiment, we use a dataset that contains data records like (appliction; URL; timestamp), meaning that the application accessed the URL at time given by timestamp. We use the Georgia Tech data as if it were collected as such.

Details of the datasets can be found in Reference [1].

Now we report the experimental results in DRT (Data Replication Throughput) and AQL (Application Query Latency).

Figure 7 plots B2CSM's DRT in N-CSM, T-CSM and A-CSM experiments using the real-world datasets mentioned above. We observe that the throughput varies with CSM scenarios. We observe that throughput of T-CSM is significantly different the throughputs of N-CSM and A-CSM. This is caused by the fact that the T-CSM data is quite different from the NCSM and A-CSM data as follows. The T-CSM data volume is large and the volumes of data units vary substantially because some data units contain more empty elements than others (recalling that T-CSM data is about alerts); in contrast, NCSM data and A-CSM data are uniformly distributed (i.e., volumes of data units are about the same). This explains why T-CSM has a lower throughput. From the throughput, we observe that after the transaction arrival rate exceeds 4, the throughput stays stable, especially for N-CSM and A-CSM; this may be caused by the limited compute resource on the full nodes (i.e., VMs) in our experiments. In T-CSM, we observe an "abnormal" throughput at transaction arrival rate 4 and data unit of 4 x 4 (i.e., 102 KBytes per unit); this may be caused by the limited compute resource at the full nodes and the cumulative effect of non-uniform distribution in the units' data volumes. Nevertheless, the overall trend is similar to the trend that is observed from B2CSM blockchain's vanilla write throughput, which reinforces Insight 1.

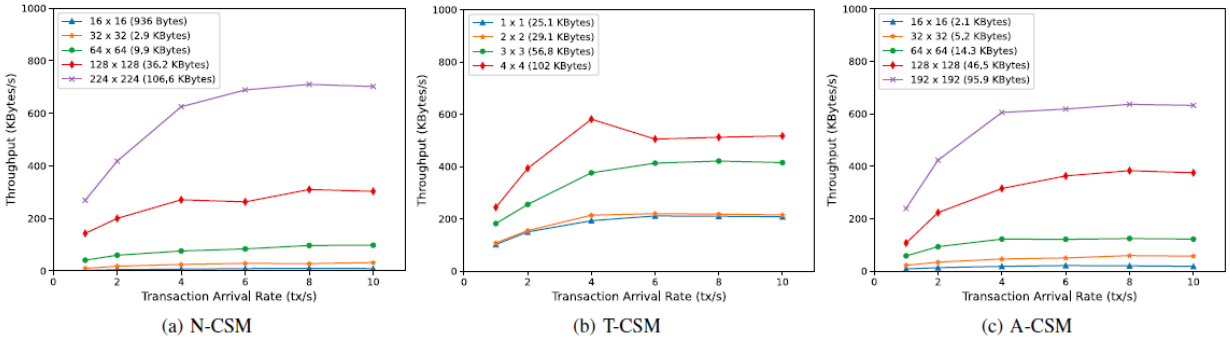
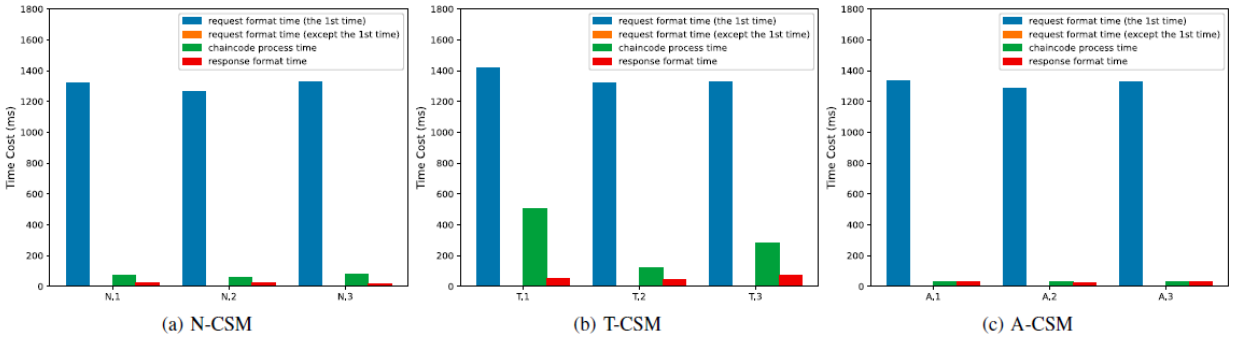


Figure 7. B2CSM's DRT measured in experiments N-CSM, T-CSM, and A-CSM, averaged over five independent experimental runs.

Figure 8 plots B2CSM's AQL in N-CSM, T-CSM and A-CSM experiments with the real-world datasets mentioned above. We observe the following: (i) For the request formatting time, it takes about 1.4 seconds for the first invocation of a CSM function, but much smaller time for next invocation of a CSM function. This is because the former requires to initializing a Hyperledger Fabric client object on behalf of the B2CSM App before connecting to the blockchain network; whereas, the later can simply reuse the object created by the former. (ii) For the chaincode processing time, the time cost varies for different invocations of CSM functions. (iii) The response time is relatively stable (i.e., varies only slightly).



(a) N-CSM (b) T-CSM (c) A-CSM

Figure 8. B2CSM's AQL measured in experiments N-CSM, T-CSM, and A-CSM, averaged over five independent experimental runs.

Insight 2: The response delay can be largely attributed to the creation of a Hyberledger Facbric client object corresponding to a CSM function invoked from a B2CSM App for the first time. Reducing this time will substantially improve the response time.

G. Discussion

The present study has multiple limitations. First, N-CSM, T-CSM and A-CAM collectively may not cover all CSM functions. Future research needs to identify other CSM functions. Second, it would be useful to develop a visualization system to present the results of B2CSM. Third, future research needs to enhance B2CSM's blockchain-related functions. For example, in the current design, chaincode installation is achieved by using tools such as Postman, it would be better to provide a self-contained package for installing B2CSM. Fourth, future research needs to investigate how to cope with replicating large volume of data to the blockchain because G(t), at least for N-CSM and T-CSM can be very large in volume. The current method we use (i.e., splitting a large G(t) into small chunks) may not work well for extremely large G(t). Fifth, the participating defenders of B2CSM can see each other's data G(t). This can be a concern when the data is sensitive. Therefore, it is an important problem to protect data secrecy while allowing the defenders to accomplish the task of CSM.

5.0 CONCLUSIONS

The focus of this project is to identify a killer application in leveraging the blockchain technology to Cyber Security Management (CSM). We propose to design and implement a blockchain-based cyber security management (B2CSM) prototype system. We define a set of queries that would be of interest to cyber defenders in practice, while using some real-world datasets to demonstrate the usefulness of B2CSM framework. To the best, of our knowledge this is the first effort to investigate blockchain's use for cyber security management.

6.0 REFERENCES

1. Songlin He, Eric Ficke, Mir Mehedi Pritom, Huashan Chen, Qiang Tang, Qian Chen, Marcus Pendleton, Laurent Njilla, and Shouhuai Xu. *Method and System for Blockchain-Based Cyber Security Management*. To be submitted for patent application and paper publication. March 2020. (This is the full version of the present report)
2. Huashan Chen, Marcus Pendleton, Laurent Njilla, and Shouhuai Xu. *A Survey on Ethereum Systems Security: Vulnerabilities, Attacks and Defenses*. ACM Computing Survey. Accepted for publication, March 2020.
3. Satoshi Nakamoto. *Bitcoin: A peer-to-peer electronic cash system*. http://www.bitcoin.org/bitcoin.pdf, 2008.
4. T. Salman, M. Zolanvari, A. Erbad, R. Jain, and M. Samaka. *Security services using blockchains: A state of the art survey*. IEEE Communications Surveys Tutorials, pages 1–1, 2018.
5. Mustafa Al-Bassam. *SCPKI: a smart contract-based PKI and identity system*. In Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts, pages 35–40. ACM, 2017.
6. Lei Xu, Lin Chen, Zhimin Gao, Xinxin Fan, Kimberly Doan, Shouhuai Xu, and Weidong Shi. *KCRS: A blockchain-based key compromise resilient signature system*. In Proc. BlockSys'2019, 2019.
7. Shouhuai Xu and Moti Yung. *Expecting the unexpected: Towards robust credential infrastructure*. In International Conference on Financial Cryptography and Data Security, pages 201–221. Springer, 2009.
8. Lei Xu, Lin Chen, Zhimin Gao, Shouhuai Xu, Weidong Shi. *EPBC: Efficient Public Blockchain Client for Lightweight Users*. SERIAL@Middleware 2017: 1:1-1:6
9. K. Christidis and M. Devetsikiotis. *Blockchains and smart contracts for the Internet of Things*. IEEE Access, 4:2292–2303, 2016.
10. M. S. Ferdous, A. Margheri, F. Paci, M. Yang, and V. Sassone. *Decentralised runtime monitoring for access control systems in cloud federations*. In 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), pages 2632–2633, June 2017.
11. Yuzhe Tang, Qiwu Zou, Ju Chen, Kai Li, Charles A. Kamhoua, Kevin A. Kwiat, and Laurent Njilla. *Chainfs: Blockchain-secured cloud storage*. In 11th IEEE International Conference on Cloud Computing, CLOUD 2018, San Francisco, CA, USA, July 2-7, 2018, pages 987–990, 2018.
12. Xueping Liang, Sachin Shetty, Deepak K. Tosh, Charles A. Kamhoua, Kevin A. Kwiat, and Laurent Njilla. *Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability*. In Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGRID 2017, Madrid, Spain, May 14-17, 2017, pages 468–477, 2017.

13. Danda B. Rawat, Laurent Njilla, Kevin A. Kwiat, and Charles A. Kamhoua. *ishare: Blockchain-based privacy-aware multi-agent information sharing games for cybersecurity*. In 2018 International Conference on Computing, Networking and Communications, ICNC 2018, Maui, HI, USA, March 5-8, 2018, pages 425–431, 2018.

14. J. Gu, B. Sun, X. Du, J. Wang, Y. Zhuang, and Z. Wang. *Consortium blockchain-based malware detection in mobile devices*. IEEE Access, 6:12118–12128, 2018.

15. Bin Yu, Joseph K. Liu, Amin Sakzad, Surya Nepal, Ron Steinfeld, Paul Rimba, and Man Ho Au. *Platform-independent secure blockchain-based voting system*. In Information Security - 21st International Conference, ISC 2018, Guildford, UK, September 9-12, 2018, Proceedings, pages 369–386, 2018.

16. Muneeb Ali, Jude Nelson, Ryan Shea, and Michael J. Freedman. *Blockstack: A global naming and storage system secured by blockchains*. In Proceedings of the 2016 USENIX Conference on Usenix Annual Technical Conference, USENIX ATC '16, pages 181–194, 2016.

17. Marcus Pendleton. *System Call Anomaly Detection in Multi-threaded Programs*, PhD Thesis, Nov. 2017

18. G. Fernandez. *Deep Learning Approaches for Network Intrusion Detection*, MS Thesis, UTSA, May 2019.

19. Richard Garcia Lebron. *A Framework for Characterizing Cyber Attack Reconnaissance Behaviors*, PhD Dissertation, July 2019.

20. L. Xu. *Detecting and Characterizing Malicious Websites*. PhD Thesis, 2014.

21. Z. Zhan. *A Statistical Framework for Analyzing Cyber Threats*. PhD Thesis, 2014.

22. Deqing Zou, Sujuan Wang, Shouhuai Xu, Zhen Li, and Hai Jin. *μVulDeePecker: A Deep Learning-Based System for Multiclass Vulnerability Detection*. CoRR abs/2001.02334 (2020)

23. Zhen Li, Deqing Zou, Shouhuai Xu, Zhaoxuan Chen, Yawei Zhu, and Hai Jin. *VulDeeLocator: A Deep Learning-based Fine-grained Vulnerability Detector*. CoRR abs/2001.02350 (2020)

24. Richard Garcia-Lebron, David J. Myers, Shouhuai Xu, and Jie Sun. *Node diversification in complex networks by decentralized colouring*. J. Complex Networks 7(4): 554-563 (2019)

25. Deqiang Li, Qianmu Li, Yanfang Ye, and Shouhuai Xu. *Enhancing Robustness of Deep Neural Networks Against Adversarial Malware Samples: Principles, Framework, and AICS'2019 Challenge*. CoRR abs/1812.08108 (2018)

26. Yujie Fan, Yiming Zhang, Shifu Hou, Lingwei Chen, Yanfang Ye, Chuan Shi, Liang Zhao, and Shouhuai Xu. *iDev: Enhancing Social Coding Security by Cross-platform User Identification Between GitHub and Stack Overflow*. IJCAI 2019: 2272-2278

27. Gabriel C. Fernandez and Shouhuai Xu. *A Case Study on using Deep Learning for Network Intrusion Detection*. MILCOM 2019: 1-6

28. Eric Ficke, Kristin M. Schweitzer, Raymond M. Bateman, and Shouhuai Xu. *Analyzing Root Causes of Intrusion Detection False-Negatives: Methodology and Case Study*. MILCOM 2019: 1-6

29. Jose David Mireles, Eric Ficke, Jin-Hee Cho, Patrick M. Hurley, and Shouhuai Xu. *Metrics Towards Measuring Cyber Agility*. CoRR abs/1906.05395 (2019)

30. Maochao Xu, Kristin M. Schweitzer, Raymond M. Bateman, and Shouhuai Xu. *Modeling and Predicting Cyber Hacking Breaches*. IEEE Trans. Information Forensics and Security 13(11): 2856-2871 (2018)

31. Moustafa Saleh, Tao Li, and Shouhuai Xu. *Multi-context features for detecting malicious programs*. J. Comput. Virol. Hacking Tech. 14(2): 181-193 (2018)

32. Yanfang Ye, Shifu Hou, Lingwei Chen, Xin Li, Liang Zhao, Shouhuai Xu, Jiabin Wang, and Qi Xiong. *ICSD: An Automatic System for Insecure Code Snippet Detection in Stack Overflow over Heterogeneous Information Network*. ACSAC 2018: 542-552

33. Richard B. Garcia-Lebron, Kristin M. Schweitzer, Raymond M. Bateman, and Shouhuai Xu. *A Framework for Characterizing the Evolution of Cyber Attacker-Victim Relation Graphs*. MILCOM 2018: 70-75

34. Eric Ficke, Kristin M. Schweitzer, Raymond M. Bateman, and Shouhuai Xu. *Characterizing the Effectiveness of Network-Based Intrusion Detection Systems*. MILCOM 2018: 76-81

35. Zhen Li, Deqing Zou, Shouhuai Xu, Xinyu Ou, Hai Jin, Sujuan Wang, Zhijun Deng, and Yuyi Zhong. *VulDeePecker: A Deep Learning-Based System for Vulnerability Detection*. NDSS 2018

36. Maochao Xu, Lei Hua, and Shouhuai Xu. *A Vine Copula Model for Predicting the Effectiveness of Cyber Defense Early-Warning*. Technometrics 59(4): 508-520 (2017)

37. Moustafa Saleh, E. Paul Ratazzi, and Shouhuai Xu. *A control flow graph-based signature for packer identification*. MILCOM 2017: 683-688

38. Zhen Li, Deqing Zou, Shouhuai Xu, Hai Jin, Hanchao Qi, and Jie Hu. *VulPecker: an automated vulnerability detection system based on code similarity analysis*. ACSAC 2016: 201-213

39. Jose David Mireles, Jin-Hee Cho, and Shouhuai Xu. *Extracting attack narratives from traffic datasets*. CyCon U.S. 2016: 118-123

40. Zhenxin Zhan, Maochao Xu, and Shouhuai Xu. *Predicting Cyber Attack Rates With Extreme Values*. IEEE Trans. Information Forensics and Security 10(8): 1666-1677 (2015)

41. Shouhuai Xu, Wenlian Lu, Li Xu, and Zhenxin Zhan. *Adaptive Epidemic Dynamics in Networks: Thresholds and Control*. ACM Trans. Auton. Adapt. Syst. 8(4): 19:1-19:19 (2014)

42. Li Xu, Zhenxin Zhan, Shouhuai Xu, and Keying Ye. *An evasion and counter-evasion study in malicious websites detection*. CNS 2014: 265-273

43. Zhenxin Zhan, Maochao Xu, and Shouhuai Xu. *A Characterization of Cybersecurity Posture from Network Telescope Data*. INTRUST 2014: 105-126

44. Zhenxin Zhan, Maochao Xu, and Shouhuai Xu. *Characterizing Honeypot-Captured Cyber Attacks: Statistical Framework and Case Study*. IEEE Trans. Information Forensics and Security 8(11): 1775-1789 (2013)

45. Jose Andre Morales, Michael Main, Weiliang Luo, Shouhuai Xu, and Ravi S. Sandhu. *Building malware infection trees*. MALWARE 2011: 50-57

46. Erhan J. Kartaltepe, Jose Andre Morales, Shouhuai Xu, and Ravi S. Sandhu. *Social Network-Based Botnet Command-and-Control: Emerging Threats and Countermeasures*. ACNS 2010: 511-528

47. Jose Andre Morales, Ravi S. Sandhu, and Shouhuai Xu. *Evaluating detection and treatment effectiveness of commercial anti-malware programs*. MALWARE 2010: 31-38

48. Li Xu, Zhenxin Zhan, Shouhuai Xu, and Keying Ye. *Cross-layer detection of malicious websites*. CODASPY 2013: 141-152

49. Yujuan Han, Wenlian Lu, and Shouhuai Xu. *Preventive and Reactive Cyber Defense Dynamics with Ergodic Time-dependent Parameters Is Globally Attractive*. CoRR abs/2001.07958 (2020)

50. Zongzong Lin, Wenlian Lu, and Shouhuai Xu. *Unified Preventive and Reactive Cyber Defense Dynamics Is Still Globally Convergent*. IEEE/ACM Trans. Netw. 27(3): 1098-1111 (2019)

51. Ren Zheng, Wenlian Lu, and Shouhuai Xu. *Preventive and Reactive Cyber Defense Dynamics Is Globally Stable*. IEEE Trans. Network Science and Engineering 5(2): 156-170 (2018)

52. Maochao Xu, Gaofeng Da, and Shouhuai Xu. *Cyber Epidemic Models with Dependences*. Internet Mathematics 11(1): 62-92 (2015)

53. Shouhuai Xu, Wenlian Lu, and Hualun Li. *A Stochastic Model of Active Cyber Defense Dynamics*. Internet Mathematics 11(1): 23-61 (2015)

54. Ren Zheng, Wenlian Lu, and Shouhuai Xu. *Active cyber defense dynamics exhibiting rich phenomena*. HotSoS 2015: 2:1-2:12

55. Gaofeng Da, Maochao Xu, and Shouhuai Xu. *A new approach to modeling and analyzing security of networked systems*. HotSoS 2014: 6

56. Wenlian Lu, Shouhuai Xu, and Xinlei Yi. *Optimizing Active Cyber Defense*. GameSec 2013: 206-225

57. Maochao Xu and Shouhuai Xu. *An Extended Stochastic Model for Quantitative Security Analysis of Networked Systems*. Internet Mathematics 8(3): 288-320 (2012)

58. Shouhuai Xu, Wenlian Lu, and Li Xu. *Push- and pull-based epidemic spreading in networks: Thresholds and deeper insights*. ACM Trans. Auton. Adapt. Syst. 7(3): 32:1-32:26 (2012)

59. Shouhuai Xu, Wenlian Lu, and Zhenxin Zhan. *A Stochastic Model of Multivirus Dynamics*. IEEE Trans. Dependable Sec. Comput. 9(1): 30-45 (2012)

60. Xiaohu Li, Timothy Paul Parker, and Shouhuai Xu. *A Stochastic Model for Quantitative Security Analyses of Networked Systems*. IEEE Trans. Dependable Sec. Comput. 8(1): 28-43 (2011)

61. Jose David Mireles, Eric Ficke, Jin-Hee Cho, Patrick M. Hurley, and Shouhuai Xu. *Metrics Towards Measuring Cyber Agility*. IEEE Trans. Information Forensics and Security 14(12): 3217-3232 (2019)

62. J. Cho, S. Xu, P. Hurley, M. Mackay, T. Benjamin, and M. Beaumont. *STRAM: Measuring the Trustworthiness of Computer-based Systems*. ACM Computing Survey, 2019.

63. Pang Du, Zheyuan Sun, Huashan Chen, Jin-Hee Cho, and Shouhuai Xu. *Statistical Estimation of Malware Detection Metrics in the Absence of Ground Truth*. IEEE Trans. Information Forensics and Security 13(12): 2965-2980 (2018)

64. Huashan Chen, Jin-Hee Cho, and Shouhuai Xu. *Quantifying the security effectiveness of firewalls and DMZs*. HotSoS 2018: 9:1-9:11

65. Huashan Chen, Jin-Hee Cho, and Shouhuai Xu. *Quantifying the security effectiveness of network diversity: poster*. HotSoS 2018: 24:1

66. John Charlton, Pang Du, Jin-Hee Cho, and Shouhuai Xu. *Measuring Relative Accuracy of Malware Detectors in the Absence of Ground Truth*. MILCOM 2018: 450-455

67. Marcus Pendleton, Richard Garcia-Lebron, Jin-Hee Cho, and Shouhuai Xu. *A Survey on Systems Security Metrics*. ACM Comput. Surv. 49(4): 62:1-62:35 (2017)

68. Jin-Hee Cho, Patrick M. Hurley, and Shouhuai Xu. *Metrics and measurement of trustworthy systems*. MILCOM 2016: 1237-1242

69. Yujuan Han, Wenlian Lu, Shouhuai Xu. *Characterizing the power of moving target defense via cyber epidemic dynamics*. HotSoS 2014: 10

70. S. Xu. *Cybersecurity Dynamics: A Foundation for the Science of Cybersecurity*. A book chapter in Springer book entitled Proactive and Dynamic Network Defense, 2019.

71. Shouhuai Xu. *Emergent behavior in cybersecurity*. HotSoS 2014: 13

72. Shouhuai Xu. *Cybersecurity dynamics*. HotSoS 2014: 14

73. Weiqi Dai, Hai Jin, Deqing Zou, Shouhuai Xu, Weide Zheng, and Lei Shi. *TEE: a virtual DRTM based execution environment for secure cloud-end computing*. ACM Conference on Computer and Communications Security 2010: 663-665

74. Qingji Zheng and Shouhuai Xu. *Verifiable Delegated Set Intersection Operations on Outsourced Encrypted Data*. IC2E 2015: 175-184

75. Qingji Zheng, Shouhuai Xu, and Giuseppe Ateniese. *VABKS: Verifiable attribute-based keyword search over outsourced encrypted data*. INFOCOM 2014: 522-530

76. Weiqi Dai, T. Paul Parker, Hai Jin, and Shouhuai Xu. *Enhancing Data Trustworthiness via Assured Digital Signing*. IEEE Trans. Dependable Sec. Comput. 9(6): 838-851 (2012)

77. Qingji Zheng, Shouhuai Xu, and Giuseppe Ateniese. *Efficient query integrity for outsourced dynamic databases*. CCSW 2012: 71-82

78. Qingji Zheng and Shouhuai Xu. *Secure and efficient proof of storage with deduplication*. CODASPY 2012: 1-12

79. Shouhuai Xu, Xiaohu Li, Timothy Paul Parker, and Xueping Wang. *Exploiting Trust-Based Social Networks for Distributed Protection of Sensitive Data*. IEEE Trans. Information Forensics and Security 6(1): 39-52 (2011)

80. Qingji Zheng and Shouhuai Xu. *Fair and dynamic proofs of retrievability*. CODASPY 2011: 237-248

81. Xuhua Ding, Gene Tsudik, and Shouhuai Xu. *Leak-free mediated group signatures*. Journal of Computer Security 17(4): 489-514 (2009)

82. Justin Leonard, Shouhuai Xu, and Ravi S. Sandhu. *A First Step towards Characterizing Stealthy Botnets*. ARES 2009: 106-113

83. Justin Leonard, Shouhuai Xu, and Ravi S. Sandhu. *A Framework for Understanding Botnets*. ARES 2009: 917-922

84. Shouhuai Xu, Xiaohu Li, and T. Paul Parker. *Exploiting social networks for threshold signing: attack-resilience vs. availability*. AsiaCCS 2008: 325-336

85. Shouhuai Xu. *On the security of group communication schemes*. Journal of Computer Security 15(1): 129-169 (2007)

86. Keith Harrison and Shouhuai Xu. *Protecting Cryptographic Keys from Memory Disclosure Attacks*. DSN 2007: 137-143

87. Shouhuai Xu. *On the security of group communication schemes based on symmetric key cryptosystems*. SASN 2005: 22-31

88. Xuhua Ding, Gene Tsudik, and Shouhuai Xu. *Leak-Free Group Signatures with Immediate Revocation*. ICDCS 2004: 608-615

89. Gene Tsudik and Shouhuai Xu. *Accumulating Composites and Improved Group Signing*. ASIACRYPT 2003: 269-286

APPENDIX A – PUBLICATIONS AND PRESENTATIONS

- Songlin He, Eric Ficke, Mir Mehedi Pritom, Huashan Chen, Qiang Tang, Qian Chen, Marcus Pendleton, Laurent Njilla, and Shouhuai Xu. *Method and System for Blockchain-Based Cyber Security Management*. To be submitted for patent application and paper publication. March 2020. (This is the full version of the present report)
- Huashan Chen, Marcus Pendleton, Laurent Njilla, and Shouhuai Xu. *A Survey on Ethereum Systems Security: Vulnerabilities, Attacks and Defenses*. ACM Computing Survey. Accepted for publication in March 2020.
- Shouhuai Xu. B2CSM: *Blockchain-Based Cyber Security Management: Design, Analysis, and Prototype Implementation*. Project Kick-off Meeting presentation. 2019.
- Shouhuai Xu. B2CSM: *Blockchain-Based Cyber Security Management: Design, Analysis, and Prototype Implementation*. Project Final Briefing presentation and **demo**, AFRL/Rome, January 27, 2020.

LIST OF SYMBOLS, ABBREVIATIONS, AND ACRONYMS

A-CSM	Application-centric Cyber Security Management
AGTSR	Annotated Graph Time Series Representation
AI/ML	Artificial Intelligence/Machine Learning
APT	Advanced Persistent Threat
AQL	Application Query Latency
B2CSM	Blockchain-Based Cyber Security Management
BFT	Byzantine Fault-Tolerance
CA	Certificate Authority
CFT	Crash Fault-Tolerance
CPS	Cyber-Physical System
CSM	Cyber Security Management
CSMA	Cyber Security Management App
DMZ	DeMilitarized Zone
DoS	Denial-of-Service
DRT	Data Replication Throughput
HIDS	Host-based Intrusion Detection System
IoT	Internet-of-Things
IP	Internet Protocol
LAN	Local Area Network
MSP	Membership Service Providers
N-CSM	Network-centric Cyber Security Management
NIDS	Network-based Intrusion Detection System
OSN	Ordering Service Nodes
T-CSM	Tools-centric Cyber Security Management
TCP/UDP	Transmission Control Protocol/ User Datagram Protocol
URL	Uniform Resource Locator
USMA	US Military Academy
VM	Virtual Machine
WAN	Wide Area Network