

EXTRUST: Reducing Exploit Stockpiles with a Privacy-Preserving Depletion System for Inter-State Relationships

Thomas Reinhold*, Philipp Kuehn*, Daniel Günther†, Thomas Schneider† and Christian Reuter*

*Science and Technology for Peace and Security (PEASEC)

Technical University of Darmstadt, Germany

†Cryptography and Privacy Engineering Group (ENCRYPTO)

Technical University of Darmstadt, Germany

Abstract—Cyberspace is a fragile construct threatened by malicious cyber operations of different actors, with vulnerabilities in IT hardware and software forming the basis for such activities, thus also posing a threat to global IT security. Advancements in the field of artificial intelligence accelerate this development, either with artificial intelligence enabled cyber weapons, automated cyber defense measures, or artificial intelligence-based threat and vulnerability detection. Especially state actors, with their long-term strategic security interests, often stockpile such knowledge of vulnerabilities and exploits to enable their military or intelligence service cyberspace operations. While treaties and regulations to limit these developments and to enhance global IT security by disclosing vulnerabilities are currently being discussed on the international level, these efforts are hindered by state concerns about the disclosure of unique knowledge and about giving up tactical advantages. This leads to a situation where multiple states are likely to stockpile at least some identical exploits, with technical measures to enable a depletion process for these stockpiles that preserve state secrecy interests and consider the special constraints of interacting states as well as the requirements within such environments being non-existent. This paper proposes such a privacy-preserving approach that allows multiple state parties to privately compare their stock of vulnerabilities and exploits to check for items that occur in multiple stockpiles without revealing them so that their disclosure can be considered. We call our system EXTRUST and show that it is scalable and can withstand several attack scenarios. Beyond the intergovernmental setting, EXTRUST can also be used for other zero-trust use cases, such as bug-bounty programs.

Index Terms—Exploit, Vulnerability, Arms Control, Cyberspace, Blockchain, Multi-Party Computation

I. INTRODUCTION

The threat of malicious cyber activities is omnipresent and state actors are becoming an increasingly important part of this development [1], [2], either due to the progressing militarization of cyberspace [3], [4] or due to cyber espionage operations [5], [6]. At the same time, advancements in the field of artificial intelligence (AI) are being used to automate cyber defence measures [7], to develop AI enabled cyber weapons [8], or to detect and predict software threats and vulnerabilities [9], [10]. In particular, knowledge of vulnerabilities is an integral part in most of these cyber operations to breach foreign IT-protection measures, and intelligence services and military forces stockpile such critical information without disclosing it for rectification [11], [12]. However, any

serious and capable exploit withheld by a state for its own purposes becomes a potential threat for everyone, including the state itself, its economy, and civil society [13] as the exploit *EternalBlue* exemplified in 2017 [14], [15].

One way out of this dilemma is a so-called vulnerability equity process (VEP) [16], an institutionalized measure to regularly assess the criticality of stockpiled exploits and vulnerabilities to (re)consider their disclosure that could take place under the leadership of extra-national entities, such as the United Nations (UN) [17]. A major obstacle for such an approach is the reluctance of participating parties to disclose sensitive information about their own capabilities, which is generally seen as giving up tactical advantages, effectively resulting in an international arms race for offensive cyber capabilities [18]. Historically, such situations have been countered by efforts to reach mutual agreements between states on arms control and reduction measures, *i.e.*, treaty-based agreements to limit the risks of proliferation of weapon-enabled technology, to prevent its use with potentially disastrous consequences, or to reduce the risks of conflict arising by mistake or technological failures [19].

With regard to the depicted development in cyberspace, early political approaches to mitigate these threats have been proposed by the UN [20], the OSCE [21], and other organizations. But although first important steps towards an effective cyber arms control, like the exchange of threat information [22], [23], have been established, they are not suitable for limiting or reverting the aforementioned international cyber arms race of vulnerability stockpiling. So far, no proposal focuses on this specific challenge and the particular constraints of state actors, with their requirements of confidentiality, their potential mutual mistrust, and individual security concerns [24].

In this paper, we propose a technical solution called EXTRUST based on a multi-party computation approach that allows multiple actors to compare vulnerability stockpiles for matching entries while preserving their confidentiality. This includes an approach for the unique machine-readable identification of exploits that allows them to be checked for matches. Our solution is designed for a zero-trust environment and does not rely on any preconditions of trust in

advance or assumptions of good nature. This contributes to the development of measures for an international agreement to deplete vulnerabilities while circumventing the problems and impediments of intergovernmental cooperation.

Beside this contribution, this paper further aims to provide an example of how politics is – sometimes – in need of technical solutions, in this case even for challenges regarding international security. As computer scientists and engineers are the experts on the domain of cyberspace, shaping it by developing software or even defining its constraints and rules themselves, we would like to encourage taking the responsibility that this entails seriously and support the peaceful development of this globally shared domain.

The paper is structured as follows: Subsequent to this introduction, Section II presents related work and elaborates the research gap. Section III analyzes the requirements of EXTRUST both on a conceptual and IT security level. Section IV discusses how vulnerabilities can be uniquely described in a machine-readable form that allows their comparison. Section V presents a Blockchain-based prototype approach that exemplifies the intended system and its requirements and discusses the challenges for a applicable EXTRUST implementation. Section VI presents our contribution of a privacy-preserving exploit depletion system for zero-trust relationships using multi-party computation. Section VII discusses the approach and evaluates it against the requirements. It also presents different application scenarios beyond state actors. Section VIII concludes this paper and provides directions for future work. In order to maintain readability, the technical details can be found in the Annex in Section VIII.

II. RELATED WORK

Since our paper covers and combines different computer science topics, this section summarizes the existing work on malware identification (Section II-A), vulnerability mitigation (Section II-B), and promising cryptographic protocols (Section II-C). Based on these descriptions, the research gap is described (Section II-D), which is closed by our approach.

A. Vulnerability Terminology and Malware Identification Methods

An important prerequisite for comparing exploits – as the core of a depletion system – is the ability to create deterministic vulnerability descriptions. Early attempts were based on the creation of so-called malware *signatures* [25], which function like a *fingerprint*. Current malware detection approaches use a different approach that is either based on the entire binary code of the malware, *i.e.*, the exploit and payload [26], the compromised storage to create signatures [27] or even artificial intelligence measures to automatically generate descriptions via a Common Vulnerability Scoring System (CVSS) prediction [28]. Beside the actual detection of malware, other research area focuses on the description and identification of exploited vulnerabilities. The popular national vulnerability database (NVD) provides a semi-structured database of known vulnerabilities [29], however, Dong *et al.* [30] showed that the

NVD entries are inconsistent compared to other vulnerability databases. Compared to the common vulnerabilities and exposures (CVE), the NVD entries differ in their announced project names or versions. Alternative approaches were introduced by Sadique *et al.* [31] with the *Structured Threat Information eXpression (STIX)* and the *Vocabulary for Event Recording and Incident Sharing Framework (VERIS)* [32] that can be used to describe, share, and publish threat information. Both definitions, STIX and VERIS, offer a syntax for different types of threats, including malware, exploits, and vulnerabilities. Some entry fields in NVD, STIX, and VERIS may contain unstructured information that undermines unique descriptions. Martin *et al.* [33] propose the common weakness enumeration (CWE), a dictionary of weakness classes that can be used to classify vulnerabilities, an approach we use in Section IV to identify vulnerabilities.

B. Vulnerability Mitigation & External Depletion Measures

Vulnerability research and mitigation methods have been a topic in IT security for several decades [34], [35], [36], [37]. One measure are so-called *bug-bounty programs* [38] like e.g. *HackerOne* [39], which aim to attract IT security practitioners to penetrate advertised systems and services and report loopholes in software or services. Other programs are run by Mozilla, Facebook, and Microsoft [40], [41], [42] or *Project Zero* [43] by Google, which focuses on the search for zero-day vulnerabilities. These programs, which we further refer to as *external depletion measures*, aim to identify vulnerabilities in popular IT products to disclose them to the producers and get them fixed as a depletion measure.

In contrast, *internal depletion measures* focus on an actor's secret exploit stockpile of already known, but not yet disclosed vulnerability information. Such measures have not yet been proposed before for the given application context of interstate cooperation and international security.

Practical approaches at this international, intergovernmental level have so far been limited to transparency and confidence-building, rather than arms control and the non-proliferation or disarmament of malicious cyber tools. [19].

C. Cryptographic Protocols

Our EXTRUST system is related to well-studied cryptographic protocols like multi-party computation (cf. Section II-C1), private set intersection (cf. Section II-C2), and trusted hardware (cf. Section II-C3). These approaches are further elaborated in the following.

1) *Multi-Party Computation (MPC)*: The first approaches to multi-party computation (MPC) of functions represented as a Boolean circuit were proposed by Yao [44] for $N = 2$ parties with constant round complexity, and by Goldreich, Micali, and Wigderson (GMW) [45] for any number of parties N with round complexity linear in the depth of the Boolean circuit. Beaver, Micali, and Rogaway (BMR) [46] extended Yao's protocol to the multi-party case while maintaining the linear round complexity. Based on this initial work, many research projects followed, showing the practical feasibility of MPC for many privacy-preserving applications, such as auctions [47],

set intersection [48], and machine learning [49]. Kamara *et al.* presented an outsourcing technique [50], which allows N parties to outsource the MPC protocol to $n \ll N$ parties.

2) *Private Set Intersection (PSI)*: Private Set Intersection (PSI) has been proposed to identify malware (cf. Section II-A) in a single client and server environment [51]. A recent survey and performance comparison of different PSI protocols by Pinkas *et al.* [48] demonstrates that the approach proposed by Pinkas, Rosulek, Trieu and Yanai [52] is currently the fastest PSI protocol which can handle malicious security. In our proposed application context, we have multiple parties, hence we are mainly interested in multi-party PSI. Multi-party PSI protocols with passive security are applied by Kolesnikov *et al.* [53] and Inbar *et al.* [54]. A scalable, maliciously-secure multi-party PSI protocol is presented by Hazay and Venkatasubramanian [55]. Huang *et al.* [56] use a general MPC framework to privately compute the set intersection between two parties.

3) *Trusted Execution Environment (TEE)*: Another promising approach for a privacy-preserving exploit depletion system is to securely isolate the execution into a *trusted execution environment* (TEE) [57], that allows untrusted data to be computed in a secure environment that is isolated from all other executions running on the same machine, where it is protected against manipulation and disclosure. TEEs are omnipresent in all Intel processors from the 6th generation upwards as *Intel Software Guard Extension* (SGX). Although many works use Intel SGX for efficient secure multi-party computation [58], [59], [60], [61], [62], TEEs are not suitable for applications when states are involved, since this would require that state actors trust the hardware-producing countries not to manipulate the TEEs, *e.g.*, by including backdoors.

D. Research Gap

Above all, practical measures are a mandatory aspect of potential arms control and disarmament treaties, as history and insights into former weaponized technologies have shown [63]. Existing IT methods such as Multi-PSI [55] (cf. Section II-C2) and secure hardware [62] (cf. Section II-C3) have not been applied to exploit depletion, especially regarding the demands and particular constraints of an interstate zero-trust environment. Such a protocol for pairwise PSI among N parties, as required for a privacy-preserving exploit depletion system, is currently not available. Thus, our approach EXTRUST proposes a Boolean circuit that implements the desired functionality via MPC (cf. Section VI).

III. REQUIREMENTS ANALYSIS

In this section, the requirements of EXTRUST are analyzed as a system for reducing exploit stockpiles, resulting from the chosen context of interstate relations. This list is divided into conceptual requirements derived from the specific constraints of the context of arms control, as well as the IT security requirements in combination with the selection of the adversary model.

A. Conceptual Requirements

As mentioned above, this paper focuses on cases in which two or more parties stockpile vulnerabilities and exploits. This reflects the character of arms control treaties, whose “practical” part of active mutual control or (limited) cooperation measures are always based on bi- or multilateral agreements [64] between a small group of states. Based on a rational choice consideration [65], our approach builds upon the following two premises, that we consider to be reflected by states that stockpile vulnerabilities [66], as they resemble the considerations behind a vulnerability equity process [16]. Firstly, we consider states to be aware, that withholding a vulnerability poses a potential threat to their own IT systems. Secondly, we consider that a vulnerability which is known to more than one state is more likely to be considered a candidate for disclosure, because its intended effect is probably ineffective or at least uncertain and because disclosing the vulnerability results in publicly available security patches that support the state’s own IT security and also renders the vulnerability worthless for everyone else.

On the other hand, all vulnerabilities are high-value assets for the stockpiling party. Given the context of state interaction, each party will try to avoid revealing any information that can lead to the loss of tactical advantages, while trying to extend these advantages by gaining information about the other parties. In addition, arms control measures are established in times of political tensions to avoid the outbreak of armed conflict. Based on these assumptions, we consider that EXTRUST has to operate in a zero-trust environment in which parties have to be incentivized to cooperate, while at the same time assuming that other parties are either extremely reluctant to disclose information, attempt to gain information for their own interest, or are otherwise dishonest regarding their cooperation and activities.

With these considerations in mind, EXTRUST aims to require as little cooperation as possible due to this zero-trust environment. This means that each party discloses only the absolutely necessary amount of information, thereby retaining all specific information about capacities and capabilities. Additionally, each party should be able to perform its own check for intersections at any time without relying on further cooperation, dedicated data exchange, or any form of superordinate institution. Furthermore, information already provided should not be allowed to be altered, deleted, or corrupted.

In light of this context, the necessary measure needs to fulfil the following conceptual requirements (RC):

- RC1 The measure has to enable parties to add information about vulnerabilities and exploits.
- RC2 Intersection checks have to be able to be performed by either party at any time without having to obtain the consent of the other parties involved. A match is considered as such if at least two different participating parties have submitted identical information about vulnerabilities or exploits.
- RC3 The system has to send feedback when it detects an intersection match.
- RC4 Although real-time computability is not strictly neces-

sary for processes that are usually politically slow, such as arms control measures, the system needs to be scalable with respect to the number of parties so that parties can join or leave at any time. While previous arms control treaties are usually established in a small circle of state actors that participate in mutual control measures, indicating there could be up to $N = 5$ participating parties in a real-world arms control scenario, this should not be the upper bound of our system.

RC5 The system should be operated decentralized and not require a specific neutral authority to operate or maintain the system.

B. Adversary Model

The two most common adversary models are semi-honest (passive) and malicious (active) adversaries [67]. While semi-honest adversaries follow the underlying rules and procedures (in technical terms the so-called *protocol*) and try to extract as much information as possible from the transcript, malicious adversaries may arbitrarily deviate from the agreed rules. Given the zero-trust environment in the context of EXTRUST, we consider an active or malicious attacker as adversary model. Although technical security measures that protect against semi-honest adversaries are more efficient than those against malicious adversaries, we must consider state actors that might maliciously manipulate arms control computations and outcomes. Additionally, we assume a dishonest majority, *i.e.*, up to $N - 1$ parties may be malicious. The motivational scenario of EXTRUST is a highly security critical one in which top secret information may be exchanged. Hence, it should withstand several passive attacks, like eavesdropping, and also be shielded against active attacks, such as flooding or brute-force attacks. We have therefore chosen the model of the stronger adversary in contrast to the passive, semi-honest adversary. This decision also covers the application context of the zero-trust relationship between the actors involved.

C. Technical and Security Requirements

In addition to the conceptual requirements, the approach must meet additional security expectations to provide an applicable and secure measure of exploit depletion in a zero-trust environment. The requirements reflect the need for confidentiality and are important to motivate stakeholders to participate. These technical and security requirements (RS) are:

- RS1 The system must ensure the confidentiality of vulnerability or exploit information against any party.
- RS2 Submitted data should not be able to be withdrawn, modified, or corrupted by any party.
- RS3 The system needs to prevent false positive intersection results.

In the following, after discussing the identification of vulnerabilities as a necessary prerequisite of our system, we present a prototype solution for EXTRUST that addresses these requirements and illustrates its inherent challenges. Afterwards, we present our contribution of a MPC-based EXTRUST.

```
{
  "cpe": "cpe:2.3:o:tp-link:wdr7400_firmware:-:*:*:*
  ↪ :*:~::~*",
  "cwe": 120,
  "fun": "copy_msg_element"
}
```

Listing 1. Vulnerability Identifier for CVE-2020-28877

IV. IDENTIFIER OF VULNERABILITIES

In this section, we propose a unique, machine-readable identification method for vulnerabilities to be able to match them. The mathematical description of the required properties and the associated challenges can be found in the Annex and are referenced here.

A. Machine-Readable Vulnerability Identifier

At its core, EXTRUST privately matches vulnerabilities or exploits of different parties. This requires using a vulnerability description method that results in the same machine-readable descriptions for the same vulnerability¹. An established approach to describe and thus identify vulnerabilities is provided by vulnerability databases like the NVD. The NVD's entries, for example, contain information used for identification. Their semi-structured format, however, makes it practically impossible for individuals to independently create the same identifier for a vulnerability. Therefore, we use the approach of Kuehn *et al.* [68] to achieve uniqueness, *i.e.*, we adjust the NVD's entry information by removing any free-form pairs and pairs that provide no information about the vulnerability itself (*e.g.*, the CVE-ID), align the structured information with the vulnerability descriptions, and add information about the vulnerable function, extracted from the vulnerability description.

The remaining fields are CWE and common platform enumeration (CPE) with the addition of the vulnerable function, which are structured and algorithmically comparable. The CWE [69] defines hierarchical layers of vulnerability weakness classes, while the CPE [70] provides a machine-readable way to describe platforms. If a vulnerability affects multiple platforms, we use separate vulnerabilities for each affected platform. The resulting vulnerability identifier is depicted in Listing 1 (for CVE-2020-28877).

B. Analysis

Using a simple object notation for the vulnerability identifier offers flexibility and extensibility, and by adding CPE and CWE as well as the vulnerable function as core elements, identifiers can be specific enough to create matching values when different actors describe and submit the same vulnerability or exploit. This is essential to identify matching vulnerabilities.

The main limitation of the vulnerability identifier's definition is based on a trade-off between the properties *accuracy* and *ambiguity*. Currently, it is still possible to describe two

¹See the Annex section "Required properties for a machine-readable vulnerability identifier".

different vulnerabilities with the same identifier, or to use two different identifiers for the same vulnerability². This leads to false positives (two different vulnerabilities are mapped to one identifier) or false negatives (the same vulnerability is mapped to two different identifiers), respectively, depending on the level of detail implemented into the identifier. However, there are possibilities to adjust the identifier definition accordingly. Increasing the amount of information captured by the identifier makes the identifier more specific but introduces more ambiguity, *i.e.*, false negatives. Parameters to be added are the common vulnerability scoring system (CVSS) parameter information (*e.g.*, impact information) or the vulnerable path (*i.e.*, the filename in which the vulnerability resides) [68]. Another way to adjust the identifier is the CWE's hierarchy depth. CWE classes are hierarchically ordered and thus offer generalization or specification. Including relations of the used CWE class increases the specificity of the identifier and could help to circumvent cases where identifiers use different CWE subclasses of the same top level class. At this point, we want to stress that in the presented scenario (cyber arms control) false positives must be avoided, while false negatives are tolerable. If false positives are a common problem in such a system, it would drastically lose acceptance among states that are still interested in stockpiling vulnerabilities.

The size of the proposed identifier space is restricted by the number of CWE classes, the size of the CPE directory, and the possible function names, which serve as secret information. Individually, these spaces can be approximated in their size. For the space of possible function names FN , we assume a clean coding style, *i.e.*, function names are descriptive and use at most three English words with any kind of connector (*e.g.*, camel case or underscores), which results in $\approx 2^{81}$ identifiers³.

As argued, the presented approach is sufficient to describe vulnerabilities uniquely. It serves our needs with a trade-off in detail that avoids both different vulnerabilities being described by the same identifier as well as the same vulnerability being described with different identifiers. Based on the current limitation of the identifier space, brute-force attacks remain a problem and efforts should be made to increase the identifier space. As an alternative to the proposed definition of identifiers, our system EXTRUST can work with any other scheme that is concise, structured, and unambiguous.

V. EXTRUST USING BLOCKCHAIN

To illustrate the challenges involved in implementing a privacy-preserving exploit depletion system, we have chosen a simple, straightforward prototype based on a Blockchain implementation, referred to hereafter as *BC-based EXTRUST*. Although this approach entails security flaws from a theoretical perspective, we want to use this prototype to illustrate, test, and analyze possible solutions regarding the requirements and the proposed depletion process, as an introduction for our multi-party computation-based approach presented in Section VI. This section presents the architecture and proof-of-concept implementation of this prototype and concludes with

a discussion of the requirements met as well as the identified constraints.

A. System Architecture and Procedure

In terms of conceptual requirements, BC-based EXTRUST should run in a *distributed setting with no central trusted authority*, with a *complete, secured, and tamper-resistant history* of all submitted information and should allow *asynchronous intersection checks* that can be performed by each participating party independently.

We have developed a prototype based on a private Blockchain technology [71], [72] that provides all of these features. A private Blockchain is a distributed chain of *blocks* containing *transactions*, where each block references its previous block via hard-to-calculate mathematical challenges and cryptographic hashes to reference the block. This provides a tamper-proof history of all submissions, as any modification would invalidate the adjacent entries. The data storage part of a Blockchain, the so-called ledger, is replicated to all participants and automatically synced between them. In private networks, access to it is walled by an access control manager (ACM). The interfaces for interaction with the ledger are called *smart contracts*. With regard to the system architecture, the ledger provides the storage space, the smart contract is responsible for the submission and comparison mechanism, and the ACM controls the access as well as the different layers of interaction permissions via roles and associated authorizations. To maintain the confidentiality of the submitted vulnerability identifiers, we secured the information using cryptographic hash functions [73].

The overall procedure begins with the setup of the Blockchain instance (*nodes*) by each participating party and their interconnection to build an evenly distributed network. To submit a vulnerability, the vulnerability identification method we propose in Section IV is used to create an identifier for the specific vulnerability, which is then cryptographically secured using a hash function and finally stored in BC-based EXTRUST. Afterwards, any participating party can perform a transaction, which checks for intersections between all hashes stored in the ledger and logs the output on the ledger. This way, a history of all actions performed is ensured, which is accessible to any involved party, including intersections. Nevertheless, parties that do not know the plaintext vulnerability identifier cannot obtain any information other than the fact that an intersection occurred.

B. Implementation

To focus on developing a proof-of-concept implementation of BC-based EXTRUST, we decided to utilize a private Blockchain framework [74] as it provides all relevant tools for the interaction of the actors with the system, the data structures for storing information, and all necessary data operations for reading, writing, and verifying information within the stored data. We have selected the *Hyperledger Fabric* [75] Blockchain framework because it is open source, actively maintained and well documented, and provides the rapid prototyping

²See the Annex section "Ambiguous vulnerability identifier".

³See the Annex section "Approximation of the vulnerability identifier space".

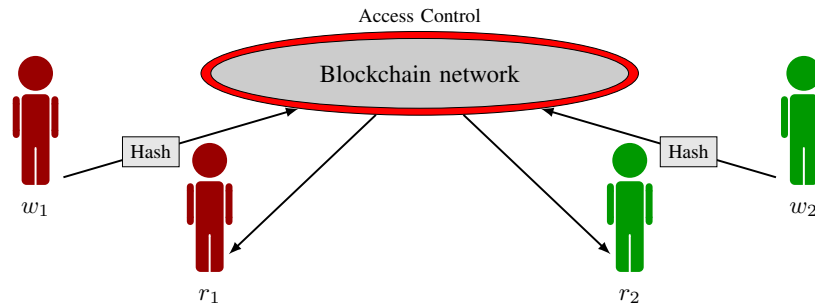


Fig. 1. System architecture for BC-based EXTRUST. r_i and w_i denote readers and writers of actor i .

environment *Hyperledger Composer* [76] with a boilerplate implementation for each part of the Blockchain network.

With regard to the permissions of participants using BC-based EXTRUST, we envision two roles: Readers, who can read the entire ledger and perform the transaction that checks for matching items; and writers, who can only submit items (see Fig. 1). This restriction is only necessary due to the use of our chosen framework⁴, otherwise, a party's submission may be intercepted and copied by other parties. The theoretical concept does not require this separation, because no party would be able to access other parties' information.

The items which are submitted and stored into the ledger are the vulnerability identifiers, as described in (Section IV). As the plain vulnerability identifier must never be inserted into the Blockchain network to prevent its exposure to all parties involved, it is obscured before being submitted. We generate a cryptographic hash of a normalized JSON representation of the vulnerability identifier via SHA3-512 ([77]), following the NIST's policy on hash functions [78]. This provides a 256-bit security level.

To interact with BC-based EXTRUST, the prototype system provides two transactions: The simple submission of hashed vulnerability identifiers and the transaction that checks the stored hashes within the ledger and triggers an event along with references to matching items, `checkIntersections`⁵.

We want to stress that this prototype implementation does not yet take performance into account, as this is no core requirement of EXTRUST and its proposed arms control application.

C. Discussion of BC-based EXTRUST

As indicated earlier, the development of IT measures is a novel approach in the field of technical tools for cyber arms control that has to balance conflicting objectives to a certain extent. For arms control, the aspect of minimum requirements for cooperation between the parties is essential, as it establishes the lowest possible barrier for participation. This is crucial for situations such as the intended one, in which trust cannot be assumed as a given motivation for cooperation. In previous treaties, this often meant a certain

degree of pragmatism regarding the acceptance of "gray areas" and the possibility of non-compliance. The opposite objective is the requirement of technically secure solutions, as this too provides important incentives for participation. This in turn is likely to result in protocol specifications creating operational conditions that potential participants are not prepared to accept.

Considering the requirements, the Blockchain approach provides a manipulation-proof and distributed storage of all submitted information. Calculations are distributed and performed independently, thereby mitigating the need for a trusted third party to maintain the shared information, as well as any other form of cooperation beyond the actual submission. The system can include additional parties without adjustments or significant impact on the performance of the system, beyond the network capacity necessary to synchronize the stored information [79]. In addition, the processing of submissions is not time-critical, which is considered a bottleneck for massive, high-traffic Blockchain applications [80], [81]. By securing vulnerability identifiers, the confidentiality of the information is – at least theoretically – maintained both in submission and in intersection detection.

On the other hand, the Blockchain-based prototype has serious IT security issues, both for active attackers (like non-participating state parties that try to break the system to gain advantages and reveal secret information) and fraudulent, semi-honest state participants that try to gain information which goes beyond the agreed exchange. Notably, the information contained in the distribution of the ledger is vulnerable to brute-force attacks by testing hashes, as foreign countries could generate possible vulnerability descriptions and test them against their local ledger. The PSI literature has demonstrated that private elements cannot be hidden by simple hashing [82], [83]. The probability of creating an existing hash is based on the size of the identifier space and influenced by the number of its properties and values. As the identifier space of BC-based EXTRUST is very small (28 bit, cf. Section IV-B) brute-force attacks are very efficient and can be successfully exploited. In addition, the brute-force attack is completely local since states have a local copy of the whole ledger. Consequently, states would not even notice if a brute-force attack was exploited to find all ledger vulnerabilities. The brute-force attack can be slowed down (but not prevented) by using a difficult to parallelize hash function

⁴As framework, we chose Hyperledger Composer.

⁵See the Annex section "The `checkIntersections` transaction of BC-based EXTRUST".

such as *Argon2* [84]. Extending the identifier space for the vulnerability by more complex identifier descriptions is not an option either, as this increases the probability of describing the same vulnerability differently.

The Blockchain also faces other attack scenarios, such as the so-called 51% attack, which allows attackers to manipulate the ledger [85]. Attackers could also use more subtle ways to create intersections to test foreign submissions by creating *fictional vulnerabilities* for rare software systems based on clever and informed guesses. This could also be used for targeted vulnerability suppression if a participating party creates and submits specific vulnerabilities, intentionally wrongfully signalling its possession to force the vulnerability to be disclosed. In addition, a dishonest state party could clone and resubmit hashes under its own flag, which would also cause BC-based EXTRUST to false signal to the original submitter that this particular vulnerability can be eliminated. However, such a cheat gives the attacker only a slight advantage, as they do not know what the cloned vulnerability information contains, and are likely to attract attention if performed regularly. A final IT security issue concerns passive adversaries that gain access to the ledger, as well as the complete disclosure of the ledger to non-involved third parties. Besides the brute-force attack, the attacker will be able to learn which hashes belong to which party via timing correlations, detecting the amount of different participating actors as well as the amount of submitted hashed items stored by each actor.

The BC-based EXTRUST prototype has shown that it provides the conceptual requirements that arise from the arms control context. Regarding the attack scenarios described, it is important to emphasize that for this application, any attempt to attack or misuse the system is contrary to the principles of the confidence-building aspect of such a mutual measure and its political signal of de-escalation. It is further expected that all parties comply with the defined rules to at least achieve a positive outcome for their own national security. Nevertheless, this expectation needs to rest upon a secure protocol that inherently prevents fraud and guarantees the promised confidentiality.

The following section presents the approach of MPC-based EXTRUST, an arms control measure that provides this level of security.

VI. EXTRUST USING MULTI-PARTY COMPUTATION

This section presents our approach for an MPC-based EXTRUST to develop an exploit depletion system under the conditions of an untrusted environment that fulfils the discussed conceptual and security requirements (Section III), while avoiding the security problems that our prototype revealed (Section V). This approach is based on an interactive *Multi-Party Computation* (MPC) protocol as grounds for our MPC-based EXTRUST architecture. In the following, we present the concept and design of MPC-based EXTRUST technical details can be found in the Annex in section "PSI-variant Boolean circuit for multiple parties".

Secure MPC [86], [87] enables N parties to securely compute a commonly agreed public function f on their respective

secret inputs x_1, \dots, x_N without revealing anything other than the result of the calculated function $f(x_1, \dots, x_N)$. MPC guarantees that each of the N parties will not learn any information (e.g., input from the other parties or intermediate results of the computation) other than what a party would learn in the ideal world with a trusted third party. In the ideal world, all parties send their inputs x_1, \dots, x_N secretly to a trusted third party, which then locally computes the function $f(x_1, \dots, x_N)$ and broadcasts the result to the N parties. In the proposed context of arms control, even if such a trusted third party existed (e.g., in the UN framework), it would probably not be accepted by all state actors or, at the very least, would raise the barrier to participation in the proposed measure (see Section III-A).

In MPC, the function f that shall be computed is represented as a *Boolean circuit*. A Boolean circuit is a logical function whose operations are so-called *Boolean gates*. A Boolean gate takes a set of Boolean inputs (i.e., either 0 or 1) and computes one Boolean output. We represent our MPC-based EXTRUST functionality as a Boolean circuit as efficient cryptographic protocols exist that can securely evaluate Boolean circuits. A Boolean circuit consists of inputs, outputs, and Boolean gates that have two inputs and one output in the Boolean set $\{0, 1\}$. The input of a gate can either be one of the inputs of the Boolean circuit or an output of a previous gate. In MPC, Boolean circuits usually only consist of AND and XOR gates, as any functionality can be realized using these two gate types. A two-input AND gate outputs '1' if both of its inputs are set to '1', while a two-input XOR gate outputs '1' if exactly one (but not both) of its inputs is set to '1'. For the actual algorithm that processes the submitted information and checks for collisions – the so-called *protocol* – we use the BMR protocol [46] and refer to Braun et al. [88] for a detailed protocol description. We further use a well-established outsourcing technique [50] to distribute the information processing for a group of N parties to $n \ll N$ parties. This setting for our MPC approach is shown in Fig. 2. In summary, a subset of n from N (state) parties interactively run an MPC protocol on a Boolean circuit, which computes the functionality of EXTRUST. This setting allows us to evaluate the functionality of EXTRUST in a privacy-preserving manner, while reducing the number of active parties that are fully involved in the computation ensures the scalability of MPC-based EXTRUST.

The protocol requires that the parties have sorted their inputs locally before they are fed into the MPC protocol. To verify this, we use the Boolean circuit and open intermediate values so that the parties can abort the protocol execution if a malicious party has not sorted its inputs correctly. When opening these values, the remaining intermediate values before and after the opening process must be protected to allow further secure computation with them. Many efficient maliciously-secure MPC protocols provide this property, known as *reactive MPC*, e.g., [89], [90], [91], [92], [93]. Apart from checking correctly sorted sets, we use reactive MPC to maintain the state of the secretly shared inputs after the end of a protocol run, so that submitted vulnerabilities do not need to be secretly shared again in the next iteration.

This MPC approach allows us to develop a privacy-

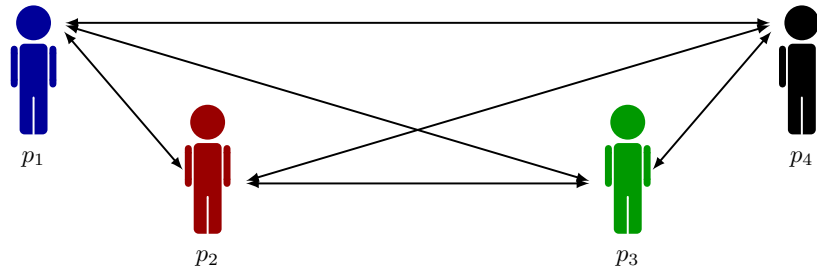


Fig. 2. MPC setting with four participating parties p_1, \dots, p_4 .

preserving exploit depletion system that fulfils the requirements of the arms control context (Section III).

A. System Architecture

Our complete MPC-based EXTRUST architecture works as follows (see Fig.2): N parties try to find intersections of their own identified vulnerabilities between themselves and at least one other actor. These N actors securely evaluate a Boolean circuit (cf. Section II-C1), consisting of AND and XOR gates⁶, that takes as input the known vulnerabilities and exploits of the actors, which are represented as hash values (cf. Section IV), compares them to find intersections, and finally outputs all intersections found to the respective parties. This circuit, however, is not constant over the lifetime of MPC-based EXTRUST as it depends on the number of parties N (states can be added/removed) and inputs u (vulnerabilities can be added). The participating parties perform an initial MPC protocol prior to the actual execution to determine the maximum number of vulnerabilities among all parties, which then determines the number of inputs for the Boolean circuit that is evaluated by the MPC protocol. Now that every party knows the number of inputs to the Boolean circuit, each party in the fixed subset n of the N parties locally compiles the Boolean circuit that is inserted into the MPC protocol, *i.e.*, no further interaction is required by the parties to agree on the Boolean circuit. Malicious-secure MPC protocols ensure that parties, who compiled a fake Boolean circuit that does not compute the agreed functionality, are identified by the other parties. Regarding the already submitted vulnerabilities, a party cannot revoke or modify submitted information because they remain in the input list of the N actors. The parties can opt in and out by sending a notification message to the N servers. Only the inputs of the participating parties that are logged in are taken into account for the computation.

1) Complexity and optimization of the Boolean circuit:

For N parties, state-of-the-art MPC protocols require sending and receiving $\mathcal{O}(N)$ messages for each AND gate in the Boolean circuit [46], while XOR gates can be computed locally without any interaction between the parties [94]. Consequently, we optimize the number of AND gates in our Boolean circuit that is evaluated via MPC. In order to prevent

⁶See the illustrating figure for regarding the section "PSI-variant Boolean circuit for multiple parties" in the Annex.

the concrete set sizes of the individual parties from being leaked, we specify an upper limit u that determines how many inputs a party feeds into the circuit. If a party has fewer than u inputs, it fills the missing inputs with random dummy values, which will not represent any vulnerability and thus will not occur in any intersection as the probability that two parties independently choose the same random dummy values is negligible. On a high level, every party inputs two unique keys – k_0 and k_1 – for each of its vulnerabilities into the Boolean circuit. The Boolean circuit outputs k_1 if this vulnerability is part of an intersection or k_0 if only the respective party knows this vulnerability. Although the resulting keys are leaked to all parties, only the party who input the keys learns any information about the intersecting identifiers of their vulnerability.

2) Using Private Set Intersection to calculate collisions:

To calculate the intersection of different stockpiles, we use the Private Set Intersection (PSI) protocol. PSI allows two parties to securely compute the intersection of their private sets without leaking any information about set elements that are not part of the intersection to the other participating party.

Multi-party PSI [55] extends the PSI functionality to more than two parties, *i.e.*, the parties jointly compute the overall intersection of all their input sets without leaking any information of set elements that are not included in the intersection. Unfortunately, multi-party PSI only outputs the set intersection of *all* input sets. However, in our exploit depletion system we search for intersections between *at least* two sets. A possible solution to this is to implement two-party PSI protocols between each pair of parties. However, this would require a quadratic number of protocol runs in the number of parties. Even more critically, this approach would reveal *which* other party has a common vulnerability. Instead, we use a generic MPC-based approach for our MPC-based EXTRUST application that is based on Huang *et al.*'s [56] Boolean circuit for two-party PSI which we extended into a multiple parties variant.

3) *Instantiation:* There are many MPC frameworks based on secret sharing and/or garbled circuits, *e.g.*, [95], [96], [97], [98], [99], [100], [101], [102]. Table I lists and compares several MPC frameworks with malicious security.

Since untrusted actors deal with highly sensitive information, we need security against malicious parties actively

Framework	# Parties N	Threshold t
ABY ³ [101]	3	1
Sharemind [100]	3	1
ASTRA [99]	3	1
BLAZE [98]	3	1
Trident [97]	4	1
MOTION [88]	≥ 2	1
SCALE-MAMBA [96]	≥ 2	$N - 1$
MP-SPDZ [95]	≥ 2	$N - 1$

TABLE I

COMPARISON OF MPC FRAMEWORKS THAT ARE SECURE AGAINST MALICIOUS ADVERSARIES, COMPUTE ON BOOLEAN CIRCUITS AND ALLOW UP TO t CORRUPTIONS.

# Vulnerabilities \ # States	2	5	10	15
100	2	14	62	146
500	4	31	134	314
1000	7	49	210	492

TABLE II

RUNTIME IN MINUTES OF MPC-BASED EXTRUST FOR VARIOUS NUMBERS OF MAXIMUM VULNERABILITIES AND STATES.

manipulating the computation to either learn more information or prevent other parties from receiving the correct output.

Current MPC frameworks that meet these requirements are MP-SPDZ [95] and SCALE-MAMBA [96]. We recommend the use of *MP-SPDZ*, which implements, among other protocols, the constant-round BMR protocol [46], which has benefits over the multi-round protocols of SCALE-MAMBA in high-latency networks. BMR is secure against malicious parties and a dishonest majority (*i.e.*, up to $N - 1$ parties can be corrupted). If the number of computing servers is fixed to $N = 3$ one can use ABY3 [101], Sharemind [100], ASTRA [99], or BLAZE [98]; if the number is fixed to $N = 4$ parties, Trident [97] can be utilized. The MOTION framework [88] allows MPC among any number of parties N , however, it does not fulfil the full-threshold requirement of $t = N - 1$. Table I shows an overview of the mentioned MPC frameworks.

B. Feasibility of MPC-based EXTRUST implementation

MPC-based EXTRUST completely relies on the security properties of the underlying multi-party computation (MPC) framework. While most MPC frameworks are implemented for academia usage, Bosch developed *Carbyne Stack* an open-source cloud stack for scalable MPC applications [103] that is also suited for real-world usage. As the name suggests, the long-term plan is to make this MPC framework scalable for many participating parties. As this entire project is open-source, a group of states can use their implementation as basis MPC-based EXTRUST.

C. Evaluation of the scalability of MPC-based EXTRUST

In this section, we estimate the feasibility and scalability of our MPC-based EXTRUST. Since we know the complexity of our Boolean circuit, we can estimate the scalability of MPC-based EXTRUST.

In a realistic setting of our proposed application context of arms control, we have the following parameters

for our benchmarks in Table II: number of parties / states $N \in \{2, 5, 10, 15\}$, maximum number of inputs $u \in \{500, 1000, 1500\}$, and length of vulnerability identifier hashes $\sigma = 256$ bit. With these parameters, the size of our Boolean circuit is $\approx 4.8 \cdot 10^7$ ANDs.

To estimate the runtime of our system, we generate a random circuit with the same number of AND gates and two XOR gates per AND gate. Since XOR gates can be evaluated in the BMR protocol without any communication [94], it is less important to determine the exact number of XOR gates, as communication is the bottleneck of MPC.

For malicious MPC with a dishonest majority, as required by our adversary model presented in Section III-B, we use the constant-round BMR protocol [46] using the MASCOT protocol [104] to compute the garbled tables as implemented in the MP-SPDZ framework [95]. To conduct our experiments, we use five servers, each equipped with an Intel Core i9 processor with 2.8 GHz and 128 GB DDR4-RAM. The round-trip network latency in our simulated WAN setting is about 100 ms and the bandwidth 90 Mbit/sec. We take the average runtime of three executions.

The execution time of our circuit is about 31 minutes. This is an acceptable runtime for governmental actors, as the protocol is run daily or weekly. However, the size of the Boolean circuit and the cost of computing each AND gate are quadratic in the number of servers N . Therefore, our scheme will not scale for a large number of parties $N \gg 10$.

We can improve the scalability for these scenarios by outsourcing the computation to $n \ll N$ non-colluding servers [50]. Here, the N parties distribute their input to the n servers, which together run the MPC protocol and distribute the result. An advantage of this method is that all $N \gg n$ parties may be malicious as long as they can trust that the n servers are not colluding. This improves the cost of computing an AND gate to $\mathcal{O}(n^2)$.

VII. DISCUSSION

This section will discuss our approach. As the main contribution of this paper is the MPC-based EXTRUST, the BC-based EXTRUST prototype is not covered here, as it was discussed in Section V-C. In the following, we will analyze our MPC-based EXTRUST regarding the conceptual requirements RC1 - RC5 (Section VII-A) and the security requirements RS1 - RS3 (Section VII-B) necessary to create incentives for states to participate. This section also reviews the scenarios in which EXTRUST can be of use, followed by an outlook on possible future applications (Section VII-C). An overview of which conceptual and security requirements are fulfilled by MPC-based EXTRUST and BC-based EXTRUST, respectively, is provided in Table III. The bracketed checkmarks in the table highlight requirements, that are only fulfilled if we can exclude the 51% attack against Blockchains.

A. Conceptual Requirements

The MPC-based EXTRUST architecture allows participating parties to input information about their known vulnerabilities and exploits without openly revealing sensitive information to other parties (RC1). The output of the computation is

Requirement	BC-based EXTRUST	MPC-based EXTRUST
RC1	(✓)	✓
RC2	✓	✗
RC3	✓	✓
RC4	✓	✓
RC5	✓	✓
RS1	✗	✓
RS2	(✓)	✓
RS3	✓	✓

TABLE III

COMPARISON OF WHICH CONCEPTUAL RC1 - RC5 AND SECURITY REQUIREMENTS RS1 - RS3 ARE FULFILLED BY BC-BASED EXTRUST AND MPC-BASED EXTRUST.

the matching vulnerabilities and exploits between the parties (RC3). Since the output is computed interactively between the parties, MPC-based EXTRUST is entirely decentralized and does not require a trusted third party (RC5).

The solution is theoretically scalable to $N = 10$ parties (RC4). However, the more parties are involved in the protocol, the more inputs and data have to be exchanged between these parties, *i.e.*, the approach has a complexity $\mathcal{O}(N^2)$, but usually $N \approx 10$ (cf. Section III-A). However, as explained in RC4, real time computability is not a critical requirement and longer computation times are no problem for such highly politically organized processes like arms control, which often require days or weeks for the full formal process and the involvement of all necessary stakeholders. In Section VI-C, we propose to outsource [50] the computation to $n \ll N$ parties, which improves the performance of MPC-based EXTRUST. Considering the context of arms control, such a scenario is only applicable and likely if the outsourced computation is performed by neutral institutions that are not involved in the arms control measure itself, since in this way none of the parties involved need to trust that the other participants will not share information outside the protocol. Such delegation is not uncommon for arms control measures. An example is the Joint Comprehensive Plan of Action (JCPOA), a multilateral treaty known as the Iran nuclear deal [105] between Iran, China, France, Russia, the United Kingdom, and Germany. The International Atomic Energy Agency (IAEA) manages and organizes all aspects of this treaty via independent bureaus, entrusted laboratories, UN working groups, and neutral experts for investigation field trips. Regardless, for practical arms control measures as our proposed depletion system, the amount of involved parties usually does not exceed a single-digit number and is often established between a small group of states.

Unfortunately, requirement RC2 is not met because intersection checks now require interaction, as the participating parties are required to exchange data. However, exactly this property of EXTRUST is the key to avoid local brute-force attacks, to which BC-based EXTRUST is vulnerable (cf. Section V). Although, in the context of arms control, the minimum threshold for cooperation to which states must commit provides an incentive to join the measure, this requirement is not mandatory to practically operate EXTRUST. As a privacy-preserving arms control measure is more critical than the desire to independently check for intersections, we consider this

a weak limitation that does not undermine the practical value of our approach, especially when considering that EXTRUST fulfils all other conceptual requirements.

B. Security Requirements

MPC-based EXTRUST fulfils all three security requirements presented in Section III-C. A notable advantage of MPC-based protocols is that the participating parties can only derive information from their own inputs and the outputs received, *i.e.*, the parties do not learn more information in the MPC-based EXTRUST than in EXTRUST with a trusted third party that receives the inputs from all parties and outputs the intersections. This means that no more information is revealed in the protocol transcript than an adversary would learn in the ideal world. In contrast to BC-based EXTRUST from Section V, local attacks (*e.g.*, brute-forcing specific hash values) are not possible in MPC-based EXTRUST. In addition, an adversary is not able to copy vulnerabilities or exploits from other parties to output an invalid intersection because the inputs are inaccessible to the other parties. Thus, requirement RS1 is completely fulfilled.

Once the inputs are submitted in the MPC protocol, the parties are not able to withdraw or modify them (RS2). A situation in which all state actors jointly manipulate the protocol will never happen, since they could otherwise share their vulnerabilities in plain anyway.

False positive intersection (RS3) results are possible with a negligible probability. A false positive is possible if two different vulnerabilities are mapped to the same hash value. Since we use a collision-resistant hash function, the probability of other collision scenarios is negligible. In addition, a false positive may occur if two parties independently choose the same key identifiers. Due to the usage of 256 bit key identifiers and a robust random generator, the probability of this situation is negligible as well.

Above all, the security and confidentiality of the assets to be shared are key incentives for establishing an arms control measure. As MPC-based EXTRUST fulfils all security requirements, it is suitable for a real world application without discouraging states from using it.

C. Further Application Scenarios

Beside the proposed context, EXTRUST can also be useful in other application scenarios, some of which will be discussed in the following.

At present, our approach concentrates exclusively on state actors as addressees. However, organizations or individuals might also be interested in using such a system. As explained in Section II-B, **bug-bounty programs** and **vulnerability research projects** have similar goals: to reduce the spread of vulnerabilities to secure systems. Here, using the aggregated information from the external stockpile depletion measures and integrating it into EXTRUST can increase the speed of detection of matching rediscoveries in stockpiles. This can be achieved by using writers for selected public services or other institutions that intend to contribute to cybersecurity, which feed their hashed vulnerability identifiers into EXTRUST. In

such a setting, the hashing of information is as important as in ExTRUST's motivational scenario to prevent the material from being disseminated for malicious cyber operations. The use of ExTRUST in a purely corporate environment is probably not possible, as organizations like Zerodium [106] are primarily looking for exploits to sell. A similar bug-bounty related approach could focus on examining discovered, potential **zero day vulnerabilities** against other submitted but not yet publicly disclosed vulnerabilities. The history of submissions would allow submitting actors to claim their first-submitted-reward later on, once the information is disclosed. In this way, the first finder could be paid out without the hackers having to reveal their discovery in advance.

VIII. CONCLUSION AND FUTURE WORK

Given the continuous developments in the field of cybersecurity and especially the expected advantages of using artificial intelligence measures to detect, mitigate or even defend against cyber threats, the exclusive knowledge of vulnerabilities is an essential component for state actors to stay ahead of competitors. Under the assumption, that this undermines national as well as international cybersecurity, our paper focused on the depletion of vulnerabilities and exploits that are being stockpiled by state actors. While the disclosure of vulnerabilities at the national level through regulatory processes is becoming more and more of an issue, cooperation on disclosure at the bilateral or multilateral level is still lacking. We discussed that an important obstacle to such measures is the comprehensible restraint of states to give up their accumulated intelligence information in order to compare stockpiles and unnecessarily reveal unique exploits or other secret assets.

To develop a technical measure in such a zero-trust scenario, we identified structural as well as IT security requirements for the detection of intersections in different exploit stockpiles. Based on these, we discussed and designed (i) a novel identification scheme for vulnerabilities and exploits and (ii) an external, privacy-preserving exploit depletion system named ExTRUST. We have identified the requirements for this depletion system for zero-trust relationships and shown that the technical security requirements could hamper the political incentives for states to cooperate. We have illustrated this challenge by developing a prototype for a depletion system based on a Blockchain. The presented MPC-based ExTRUST system handles this dualism by focusing on the IT security of a depletion system while fulfilling most of the conceptual requirements. It stores the detected intersections, while the submitted vulnerabilities are protected by the MPC protocol and thus remain hidden from all involved actors. However, one limitation of this approach is that it is vulnerable to secret agreements by multiple actors, as they could add vulnerabilities and remove them from the intersection – an edge case that is not an option in the proposed arms control context. We also argued that MPC-based ExTRUST is currently not able to fulfil all conceptual requirements, as participating states need to explicitly cooperate and share obfuscated information,

which could be a disadvantage regarding its implementation. Nevertheless, we have shown that the strength of the MPC protocol lies in the fact that an adversary cannot obtain more information from the joint computation than if a trusted third party were to compute the intersections. The ExTRUST system uses a novel exploit identifier and discussed how this identifier could be improved in different scenarios to address the trade-off between the uniqueness and ambiguity of the properties. We believe that this provides a secure measure which fulfils the state's need for secrecy and yet at the same time can contribute to the reduction of vulnerability stockpiles to foster the public IT security through the disclosure of vulnerabilities. We discussed further application scenarios beyond the specific context of cyber arms control with different parties comparing their vulnerability stockpiles. We demonstrated that such an approach could be facilitated for external depletion measures such as bug-bounty programs. Such measures could potentially be extended so that even private actors could contribute to the internal exploit stockpile depletion process by adding external information about the depletion into ExTRUST.

As discussed, further evaluation and study of our concept is recommended, in particular in terms of the definition of the identifier. We discussed that a current limitation of the identifier is the necessity to find a sweet spot in the accuracy regarding the description of a security vulnerability that prevents duplicate descriptions of the same identifier while avoiding an unnecessary and potentially problematic generalization. Future work should analyze the relationship between the uniqueness and ambiguity of the characteristics of the identifier, the size of the identifier space, and – on a practical level – whether security experts independently create matching identifiers for the same vulnerability. Further work should focus on the possibility, the role and the security requirements of a trusted third party like the UN to calculate stockpile intersections, to circumvent the current necessity of cooperation between potentially opposing state actors. In addition, it would be interesting to implement ExTRUST as an actual measure between state parties to monitor its real world usage, its perception of the systems security and usability by the participating states as well its impact on their vulnerability disclosure considerations.

Due to the high political relevance of our proposal, we hope that this approach can be an inspiration to computer science and engineering to reflect on the ethical responsibility for the domain of cyberspace and its peaceful development and that future interdisciplinary work in this area will bring together researchers from privacy, IT security, and peace and conflict research.

ACKNOWLEDGEMENTS

This research work has been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1119 (CROSSING) – 236615297. It was co-funded by GRK 2050 Privacy & Trust/251805230, the German Federal

Ministry of Education and Research and the Hessian Ministry of Higher Education, Research, Science and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE, as well as the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 850990 PSOTI).

REFERENCES

- [1] C. Reuter, *Information Technology for Peace and Security - IT-Applications and Infrastructures in Conflicts, Crises, War, and Peace*. Wiesbaden, Germany: Springer Vieweg, 2019. [Online]. Available: <https://www.springer.com/de/book/9783658256517>
- [2] K. Giles and K. Hartmann, "“silent battle” goes loud: Entering a new era of state-avowed cyber conflict," in *11th International Conference on Cyber Conflict (CyCon)*. IEEE, 2019, pp. 1–13.
- [3] J. A. Lewis and G. Neuneck, *The cyber index: international security trends and realities*. UNIDIR, 2013.
- [4] R. Koch and M. Golling, "Silent Battles: Towards Unmasking Hidden Cyber Attack," in *11th International Conference on Cyber Conflict*, 2019, pp. 1–20.
- [5] R. Buchan, *Cyber Espionage and International Law*. Hart Publishing, 2018. [Online]. Available: <https://www.bloomsburyprofessional.com/uk/cyber-espionage-and-international-law-9781782257363/>
- [6] I. Georgieva, "The unexpected norm-setters : Intelligence agencies in cyberspace," *Contemporary Security Policy*, vol. 0, no. 0, pp. 1–22, 2019. [Online]. Available: <https://doi.org/10.1080/13523260.2019.1677389>
- [7] N. Dhir, H. Hoeltgebaum, N. Adams, M. Briers, A. Burke, and P. Jones, "Prospective artificial intelligence approaches for active cyber defence," 2021.
- [8] T. Reinhold and C. Reuter, *Cyber Weapons and Artificial Intelligence: Impact, Influence and the Challenges for Arms Control*. Springer International Publishing, 2022, pp. 145–158. [Online]. Available: https://doi.org/10.1007/978-3-031-11043-6_11
- [9] R. L. Russell, L. Kim, L. H. Hamilton, T. Lazovich, J. A. Harer, O. Ozdemir, P. M. Ellingwood, and M. W. McConley, "Automated vulnerability detection in source code using deep representation learning," 2018. [Online]. Available: <https://arxiv.org/abs/1807.04320>
- [10] A. Amarasinghe, W. Wijesinghe, D. Nirmana, A. Jayakody, and A. Priyankara, "Ai based cyber threats and vulnerability detection, prevention and prediction system," in *2019 International Conference on Advancements in Computing (ICAC)*, 2019, pp. 363–368.
- [11] L. Ablon and A. Bogart, *Zero days, thousands of nights: The life and times of zero-day vulnerabilities and their exploits*. Rand Corporation, 2017.
- [12] J. Rovner, "The Intelligence Contest in Cyberspace," 2020. [Online]. Available: <https://www.lawfareblog.com/intelligence-contest-cyberspace>
- [13] NATO CCDCOE, "Recent Cyber Events and Possible Implications for Armed Forces," NATO Cooperative Cyber Defence Centre of Excellence, Tech. Rep., 2020.
- [14] M. Schulze and T. Reinhold, "Wannacry About the Tragedy of the Commons? Game-Theory and the Failure of Global Vulnerability Disclosure," in *European Conference on Cyber Warfare and Security (ECCWS'18)*. Academic Conferences Ltd, 2018, pp. 454–463.
- [15] C. Cimpanu, "Kaspersky identifies mysterious APT mentioned in 2017 Shadow Brokers leak," <https://www.zdnet.com/article/kaspersky-identifies-mysterious-apt-mentioned-in-2017-shadow-brokers-leak/>, 2019, [Last retrieved 30-Aug-2020].
- [16] R. Milch, I. Pernice, S. Romanosky, K. Lewinski, S. Shackelford, P. Rosenzweig, T. Christakis, P. Swire, J. Healey, S. Herpig, J. Pohle, S. Zatkan, and E. Wenger, "Building common approaches for cybersecurity and privacy in a globalized world," *SSRN Electronic Journal*, 01 2019.
- [17] M. Schulze, "Quo Vadis Cyber Arms Control? – A Sketch of an International Vulnerability Equities Process and a 0-Day Emissions Trading Regime," in *Science Peace Security '19: Proceedings of the Interdisciplinary Conference on Technical Peace and Security Research*. TU Darmstadt, 2019, pp. 24–40.
- [18] R. J. Harknett and M. Smeets, "Cyber campaigns and strategic outcomes," *Journal of Strategic Studies*, vol. 45, no. 4, pp. 534–567, 2020. [Online]. Available: <https://doi.org/10.1080/01402390.2020.1732354>
- [19] T. Reinhold and C. Reuter, "Arms Control and its Applicability to Cyberspace," in *Information Technology for Peace and Security*. Springer, 2019, pp. 207–231.
- [20] UN-GGE, "Report of the Group of Governmental Experts on Developments in the Field of Information and Telecommunications in the Context of International Security," <https://digitallibrary.un.org/record/799853>, 2015, [Last retrieved 30-Aug-2020].
- [21] OSCE, "OSCE Confidence-Building Measures to reduce the risks of conflict stemming from the use of Information and Communication Technologies," <https://www.osce.org/pc/227281>, 2016, [Last retrieved 30-Aug-2020].
- [22] C. Sauerwein, C. Sillaber, A. Musmann, and R. Breu, "Threat intelligence sharing platforms: An exploratory study of software vendors and research perspectives," in *Proceedings der 13. Internationalen Tagung Wirtschaftsinformatik (WI 2017)*. WI, 2017, pp. 837–851.
- [23] P. Kuehn, T. Riebe, L. Apelt, M. Jansen, and C. Reuter, "Sharing of cyber threat intelligence between states," *Sicherheit & Frieden*, vol. 38, no. 1, pp. 22–28, 7 2020.
- [24] T. Reinhold and C. Reuter, "From Cyber War to Cyber Peace," in *Information Technology for Peace and Security*. Springer, 2019, pp. 139–164.
- [25] F. Cohen, "Computer viruses: theory and experiments," *Computers & Security (C&S'87)*, vol. 6, no. 1, pp. 22–35, 1987.
- [26] D. Kirat and G. Vigna, "Malgene: Automatic extraction of malware analysis evasion signature," in *Computer and Communications Security (CCS'15)*. ACM, 2015, pp. 769–780.
- [27] R. Petrik, B. Arik, and J. M. Smith, "Towards architecture and os-independent malware detection via memory forensics," in *Computer and Communications Security (CCS'18)*. ACM, 2018, pp. 2267–2269.
- [28] P. Kuehn, D. N. Relke, and C. Reuter, "Common Vulnerability Scoring System Prediction Based on Open Source Intelligence Information Sources," *Computers & Security*, 2023.
- [29] Mitre, "Common vulnerabilities and exposures," <https://cve.mitre.org/>, 2005, [Last retrieved 30-Aug-2020].
- [30] Y. Dong, W. Guo, Y. Chen, X. Xing, Y. Zhang, and G. Wang, "Towards the detection of inconsistencies in public security vulnerability reports," in *USENIX Security'19*. USENIX, 2019, pp. 869–885.
- [31] F. Sadique, S. Cheung, I. Vakiliinia, S. Badsha, and S. Sengupta, "Automated structured threat information expression (stix) document generation with privacy preservation," in *Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON'18)*. IEEE, 2018, pp. 847–853.
- [32] VERIS, "Vocabulary for Event Recording and Incident Sharing Framework," <http://veriscommunity.net/>, 2019, [Last retrieved 30-Aug-2020].
- [33] B. Martin, M. Brown, A. Paller, D. Kirby, and S. Christey, "CWE," Mitre, Tech. Rep., 2011.
- [34] A. One, "Smashing the stack for fun and profit," *Phrack magazine*, vol. 7, no. 49, pp. 14–16, 1996.
- [35] W. You, P. Zong, K. Chen, X. Wang, X. Liao, P. Bian, and B. Liang, "SemFuzz: Semantics-based Automatic Generation of Proof-of-Concept Exploits," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, Oct. 2017, pp. 2139–2154. [Online]. Available: <https://doi.org/10.1145/3133956.3134085>
- [36] N. Carlini and D. Wagner, "{ROP} is Still Dangerous: Breaking Modern Defenses," in *23rd USENIX Security Symposium (USENIX Security 14)*, 2014, pp. 385–399. [Online]. Available: <https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/carlini>
- [37] H. Shacham, "The geometry of innocent flesh on the bone: return-into-libc without function calls (on the x86)," in *Proceedings of the 14th ACM conference on Computer and communications security*, ser. CCS '07. New York, NY, USA: Association for Computing Machinery, Oct. 2007, pp. 552–561. [Online]. Available: <https://doi.org/10.1145/1315245.1315313>
- [38] M. Zhao, J. Grossklags, and K. Chen, "An exploratory study of white hat behaviors in a web vulnerability disclosure program," in *Security Information Workers (SIW'14)*. ACM, 2014, pp. 51–58.
- [39] N. Perlroth, "HackerOne Connects Hackers with Companies, and Hopes for a Win-Win," <https://www.nytimes.com/2015/06/08/technology/hackeron-one-connects-hackers-with-companies-and-hopes-for-a-win-win.html>, 2015, [Last retrieved 30-Aug-2020].
- [40] Mozilla, "Security Bug Bounty Program," <https://www.mozilla.org/en-US/security/bug-bounty/>, 2017, [Last retrieved 30-Aug-2020].
- [41] Facebook, "Facebook White-Hat," <https://www.facebook.com/whitehat/>, 2018, [Last retrieved 30-Aug-2020].

- [42] S. Zimmerman, "Microsoft Announces Windows Bug Bounty Program and Extension of Hyper-V Bounty Program," <https://www.xda-developers.com/microsoft-windows-bug-bounty/>, 2017, [Last retrieved 30-Aug-2020].
- [43] C. Evans, "Announcing Project Zero," <https://googleprojectzero.blogspot.com/2014/07/announcing-project-zero.html>, 2014, [Last retrieved 30-Aug-2020].
- [44] A. C. Yao, "How to generate and exchange secrets (extended abstract)," in *Symposium on Foundations of Computer Science (FOCS'86)*. IEEE, 1986, pp. 162–167.
- [45] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game or A completeness theorem for protocols with honest majority," in *Symposium on Theory of Computing (STOC'87)*. ACM, 1987, pp. 218–229.
- [46] D. Beaver, S. Micali, and P. Rogaway, "The round complexity of secure protocols (extended abstract)," in *Symposium on Theory of Computing (STOC'90)*. ACM, 1990, pp. 503–513.
- [47] P. Bogetoft, D. L. Christensen, I. Damgård, M. Geisler, T. P. Jakobsen, M. Krøigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter, M. I. Schwartzbach, and T. Toft, "Secure multiparty computation goes live," in *Financial Cryptography (FC'09)*. Springer, 2009, pp. 325–343.
- [48] B. Pinkas, T. Schneider, and M. Zohner, "Scalable private set intersection based on OT extension," *ACM Transactions on Privacy and Security (TOPS)*, vol. 21, no. 2, pp. 1–35, 2018.
- [49] A. Mirhoseini, A. R. Sadeghi, and F. Koushanfar, "CryptoML: Secure outsourcing of big data machine learning applications," in *International Symposium on Hardware Oriented Security and Trust (HOST'16)*. IEEE, 2016, pp. 149–154.
- [50] S. Kamara, P. Mohassel, and M. Raykova, "Outsourcing multi-party computation," <https://eprint.iacr.org/2011/272>, 2011, [Last retrieved 30-Aug-2020].
- [51] Á. Kiss, J. Liu, T. Schneider, N. Asokan, and B. Pinkas, "Private set intersection for unequal set sizes with mobile applications," in *Privacy Enhancing Technologies Symposium (PoPETs'17)*. De Gruyter, 2017, pp. 177–197.
- [52] B. Pinkas, M. Rosulek, N. Trieu, and A. Yanai, "PSI from PaXoS: Fast, malicious private set intersection," in *Theory and Applications of Cryptographic Techniques (EUROCRYPT'20)*, A. Canteaut and Y. Ishai, Eds. Springer, 2020, pp. 739–767.
- [53] V. Kolesnikov, N. Matania, B. Pinkas, M. Rosulek, and N. Trieu, "Practical multi-party private set intersection from symmetric-key techniques," in *Computer and Communications Security CCS'19*. ACM, 2017, pp. 1257–1272.
- [54] R. Inbar, E. Omri, and B. Pinkas, "Efficient scalable multiparty private set-intersection via garbled bloom filters," in *Security and Cryptography for Networks (SCN'18)*, 2018.
- [55] C. Hazay and M. Venkitasubramaniam, "Scalable multi-party private set-intersection," in *Public Key Cryptography (PKC'17)*. Springer, 2017, pp. 175–203.
- [56] Y. Huang, D. Evans, and J. Katz, "Private set intersection: Are garbled circuits better than custom protocols?" in *Symposium on Network and Distributed System Security (NDSS'12)*. The Internet Society, 2012.
- [57] I. Anati, S. Gueron, S. Johnson, and V. Scarlata, "Innovative technology for cpu based attestation and sealing," in *Hardware and Architectural Support for Security and Privacy (HASP'13)*. Citeseer, 2013.
- [58] P. Koerberl, V. Phegade, A. Rajan, T. Schneider, S. Schulz, and M. Zhdanova, "Time to rethink: Trust brokerage using trusted execution environments," in *Trust and Trustworthy Computing TRUST'15*. Springer, 2015, pp. 181–190.
- [59] D. Gupta, B. Mood, J. Feigenbaum, K. R. B. Butler, and P. Traynor, "Using intel software guard extensions for efficient two-party secure function evaluation," in *Financial Cryptography (FC'16)*. Springer, 2016, pp. 302–318.
- [60] K. A. Küçük, A. Paverd, A. C. Martin, N. Asokan, A. Simpson, and R. Ankele, "Exploring the use of intel SGX for secure many-party applications," in *Proceedings of the 1st Workshop on System Software for Trusted Execution (SysTEX'16)*. ACM, 2016, pp. 5:1–5:6.
- [61] R. Bahmani, M. Barbosa, F. Brasser, B. Portela, A. Sadeghi, G. Scerri, and B. Warinschi, "Secure multiparty computation from SGX," in *Financial Cryptography (FC'17)*. Springer, 2017, pp. 477–497.
- [62] S. Felsen, Á. Kiss, T. Schneider, and C. Weinert, "Secure and private function evaluation with intel SGX," in *Cloud Computing Security Workshop (CCSW'19)*. ACM, 2019, pp. 165–181.
- [63] J. Goldblat, *Arms Control: The New Guide to Negotiations and Agreements*, 2nd ed. SAGE, 2002.
- [64] T. Reinhold and C. Reuter, "Verification in Cyberspace," in *Information Technology for Peace and Security*. Wiesbaden, Germany: Springer Fachmedien, 2019.
- [65] B. Zangl and M. Zürn, "Theorien des rationalen handelns in den internationalen beziehungen," in *Rational Choice in der Politikwissenschaft: Grundlagen und Anwendungen*. VS Verlag für Sozialwissenschaften, 1994, pp. 81–111.
- [66] A. I. Kraus, O. Frazer, L. Kirchhoff, T. Kyselova, S. J. A. Mason, and J. Palmiano Federer, "Dilemmas and trade-offs in peacemaking: A framework for navigating difficult decisions," *Politics and Governance*, vol. 7, no. 4, pp. 331 – 342, 2019.
- [67] D. Evans, V. Kolesnikov, and M. Rosulek, "A pragmatic introduction to secure multi-party computation," *Found. Trends Priv. Secur.*, vol. 2, no. 2-3, pp. 70–246, 2018.
- [68] P. Kuehn, M. Bayer, M. Wendelborn, and C. Reuter, "OVANA: An approach to analyze and improve the information quality of vulnerability databases," in *Proceedings of the 16th International Conference on Availability, Reliability and Security*. ACM, 2021, p. 11.
- [69] Mitre, "CWE - Research Concept," <https://cwe.mitre.org/data/definitions/1000.html>, 2019, [Last retrieved 30-Aug-2020].
- [70] B. A. Cheikes, K. A. Kent, and D. Waltermire, *Common platform enumeration: Naming specification version 2.3*. US Department of Commerce, National Institute of Standards and Technology, 2011.
- [71] S. S. Gupta, *Blockchain*. John Wiley & Sons, Inc, 2017.
- [72] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *International Journal of Web and Grid Services (IJWGS'18)*, vol. 14, no. 4, pp. 352–375, 2018.
- [73] J. Katz and Y. Lindell, *Introduction to modern cryptography*. Chapman and Hall/CRC, 2014.
- [74] A. Davies, "Private Blockchain: 2019 Implementation Guide - DevTeam.Space," <https://www.devteam.space/blog/private-blockchain-implementation-guide>, 2018, [Last retrieved 30-Aug-2020].
- [75] D. Voell, F. L.-N. Gaski, R. Jagadeesan, R. Khasanshyn, H. Montgomery, S. Teis, T. Blummer, M. K. Katipalli, and M. Bowman, "Hyperledger whitepaper," <https://blockchainlab.com/pdf/Hyperledger%20Whitepaper.pdf>, 2016, [Last retrieved 30-Aug-2020].
- [76] Hyperledger, "Hyperledger Composer Overview," <https://www.hyperledger.org/wp-content/uploads/2017/05/Hyperledger-Composer-Overview.pdf>, 2017, [Last retrieved 30-Aug-2020].
- [77] OpenSSL, "openssl-dgst, dgst - perform digest operations," <https://www.openssl.org/docs/man1.1.1/man1/openssl-dgst.html>, 2020, [Last retrieved 30-Aug-2020].
- [78] NIST, "Nist policy on hash functions," <https://csrc.nist.gov/projects/hash-functions/nist-policy-on-hash-functions>, 2015, [Last retrieved 30-Aug-2020].
- [79] S. Nadeem, "If we lived in a bitcoin future , how big would the blockchain have to be?" <https://link.medium.com/5pi6wxXqN2>, 2019, [Last retrieved 30-Aug-2020].
- [80] U. W. Chohan, "The Limits to Blockchain? Scaling vs. Decentralization," *SSRN Electronic Journal*, 2019.
- [81] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, D. Song, and R. Wattenhofer, "On scaling decentralized blockchains," in *Financial Cryptography (FC'16)*. Springer, 2016, pp. 106–125.
- [82] L. Demir, A. Kumar, M. Cunche, and C. Lauradoux, "The pitfalls of hashing for privacy," *IEEE Communications Surveys & Tutorials (CST'18)*, vol. 20, no. 1, pp. 551–565, 2018.
- [83] D. Kales, C. Rechberger, T. Schneider, M. Senker, and C. Weinert, "Mobile private contact discovery at scale," in *USENIX Security'19*. USENIX, 2019, pp. 1447–1464.
- [84] A. Biryukov, D. Dinu, and D. Khovratovich, "Argon2: new generation of memory-hard functions for password hashing and other applications," in *European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2016, pp. 292–302.
- [85] C. Ye, G. Li, H. Cai, Y. Gu, and A. Fukuda, "Analysis of Security in Blockchain: Case Study in 51%-Attack Detecting," in *5th International Conference on Dependable Systems and Their Applications (DSA)*, 2018, pp. 15–24.
- [86] D. W. Archer, D. Bogdanov, Y. Lindell, L. Kamm, K. Nielsen, J. I. Pagter, N. P. Smart, and R. N. Wright, "From keys to databases - real-world applications of secure multi-party computation," *The Computer Journal*, vol. 61, no. 12, pp. 1749–1771, 2018.
- [87] D. Evans, V. Kolesnikov, and M. Rosulek, "A pragmatic introduction to secure multi-party computation," *Foundations and Trends in Privacy and Security*, vol. 2, no. 2-3, pp. 70–246, 2018.

- [88] L. Braun, D. Demmler, T. Schneider, and O. Tkachenko, "MOTION - a framework for mixed-protocol multi-party computation," *ACM Transactions on Privacy and Security (TOPS)*, vol. 2, no. 2-3, 2021, to appear.
- [89] I. Damgård and C. Orlandi, "Multiparty computation for dishonest majority: From passive to active security at low cost," in *Cryptology Conference (CRYPTO'10)*. Springer, 2010, pp. 558–576.
- [90] R. Bendlin, I. Damgård, C. Orlandi, and S. Zakarias, "Semi-homomorphic encryption and multiparty computation," in *Theory and Applications of Cryptographic Techniques (EUROCRYPT'11)*. Springer, 2011, pp. 169–188.
- [91] J. B. Nielsen, P. S. Nordholt, C. Orlandi, and S. S. Burra, "A new approach to practical active-secure two-party computation," in *Cryptology Conference (CRYPTO'12)*. Springer, 2012, pp. 681–700.
- [92] I. Damgård, V. Pastro, N. P. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Cryptology Conference (CRYPTO'12)*. Springer, 2012, pp. 643–662.
- [93] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl, and N. P. Smart, "Practical covertly secure MPC for dishonest majority - or: Breaking the SPDZ limits," in *European Symposium on Research in Computer Security (ESORICS'13)*. Springer, 2013, pp. 1–18.
- [94] Y. Lindell, N. P. Smart, and E. Soria-Vazquez, "More efficient constant-round multi-party computation from BMR and SHE," in *Theory of Cryptography (TCC'16) (B1)*. Springer, 2016, pp. 554–581.
- [95] M. Keller, "MP-SPDZ," <https://github.com/data61/MP-SPDZ>, 2016, [Last retrieved 30-Aug-2020].
- [96] A. Aly, M. Keller, D. Rotaru, P. Scholl, N. P. Smart, and T. Wood, "SCALE-MAMBA," <https://homes.esat.kuleuven.be/~%7Ensmart/SCALE/>, 2018, [Last retrieved 30-Aug-2020].
- [97] R. Rachuri and A. Suresh, "Trident: Efficient 4PC framework for privacy preserving machine learning," in *Symposium on Network and Distributed System Security (NDSS'20)*. The Internet Society, 2020.
- [98] A. Patra and A. Suresh, "BLAZE: blazing fast privacy-preserving machine learning," in *Symposium on Network and Distributed System Security (NDSS'20)*. The Internet Society, 2020.
- [99] H. Chaudhari, A. Choudhury, A. Patra, and A. Suresh, "ASTRA: high throughput 3pc over rings with application to secure prediction," in *Proceedings of the 2019 ACM SIGSAC Conference on Cloud Computing Security Workshop (CCSW'19)*. ACM, 2019, pp. 81–92.
- [100] D. Bogdanov, S. Laur, and J. Willemsen, "Sharemind: A framework for fast privacy-preserving computations," in *European Symposium on Research in Computer Security (ESORICS'08)*. Springer, 2008, pp. 192–206.
- [101] P. Mohassel and P. Rindal, "Aby³: A mixed protocol framework for machine learning," in *Computer and Communications Security (CCS'18)*. ACM, 2018, pp. 35–52.
- [102] D. Demmler, T. Schneider, and M. Zohner, "ABY - A framework for efficient mixed-protocol secure two-party computation," in *Symposium on Network and Distributed System Security (NDSS'15)*. The Internet Society, 2015.
- [103] Bosch, "Carbyne Stack: Cloud Native Secure Multiparty Computation," <https://carbynestack.io>, 2021, [Last retrieved 28-Oct-2022].
- [104] M. Keller, E. Orsini, and P. Scholl, "MASCOT: faster malicious arithmetic secure computation with oblivious transfer," in *Computer and Communications Security (CCS'16)*. ACM, 2016, pp. 830–842.
- [105] G. Maslov and R. Ustinov, "IAEA Verification Activities: A Tool of Building Trust or Building Up Pressure?" in *Russia in Global Affairs*, 2020, [Last retrieved 30-Aug-2020].
- [106] ZERODIUM, "Zerodium - the leading exploit acquisition platform for premium zero-days and advanced cybersecurity capabilities." <https://www.zerodium.com/>, 2019, [Last retrieved 30-Aug-2020].

ANNEX

The Annex is made available in a separate electronic file that will be an addendum to the main article.