Hindawi
Security and Communication Networks
Volume 2020, Article ID 8856379, 9 pages
https://doi.org/10.1155/2020/8856379

Research Article

Oblivious Inspection: On the Confrontation between System Security and Data Privacy at Domain Boundaries

Jorge Sancho, José García, and Álvaro Alesanco

Aragón Institute of Engineering Research (I3A), University of Zaragoza, Zaragoza 50009, Spain

Correspondence should be addressed to Jorge Sancho; jslarraz@unizar.es

Received 3 April 2020; Revised 4 August 2020; Accepted 10 September 2020; Published 22 September 2020

Academic Editor: Leandros Maglaras

In this work, we introduce the system boundary security vs. privacy dilemma, where border devices (e.g., firewall devices) require unencrypted data inspection to prevent data exfiltration or unauthorized data accesses, but unencrypted data inspection violates data privacy. To shortcut this problem, we present Oblivious Inspection, a novel approach based on garbled circuits to perform a stateful application-aware inspection of encrypted network traffic in a privacy-preserving way. We also showcase an inspection algorithm for Fast Healthcare Interoperability Resources (FHIR) standard compliant packets along with its performance results. The results point out the importance of the inspection function being aligned with the underlying garbled circuit protocol. In this line, mandatory encryption algorithms for TLS 1.3 have been analysed observing that packets encrypted using Chacha20 can be filtered up to 17 and 25 times faster compared with AES128-GCM and AES256-GCM, respectively. All together, this approach penalizes performance to align system security and data privacy, but it could be appropriate for those scenarios where this performance degradation can be justified by the sensibility of the involved data such as healthcare scenarios.

1. Introduction

Data leakage and exfiltration are one of the top security concerns in modern information systems. The consequences of data exfiltration are huge for companies; information is the most valuable resource a company has. However, these consequences elevate to disastrous when sensible data, such as health-related one, are involved due to the consequent legal implications [1]. For decades, companies have addressed this problem by holding this sensible information within their boundaries. However, in the last decade, their doors have been opened to take advantage of all the advances that are taking place in the ICT industry; information exchange between partners has been proved to suppose an important strategic advantage (e.g., improvements in global supply chain). This is not different for healthcare institutions where the patient-generated resources and the cross-organizational healthcare data sharing seem to be the key for the future of medicine. Open data sharing jointly with the latest data processing technologies, such as big data and machine learning, would help to enable preventive medicine and early detection of illness conditions by improving care delivery and lowering costs [2].

This unstoppable new connected environment makes it impossible for organizations to keep as isolated silos, experimenting the need to open their boundaries to take advantage of all these changes. However, with these new opportunities also come new risks and threats and the adoption of this connected scenario must be performed carefully, without falling into security concerns [3]. Firewalls, as key components for system security, are usually placed at domain borders to take care in the perimeter security. They inspect the traffic that flows along the system boundaries to avoid many threats: malware reaching inside the system, confidential information being compromised, and so on. However, the use of end-to-end data encryption between communicating peers (providing confidentiality, integrity, and authenticity to data) limits the information that the firewall can use to decide whether a packet is legitimate or not. In these cases, the application data are encrypted so that firewall can just inspect the network information (e.g., IP addresses and ports) that is transmitted

without encryption. Some existent approaches such as SSL inspection propose to use a man-in-the-middle-like controlled (by the system administrator) attack [4] allowing the firewall to decrypt the whole packet and inspect its content. However, this approach presents several drawbacks, being the most important that the data privacy is compromised making it unacceptable for several scenarios. Hence, the system boundary security vs. privacy dilemma arises; if more information is used to decide whether a packet might suppose a security concern, more reliable would be the decision (security) but more information available compromises the confidentiality of the data flow (privacy).

In the past few years, data privacy has attracted much attention and new methods to perform computation on private data have appeared. These solutions can be classified into two groups: those ones that rely on Trusted Execution Environments (TEEs) and others that rely on Cryptographic Primitives. A TEE is a secure area inside a main processor able to guarantee the confidentiality and integrity of the code and data loaded on it. Main examples of TEEs are Intel Software Guard Extensions (SGX) and ARM TrustZone. The second group includes methods (also known as protocols) that use Cryptographic Primitives to perform some privacy-preserving computations, but the use of these protocols comes at a cost: a decrease in the overall system performance [5]. In practice, one of the most widely used protocols is garbled circuits (GC), which is a general purpose, two-party secure computation protocol. The use of this protocol allows two parties to jointly compute the output of a function without learning anything about the other party's inputs or intermediate results. In garbled circuits, as in other secure computation protocols, some parties might be corrupted and they would try to extract information related with the inputs of the other parties. This behavior could be classified within two adversaries' models, each with its own security concerns. The semihonest adversary model, which provides passive security, assumes that the adversary will cooperate to gather all information leaked from the protocol execution without deriving from the agreed protocol. In the malicious adversary model, which provides active security, the adversary may arbitrarily deviate from the protocol execution in its attempt to cheat. The only thing that an adversary can do in the case of dishonest majority is to cause the honest parties to abort having detected cheating.

Thus, the question raised is whether secure computation protocols could sort out the system boundary security vs. privacy dilemma, allowing the firewall to inspect the encrypted information (looking for threats) while information privacy remains unaltered. Some challenges reside in the design of these privacy-preserving inspection methods. Most relevant challenges include that proposed privacy-preserving inspection methods must fit in existent network architectures with minimal modifications; they should provide similar functionalities to standard firewall devices, achieving similar performance and without requiring additional security assumptions. Although some previous works already talk about privacy-preserving middleboxes [6], all of them have important related drawbacks. They require heavy modification on network architectures for their use, provided functionalities are quite limited (typically only allow to perform pattern or range matching), and they require making some security assumptions (at least one of the communication peers, the client, or the server cannot be compromised) that cannot be guaranteed in most scenarios.

In this paper, we present Oblivious Inspection, the first privacy-preserving inspection method able to keep track of the communication state at the application level while is also able to understand some protocols and parse them so that signatures or rules can specifically address certain fields in the protocol. In order to show the interest of the proposed method, an application scenario where clinical information is involved has been proposed and a proof of concept has been implemented to test the inspection of packets compliant with the Fast Healthcare Interoperability Resources (FHIR) standard.

2. Related Work

In the past few years, there have been advances in providing middleboxes with the appropriate mechanisms to improve both system security and data privacy at the same time. First approaches, which are based on Cryptographic Primitives, could be found in [7, 8]. These works use searchable encryption to check whether some patterns of interest are present or not in the ciphered payload. To that end, the packet payload is first tokenized and then these tokens are matched against the rule patterns. Those studies do not show interest not only on the confidentiality of the ciphered data but also on the rules. The rules have to be protected given that in some cases can be intellectual property (e.g., commercial IDS rules). In [7], only the content of the rules is protected while some metadata (inspection fields, offsets, and number of patterns) remain unencrypted. This could leak some confidential information and is addressed in [8]. Although those works pointed into a promising research direction, functionality achievable with those methods is quite limited since they only allow us to look for patterns (i.e., do not support range matching nor regular expressions) inside the ciphered payload.

In [9, 10], the authors proposed novel methods to provide new functionalities to middleboxes. These improvements enable them to match encrypted values against encrypted prefixes and ranges (e.g., check if a port belongs to a range). This can be used to perform tasks such as load balancing, NAT, or traffic classification. In [9], the authors proposed a new encryption scheme called PrefixMatch and used it to perform the secure range matching. Some metadata about the packet headers are leaked in this system. In [10], a different approach based on secret sharing secure multiparty computation (MPC) is taken. This approach is limited to substring comparison and leaks information about which bits of the incoming packet do not match.

Zhou et al. exposed in their work [11] the importance of privacy-preserving packet filtering for Internet of Things (IoT) scenarios. They state that message content filtering would avoid duplicate packet transmission which would reduce both computational and communication cost. To that end, they proposed the use of a prefilter [12], an efficient

privacy-preserving relay filtering scheme for Delay-Tolerant Networks (DTNs) in vehicular IoT communication.

Other works [13, 14] presented methods that rely on a TEE, Intel SGX, to ensure the confidentiality and integrity of the middlebox. However, although approaches based on the TEE are really promising, it is still necessary to trust the hardware manufacturer and deal with possible security issues related to the technology that could appear (e.g., foreshadow [15]), which may not be acceptable in all cases.

Finally, the authors presented a prior study [16] that explores the idea of using garbled circuits to inspect encrypted traffic. However, this is a very preliminary work that only considered some basic functionality without caring about some essential aspects such as the state of the connection, information about the user making the request, or how policies are compartmented. Moreover, performance results were very scarce and it does not face how the involved parties interact between them to perform the inspection process.

3. Scenario and Assumptions

In this section, we first present the target scenario where the proposed inspection method would be applied. After that, we discuss the assumptions that have been taken for the design of the inspection function as well as for the obtention of the performance results.

3.1. Scenario. In this work, we focus on a connected healthcare scenario (see Figure 1) where information is stored into an Electronic Health Record (EHR) placed inside of a trusted domain (e.g., healthcare institution) and some clients, placed outside, request access to the stored information. Different kinds of clients might attempt to access the EHR with different purposes. Thanks to the widespread adoption of wearables, a huge amount of patient's information, such as vital signs and fitness data, is collected daily. Since some of these devices can acquire this information with enough accuracy to be used with clinical purposes [17], many advantages could be obtained from these patient-generated resources if available for the healthcare institution (i.e., they are in the EHR). Thus, patients should be able to access the EHR not only to see their history but also to feed it with new data.

To obtain real value from the collected data, some processing techniques must be applied. Machine learning and big data techniques are providing promising results in the field of the health informatics [18], but tons of data are required to develop and validate these kinds of methods. Thus, researchers should be allowed to access the patient's record, after the corresponding process of anonymization, to perform their studies. Finally, in this highly connected world, it is not unusual that users desire to share some information with their family or friends. A patient who stores his fitness data, his weight progression, or other related data might be able to delegate access to a third-party application (e.g., social network) to retrieve authorized information from the EHR and share it with his contacts.

In such heterogeneous scenario with different parties accessing patient's information with different profiles, purposes, and security restrictions, it might be challenging to avoid information leakages to undesired locations. A firewall placed at the boundary of the trusted domain might inspect all the data packets coming from/to the EHR and ensure that they meet the security policies defined by the security administrator. For example, dropping resources containing personally identifiable information when destined to a research institution or blocking data traffic that contains sensitive information when destined to a social network.

3.2. Assumptions. In this scenario, we would assume that all the information is exchanged between the EHR and its clients using the most recent standard from the Health Level Seven (HL7) International Organization, FHIR. This standard defines a set of data models which can represent a wide range of healthcare-related aspects, both clinical and administrative. Instances of these data models, which are named resources, are used to exchange and/or store the data using different serializations formats, such as the JavaScript Object Notation (JSON), which would be used in this case. A JSON object is a data structure that holds unordered name/value pairs, where names are strings and values might be basic (string, numeric, true, false, or null) or complex types (lists of basic types of another object). The FHIR standard also defines an Application Programming Interface (API) based on Representational State Transfer (REST) that can be used to retrieve, create, update, and delete the resources from the EHR. This kind of API uses the HyperText Transfer Protocol (HTTP) for message exchange, in which the preferred way to assert identities is by means of JSON Web Tokens (JWTs), as is also suggested by the FHIR team. JWTs are a JSON-formatted set of claims stating things such as the user identity, the organization he/she belongs to, or his/her role there. The token is signed, encoded in base64, and included as the Authorization header value in the HTTP packet. An example of a request to create an observation resource is shown in Figure 2. In this message, the request line, which indicates the type of operation and the resource involved, is shaded in green, the request headers are shaded in blue, and the request body is shaded in orange.

Finally, Transport Layer Secure (TLS) is used to provide confidentiality to data flows between clients and the EHR. In this work, we will assume that TLS 1.3 would be used since it is the most recent version of the standard which comes with remarkable security improvements. When using TLS, different cipher suits which are combinations of Authenticated Encryption with Associated Data (AEAD) and HMAC-based Key Derivation Function (HKDF) algorithms could be used. The cipher suit for each session is negotiated during the TLS handshake.

4. Methodology

The proposed inspection method, Oblivious Inspection, allows us to perform a stateful, application-aware inspection of encrypted packets relying on garbled circuits [19] to

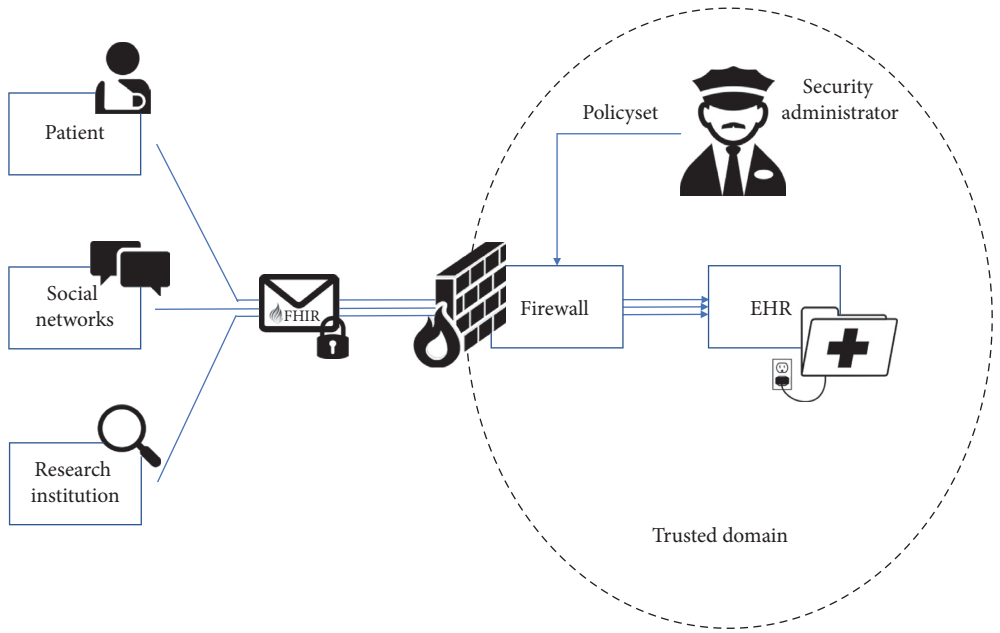

Figure 1: Reference scenario: several clients require access to the EHR from different locations and for different purposes.

```
POST /Observation HTTP/1.1

Host: fhir-server.org
User-Agent: Mozilla/4.0
Content-Type: application/fhir+json;charset = UTF-8
Content-Length: 450
Authorization: eyJhbGciOiJIUzI1N … R5cCI6IkpXVCJ9
```

```
{
  "resourceType": "Observation",
  "_id": "1234",
  "status": "final",
  "code": {
    "coding": [
      {
        "system": "http://snomed.info/sct",
        "code": "27113001",
        "display": "Body weight"
      }
    ]
  },
  "subject": {
    "reference": "Patient/5678"
  },
  "effectiveDateTime": "2016-03-28",
  "valueQuantity": {
    "value": 185,
    "unit": "lbs",
    "system": "http://unitsofmeasure.org",
    "code": "[lb_av]"
  }
}
```

Figure 2: Request to create an observation resource.

ensure the privacy of the inspected data. Garbled circuits are a cryptographic protocol that enables two-party secure computation, which means that two mistrusting parties can jointly evaluate an arbitrary function while no information about one party's private input nor intermediate results are leaked to the other party. The garbled circuits protocol consists of 6 steps (see Figure 3). First, the function that would be evaluated in a private way is defined as a Boolean circuit known by both parties. Then, one party, the garbler, assigns a k-bit label randomly generated to 0 and 1 values in each circuit wire (i.e., for input X wire, X_0 and X_1 labels). New truth tables are generated for each gate in the circuit using these labels, and finally the garbled tables are generated, encrypting each output in the new truth table using its inputs as keys. Garbled tables and labels corresponding to garbler's inputs (X_0 and Y_1 in Figure 3) are sent to the other party, the evaluator. The evaluator obtains the labels corresponding to his own inputs (Z_0) through Oblivious Transfer with the help of the garbler. After that, the evaluator goes through all gates trying to decrypt the rows of the garbled tables but just one row could be decrypted correctly, obtaining the label corresponding to the output of the gate. After the evaluation of the whole circuit, the evaluator obtains the label associated to the output (F_1). Finally, the garbler and evaluator cooperate to learn the value represented by the output label, which is the result of the function.

When the Oblivious Inspection is applied to the proposed eHealth scenario, the role of the garbler would be played by the EHR while the firewall would be the evaluator. The Oblivious Inspection is performed each time when a new packet reaches the firewall. However, before the Oblivious Inspection could be performed for the first time, some configurations are required. Since the Oblivious Inspection relies on garbled circuits, two parties are involved in the inspection process. Thus, each time the inspection process is about to start, the firewall must know how to

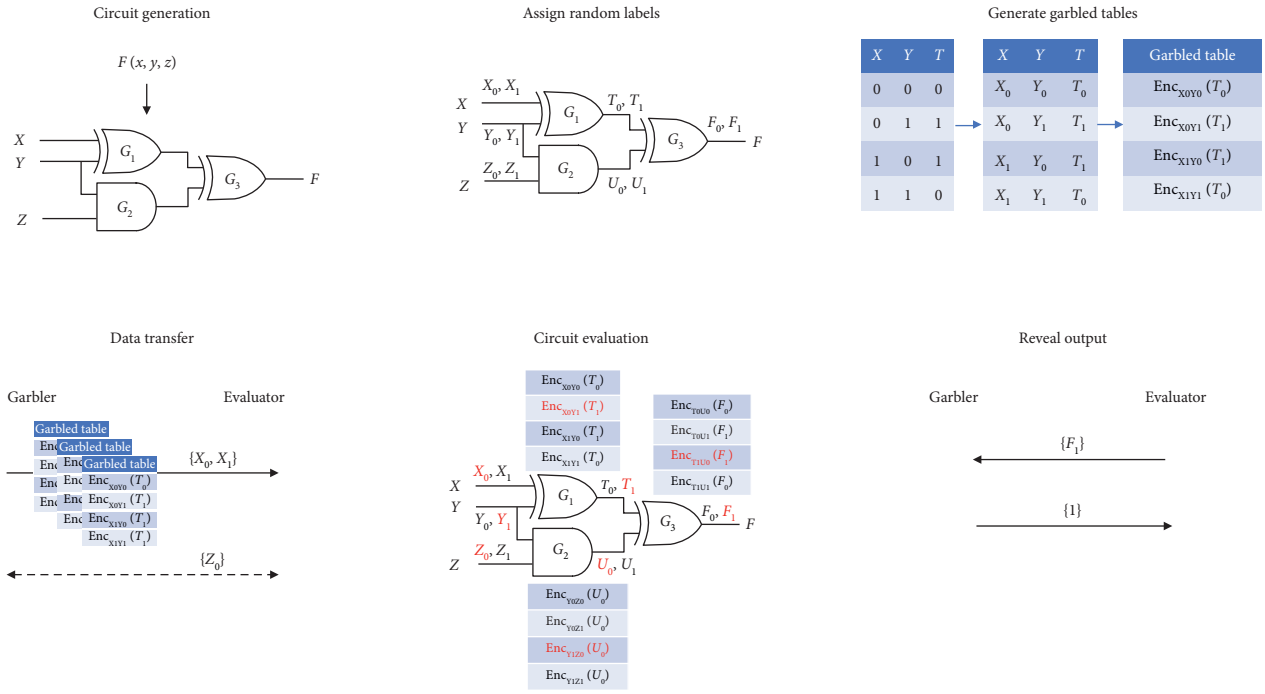

FIGURE 3: Garbled circuit protocol: steps that compose the underlying multiparty computation protocol.

notify the EHR about it and jointly perform the inspection. To that end, the EHR is configured to be waiting for Oblivious Inspection Start Request on a concrete port and the corresponding configuration (IP address and port) is provided to the firewall beforehand.

In the normal operation mode, each time a packet reaches the firewall, the workflow shown in Figure 4 is triggered to decide whether the packet should be forwarded or not. First, the firewall checks if the packet source or destination addresses belong to a registered EHR. If this is the case, the firewall will send an Oblivious Inspection Start Request to its previously configured peer. This Oblivious Inspection Start message is sent over a new TLS session and includes IP addresses and ports (both source and destination) of the packet to be evaluated. This information would be used by the EHR to identify the flow to which the packet belongs and obtain the session key used for its encryption. If everything goes as expected, both firewall and EHR jointly evaluate the inspection function (right side of Figure 4) using the garbled circuit protocol explained previously. Information required to evaluate the inspection function (garbled tables, labels of EHR private inputs, and all information exchange related to the oblivious transfer) is transmitted over the new established TLS session. After the inspection has taken place, both parties learn whether the packet meets the defined policies or not. If true, the firewall would forward the packet to its destination. If any of these steps do not result as expected (the firewall does not recognize any of the packet addresses as a registered EHR, the EHR do not recognize the flow to which the packet belongs, or the packet does not meet the policies), the packet is dropped and the firewall sends a reset (RST) message to the source of the rejected packet.

4.1. Inspection Function. The inspection function depends on the application traffic being inspected, and the achievable functionality would be limited by the performance overhead assumable on each scenario. In order to show the potential of the Oblivious Inspection, a function to evaluate some relevant aspects of the FHIR compliant packets has been proposed (see Figure 4, right side). To evaluate this inspection function, the session key and the ciphered packet would be provided as private inputs by the EHR and the firewall, respectively. The firewall would also provide an auxiliary key that would be used, jointly with the session key, to derive the state key.

When the inspection of a new packet begins, it is first decrypted using the session key (note that evaluating the whole inspection function inside the garbled circuit protocol guarantees that no party would have access to the decrypted message). After that, behavior differs when the packet is a request or a response. If the packet is a request, the request line is parsed and information about the requested operation (read, create, delete, etc.) and the targeted resource (resource type and id) is gathered and saved as state information. After that, the HTTP headers are parsed and inspected to assert that only allowed headers are present. If the Authorization header is present, it is then processed to collect some useful information from it. In the proposed scenario, the Authorization header would be a JWT, so that it would be firstly decoded from base64 and then its integrity would be validated by checking its signature, ensuring that it has not been tampered after it was issued. If everything goes as expected, some claims (such as the user identity, the organization he/she belongs, or his/her role there) are extracted from the token and saved as state information. After the header's inspection, the state information is encrypted using the state

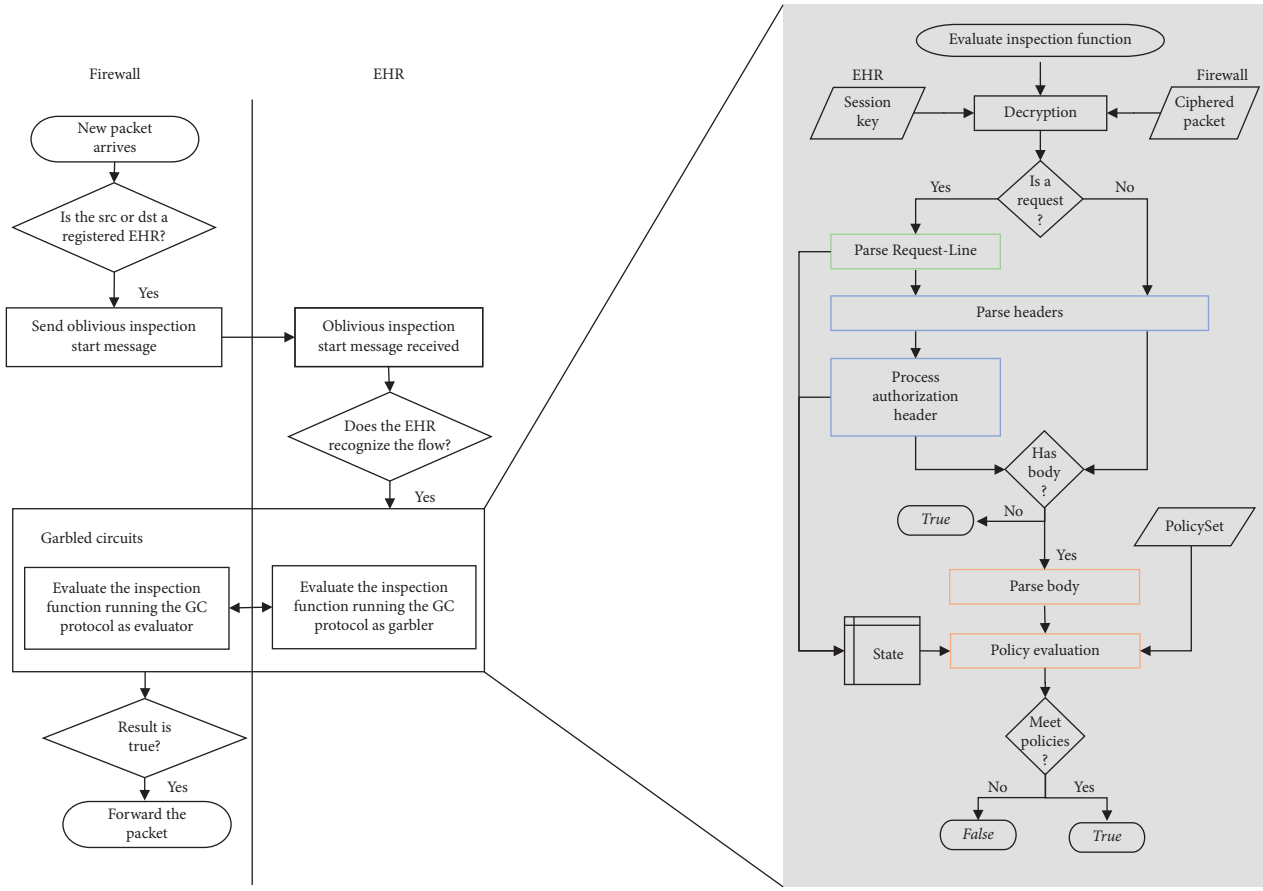

Figure 4: Oblivious Inspection: flowchart of the Oblivious Inspection evaluation.

key and returned as protocol output. If the request has not a body (i.e., a read operation), the inspection would be finished and the packet forwarded. Otherwise, the body of the HTTP packet would be inspected now. The body would be parsed to assert that it is a FHIR resource serialized in the JSON format. To that end, names in the resource are matched against a dictionary of FHIR keywords, which content would depend on the resource type information obtained from the request line. After that, attributes are extracted from the resource. In this study, we refer as "attribute" to each pair of the nested names that point to a basic value type and the value itself. Some examples of attributes extracted from the FHIR resource shown in Figure 2 would be (resourceType, "Observation") and (code.coding.display, "Body weight"). At this point, the resource being inspected is checked to assert that is coherent with the resourceType and resource id information obtained from the request line, ensuring that only resources that have previously been requested leave the perimeter.

Finally, the policy evaluation takes place. Policies are compartmented depending on the role of the user making the request (obtained from the JWT) and the resource type being inspected. For each combination of resourceType and role, a PolicySet is defined, which contains a set of policies which in turn contains a set of rules. Each rule defines a condition about an attribute of the resource (e.g., the value of "code.coding.display" must be "Body weight" or "subject.reference" must not be present in the resource). If all the rules in a policy are successfully matched, the policy is met. If at least one policy in the policy set is met, the policy evaluation is considered as passed and the decision would be to forward the packet.

If the packet is not a request but a response, the parse of the request line and the Authorization header processing steps would be skipped while the rest of the inspection would be performed as already explained for the requests. The additional information required for the policy evaluation (i.e., resource type and user role), which is gathered from the skipped steps when a request is being inspected, would be now obtained from the state information instead. To that end, the firewall would provide as protocol input the encrypted state resulting from the inspection of the request that has triggered the response being inspected, which would be decrypted using the state key.

5. Results

In order to evaluate the proposed Oblivious Inspection, we have implemented the proposed inspection function using the Obliv-C framework [20]. Several analyses have been performed to assess the viability of using this kind of methods in real-world scenarios. The performance results have been calculated using Amazon Web Services (AWS) EC2 c5.large instances, which have 2 virtual CPUs and 4 GB

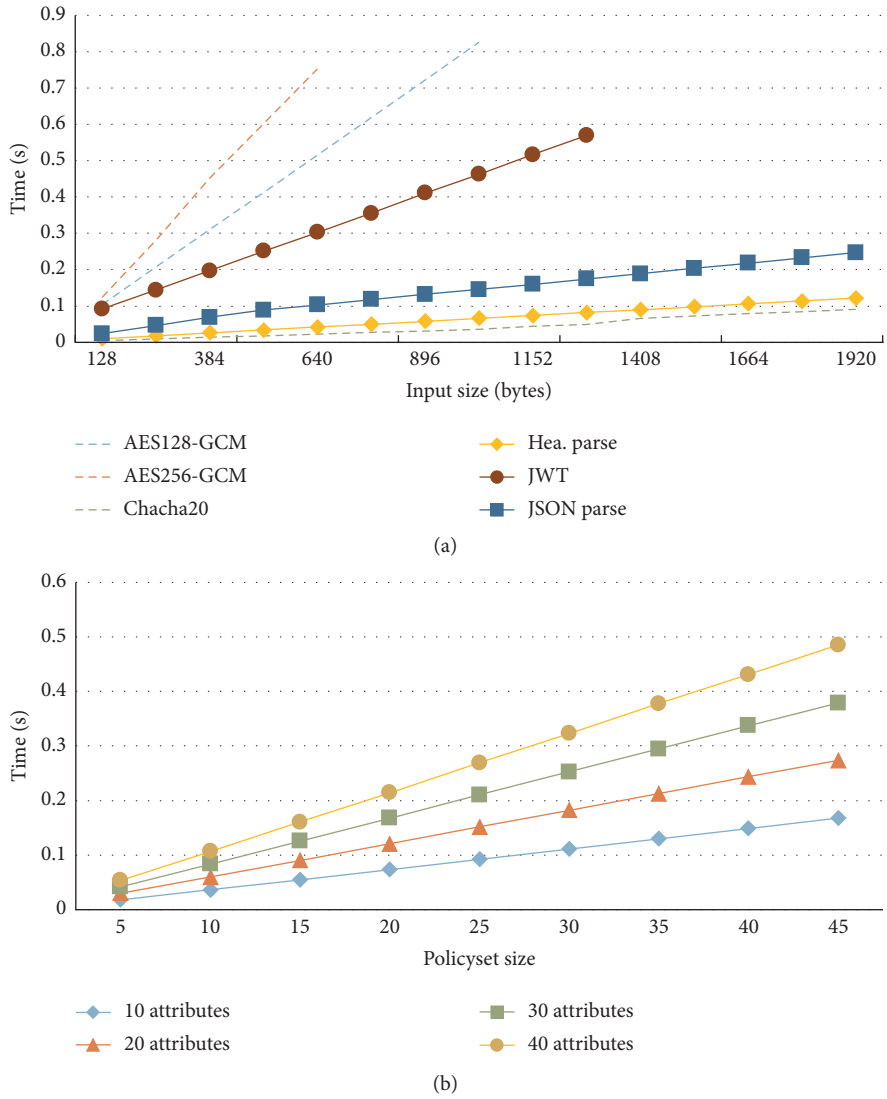

Figure 5: Performance results: they have been calculated using Amazon Web Services (AWS) EC2 c5.large instances, which have 2 virtual CPUs and 4 GB RAM.

RAM. One instance has been used for each party (garbler and evaluator) needed to consider the communication cost required by the garbled circuit protocol in a realistic way.

The times taken to perform different tasks that compose the proposed inspection function under different conditions have been analysed (see Figure 5). In the left side, the figure shows the times associated with packet decryption (dashed lines), the header parse and inspection (green line), Authorization header processing (brown line), and JSON content parsing (blue line) when the input size ranges from 0 to 2000 bytes. Decryption performance has been analysed for the three mandatory AEADs defined in the TLS 1.3 standard (AES128-GCM, AES256-GCM, and Chacha20). As we can see in the figure, times required to decrypt the ciphertext when an AES-based algorithm is used is significantly higher than the required time when using Chacha20 (close to 17 times for AES128-GCM and 25 times for AES256-GCM). Therefore, the encryption scheme used may condition the viability of using this inspection method. The Authorization header processing times include the time required to decode the JWT from its base64 encoding, to verify its signature using the HMAC-SHA256 algorithm, and to extract the subject id, the role, and organization claims from the token. Times required to perform all tasks shown in this figure are linear with the input size. On the other hand, times associated with the policy evaluation are shown in the right side of Figure 5. Times for different policy set sizes (5 to 45 policiey per policy set) and number of attributes inside the resource (10, 20, 30, and 40 attributes) have been calculated, assuming a fixed number of 10 rules per policy in all cases. Times used for the policy evaluation are linear with both the number of attributes extracted from the resource and the size of the policy set being enforced.

A proof of concept of the Oblivious Inspection has been implemented using the NGINX server as the EHR and the Click modular router as the firewall. Both have been

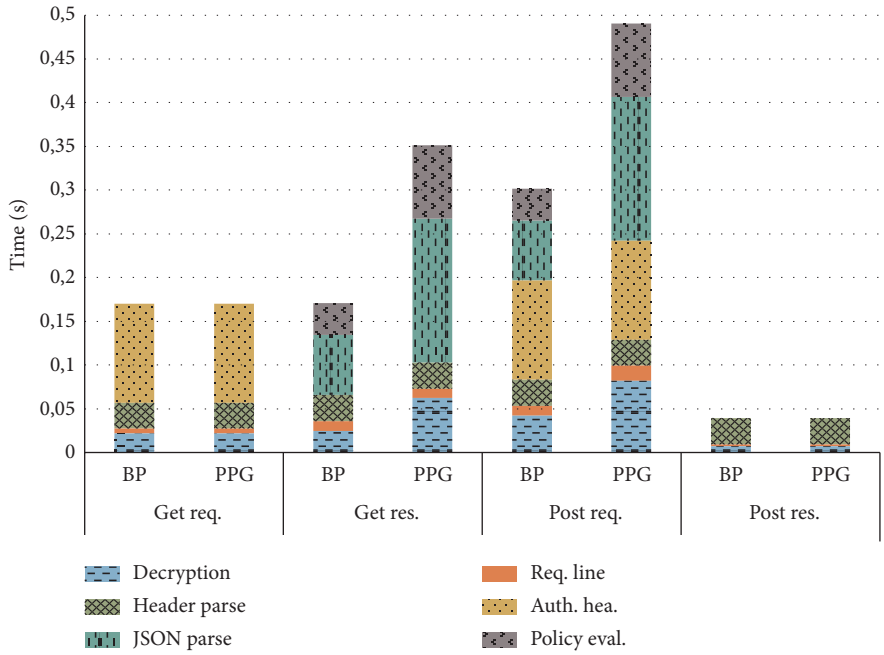

Figure 6: Package inspection times: inspection times to read or create a photoplethysmogram and blood pressure observations.

modified to support the workflow shown in Figure 4. Times required to inspect some FHIR packets exchanged for some frequent operations along the patient's daily illness management are shown in Figure 6. These packets include requests and responses required to read and create two different kinds of resources. To obtain these times, some reasonable assumptions have been performed. First, a fixed HTTP header size of 600 bytes for requests and 300 for responses is assumed in all cases, where the 300-byte difference is because the Authorization header is only included in request packets. Two different resources have been analysed: a blood pressure (BP) observation (scalar values) which has a size of 400 bytes and a 10-second plethysmography (PPG) observation (low frequency unidimensional signal) which has a size of 1200 bytes. The number of attributes inside of each resource has been set to 15 and 30 for the BP and PPG resources, respectively. Finally, the size of the policy set being enforced for each combination of role and resource type would be a maximum of 10 policies containing 10 rules each. These times are decomposed in times for the different tasks to show the impact of each task in the overall inspection time. From the figure, we can see that the whole time (including request and response inspection times) required to read the already defined resources would be 340 and 521 ms, while times required to create them would be 340 and 529 ms for BP and PPG observations, respectively.

6. Conclusions

In this paper, we have presented Oblivious Inspection, the first stateful application-aware inspection method that cares about the privacy of the inspected data without relying on TEEs. Some key benefits can be associated with the proposed inspection method. First, it can work over a standard implementation of TLS with minimal modification on the EHR (server side) being seamless for clients, which eases its deployment in real-world scenarios. Second, it can be used to inspect not only plain text but also structured data, enabling a wide spectrum of applications. It is also able to keep track of the connection state at application level, which is crucial to inspect new-generation protocols such as HTTP/2, which is already replacing the previous version due to its performance improvement. Finally, with the proposed method, the firewall can be sure that a given packet matches or not the defined rules before forwarding it without relying on any other party.

An inspection function to evaluate FHIR compliant packets along with its performance results has been presented. In view of the performance results, the authors conclude that although the performance penalty introduced by using the proposed method could currently be too high for a general application use, the sensibility of the involved data could justify its use in certain scenarios, such as healthcare ones. However, there are several fronts that could be attacked to improve this performance. First, research on MPC friendly cryptographic algorithms and information serialization formats is needed. Second, developing MPC frameworks that take advantage of the parallelism provided by hardware such as GPUs or FPGAs would improve the execution speed by orders of magnitude [21]. With the adoption of these measures, the performance degradation could be expected to significantly decrease, thus allowing the widespread adoption of these solutions.

All together, this work shows how multiparty computation could be used to address the confrontation between system security and data privacy at the domain boundaries. The proposed inspection method would help to keep the private information away from the eyes of a curious network

administrator or from an intruder taking control of the firewall (which is usually facing the Internet) without preventing the firewall from providing its crucial functionality to keep systems secure.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this article.

Acknowledgments

This research was funded by the Ministerio de Economía, Industria y Competitividad from Gobierno de España, European Regional Development Fund (TIN2016-76770-R), Gobierno de Aragón and FEDER "Construyendo Europa desde Aragón" (T31_20R), and Ministerio de Educación, Cultura y Deporte from Gobierno de España via a doctoral grant to the first author (FPU15/04841).

References

[1] Regulation (EU) 2016/679 of the European Parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/EC (general data protection regulation) (text with EEA relevance).

[2] O. Enaizan, A. A. Zaidan, N. H. M. Alwi et al., "Electronic medical record systems: decision support examination framework for individual, security and privacy concerns using multi-perspective analysis," *Health and Technology*, vol. 10, no. 3, pp. 795–822, 2020.

[3] M. Hussain, A. A. Zaidan, B. B. Zidan et al., "Conceptual framework for the security of mobile health applications on android platform," *Telematics and Informatics*, vol. 35, no. 5, pp. 1335–1354, 2018.

[4] T. Radivilova, L. Kirichenko, D. Ageyev, M. Tawalbeh, and V. Bulakh, "Decrypting SSL/TLS traffic for hidden threats detection," in *Proceedings of the 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, IEEE, Kyiv, Ukraine, pp. 143–146, May 2018.

[5] D. Evans, V. Kolesnikov, and M. Rosulek, "A pragmatic introduction to secure multi-party computation," *Foundations and Trends® in Privacy and Security*, vol. 2, pp. 2-3, 2018.

[6] C. Wang, X. Yuan, Y. Cui, and K. Ren, "Toward secure outsourced middlebox services: practices, challenges, and beyond," *IEEE Network*, vol. 32, no. 1, pp. 166–171, 2017.

[7] J. Sherry, L. Chang, R. Popa, and S. Ratnasamy, "Blindbox: deep packet inspection over encrypted traffic," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, pp. 213–226, London, UK, August 2015.

[8] X. Yuan, X. Wang, J. Lin, and C. Wang, "Privacy-preserving deep packet inspection in outsourced middleboxes," in *Proceedings of the IEEE INFOCOM 2016-the 35th Annual IEEE International Conference on Computer Communications*, IEEE, San Francisco, CA, USA, pp. 1–9, April 2016.

[9] C. Lan, J. Sherry, R. Popa, S. Ratnasamy, and Z. Liu, "Embark: securely outsourcing middleboxes to the cloud," in *Proceedings of the 13th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 16)*, pp. 255–273, Santa Clara, CA, USA, March 2016.

[10] H. J. Asghar, L. Melis, C. Soldani, E. De Cristofaro, M. A. Kaafar, and L. Mathy, "Splitbox: toward efficient private network function virtualization," in *Proceedings of the 2016 Workshop on Hot Topics in Middleboxes and Network Function Virtualization*, pp. 7–13, Florianopolis, Brazil, August 2016.

[11] J. Zhou, Z. Cao, X. Dong, and A. V. Vasilakos, "Security and privacy for cloud-based IoT: challenges," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 26–33, 2017.

[12] R. Lu, X. Lin, T. Luan et al., "Prefilter: an efficient privacy-preserving relay filtering scheme for delay tolerant networks," in *Proceedings of the 2012 Proceedings IEEE INFOCOM*, pp. 1395–1403, IEEE, Orlando, FL, USA, March 2012.

[13] L. Schiff and S. Schmid, "PRI: privacy preserving inspection of encrypted network traffic," in *Proceedings of the 2016 IEEE Security and Privacy Workshops (SPW)*, IEEE, San Jose, CA, USA, pp. 296–303, May 2016.

[14] J. Han, S. Kim, J. Ha, and D. Han, "Sgx-box: enabling visibility on encrypted traffic using a secure middlebox module," in *Proceedings of the First Asia-Pacific Workshop on Networking*, pp. 99–105, Hong Kong China, August 2017.

[15] J. Van Bulck, M. Minkin, O. Weisse et al., "Foreshadow: extracting the keys to the intel {SGX} kingdom with transient out-of-order execution," in *Proceedings of the 27th {USENIX} Security Symposium ({USENIX} Security 18)*, pp. 991–1008, Baltimore, MD, USA, August 2018.

[16] J. Sancho, G. L. Mikkelsen, J. Lindstrøm, J. García, and Á. Alesanco, "On the privacy enhancement of in-transit health data inspection: a preliminary study," in *Proceedings of the Mediterranean Conference on Medical and Biological Engineering and Computing*, Springer, Coimbra, Portugal, pp. 855–860, September 2019.

[17] D. Hernando, S. Roca, J. Sancho, Á. Alesanco, and R. Bailón, "Validation of the apple watch for heart rate variability measurements during relax and mental stress in healthy subjects," *Sensors*, vol. 18, no. 8, p. 2619, 2018.

[18] T. Zheng, W. Xie, L. Xu et al., "A machine learning-based framework to identify type 2 diabetes through electronic health records," *International Journal of Medical Informatics*, vol. 97, pp. 120–127, 2017.

[19] S. Yakoubov, *A Gentle Introduction to Yao's Garbled Circuits*, Boston Univeristy, Boston, MA, USA, 2019.

[20] S. Zahur and D. Evans, "Obliv-C: a language for extensible data-oblivious computation," *IACR Cryptology ePrint Archive*, vol. 2015, p. 1153, 2015.

[21] X. Fang, S. Ioannidis, and M. Leeser, "Secure function evaluation using an FPGA overlay architecture," in *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pp. 257–266, Monterey, CA, USA, February 2017.