A Parser-Based Data Collector for Intrusion Detection

Grégor Quétel, Eric Alata, Pierre-François Gimenez, Laurent Pautet, Thomas Robert

A parser-based data collector for intrusion detection

Grégor Quétel[1], Eric Alata[2], Pierre-François Gimenez[3], Laurent Pautet[1], and Thomas Robert[1]

[1]LTCI, Télécom Paris, Institut Polytechnique de Paris, {firstname.lastname}@telecom-paris.fr
[2]LAAS CNRS, eric.alata@laas.fr
[3]CentraleSupélec, Univ. Rennes, IRISA, pierre-francois.gimenez@centralesupelec.fr

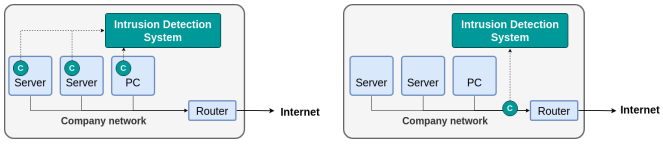
***Abstract*—Intrusion detection systems often struggle to identify attacks directed at applications. A contributing factor is the various syntactical forms these attacks can take. This paper introduces a methodology to design and adapt applicative data collectors (DCs) to software projects by integrating them into the application's parsers. This data collector aims to enhance applications' security by providing semantic information to intrusion detection mechanisms.**

***Index Terms*—Application-level data collector, Semantics, Intrusion Detection System**

I. Introduction

Intrusion Detection Systems (IDS) are tools designed to monitor information systems and detect any abnormal content. Typically, an information system is equipped with DCs positioned to protect valuable assets. They transmit their observations to a central intrusion detection mechanism, which analyzes the received data and triggers alerts in the event of detecting abnormal behavior. IDSes based on observations at the network or host level fail to detect attacks targeting applications. The former because the traffic is mostly encrypted nowadays [1], the latter because observations made at the host level fail to reflect events taking place at the application level [2]. We intend to detect attacks targeting applications and therefore intend to position our DC at this abstraction level. Attacks directed towards applications generally have multiple syntactical variants with equivalent semantics: they possess the same impact on an application but are expressed differently. For instance, with MySQL, the queries `Select * from users where role='admin';` and `Select * from users where role='adm' 'in';` are semantically equivalent but syntactically different. Existing works that base their analysis on applicative data mostly focus on syntactical features such as string matching or signature analysis.

However, the malicious nature of inputs is fundamentally determined by their impact on the system. Producing information regarding the impact of user input on an application rather than the specifics of how it is formulated would allow better anomaly detection [3]. Furthermore, prevalent approaches are usually integrated into the internals of one specific application and can hardly be applied to a different software project. In this paper, we present the methodology to integrate data collectors in applications' parsers to produce application-level information for security purposes. Specifically, we focus on generating semantic information associated with the application inputs. The DC utilizes the application codebase to minimize integration costs and to automate most of its deployment. The paper is organized as follows. Section II presents widespread data collection approaches. Section III describes the characteristics of our DC. Finally Section IV highlights the goal of the first author's PhD thesis.



(a) Host-based architecture (b) Network-based architecture

Fig. 1: The position of DCs for intrusion detection

II. Related work

Generally, IDSes are characterized according to the nature of the data they utilize to perform detection. DCs most often provide either network or host information to the detection mechanism[1], [4]. Network-based IDS (NIDS) DCs typically consist of dedicated networking hardware added to a network to monitor a multitude of hosts. Their ability to detect application-level attacks is hindered by increasingly encrypted traffic. Most often, Host-based IDS (HIDS) collection mechanisms involve a software component (agent) that analyzes the system on which they are deployed. An agent is installed on each host within a network that needs to be monitored. Information provided to the detection mechanism usually consists of system call sequences, file system access or system logs. Figure 1 illustrates the difference in DC positioning between HIDS and NIDS. Application-based IDS utilize their domain knowledge to provide information about events taking place at the application level [1], [5]. Typically, they consist of the native audit functions provided by applications. However, these mechanisms impact the performance of applications because they are not optimized for production deployment. Finally, these features are rendered useless in the scenario where a privileged user has been compromised, as they can disable them. [1], [5].

III. Approach

We present a data collector integrable to multiple applications with minimal engineering integration costs. The DC produces information characterizing each application input and is strategically positioned within parsers. Parsers break down inputs into structured data using a specification on how to

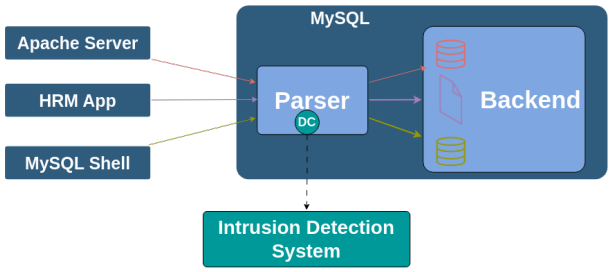


Fig. 2: The position of our DC in a typical information system

analyze and verify the syntax of the input: the grammar file. This grammar file is composed of parsing rules. By inferring the semantics associated with each rule before the application deployment, we can at runtime, construct each parser input's associated semantic.

A. Position within applications

Information systems are made of the interconnection of different applications. Consequently, when an attacker targets an application or the resources it manages, they have access to multiple means to carry out their attack. The parser is an application component that remains unavoidable regardless of the input's origin. We ensure to produce information for every application input by placing our DC into this component. Finally, most often, the parser is automatically generated using parser generator tools such as Bison or ANTLR, which facilitates the generalization of our approach. In Fig 2 we illustrate the positioning of our parser-based DC.

B. Information produced by the data collector

The data collector position allows it to transmit both semantical and syntactical information such as details about utilized keywords or processing rules for each received input, to the intrusion detection mechanism. However, our methodology focuses on the production of semantic information. Existing formal semantic approaches are mostly designed for safety and correctness purposes. Therefore, we defined different *semantic meta-models* adapted to our security concerns that aim to abstract the impact of inputs through different characteristics:

- Meta-model A - $< action >$: at the most basic abstraction level, each input is characterized by an action: a verb in natural language describing the impact of the query on the system, e.g. `create`, `delete`, `modify`.
- Meta-model B - $< action, object >$: with this model, we also provide the information of the object on which the action is performed, e.g. `file`, `user`, `database`.

Applications we plan to instrument belong to different software categories and allow different actions on a system. For instance, a Database Management System allows creating, deleting, and modifying data, while a web server allows receiving and sending data. Actions and object classes relevant to the former might not be relevant to the latter. Therefore, for each project (or project family), a relevant *semantic model* will be instantiated (i.e. an application using the meta-model B will need to be instantiated with action and object classes). Subsequently, given the semantic model, we plan to have our data collector automatically associate a *semantic* to each parser input.

C. Deployment

Automatically generated parsers are built from a standardized grammar file containing the parser rules. Therefore, we can easily inject into the grammar file, code consisting of the semantics associated with each rule. Additionally, we inject a routine that aggregates them to construct the overall input semantic and transmit it to the detection mechanism.

IV. Conclusion and Goal of the PhD

In this paper, we introduced the idea of a novel approach to produce applicative-level information for detection purposes. The Ph.D. thesis of the first author centers on designing the described DC and the associated intrusion detection mechanism. Before providing DC prototypes two major challenges must be addressed. First, because many security tools are simply ignored due to their deployment complexity, we plan to define a process to automatically instantiate the semantic model for each application. Then we plan to establish a methodology whereby, given a *semantic model* and a set of parser rules, we can automatically associate a semantic to each rule. To perform this task we can leverage software project information such as grammar files, application codebase or online documentation. Various techniques are available to infer a label from these sources such as the computation of similarity measures on text data or automatic source code summarization on codebase [6].

Acknowledgments

This work has been partially supported by the French National Research Agency under the France 2030 label (Superviz ANR-22-PECY-0008). The views reflected herein do not necessarily reflect the opinion of the French government.

References

[1] R. A. Bridges, T. R. Glass-Vanderlan, M. D. Iannacone, M. S. Vincent, and Q. (Chen, "A Survey of Intrusion Detection Systems Leveraging Host Data," *ACM Computing Surveys*, 2019.

[2] X. Jin and S. L. Osborn, "Architecture for Data Collection in Database Intrusion Detection Systems," in *Secure Data Management*, 2007.

[3] S. Mathew, M. Petropoulos, H. Q. Ngo, and S. Upadhyaya, "A Data-Centric Approach to Insider Attack Detection in Database Systems," in *Recent Advances in Intrusion Detection*, 2010.

[4] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, 2019.

[5] X. Jing, Z. Yan, and W. Pedrycz, "Security Data Collection and Data Analytics in the Internet: A Survey," *IEEE Communications Surveys & Tutorials*, 2019.

[6] C. Zhang, J. Wang, Q. Zhou, *et al.*, "A Survey of Automatic Source Code Summarization," *Symmetry*, 2022.