Fuzzy Rules Based Solution for System Administration Security Management via a Blockchain

Arnaud Castelltort, Antoine Chabert, Nicolas Hersog, Anne Laurent, Michel Sala

▶ **To cite this version:**

Arnaud Castelltort, Antoine Chabert, Nicolas Hersog, Anne Laurent, Michel Sala. Fuzzy Rules Based Solution for System Administration Security Management via a Blockchain. BLOCKCHAIN, Jun 2019, Ávila, Spain. lirmm-02085775

HAL Id: lirmm-02085775

https://hal-lirmm.ccsd.cnrs.fr/lirmm-02085775

Submitted on 7 Oct 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Fuzzy Rules Based Solution for System Administration Security Management via a Blockchain

Arnaud Castelltort[1], Antoine Chabert[2], Nicolas Hersog[2], Anne Laurent[1], and Michel Sala[1]

[1] Univ Montpellier, LIRMM, CNRS, Montpellier, France
{firstname.lastname}@umontpellier.fr,
WWW home page: http://www.lirmm.fr
[2] ChainHero, Montpellier, France
nicolas.hersog@chainhero.io,
WWW home page: www.chainhero.io

Abstract. Digital transformation has led to the fact that almost all organizations and companies are provided with internal private networks and manage sensitive data and applications. In this context, system administrators are superusers who can access all this sensitive material. As it is known that many frauds are caused by internal actions, we argue that it is important to be provided with strong automated logging systems even for superusers. For this purpose, blockchains are an efficient solution as they cannot be overwritten by the system administrators. However, as it is not efficient to store all the actions, we introduce in this paper a novel system based on fuzzy rules in order to efficiently manage the system logging system in a blockchain.

Keywords: blockchain, fuzzy rules, system administration, log files, fraud detection.

1 Introduction

As the organizations are more and more using digital solutions to manage their internal processes and data, sensitive material is more and more managed by the information system. Security is thus a critical issue and is a major concern and expense. Information systems and infrastructures are thus protected and all operations are logged in journals. However, it has been reported by the U.S. State of Cybercrime that *"23% of electronic crime events were suspected or known to be caused by insiders"* and that *"45% of the respondents thought that damage caused by insider attacks was more severe than damage from outsider attacks [...] customer records compromised or stolen, confidential records (trade secrets or intellectual property) compromised or stolen, and private or sensitive information [...] unintentionally exposed"* [10]. As system administrators have access to sensitive information without any restriction as being *root* on the systems,

it is important to be able to monitor their operations so that they do not take benefit from their position while ensuring to cover up their activities.

For this reason, the ChainHero company has built the Blockaudit system that allows to report in a blockchain the system administrator operations. This prevents from any falsification. The architecture is described in Figure 1, the smart engine being responsible for choosing the operations being logged in the blockchain.

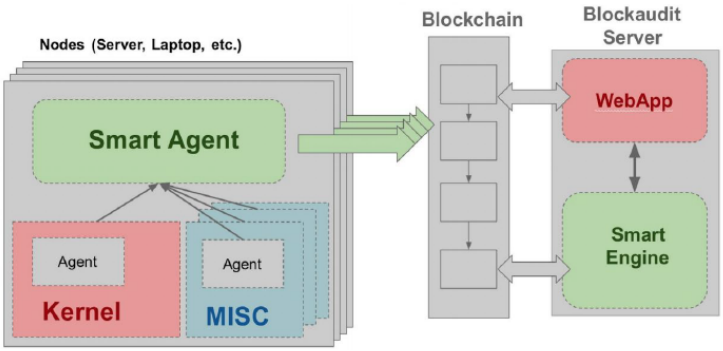

Fig. 1. Current Crisp Architecture

The Hyperledger Fabric private blockchain has been chosen by the company. The blockaudit agents are spread over the network on the critical servers and report all actions on the blockchain, using the *auditd* tool developed by RedHat and delivering information from the Linux kernel. Every agent is provided with predefined behaviours so as to determine which operations are reported, e.g., file access or network operations. Rules can be added so as to follow up specific operations, as for instance determining softwares, users or files to be taken into account. Such operations are then collected and stored in the blockchain so as to be reported for users. Rules can be associated with the level of criticality.

However, these rules are too crisp to be easily applied as the operation needs to perfectly feet the rule so as to be detected as critical, which may lead to problems.

For instance, if getting connected 5 times on a server in the night is considered as critical then an ill-intentioned person who would connect only 4 times or one minute after the night would not be discovered.

In this work, we have thus proposed to soften the rules by relying on fuzzy rules. Our solution has been implemented and tested.

2 Related Works

2.1 System Administration and Audit

Auditing systems is a key element for managing the security [12]. Several risks are associated with such insiders [5]: data leaks, access control,... Data leakage can be of different forms and data leakage prevention systems (DLPSs) have been designed [1]. Some of them are focused on internal threads, as in [8, 9].

In this work, we focus on the operations close to the Linux kernel. For this purpose, we focus on auditd which is the userspace component to the Linux Auditing System and is responsible for writing audit records [11].

In our work, we rely on fuzzy rules, described below.

2.2 Fuzzy Rules

When dealing with rules, the systems are based on a set of *if ... then...* statements where the antecedent is either true or false. However, in some cases, it is important to soften this in order to consider antecedents that are partially true. For this purpose, the fuzzy logic framework [3] is relevant. In such a framework, objects gradually belong to the fuzzy subsets with a membership degree ranging from 0 to 1. Every fuzzy set is defined over a universe and is associated with a fuzzy membership function that allows to compute this degree.

Fuzzy rules are then of the form *if A then B* where A and B are fuzzy sets, as for instance *if the number of connections is high then the level of criticity is low*. Multiple variables can be aggregated by using t-norms and t-conorms, as for instance *if the number of connections is high AND the period is night then the level of criticity is high* where *low*, *high* and *night* are fuzzy concepts described by their membership function.

Figure 2 describes a fuzzy membership function for the concept of night, the universe being the hours (from 0 to 24) in a day.

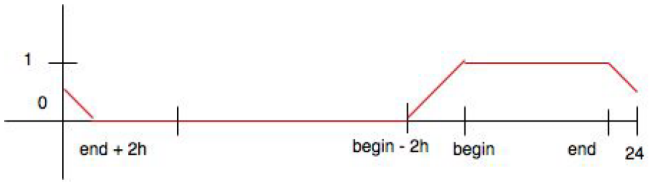

Fig. 2. Fuzzy Membership Function Describing Night and Day

Fuzzy quantifiers can be defined in the same manner in order to describe quantities and percentages with words, as for instance *many, almost 5, few, etc.*

Fuzzy concepts are used in our approach in order to describe the concepts appearing in the rules in a soft and understandable manner as crisp partitions are often impossible for users to define.

It has already been shown that it is interesting to link blockchain and artificial intelligence [7], and fuzzy logic [4, 6], as it is the case for cyber security [2]. But no work has coupled blockchains and fuzzy rules as proposed in our work and described below.

3 FBA: The Fuzzy BlockAudit approach

Our work aims at defining a solution to store the operations of the system administrators of critical servers in a secure manner. We thus rely on blockchains. The operations are taken from the *auditd* tool on Linux kernel. As we claim that it is not necessary to store all operations, we define rules that determine what operations deserve to be written in the blockchain. These rules depend on so-called *contexts* as described below.

3.1 Contexts

Contexts are used in the definition of fuzzy variables. For example, *connection* and *command* have been implemented in our solution: they are used to define the fuzzy sets of *low* and *high* for both contexts (connection and command). The fuzzy variables are then used to:

- determine if a rule must be triggered;
- retrieve the information from the event;
- compute to which extent the premise of a rule is matched;
- update the premises of the rules.

For instance, in the *connection* context, the value 10 represents a *high* number of connections whereas this value is rather *low* in the *command* context.

3.2 Fuzzy Rules

These fuzzy variables are then used in fuzzy rules. Rules are of the form:

```
RULE RuleName [ Expression ] AS CriticalityType
```

where expression is recursive and is composed of other expressions connected with logical AND, OR, XOR operators. In a rule, the premise is of the form

```
AMOUNT (context) SPEED FROM (ACCESS -> ACCESS) AT (PERIOD)
```

where:

- AMOUNT(context) is the number for triggering the rule for the given context.
- SPEED allows to follow up the events over time.
- AT allows to restrict the period.

For instance, a premise may be only valid for holiday periods. It should be noted that the definition of the periods is not easy as nights overlap 2 days and nights, and cannot be defined as a single subset of the hours between 0 and 24 using modulo 24.

A critical level is also considered. $WEAK$ critical level (as for instance *cd* commands, read and write on non sensitive files) are distinguished from $NORMAL$ critical level (as for instance change of user, access to a shared folder), $STRONG$ critical level (as for instance the mounting of a folder, the use of startup script), and $CRITICAL$ (as for instance reboot, change of audit rules, adding or deleting a user account, read and write on sensitive files).

Depending on the level of criticity, the events on premises are aggregated using various operators, as listed in Figures 3 and 4.

Fig. 3. Aggregating Degrees with AND Operator (t-norm)

Criticality	T-Norm
WEAK	f(a,b) = a * b
NORMAL	f(a,b) = 0.7 * min(a,b) + (1 - 0.7)(a + b)/2
STRONG	f(a,b) = 0.8 * min(a,b) + (1 - 0.8)(a + b)/2
CRITICAL	f(a,b) = min(a,b)

Fig. 4. Aggregating Degrees with OR Operator (t-conorm)

Criticality	T-Conorm
WEAK	f(a,b) = max(a,b)
NORMAL	f(a,b) = 0.8 * max(a,b) + (1 - 0.8)(a + b)/2
STRONG	f(a,b) = 0.7 * max(a,b) + (1 - 0.7)(a + b)/2
CRITICAL	f(a,b) = 1 - (1 - a)(1 - b)

4 Architecture and Implementation

Our solution is based on the management of the operations that trigger fuzzy rules as described previously if they are considered as worth being stored in the blockchain. The fuzzy rules are defined thanks to a DSL.

4.1 Domain Specific Language

As the rules must be easily configured by using a simple language, we use an external DSL that allows to define the rule with the following grammar.

```
RULE identifier
OL (logical operator): [AND, OR, XOR]
```

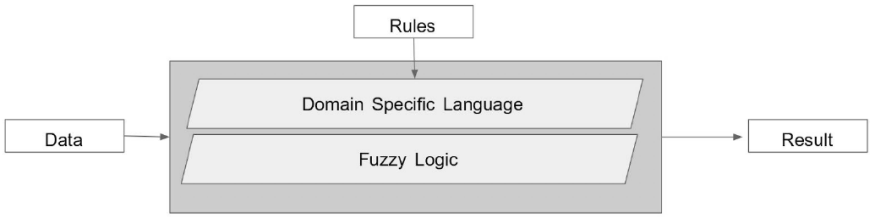

Fig. 5. Current Architecture

```
Parenthesis: [(, )]
Expression <- (Expression) OL Premise
Expression <- Prmisse OL Expression
Premise:
AMOUNT: [HAPPEN, FEW,MODERATE,  MANY]
SPEED: [SLOWLY, AVERAGE, QUICKLY]
ACCESS: [USER, ROOT, SYSTEM]
PERIOD : AT( [NIGHT, OFFICE, HOLIDAYS, WEEKEND] ).
CRITICITY
 [AS] [WEAK, NORMAL, STRONG, CRITICAL]
```

A rule always starts with

```
[RULE]
```

An example is given by Figure 6.

```
RULE exempleRule [
        FEW(connection) QUICKLY FROM(USER -> ROOT) AT(NIGHT)
        OR
        MANY(command) SLOWLY AT(NIGHT)
]
AS CRITICAL
```

Fig. 6. Example of a Rule

4.2 Architecture

Our solution is accessible through an API (Application Programming Interface) exposed on the HTTP protocol. All components are easily configurable by the users with several modules (see Figure 8): ContextsModule for the management of contexts, DSL, ParserModule for the management of events, RulesModule for the management of rules, FuzzyLogicModule for the management of fuzzy

operators. A document-based NoSQL storage backend is used as an efficient storage for raw textual data from JSON that are pre-treated.

One of the key aspect of this software architecture is the ability to be extended with context modules. Rogue behaviors are behavioral anomalies that can occur in human activities and that can thus be retrieved from human generated data. The context modules allow the user expert to override some predefined rules to adapt them to a specific context and also to extend the system with new or domain specific fuzzy terms, rules and then ultimately more rogue behavior definitions.

Figures 7 and 8 show the architecture and technologies being used.

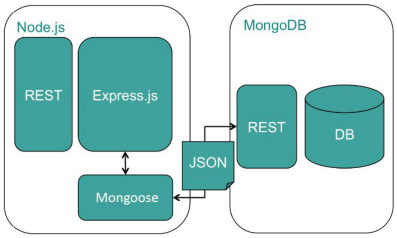

Fig. 7. Technical Architecture of the Solution

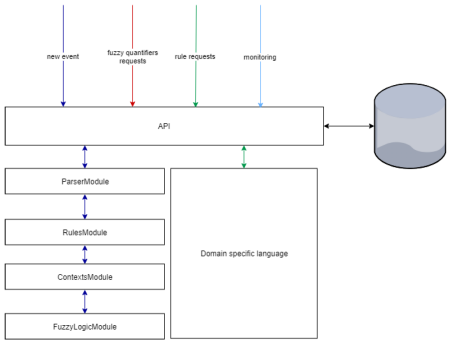

Fig. 8. Architecture of the Solution

The solution is easily pluggable in an environment with an HTTP event-based application server such as NodeJS server. It is possible to dynamically configure, add, and delete rules through the API. It does not require to reboot and restart which could allow the system to adapt itself while running (which is a further work). Data visualization and monitoring systems are available allowing the user to take decisions.

5 Conclusion

In this paper, we present an original method for managing system administrator operations logging in a blockchain by the use of fuzzy rules. This approach allows companies to prevent system administrators to take benefit from their position for attacking critical servers while ensuring to cover up their activities and to erase their traces. The solution has been implemented, tested and is in deployment in the information system of a confidential client.

Our future work includes the automatic learning of rule evolutions and the monitoring of operations outside the Linux kernel monitored by auditd.

6 Acknowledgments

The authors would like to thank Loïc Combis, Kevin Hassan, and Hugo Maitre, students from Polytech Montpellier, for their help for implementing and testing the approach.

References

1. Sultan Alneyadi, Elankayer Sithirasenan, and Vallipuram Muthukkumarasamy. A survey on data leakage prevention systems. *J. Netw. Comput. Appl.*, 62(C):137–152, February 2016.
2. Sulaiman Al Amro, Francisco Chiclana, and David A. Elizondo. Application of fuzzy logic in computer security and forensics. In David A. Elizondo, Agusti Solanas, and Antoni Martnez-Ballest, editors, *Computational Intelligence for Privacy and Security*, volume 394 of *Studies in Computational Intelligence*, pages 35–49. Springer, 2012.
3. R. E. Bellman and L. A. Zadeh. Decision-making in a fuzzy environment. *Manage. Sci.*, 17(4):B–141–B–164, December 1970.
4. Rui-Yang Chen. A traceability chain algorithm for artificial neural networks using ts fuzzy cognitive maps in blockchain. *Future Generation Computer Systems*, 80:198–210, 2018.
5. Ivan Homoliak, Flavio Toffalini, Juan Guarnizo, Yuval Elovici, and Martín Ochoa. Insight into insiders: A survey of insider threat taxonomies, analysis, modeling, and countermeasures. *CoRR*, abs/1805.01612, 2018.
6. Yosuke Kaga, Masakazu Fujio, Ken Naganuma, Kenta Takahashi, Takao Murakami, Tetsushi Ohki, and Masakatsu Nishigaki. A secure and practical signature scheme for blockchain based on biometrics. In Joseph K. Liu and Pierangela Samarati, editors, *Information Security Practice and Experience - 13th International Conference, ISPEC 2017, Melbourne, VIC, Australia, December 13-15, 2017, Proceedings*, volume 10701 of *Lecture Notes in Computer Science*, pages 877–891. Springer, 2017.
7. Tshilidzi Marwala and Bo Xing. Blockchain and artificial intelligence. *CoRR*, abs/1802.04451, 2018.
8. Lance Spitzner. Honeypots: Catching the insider threat. In *Proceedings of the 19th Annual Computer Security Applications Conference*, ACSAC '03, pages 170–, Washington, DC, USA, 2003. IEEE Computer Society.
9. Salvatore J. Stolfo, Malek Ben Salem, and Angelos D. Keromytis. Fog computing: Mitigating insider data theft attacks in the cloud. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy Workshops*, SPW '12, pages 125–128, Washington, DC, USA, 2012. IEEE Computer Society.
10. R. F. Trzeciak. Sei cyber minute: Insider threats, http://resources.sei.cmu.edu/library/asset-view.cfm?assetid= 496626, 2017.
11. H. Xu and R. Tang. Study and improvements for the real-time performance of linux kernel. In *2010 3rd International Conference on Biomedical Engineering and Informatics*, volume 7, pages 2766–2769, Oct 2010.
12. K. Zhao, Q. Li, J. Kang, D. Jiang, and L. Hu. Design and implementation of secure auditing system in linux kernel. In *2007 International Workshop on Anti-Counterfeiting, Security and Identification (ASID)*, pages 232–236, April 2007.