Session-dependent Usage Control for Big Data

Gabriele Baldi[1], Yair Diaz[2], Theo Dimitrakos[2], Fabio Martinelli[1], Christina Michailidou[1*], Paolo Mori[1], Oleksii Osliak[1] and Andrea Saracino[1]
[1]Consiglio Nazionale delle Ricerche, Istituto Informatica e Telematica, Pisa, Italy
name.surname@iit.cnr.it
[2]German Research Center, Huawei Technologies Dusseldorf GmbH, Munich, Germany
name.surname@huawei.com

Abstract

Business strategies are increasingly driven by the integrated analysis of huge volumes of heterogeneous data, coming from different sources such for example social media or Internet of Things devices. The so called Big Data are considered as relevant assets by companies and organizations, since they can be analysed to create new valuable knowledge and insights that could help managers in their strategic decisions. The full potential of Big Data could be realized if the information was coming from several distinct sources, with different characteristics and target audience. Although, data producers are not always willing to share their data with other companies due to lack of trust and the absence of a data protection framework which can be adopted in a Big Data environment. In this work, we present BigUCON, a framework which exploits the Usage Control paradigm in order to provide an enhanced, expressive and flexible authorization support for data protection within the aforementioned environment. The framework is integrated in Apache Hadoop, a software library which provides the infrastructure for storing, mining and processing large data sets through a collection of open-source software.

Keywords: Big Data, Usage Control, Access Control, Hadoop

1 Introduction

Undeniably, the interaction of the people with any kind of Internet services, from social media to e-banking, has become a daily task. Driven by the continuous proliferation of devices that can be easily connected to the Internet, including sensors, smart phones, connected cars, smart appliances, wearables and many more, we face the evolution from the Internet of Things (IoT) towards the Internet of Everything (IoE), where physical objects, services, people and processes are interacting with each other, sharing activities and data to their mutual benefit.

While this transition offers great opportunities, the datasets that are being created nowadays exceed the typical datasets both in size and complexity. Thus, the term Big Data receives during the past few years a great amount of attention both from the academia and the industry. This term refers to sets of data, both structured and unstructured, that are too large or too complex in order to analyze and process them exploiting traditional data processing techniques. In 2001 the industry analyst Doug Laney coined a currently widely used definition of Big Data based on the three V (Volume, Variety and Velocity): "Big data technologies describe a new generation of technologies and architectures designed to economically extract value from very large Volumes of a wide Variety of data, by enabling high Velocity capture, discovery, and/or analysis" [16].

Journal of Internet Services and Information Security (JISIS), volume: 10, number: 3 (August 2020), pp. 76-92
DOI: *10.22667/JISIS.2020.08.31.076*
*Corresponding author: Consiglio Nazionale delle Ricerche, Istituto Informatica e Telematica, Pisa, Italy, Tel: +393347649376

Big data are considered as assets by companies since through their analysis and process new knowledge and insights are created that could be exploited by the company or the organization and lead their managers to better business and strategic decisions. The ways that Big Data can support value creation for companies include and are not limited to:

- Creating transparency by making big data openly available for business and functional analysis (quality, lower costs, reduce time to market, etc.)
- Supporting experimental analysis in individual locations that can test decisions or approaches, such as specific market programs
- Assisting, based on customer information, in defining market segmentation at more narrow levels
- Supporting real-time analysis and decisions based on sophisticated analytics applied to data sets from customers and embedded sensors
- Facilitating computer-assisted innovation in products based on embedded product sensors indicating customer responses

The nature of Big Data can be characterized as sensitive since they usually include personal information of customers and employees as well as financial and trading information, user's profiles etc. Thus, protecting those data and ensuring that only authorized entities will have rights to perform actions over them is crucial. Although many data protection frameworks have been proposed both by academia and companies implementing traditional and enhanced access control models, they are not meant and they cannot be easily adapted to be used with Big Data. Consequently, most of the currently existing Big Data platforms integrate quite basic access control enforcement mechanisms [4].

In this paper we propose the integration of Usage Control (UCON) paradigm in Apache Hadoop software library, aiming at providing an effective and efficient framework for the protection of the datasets which are present in this environment. UCON paradigm [17, 19] is an improvement of traditional access control which enables the definition and enforcement of complex policies on resources' usage, considering continuous policy enforcement and attributes whose values change over time. As a matter of fact, UCON introduces the concept of revocation of ongoing access when the related policy is not valid anymore as a consequence of an attribute update. UCON model is a promising approach for access control in open, distributed, heterogeneous and network-connected computer environments [13] and the authors made a first integration of the model in cloud systems in [12, 2]. A relevant novelty of the proposed solution is that authorizations are organized through an hierarchy, which represents dependencies among them. The main idea is that, in case the authorization paired with a given node of the hierarchy is not valid anymore because of an attribute update, all the accesses in the subtree rooted in this specific node must be revoked.

The rest of the paper is organized as follows. Section 2 presents the theoretical background of UCON and Apache Hadoop, Section 3 is devoted to the presentation of the proposed solution, Section 4 presents the utilized use case, Section 5 reports the related work and Section 6 concludes this work.

2 Background

This section provides a brief theoretical background of Usage Control and the Apache Hadoop software library.

2.1 Usage Control

The Usage Control model extends traditional access control models mainly introducing *mutable attributes*, which represent features of subjects, resources, and environment that change their values over time [18]. Since mutable attributes change their values during the usage of an object, the usage control model allows to define policies which are evaluated before *(pre-decision)* and continuously during the access to the object (*ongoing-decision*). The continuous evaluation of the policy when the access is in progress is aimed at executing proper countermeasures (i.e interrupting the access) when the execution right is no more valid, in order to reduce the risk of misuse of resources. Hence, in UCON model it is crucial to be able to continuously retrieve the updated values of the mutable attributes, in order to perform the continuous evaluation of the policy and to promptly react to the attribute change by taking proper actions, e.g., by interrupting those ongoing accesses which are no longer authorized.

This paper takes into account the UCON system based on the XACML reference architecture presented in [2], whose architecture is shown in Figure 1. In the XACML reference architecture, the Policy

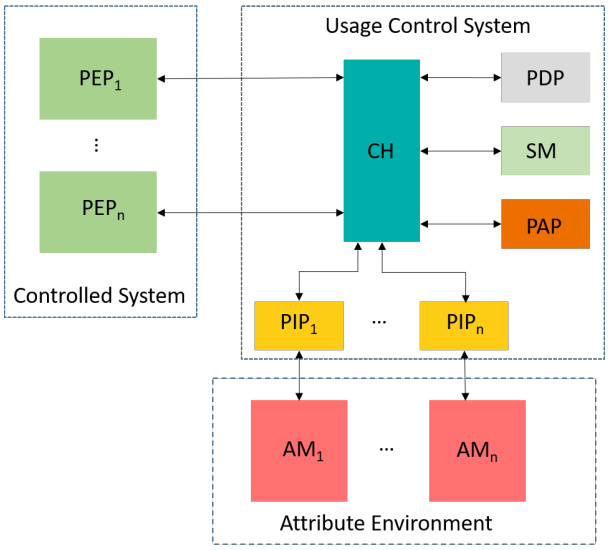


Figure 1: Usage Control System architecture.

Enforcement Points (PEPs) embedded in the controlled system intercept the execution of security relevant operations, and they invoke the Context Handler (CH), which is the frontend of the UCON system. The *Policy Information Points (PIPs)* are the components invoked by the CH to retrieve the attributes required by the Policy Decision Point (PDP) for the execution of the decision process. Attributes are managed by Attribute Managers (AMs), which provide the interfaces to retrieve their current values. Each specific scenario requires its own set of AMs to manage the attributes required for the policy evaluation (e.g, LDAP servers or SQL databases could be exploited). Hence, PIPs are properly configured in order to be able to query the specific AMs adopted in the scenario of interest for retrieving the current values of attribute and to detect their changes.

The workflow of the UCON decision process can be summarized as follows. When the subject s tries to execute a security relevant action a, PEP suspends its execution and retrieves the information related to this access (subject IDs etc.). The PEP sends the *TryAccess* message with the data previously collected to the UCON system, which performs the *pre-decision* process and returns the result to the PEP (*permitaccess* or *denyaccess*), which enforces it. If the execution of a is permitted, the PEP sends the *StartAccess* message to the UCON system as soon as a is started, to start the *ongoing-decision* phase.

Again, the UCON system performs the first evaluation of the UCON policy. From this moment on, as long as the action *a* is in progress, the UCON service evaluates the UCON policy every time an attribute changes its value. If the policy is violated, the UCON system sends the *RevokeAccess* message to the PEP, in order to take proper countermeasures.

2.2 Apache Hadoop

Apache Hadoop [1] is one of the most widely and commonly used frameworks that provides services and solutions for handling Big Data, Hadoop provides the necessary infrastructure for storing, mining, processing and analyzing large data sets through a collection of open-source, Java-based software. The applications which form the framework of Hadoop are designed to run in clustered systems and highly distributed environments and are capable of managing various forms of data, from structured such as those coming from transactions to totally unstructured piece of information such as social media posts. Figure 2 depicts the Apache Hadoop Ecosystem. The core component of Hadoop ecosystem is its dis-

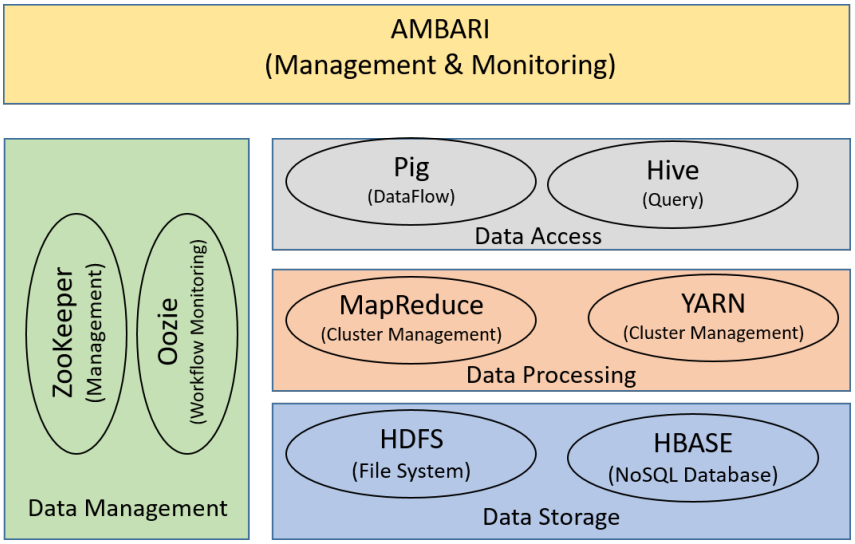


Figure 2: Apache Hadoop Ecosystem.

tributed file system, called Hadoop Distributed File System (HDFS), which is responsible to provide high-performance and fault-tolerant access to data across Hadoop's clusters. One of the main characteristics of HDFS is the support of a parallel processing of the data, since its functionality relies on the fact that the incoming information are broken down into separate blocks and distributed in different nodes of the cluster, making sure always that replicates and copies have been created on different servers. The need of a strong and flexible usage control model to regulate the access over this amount of data is the main motivation behind the integration of UCON model to Hadoop.

2.3 Apache Ranger

Apache Ranger is the access control solution provided by Hadoop, which includes an administration portal exploited by security administrators in order to write, edit, modify and delete access control policies, stored in a dedicated database.

[1] https://hadoop.apache.org/

Each one of the components of Hadoop Ecosystem, such as Knox or Hive, is configured with a Ranger Plugin. The core responsibility of this plugin is to pull the policies which correspond to the specific component from the policy database and perform a policy evaluation. Hence, a request to a Hadoop component is intercepted from its associated Ranger Plugin, which following retrieves the corresponding security policy and evaluates the request against it. The implemented solution integrates UCON in Ranger Plugin. This integration is being realized by enabling the Apache Ranger administration portal with a User Interface in order to give the possibility of defining UCON policies as it will described in more details in the next sections.

3 BigUCON

Following the overall architecture of the proposed framework will be presented.

3.1 Architecture

In this architecture, assets (i.e. services, data, etc.) are protected by PEPs whose main function is to intercept incoming access requests, and to process and transform them into enriched UCS requests to be forwarded to the UCS component for policy evaluation. Protected assets are segmented and grouped by domain, e.g. functional, security, administrative, etc., depending on the scenario/use case.

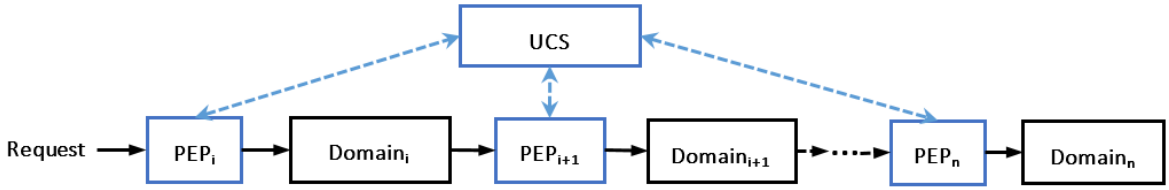


Figure 3: UCON-BigData multi-tiered architecture.

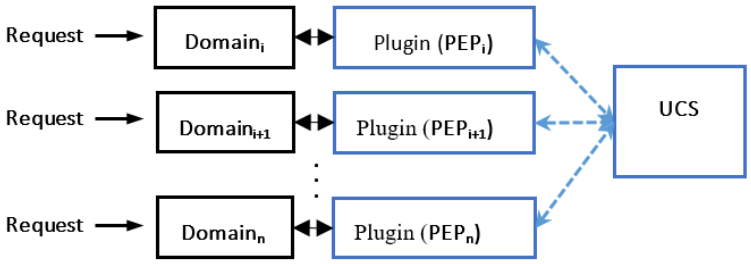


Figure 4: UCON-BigData plugin-based architecture.

Figure 3 depicts an architecture consistent with a typical multi-tiered organizational IT infrastructure deployment where access requests are intercepted before passing flow control to the protected assets. Figure 4 depicts a plugin-based architecture consistent with the Hadoop-Ranger ecosystem where the protected assets intercept the access requests and delegates the access control functionality to the plugin. In both cases access to the protected resources is determined by the UCS component.

3.2 Global context

In a typical centralized system the authorization architecture consists of a dedicated authorization service that makes evaluation decisions in response to the authorization requests made by the PEPs in the system. Each PEP establishes communication directly with the authorization service and there is single independent session to handle the authorization. However, in scenarios targeted by the BigUCON architecture is not always sufficient or adequate to have single independent session. For example, consider a scenario where an organization needs to define authorization policies to control access to its IT infrastructure: Restful APIs and database systems as shown in the multi-tiered diagram above. In the case of the Restful APIs, the authorization policies are typically defined on the basis of the organizational and administrative structure, i.e. marketing, finance, HR, etc.; and on the type of services exposed e.g. internal vs public. In the case of database systems, the authorization policies are defined based on privacy, regulatory, data governance and compliance requirements. An authorization request made to an API endpoint may in turn trigger a database request to access some data; thus creating two interrelated but separate authorization sessions started at the API interface and at the database interface, respectively. In this example there are two sets of policies that originate from different sets of requirements but whose evaluation need to co-exist in harmony from the perspective of the overall system. Within the context of BigUCON, the two sessions created are said to form an authorization or global context.

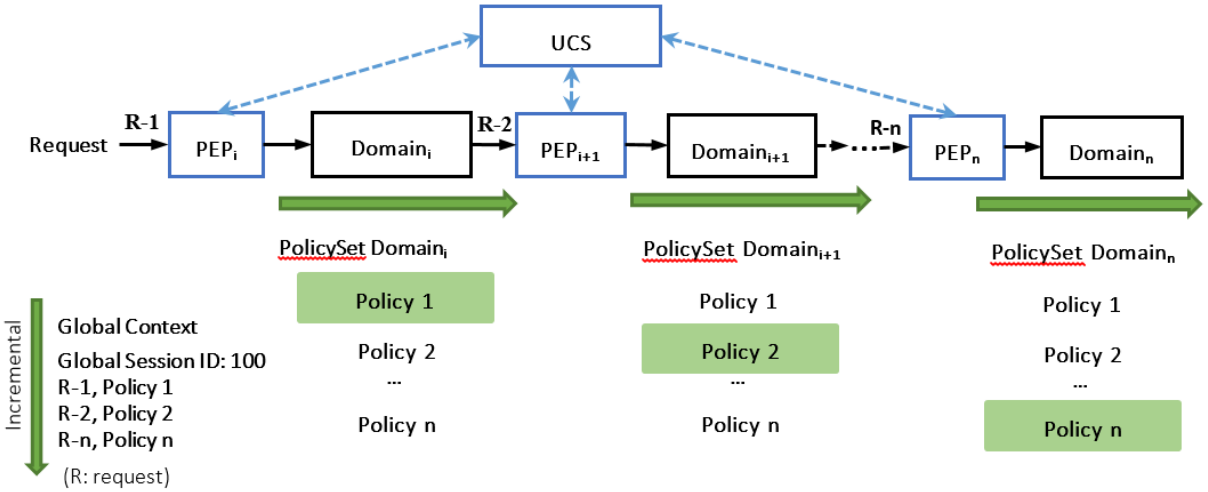


Figure 5: Global context and distributed tracing.

The global context emerges from the propagated requests from left to right, the applicable triggered policies, and the independent authorization sessions generated by the UCS. In Figure 5, the global context is represented by the green policies and request arrows.

3.3 Distributed Context Propagation

In order to be able to aggregate and build the global context about the independent sessions it is required to pass tracing metadata as the requests propagate from left to right across the components. This is known as (distributed) context propagation. The BigUCON framework supports passing tracing metadata along the initial request execution path. In Figure 5, the initial request R-1 triggers request R-2, and in turn R-2 triggers R-n as requests propagate. In this example the global session Id 100 is propagated with the requests' payload.

One of the most popular distributed context propagation specifications is OpenTracing which uses the mechanism of "baggage" to pass tracing data. On the implementation side, context propagation is achieved by means of instrumentation components deployed on the nodes of the target system. An example of instrumentation in Java consists in attaching the instrumentation components in the Java Virtual Machine (JVM) in order to hook to live threads and inject metadata. The metadata is then propagated as part of the requests made between nodes and according to the communication protocol, e.g. JDBC, HTTP, etc.

3.3.1 Authorizations Hierarchy

As previously discussed, in the proposed system the access requests are propagated among several components, having different elements of the request being evaluated at each step. This propagation creates a dependency relation among authorizations, which can be expressed as an hierarchy tree.

Figure 6 depicts a possible tree structure which represents the correlation among different authorizations for a set of services distributed across several layered domains. Let us suppose that a subject wants to perform a set of operations over a group of data. The first authorization (AuthAP) will take place once the subject performs the requests to access the platform (AP). Afterward, given that the first authorization is granted, the subject will be able to control a number of different entities (i.e. data centers, devices etc) in order to run analytics operations (AS) over the data belonging to this specific entity. Subsequent, AS will initiate a number of calls to several Big Data services (BD) for the purposes of the analysis, asking access over a set of operational data (OD). For accessing each node of the tree and performing the corresponding action a different authorization is required, which depends on the specific characteristics of the node and the resources that protects.

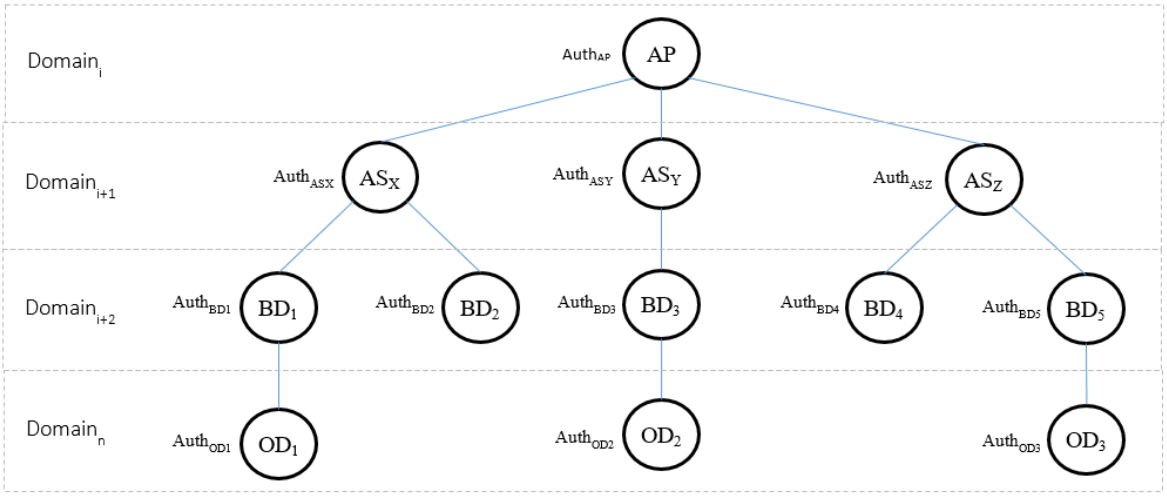


Figure 6: Inter-dependence of authorizations.

The authorizations of one level of the tree are, though, strongly depended on the authorizations provided on the upper level. Hence, if the subject, for example, does not have the right to access the platform, AuthAP is not given and the action terminates at the root of the tree. Otherwise, AuthAP is given and the access remains active as long as the subject can acquire the necessary authorization for each level. Moreover, concerning the UCON concept of access revocation, the correlation of the authorizations gives the possibility of revoking the enmeshed sessions upon a violation of a policy. More specifically, if at any point of the access a violation occurs at the root level and the subject is no longer

authorized to the platform, then all the authorizations given to this subject moving towards the tree must be also revoked. In the same way, if a policy does not stand anymore in any other node of the tree, the authorizations given to the children of this node must be revoked.

The above describes a parent-child relationship among the authorizations, which in order to be enforced, must be realized also within the security policies. An achievable way of implementing this is by exploiting the possibility of tagged policies, provided by Apache Ranger, and the obligations provided by the XACML standard. Hence, each policy will be given a tag, representative of their level (i.e. BigDataService policy), and will contain a condition and a set of obligations as those described below:

1. `Conditions`[2]: XACML conditions define the value or range of values which an attribute can acquire in order to satisfy the policy. In this case, the condition is related to the father node and contains a boolean attribute (true/false) which indicates if the father is still valid or not. If the father is invalid, and since as mentioned previously the authorization of the current level is depended on the authorization of the previous one, UCON will proceed with a revocation of the access.

2. `Obligations`: Once a violation occurs the policy which is no longer satisfied needs to inform their children that no longer stands. As stated previously, each policy monitors the status of the father node through a dedicated attribute and thus the value of this attribute needs to be updated immediately upon a change of the status. This can be achieved with the use of an obligation, by which XACML policies indicate that a decision depends on extra constraints. More specifically, having defined an obligation PDP informs the PEP upon a decision (either it is PERMIT or DENY) that it is necessary to take some more actions in order to satisfy those constraints. In the case of the inter-depended authorizations the obligations can be exploited in order to inform the PEP that the value of the aforementioned attribute needs to be changed. Thus, if the policy is satisfied and the authorization is provided, through an on-permit obligation the value will be set to True, otherwise with an on-deny obligation the value will be set to false. Consequently, the children nodes, having a PIP monitoring those attributes, will be immediately notified upon any change of the status of the father.

Listing 1 presents an example of a security policy which can be present on node BD1. As shown the policy contains a condition which checks an attribute with id ASx. This attribute is boolean and shows the current state of the father node. If it is `True`, meaning that the father is valid, then the authorization for BD1 will be given and an obligation will be triggered. Through this obligation the PEP will set the value of the attribute BD1 to true, notifying thus the children of the current node that the authorization was granted. On the contrary, if ASx is `False` then, the second obligation will be triggered and the PEP will have to set the value of BD1 to false.

3.3.2 Risk aggregation

In the tree structure described above, the notion of risk can also be incorporated in order to provide an extra level of security and a better protection of the resources. Risk can be realized as an attribute, within the security policies, related to the authorization of each level of the tree. Although, the presence of a dependency among different authorizations implies that risk of a single session cannot be considered as standalone. In fact, in a tree-based hierarchy representing the dependencies among different authorizations, the overall risk of a parent node can be calculated as a base risk, proper of the specific node authorization and the combination of the risk among its children. More specifically, taking as an example the hierarchy tree of Figure 6, each one of the leaves (OD1, OD2, OD3) can be correlated with a

[2]Here we refer to XACML conditions, which can be used to constraint the values of attributes related to users, objects and environment. Hence, they include UCON authorizations and UCON conditions

```
<Policy xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
PolicyId="policyBD1" RuleCombiningAlgId="..." Version="3.0">
<Description>Description</Description>
<Target>PolicyTarget</Target>
<Rule Effect="Permit" RuleId="rule-permit">
<Target>RuleTarget</Target>
<Condition>
  <Apply FunctionId="...:boolen-equal">
  <AttributeDesignator AttributeId="http://wso2.org/attribute/ASx" Category="" DataType="http:
      //www.w3.org/2001/XMLSchema#boolean"/>
 <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#boolean">true</
     AttributeValue>
  </Apply>
</Condition>
<ObligationExpressions>
  <ObligationExpression ObligationId="urn:oasis:names:tc:xacml:example:obligation:isValid"
      FulfillOn="Permit">
    <AttributeAssignmentExpression AttributeId="http://wso2.org/attribute/BD1" Category="">
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#boolean">true</
        AttributeValue>
    </AttributeAssignmentExpression>
  </ObligationExpression>
  <ObligationExpression ObligationId="urn:oasis:names:tc:xacml:example:obligation:isValid"
      FulfillOn="Deny">
    <AttributeAssignmentExpression AttributeId="http://wso2.org/attribute/BD1" Category="">
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#boolean">false</
        AttributeValue>
    </AttributeAssignmentExpression>
  </ObligationExpression>
</ObligationExpressions>
</Rule>
<Rule Effect="Deny" RuleId="rule-deny">
<Description>Description</Description>
</Rule>
</Policy>
```

Listing 1: Simplified example of policy including obligation in XACML 3.0.

specific level of risk, depended on the criticality of the resource. This risk will be taken into account as an attribute in the security policy of the node. In the upper level of the tree, the Big Data services, each node will also be assigned with a level of risk depending on the nature of the service and the action that the service is monitoring. In addition to this risk , the nodes of this level should also consider the risk coming from their children. Thus, for example, BD1 will include in its policy an aggregated risk derived from its base risk and the risk coming from OD1.

The hierarchical aggregation of risk is a topic frequently met in Finance and Insurance [5, 1, 6, 3] and a common exploited methodology is the one based on copulas [15]. Those studies proved that the hierarchical structure and the combination of risks at each level simplifies the procedure of showing how the dependency propagates through the structure. The aggregation approach is briefly described below and a representation is depicted in Figure 7.

Let us consider that we have a tree structure which consists of a root node and a number of branching and leaf nodes, which are denoted by a tuple $(i_1,....i_d)$ of natural numbers $i_j \in N$. The first number of the tuple indicates the parent and the second is the identifier of the node. Hence, for example the children of node N_1 will be denoted as $N_{11}, N_{12}, ..., N_{1d}$. Moreover, we define a vector of random values $X_1, ..., X_d$

with cumulative distribution functions (cdf) $F_1, ..., F_d$ respectively. For the leaf nodes X_i represents the risk values that we want to aggregate, while for the parents X_i will be given by the aggregation of the risk values of their children based on the following equation: $X_i = X_{i,1} + ... + X_{i,d}$. The definition which can be given is the following [1]:

- The random value X_i of leaf nodes has a distribution $F_i : P[X_i \leq x] = F_i(x)$ for all $x \in R$.
- For parent nodes X_i is the sum of its children.
- For parent nodes the dependence structure of its children is given by the copula C_i.

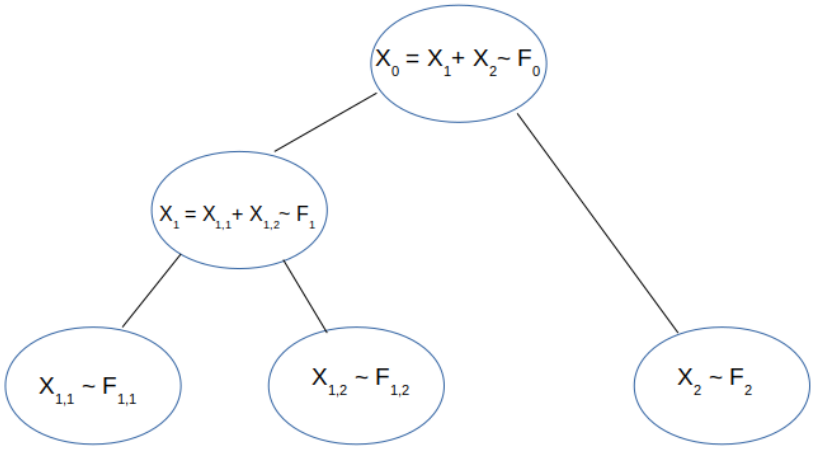


Figure 7: Hierarchical risk aggregation.

3.4 Integration of UCON in Apache Ranger

The implemented solution integrates UCON in Ranger Plugin. This integration is being realized by enabling the Apache Ranger administration portal with a User Interface in order to give the possibility of defining UCON policies. Moreover, for the evaluation of the policy and the extraction of a decision we exploit a UCON PDP. Figure 8 depicts the logical plugin-based architecture of the Apache Ranger and UCON integration. In the diagram shown two different Hadoop components (i.e. Hive and Kafka) delegate usage control to the UCS component via their corresponding plugins. Figure 9 depicts the required extension needed to enable UCON policies in Ranger Admin Tool.

Hive Ranger Plugin was modified in order to pass the evaluation control to UCON PDP upon a request interception. Ranger provides an administrative console with a set of User Interfaces alongside with a Back-end system which supports the User Interface. As depicted in the above figure, for this implementation we have integrated the necessary UCON interfaces into the Ranger Console with a Back-end service which can support them.

4 Use Case

The following use case focuses on the integration of the UCON framework in a hybrid deployment of multi-tiered services and plugin-based services. The multi-tiered services are device D (e.g. mobile, laptop, etc.) connected to network N and cloud-based application App, each protected by a PEP. The

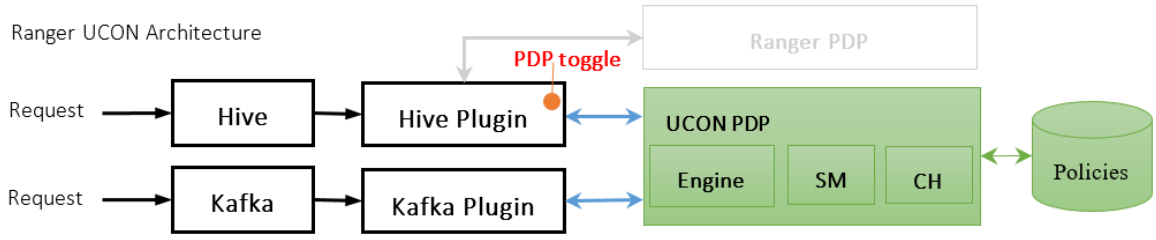


Figure 8: Integration among Ranger Plugins and UCON.

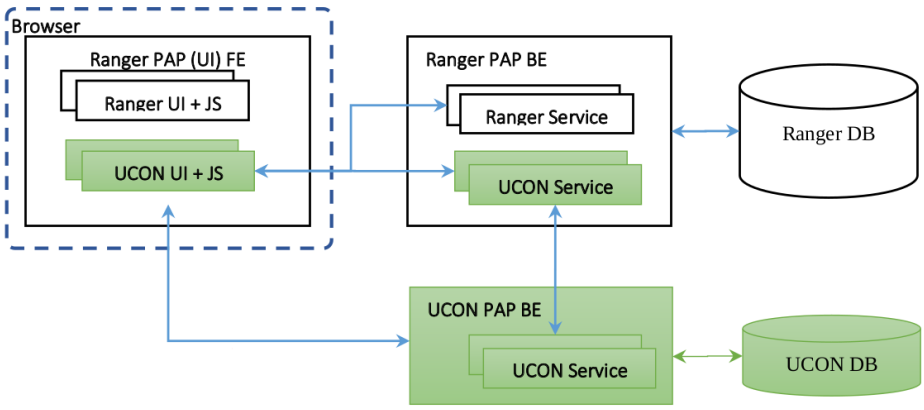


Figure 9: Enable of UCON policies in Ranger Admin Tool.

plugin-based service is Hive, a service component of the Hadoop framework, protected via a Hive Plugin configured to delegate access control to the UCS component. Figure 10 shows the high-level architecture of the setup.

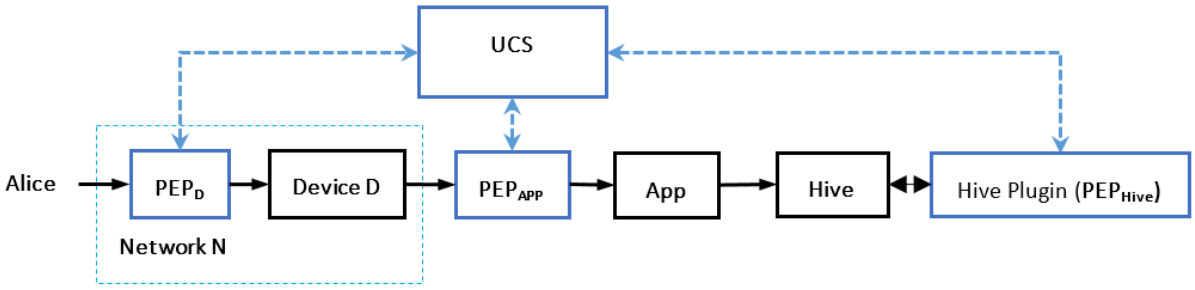


Figure 10: Use case setup.

The scenario is as follows: 1) User Alice on device D in network N is given access to data analysis application APP. 2) Alice starts a data analysis job on application APP for which she has been granted authorization on the basis of ongoing rule that relates to Risk of her device / network access. 3) Subsequent to Alice starting the analysis job, APP makes a number of calls to big data services Hive for the

purpose of the analysis. Access to Hive is given on the basis of an ongoing rule that APP is doing the analysis on behalf of Alice and on the basis of Alice's authorization. In this use case there is a model of dependencies of ongoing conditions: - AUTH1: Access to APP by Alice for the analysis is granted on the basis of an authorization that depends on attributes of Alice, D and N (e.g. security risk of device, trustworthiness of network, etc.) - AUTH2: Access to Hive by APP is given on the basis of AUTH1 which is continuously monitored and may be revoked. AUTH 2 is also continuously monitored. AUTH2 can be revoked or denied without affecting AUTH1 – e.g. if the APP attempts to read very sensitive data. APP will still be able to read less sensitive data for Alice's job. However, if AUTH1 is revoked then AUTH2 must be revoked because Alice is not authorized anymore to continue the analysis (e.g. her privileges have been revoked or her device is insecure or her network is insecure).

4.1 Experimental Evaluation

The utilized testbed for running the experiments for evaluating the proposed solution consists of a machine equipped with an Intel i7-6700 @3.40GHz, with 16GB DDR4 RAM and Ubuntu 19.10 64-bit. Worth mentioning is that all the components previously mentioned are running inside Docker containers.

In order to evaluate the proposed solution we conducted the following set of experiments. We calculated the time needed from the moment the subject conducts a query to Hive database until he/she gets back the results. Each time we increased the entries of the database, using thus datasets with 100, 1k, 5k, 10k, 50k and 100k data. For each dataset we conducted 30 queries to Hive and measured the average total time. The results are shown in Figure 11.

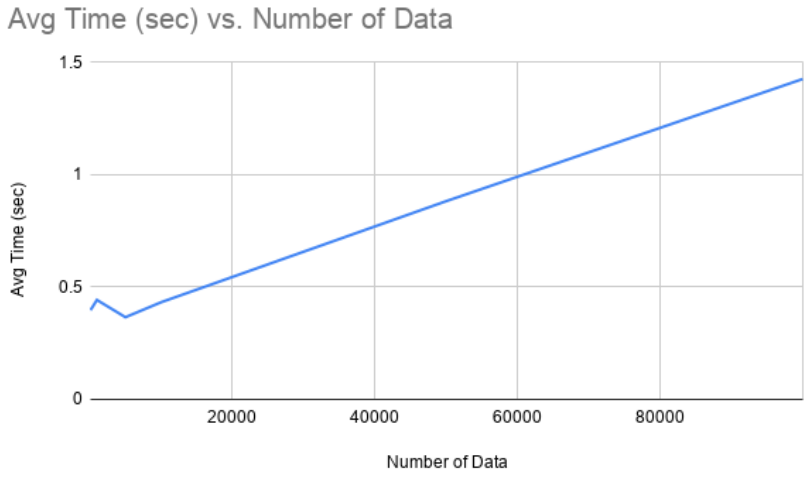


Figure 11: Average Evaluation Time.

For the evaluation of the queries we had in place a two-level hierarchy of security policies. More specifically, in accordance to the architecture presented in Figure 10, The user attempts to login onto the device D and network N. Upon evaluation of the applicable policy1 access is granted and an authorization session S1 is created which includes an ongoing condition associated to the risk level of the device D. Whenever the aggregated risk level is higher than a threshold A, access is denied and session S1 and child sessions are terminated. Once logged in, the user using application App makes an access request to the hive interface with a query to fetch a list of records. Upon evaluation of the applicable policy2 access is granted and an authorization session S2 is created which includes an ongoing condition associated to the aggregated risk level computed from the individual risks of each record. Whenever the aggregated

risk is higher than a threshold B, either there is an obligation that will prunes the number of records until an aggregated threshold is less than threshold B or there is an obligation that will simply deny access and terminate session S2. The total average time depicted on Figure 11 refers to the total time needed to evaluate the request against both of the policies.

The results show that as the size of the database becomes bigger, there is also a slightly increase in the total response time, which is something expectable considering that the framework needs to iterate through a bigger number of entries and return to the subject a bigger amount of results. Although, it is worth mentioning that even when the dataset that we consider includes 100k of data the average evaluation time does not exceed the 1.5 sec, which is not a delay able to modify the user experience.

5 Related Work

The sensitive nature of Big Data imposes the need of strong protection mechanisms in place. To this end, a number of studies propose different access control schemes, in order to regulate the access over those resources and to protect their confidentiality, integrity and availability. In more details, Gupta et al [9] present HeAC which is a multi-layer access control model designed for Hadoop ecosystem. HeAC formalizes the authorization model in Ranger and the authors combine it with an Object-Tagged Role Based Access Control so as to enable the assignment of attributes to objects based on specific tags. In their following study [10] they move on extening HeAC and they present HeABAC which is an attribute based access control model. In HeABAC they introduce more access control capabilities and in addition they demonstrate the concept of cross Hadoop service trust to secure inter-Hadoop services interaction. With respect to those works, in our implementation we exploit Usage Control as the main model for regulating the access.

Another related study is the one of Hu et al. [11], where the authors propose a general access control scheme for the process of Big Data. The main concepts that they exploit in their work is the trust lists, the security agreements and the access policies for the subjects who are already authenticated. However, they do not consider mutable attributes that they can be present in the security policies and can affect the access context. In addition to those studies a number of white papers and technical reports [22, 7, 20, 21], exist in the literature where there are presented the security requirements and the different tools, models, methods and techniques that can be exploited in order to protect Big Data.

Furthermore, other studies consider also the aspect of the encryption of the data. In more details, one of the most used forms of encryption is the Attribute Based Encryption (ABE) alongside with a policy update [8, 14] for the access control in cloud computing. Those studies are focused on providing a flexible enforcement of the policies and ensure the privacy and confidentiality of the data through the attribute encryption. In another study, relevant to the previous, Kan Yang et al [23] proposed a Data Access Control for Multi-Authority Cloud Storage (DAC-MACS) model. The authors consider a multi-authority environment where several entities are able to issue attributes. They apply a CP-ABE technique aiming at constructing an access control model, improving the decryption process and providing new algorithms for the key and ciphertext update. Table 1 provides a comparison among the existing studies in literature in terms of whether or not they use an ABAC approach, they include encryption of the data, the consider Mutability of Attributes (MoA), they consider an Authorization Hierarchy (AH) or they take into account the risk factor.

6 Conclusion

Controlling access rights on Big Data is a challenging task, due to their structure, volume and complexity. In particular, when access control is not needed only to check who has the right to access specific Big

Table 1: Comparison of the existing studies

Author	Ideology	ABAC	Encryption	MoA	AH	Risk
Gupta et al. [9]	a multi-layer access control model designed for Hadoop	No	No	No	No	No
Gupta et al. [10]	attribute based access control model with cross Hadoop service trust	Yes	No	No	No	No
Hu et al. [11]	general access control scheme based on trust lists and the security agreements	Yes	No	No	No	No
Devaraj Das et al. [7]	technical report regarding security on Hadoop	No	No	No	No	No
Fugkeaw [8]	attribute based encryption and policy updates for accessing cloud	No	Yes	No	No	No
Kan Yang et al [23]	a multi-authority cloud model where several entities issue different attributes	No	Yes	No	No	No

Data segments, but also the kind and number of operations that can be performed on them, common access control policies may not be sufficient to capture the inter-relationships and the dependencies which are generated. To model access policies, which consider those dependencies, it is necessary to design hierarchical relationship among policies at authorization and session level. In this work we have presented an approach based on UCON and tree-based policy hierarchy to manage dynamically access and usage control in an hadoop-based system. Furthermore, we have considered aggregated risk as an additional attribute related to execution of data operations, where the total risk of an operation is obtained as a composition of the risk on each specific data and the risk of the data operation itself.

We have presented an implementation which combines the UCS with the Apache Ranger access control tool for Hadoop services and demonstrated the viability of the approach with a set of performance experiments. It is worth noting that, one of the main contributions of this work is that the proposed solution incorporates into Hadoop ecosystem the ability of UCON to perform continuous monitoring once access is granted. As presented, once permission is granted UCON component continuously evaluates the list of records and based on the calculated risk the result which returns to the subject vary dynamically over time or may not be shown at all.

References

[1] P. Arbenz, C. Hummel, and G. Mainik. Copula based hierarchical risk aggregation through sample reordering. *Insurance: Mathematics and Economics*, 51(1):122–133, July 2012.

[2] E. Carniani, D. D'Arenzo, A. Lazouski, F. Martinelli, and P. Mori. Usage control on cloud systems. *Future Generation Computer Systems*, 63:37–55, October 2016.

[3] C. Cech. Copula-based top-down approaches in financial risk aggregation. `https://papers.ssrn.com/sol3/papers.cfm?abstract_id=953888` [Online; accessed on August 20, 2020], December 2006.

[4] P. Colombo and E. Ferrari. Privacy aware access control for big data: A research roadmap. *Big Data Research*, 2(4):145–154, December 2015.

[5] G. Coqueret. Second order risk aggregation with the bernstein copula. *Insurance: Mathematics and Economics*, 58:150–158, September 2014.

[6] M.-P. Côté and C. Genest. A copula-based risk aggregation model. *Canadian Journal of Statistics*, 43(1):60–81, January 2015.

[7] D. Das, O. O'Malley, S. Radia, and K. Zhang. Adding security to apache hadoop. *Hortonworks, IBM*, pages 26–36, 2011.

[8] S. Fugkeaw. Achieving privacy and security in multi-owner data outsourcing. In *Proc. of the 7th International Conference on Digital Information Management (ICDIM'12), Macau, China*, pages 239–244. IEEE, August 2012.

[9] M. Gupta, F. Patwa, and R. Sandhu. Object-tagged rbac model for the hadoop ecosystem. In *Proc. of the 31st IFIP Annual Conference on Data and Applications Security and Privacy (DBSec'17), Philadelphia, Pennsylvania, USA*, volume 10359 of *Lecture Notes in Computer Science*, pages 63–81. Springer, July 2017.

[10] M. Gupta, F. Patwa, and R. Sandhu. An attribute-based access control model for secure big data processing in hadoop ecosystem. In *Proc. of the 3rd ACM Workshop on Attribute-Based Access Control (ABAC'18), Tempe, Arizona, USA*, pages 13–24. ACM, March 2018.

[11] V. C. Hu, T. Grance, D. F. Ferraiolo, and D. R. Kuhn. An access control scheme for big data processing. In *Proc. of the 10th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom'14), Miami, Florida, USA*, pages 1–7. IEEE, October 2014.

[12] A. Lazouski, G. Mancini, F. Martinelli, and P. Mori. Usage control in cloud systems. In *Proc. of the 2012 International Conference for Internet Technology and Secured Transactions (ICITST'12), London, UK*, pages 202–207. IEEE, December 2012.

[13] A. Lazouski, F. Martinelli, and P. Mori. Usage control in computer security: A survey. *Computer Science Review*, 4(2):81–99, May 2010.

[14] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou. Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. *IEEE transactions on parallel and distributed systems*, 24(1):131–143, March 2012.

[15] R. B. Nelsen. *An introduction to copulas*. Springer Science & Business Media, 2007.

[16] B. Nelson and T. Olovsson. Security and privacy for big data: A systematic literature review. In *Proc. of the 2016 IEEE international conference on big data (Big Data'16), Washington, DC, USA*, pages 3693–3702. IEEE, December 2016.

[17] J. Park and R. Sandhu. The $UCON_{ABC}$ usage control model. *ACM Transactions on Information and System Security*, 7(1):128–174, February 2004.

[18] J. Park, X. Zhang, and R. Sandhu. Attribute mutability in usage control. In *Proc. of the 18th Annual conference on Data and Applications Security, Sitges, Catalonia, Spain*, pages 15–29. Springer-Verlag, July 2004.

[19] A. Pretschner, M. Hilty, and D. Basin. Distributed usage control. *Communications of the ACM*, 49(9):39–44, September 2006.

[20] B. Spivey and J. Echeverria. *Hadoop Security: Protecting your big data platform*. O'Reilly Media, Inc., 2015.

[21] T. White. *Hadoop: The definitive guide*. O'Reilly Media, Inc., 2012.

[22] C. Yalla, A. Gill, M. Gupta, M. H, T. McCloskey, N. Ngo, S. Tolentino, and D. Watson. Big Data: Securing Intel IT's Apache Hadoop Platform. Technical report, Intel, June 2016.

[23] K. Yang, X. Jia, and K. Ren. Secure and verifiable policy update outsourcing for big data access control in the cloud. *IEEE Transactions on Parallel and Distributed Systems*, 26(12):3461–3470, December 2014.

Author Biography

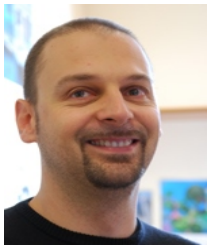
Gabriele Baldi (M.Phil. 2017) is currently a research fellow in the Trustworthy and Secure Future Internet research group at the National Research Council of Italy in Pisa. His main research topics are access and usage control, risk management of usage control, quantum algorithms and quantum key distribution protocols.

Yair Diaz currently works for Huawei as a Principal Engineer in the Trustworthy Technology and Engineering Laboratory, Munich Research Center. He has Ph.D in Information Security in Adaptive Policy-based Security Systems. He has over 10 years experience in IT consulting, software architecture and design, and software development in the IT/Security industry. He has co-author several papers and patents in security topics.

Theo Dimitrakos is a research area Director at the TTTE laboratory at German Research Center of Huawei based in Munich and a (part time) Professor of Computer Science at the University of Kent, UK. Theo has over 25 years of experience in Information and Communications Technology including 20 years of experience in Trust and Information Security. His experience spans a wide range of topics including IoT/IoV Security, Cloud Security, SOA and Web Services, SDN/NFV security, Identity and Access Management, Applied Cryptography, Privacy, Data Protection, Uncertainty Reasoning and Trust Management, Security Risk Analysis. In the past, Theo has been a Chief Researcher at Research and Innovation headquarters of British Telecom (BT), UK. Through his participation in expert groups and strategy boards, Theo has advised European Agencies, such as ENISA, industry forums such as ISF and CSA and the European Commission on both technological development and policy making issues. He has also been senior research at Rutherford Appleton Laboratory where he was involved at the realization of Grid computing and supported the W3C Office for UK and Ireland. He has authored several technical books, scientific papers, guest editions of international journals and over forty patents.

Fabio Martinelli (M.Sc. 1994, Ph.D. 1999) is a research director at the National Research council of Italy (CNR), where he is referent for cyber security project activities. He is co-author of more than four hundreds scientific papers. His main research interests involve security and privacy in distributed and mobile systems and foundations of security and trust.

Christina Michailidou (M.Eng. 2016) is currently a Ph.D student in the University of Pisa, Department of Computer Engineering and an assistant researcher in the National Research Council of Italy in Pisa. Her main research topics are access and usage control, risk management of usage control, Internet of Things (IoT) Security and Software Defined Network (SDN) security. She was an Early Stage Researcher of European Network in Cyber Security (NeCS) H2020 EU-Funded Project.



Paolo Mori received the M.Sc. in Computer Science from the University of Pisa in 1998, and the Ph.D. from the same University in 2003. He is currently a researcher at "Istituto di Informatica e Telematica" of "Consiglio Nazionale delle Ricerche". His main research interests involve trust, security and privacy in distributed environments focusing on the design and implementation of mechanisms for access/usage control, data privacy aspects of Online Social Networks, and Blockchain technology and its applications. He usually serves in the Organization and Program Committees of international conferences and workshops. In particular, he was Program Co-Chair of the 2nd-6th editions of the "International Conference of Information System Security and Privacy" (ICISSP 2016-2020). He is (co-)author of 100+ papers published on international journals and conference/workshop proceedings, and of several chapters in books on research topics. He is usually actively involved in research projects on information and communication security, such as the European Commission funded SPARTA, C3ISP, CoCoCloud, CONTRAIL, GridTrust, S3MS, and the EIT Digital funded ones such as "Trusted Data Management with Service Ecosystem" (High Impact Initiative), "Private Virtual Federator", and "Hybrid Cloud and IoT".

Oleksii Osliak (M.Eng. 2015) is a Ph.D. student at the Department of Computer Science at the University of Pisa and a research assistant at the National Research Council of Italy, Pisa. His topics of interest include, but are not limited to, the Cyber-Threat Intelligence management, Industrial Control Systems Security, Industrial Internet of Things (IIoT) Security, access control models, security policy management. He was an Early Stage Researcher of the European Network for Cybersecurity (NeCS) H2020 EU-Funded Project.

Andrea Saracino (M.Eng 2011, Ph.D 2015) is a Researcher in the Trustworthy and Secure Future Internet group at National Research Council of Italy (CNR). He coauthored more than 50 papers in mobile malware analysis, data usage control, distributed reputation models and behavioral analysis and he has been involved in the activities of EU funded projects on these topics, such as C3ISP (GA n. 700294), NECS (GA n. 675320) and EIT Digital Trusted Cloud Management. He lectured several course on mobile app security, intrusion detection and authentication at both graduate and post graduate level and is co-organizer of international workshops on these topics.