Access Control Policy Enforcement for Zero-Trust-Networking

Romans Vanickis
Software Research institute
Athlone Institute of Technology
Athlone, ireland
rvanickis@research.ait.ie

Paul Jacob
Software Research institute
Athlone Institute of Technology
Athlone, ireland
pjacob@ait.ie

Sohelia Dehghanzadeh
Software Research institute
Athlone Institute of Technology
Athlone, ireland
sdeghanzadeh@ait.ie

Brian Lee
Software Research institute
Athlone Institute of Technology
Athlone, ireland
blee@ait.ie

***Abstract*—The evolution of the enterprise computing landscape towards emerging trends such as fog/edge computing and the Industrial Internet of Things (IIoT) are leading to a change of approach to securing computer networks to deal with challenges such as mobility, virtualized infrastructures, dynamic and heterogeneous user contexts and transaction-based interactions. The uncertainty introduced by such dynamicity introduces greater uncertainty into the access control process and motivates the need for risk-based access control decision making. Thus, the traditional perimeter-based security paradigm is increasingly being abandoned in favour of a so called "zero trust networking" (ZTN). In ZTN networks are partitioned into *zones* with different levels of trust required to access the zone resources depending on the assets protected by the zone. All accesses to sensitive information is subject to rigorous access control based on user and device profile and context. In this paper we outline a policy enforcement framework to address many of open challenges for risk-based access control for ZTN. We specify the design of required policy languages including a generic firewall policy language to express firewall rules. We design a mechanism to map these rules to specific firewall syntax and to install the rules on the firewall. We show the viability of our design with a small proof-of-concept.**

Keywords—zero trust networking, risk-based access control, trust, policy enforcement, firewall, network zone, micro-segment

I. Introduction

There has been much change in enterprise computing in the last two decades with the appearance of new approaches such as cloud and edge computing, the (industrial) Internet of Things (IIoT) etc. [1]. These environments will be characterised by distributed interactions on a scale not seen heretofore with attendant high levels of complexity and dynamicity - including mobility -heterogeneity and uncertainty. We consider that the nature of these interactions will move to a combination of the current dominant stateless, or session-less, REST model and a stateful/session-based interaction, [2] [3].

The dramatic escalation in both the number and sophistication of security-attacks on business in recent years, [4], will continue to grow in coming years– a factor that merely adds to the computing environment complexity.

Access control (AC) systems will therefore need to adapt dynamically to incorporate *risk assessment* into the access control process. AC decisions will be include many factors such as the degree of trust in the user and the device, user and device situational context i.e. location, time-of-day, type of task as well as the current security threat level in the user's immediate environment, [5] . Furthermore the level of access assigned to a device or user can change over time and the AC system must be able to infer the current trust level by consulting various data sources and making decisions accordingly [6]. Researchers are therefore exploring the use of risk-based access control in many domains, [7] [8].

A main result of these trends is a move away from the traditional perimeter based security model toward the application of so called *zero trust networking* (ZTN) security models that treat the enterprise intranet with the same degree, i.e. lack, of trust as the Internet, [9]. The key element of the ZTN approach is to treat the internal network as untrusted to the same degree as the Internet. The internal network is divided into a number of network segments or *zones* each of which contains different functions and information. Each zone will have a different trust level that indicates the importance of the assets housed within the zone, [10]. In order to access an asset, a subject's trust level assignment must be equal to or greater than the zone's minimum trust level, [11].

Traffic between zones is restricted by firewalls in accordance with with the overall access control policy. Access control is also dynamic and transaction based i.e. a decision is made for each access request and rules are updated on the inter-zone firewalls as needed for each transaction. ZTN is becoming widely deployed in the commercial world cross many domains.

This paper has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement 700071

Existing examples of deployments include enterprise security e.g. Google's BeyondCorp [11] as well as campus [7], and cloud computing security [8], [12].

While these deployments show the viability of ZTN and the associated risk-based access control they do not describe in sufficient detail how to carry out policy enforcement to implement risk-based access control in ZTN. We have addressed this deficit in a previous paper [5], where we describe a policy management framework, FURZE (Fuzzy Risk Framework for ZTN), to facilitate fuzzy risk evaluation. In this paper we elaborate further on this ongoing work. In particular we define the specification of two policy languages to capture and express required authorisations, obligations and constraints to enable risk based AC for ZTN. We also demonstrate the feasibility of our approach through the implementation of a proof of concept for a particular component of the FURZE system to allow the dynamic update of firewalls in a ZTN system.

The rest of the paper is structured as follows: In Section 2 we give an overview of the FURZE system. Section 3 defines policy languages for ZTN AC. Section 4 describes the development of the FURZE firewall provisioning module (FPM). We give conclusions of the research in section 5 and outline future research challenges.

II. FURZE SYSTEM OVERVIEW

FURZE is a policy enforcement framework for risk adaptive access control (RAdAC) based on the policy modelling approach of Kandala, [13]. FURZE is specifically aimed at RAdAc for the ZTN domain. The key concept of RAdAC is the requirement to make a trade-off made between *operational need* and *security risk* when making access control decisions. Operational need can be seen as the reason for the users access request. It can be represented as a person's membership in some community of interest or an organization. In some cases operational needs will override security risk and access to otherwise restricted resources will be granted.

Kandala defines a RAdAC policy model based on the UCON (Usage Control) model of Sandhu [3]. The key innovation of UCON is the notion of "decision continuity" which means that a policy authorisation or related obligations and conditions can be transacted before, during or after the AC session/transaction. This insight is a significant addition to RAdAC as it allows adaptation to changing environment conditions as well as session based interactions. While Kandala has provided a sound abstract model for "UCON-ised" RAdAC a number of significant research questions remain open around the practical deployment of RAdAC.

Specific issues we seek to explore through the development of FURZE include i) the definition of a policy management architecture to include on-going monitoring to enable decision continuity in ZTN AC, ii) the design of an access control policy language to specify and manage decision continuity updates, iii) the design a risk evaluation function based on fuzzy logic to enable probabilistic access control decisions to be taken iv) the development of a policy language to express firewall access control list (ACL) and the design of an accompanying firewall provisioning mechanism.

The proposed FURZE policy enforcement architecture is shown in Figure 1. This contains a number of elements that form the functional, logical and linguistic basis for a policy framework for ZTN. The architecture is broadly based on the XACML policy framework, [14]. Access requests are received via a Policy Enforcement Point (PEP) such as e.g. a WiFi base station. The Context Handler (CH) coordinates the access control process including decision continuity handling. The Environment Evaluation (EE) contains plug-in components that convert session and other relevant factors into attributes that can be used as input ot risk evaluation. Examples shown include a plug-in to determine security situational awareness (see [5]), operational need and locations - for a simple access control case these components could simply default to session attributes. The Risk Evaluation Function (REF) and Access Decision Function (ADF) jointly act as a Policy Decision Point (PDP) to control access while the subject and object attributes are stored in a management database. The Topology Awareness Module (TAM) accesses the network topology map to determine the list of firewalls on the route between the point of access and the requested service. The Firewall Provisioning Module (FPM) then provisions the appropriate firewalls. These two modules play a key role in ZTN AC enforcement.

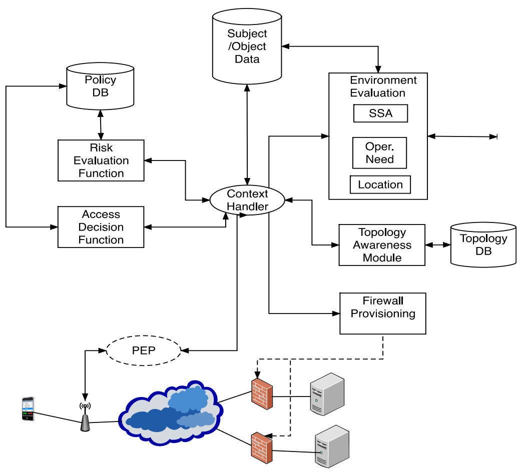

Figure 1 FURZE policy enforcement framework

In FURZE the application of decision continuity imposes a requirement on the control function to maintain session state information so that access control can adapt to reflect situational or other influencing factors that change the balance between operational need and security risk and trigger policy re-evaluation. Risk assessment is made as part of the initial authorisation predicate evaluation and, possibly, subsequently as part of either an authorisation or condition predicate evaluation that in turn has been triggered by some event during the session, [13]. Dynamic firewall provisioning is part of this ongoing session management. Firewall rules are removed from the relevant gateways when the session ends or if the risk situation changes during the course of the session for some reason.

III. FURZE Policy Languages

Two policy languages are defined as part of the FURZE framework. The first, *PAROLE*, is a language to express general AC policies while the second language *FACL* (FURZE *Firewall Access Control List*) defines generic firewall rules.

A. PAROLE

PAROLE fulfils a role similar to XACML in providing access control to network resources. However, it contains a number of enhancements to improve on shortcomings in current access control approaches, identified above. The primary additions are to enable decision continuity and access control risk assessment. PAROLE draws from a number of existing policy management approaches including XACML, [14], Fuzzy Control Language (FCL, [15] and RAdAC, [16]. PAROLE contains three main elements required to provide a complete UCON-RAdAC solution- see Listing 1.These are:

```
namespace someName{

 namespace Attributes {
   attribute att-1 { }
   attribute att-2 { }
 }

 namespace Events {
   event ev-1 { }
   event ev-2 { }
 }

Session mySession { ..}

 AuthRule auth1 { .. }
 AuthRule auth2 { .. }
  . . .

 RiskFB someFB1 { .. }
 RiskFB someFB2 { .. }

} // end namespace someName
```

Listing 1: PAROLE document

- A *Session* construct that enables ongoing control (decision continuity) for relatively long lived accesses. This notion of session differs from the more usual access control notion of session which defines a set of access permissions [14] but which does not have the notion of continuity
- An *AuthRule* construct that specifies the access control policies. This is similar to the XACML policy and rules.
- A *RiskFB* construct to determine estimation of access control risk to enable risk based access control. This construct is based on the use of the FCL and contains a set of fuzzy rules.

PAROLE is an attribute based access control (ABAC) approach to policy management and draws on XACML to define and manage attributes and other identifiers. In particular, PAROLE borrows the XACML *namespace* approach. As described in [17] a "namespace is used to declare a scope that contains a set of related objects" where, the case of XACML, objects may be attributes, subjects etc. A PAROLE specification may contain multiple namespaces, some of which may be nested. It must contain at least one root level namespace. Identifiers may be imported from one name space to another and elements of a nested namespace can be accessed via '.' notation as in Java and other languages e.g. *parentns.childns.identifier_x*. A namespace may be either *artefact-based* or *domain-based*. The artefact approach uses namespaces to organise the PAROLE structure based on *language elements* e.g. an "Attributes" namespace may be used to collect all attributes or a "Policy" namespace may be similarly used to collect policy definitions. In the domain-based approach the namespaces are used to collect artefacts according to the entities or application for which the policies are being defined e.g. a "User" namespace may be used to collect all attributes and any other identifiers for the policy *subject.* The PAROLE language is currently being specified and language tool development is part of ongoing FURZE research work.

B. FACL

FACL is a Domain Specific Language (DSL) for expressing firewall rules and from which firewall-specific filtering rules can be generated.

```
service_def::= "service" sname
   prot port_range
 port-range ::= "port" ({portid }+
       |  "from" portid  "to" portid )
zone::= "zone" zone_name ip_address
 zone_addr ::=  zone_name
      | ip_address | "any"
 ip_address ::= ip_addr+ | ip_range
fwRule::= "fwRule" fwName "{"
   ["incoming:" service_rule]
   ["outgoing:" service_rule] "}"
 service-rule::=
   sname ("allow" | "deny")
   ("to" | "from") zone_addr
   ["to" | "from" zone_addr]
 zone_action::= = ("add"|"remove)
   fwRule+ zone
```

Listing 2: FACL BNF subset

The FACL language definition contains four main *productions* (in bold in Listing 2), which are used to define the firewall entries i.e.:

- A *service* production that defines a mnemonic for a service and associates a port, or ports to the service. An example is:
  `Service http tcp port from 8080 to 8088`
- A *zone* production that defines a mnemonic for a network zone and associates a range of IP addresses with the zone. An example is:

  `zone zonABC from 192.168.0.1 to 192.168.0.5`
- A *firewall rule* production which defines the actual firewall rule based in part on the previous productions

```
fwRule fwAcl1{
 incoming:http allow to 173.41.1.2 from any
  outgoing:ssh deny to any}
```

- A *zone action* production to add or remove firewall rules to/from a zone firewall. An example is

```
add fwAcl1 to zoneABC
```

Listing 2 is a subset of the BNF for the language.

IV. FIREWALL PROVISIONING

Firewall provision is realized by a combination of the TAM and FPM components of the FURZE framework. The TAM maintains a network topology map including zone topology and firewall information. A module such as the TAM can be constructed in various ways such as by using SNMP or LLDP.

When a request for access to a service or resource has been granted the CH queries the TAM for a list of the firewalls on the path to the host or server hosting the requested resource. It then passes origin and destination information to the FPM to generate and install the required firewall rules.

A. Firewall Provisioning Module

The purpose of the FPM is to generate FACL firewall rules and to then convert these rules to firewall specific rules. It consists of four main components

- An *interface module* to the Context handler. This is designed to isolate the main FPM functions from the interfacing mechanism in order to enable flexible module distribution so that for example the interface could be a direct method call on a Java object or alternatively a REST implementation.
- The *FACL rule generator* (FRG). This module creates the FACL ruleset in response to the CH call. The design of this module is described below.
- The *FACL language processor* (FLP*)*. This module take the generated FACL rules and translates them to the specific firewall rule language(s) that are to be used. The FLP is extensible to support a range of firewall implementations.
- The *Firewall Interfacing* plug-in. This module interfaces the FPM to the individual firewall. It retrieves the firewall address from the FRG.

In the literature we observe two approaches to firewall interfacing , one based on the use of a generalised interface and one based on the language approach we have taken here. An example of the first approach is described in the Dynfire framework [7], which uses the Simple Middlebox Configuration (SIMCO) Protocol Version 3.0 – defined in RFC 4540. SIMCO define a general interface to add policy rules to any type of middle box. The authors used their own SIMCO implementation. An example of the second approach is that of AL_SAFE [8]. We have adopted the language based approach because we believe to be more portable and more easily implemented.

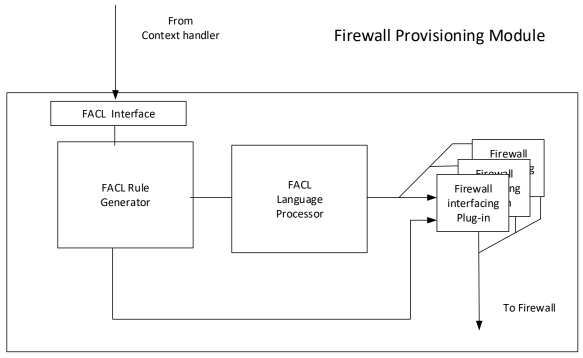

Figure 2 FURZE policy enforcement framework

1) FRG

The Firewall Rule Generator creates FACL rules. The CH transfers the relevant production parameters to the FPM in a single interface invocation. These parameters are passed to the FRG which converts them to the required set of FACL production rules. The FRG is implemented in Java. The FRG includes the use of a templating engine as the key mechanism to generate the FACL rule text.. A templating language enables a developer/designer to define a document/template containing generic text and to substitute tagged item-specific text parts through a callout to a templating engine. At runtime, the template engine replaces variables in a template file with actual values, and transforms the template into an the final file version.

The Apache Velocity templating engine is used within the FRG, [18]. The templating engine is invoked from the main FRG code component. A FACL template contains a set of statements in the Velocity templating language (VTL) that collectively define the FACL firewall rule generation. The key VTL constructs are *references* and *directives*. As the VTL user guides puts it "References begin with $ and are used to get something. Directives begin with # and are used to do something". Variables are assigned a value either internally from within the template or externally from the calling FRG Java wrapper. Directives include program control constructs as can be seen in Listing 3, which show part of the FACL VTL file, used to generate the service production.

```
  #set ($sp=" ")

  ##  Define service name e.g. "service http
    tcp port 8080"

 ##if ($sname ne "")
  #if ($port-type=="range")
   #set ($ports =from $low_port to $hi_port)
  #else
   #set ($ports = $port_list)
  #end
  service $sname $protocol port $ports
#end
```

Listing 3: VTL service rule generation

2) FLP

The rules generated by the FRG are next passed to the language processor. For the processor a parser has been developed using the PyParsing library, itself an internal DSL for developing parsers in Python. The library provides predefined functionalities and classes to create and combine parser elements. It has many predefined elements that can match common constructs, such as *Literal(), Word(), Optional(), Or(), ZeroOrMore()*, *OneOrMore()* etc. PyParsing generates a recursive-descent-like parser from a set of rules and semantic actions. Rules are defined in a syntax similar to BNF, for example – see Listing 4.

```
serviceRule = service + serviceName +
               sProtocol + servicePortRange
service = Literal(("service(")
serviceName = Word(alphanums)
port = Literal(("port(")
servicePortRange = port + Or(portIds |
    portRange)
portRange = frm + portId + to + portId
frm = Literal("from")
to = Literal("to")
```

Listing 4: FACL PyParsing grammar

Semantic actions are called when an input line is recognised as an instance of a BNF rule. They are associated with a rule using the method setParseAction(). When a semantic action is called by the PyParsing runtime, three arguments are passed, the string being recognised, the start location of the matched text and the list of tokens recognised. Semantic actions generate an in-memory representation of the input known as a semantic model. In the example in Listing 5 a service object has been created and inserted into a Python dictionary (unordered key-value pairs), with key equal to the service name and value equal to the service object. The service object stores all information about the service. Corresponding zone, firewallRule and zoneAction objects are created and inserted into dictionaries. Finally all references between objects in the dictionaries are resolved to create a semantic model. In the language processor a backend module to generate Uncomplicated Firewall (UFW) rules has been implemented. It traverses the semantic model and generates appropriate UFW rules. Other modules, to generate rules in other

```
serviceRule = service + serviceName +
               sProtocol + servicePortRange

def serviceRuleAction(s,loc,toks):
 service = Service(toks[1], toks[2],toks[3])
 self.ast.servicesDictionary[toks[1]]= service

serviceRule.setParseAction(serviceRuleAction)
```

Listing 5 FACL semantic actions

firewall languages such as IPChains, NetFilter, IPFilter, Cisco ACL, can easily be implemented

We have implemented a proof of concept – see Fig. 3 - to demonstrate the correction functioning of firewall provisioning (as FURZE is a work in progress only some parts are

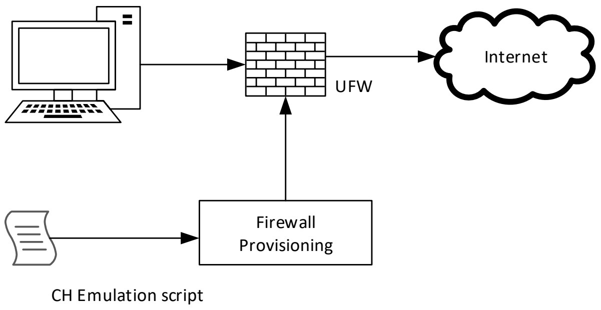

Figure 3 FPM proof of concept

implemented and the proof of concept is consequently limited). The scenario demonstrated is basic internet access. The FURZE CH module is emulated by a python script to invoke the FPM REST interface. Initially the UFW is configured by command line to disallow outgoing http requests. The CH is then executed to invoke the FPM to, in turn, to generate the FACL and UFW commands to install and revoke permissions. Successful operation is verified.

V. RELATED WORK

Google has adopted a ZTN approach for access control that, as described, seems partly similar to the work defined here [6], [11]. However they have not described the policy language, risk management or decision continuity implementation in detail. Vensmer, [7], describes Dynfire, an AC policy management framework for ZTN applied to a university campus that encompasses a number of the ideas described in our work. However it does not include either risk management nor decision continuity. Giannoku, [8], describe AL-SAFE, a ZTN AC implementation for cloud computing –however they also do not include policy language, risk management or decision continuity aspects. Approaches to network segmentation and zoning in general has been described by [10] and for virtual networking in data centres by VMware [19] and in the emerging 5G network by [20]. The underlying firewall policy management is a mature research field and a number of authors have described efforts in this direction [21],[22],[23]. However none of these authors have described access control risk management as part of their contributions

AC policy language design is also a mature area. XACML is one of the most widely deployed policy enforcement approaches [14]. However neither XACML nor it's competitors support the notion of decision continuity in a session construct. In designing FACL we have incorporated a number of ideas from these sources. FACL is designed however to be simple to use and avoids overly complex language constructs e.g. such as inherited zones ,[22].

Chen [24], developed a fuzzy logic approach to AC risk assessment that proposed the gradation of security risk as services of levels between "allow" and "deny" where each level has an associated risk mitigation countermeasure. Ni [25], investigated the applicability of fuzzy inference for risk based access control and concluded that the approach was flexible and scalable. Our work most closely matches that of Ni as we also propose the use of fuzzy inference for risk calculation – however our scope of research is somewhat wider.

VI. Conclusions

Computer networking is evolving rapidly on many fronts. Security and trust models are likewise evolving to meet these challenges. In this paper we have described a risk-based access control enforcement framework to support future security needs, in particular ZTN. We have defined the policy languages needed to support the framework and have described the design and implementation of a component to support firewall provisioning.

Future work includes the development of language tools and runtime mechanism for the PAROLE language and its integration into an existing PDP system such as Keycloak [26], to demonstrate its viability. Furthermore, the PAROLE session construct is a form of context aware access control and the concept is easily extended to IoT and edge computing application areas such as smart manufacturing.

VII. Bibliography

[1] M. Chiang and T. Zhang, "Fog and IoT: An Overview of Research Opportunities," IEEE INTERNET OF THINGS JOURNAL, vol. 3, no. 6, 2016.

[2] S. Berger, S. Vensmeer and A. Kiesel, "An ABAC-based Policy Framework for Dynamic Firewalling," in ICSNC 2012 : The Seventh International Conference on Systems and Networks Communications An, Lisbon, 2012.

[3] J. Park and R. Sandhu, "The UCON(ABC) Usage Control Model," Transactions on Information and System Security, vol. 7, no. 1, pp. 128-174, Feb 2004.

[4] R. Broadhurst, P. Grabosky, M. Alazab, B. Bouhours and S. Chon, "An Analysis of the Nature of Groups engaged in Cyber Crime," International Journal of Cyber Criminology, vol. 8, no. 1, pp. 1-20, 2014.

[5] B. Lee, R. Vanickis, F. Rogelios and P. Jacob, "Situation Awarness based Risk Adaptable Access Control in Enterprise Netowrks," in 2nd International Conference on Internet of Things, Big Data and Security (IoTBS), Porto, 2017.

[6] R. Ward and B. Beyer, "BeyondCorp A New Approach to Enterprise Security," ;login:, vol. 39, no. 6, Dec 2014.

[7] A. Vensmer and S. Kiesel, "DynFire: dynamic firewalling in heterogeneous networks," in Proceedings World Congress on Internet Security (WorldCIS), 2012.

[8] A. Giannakou, L. Rilling, C. Morin and J-L. Pasage, "ALSAFE: A Secure Self-Adaptable Application-Level Firewall for IaaS," in SEC2 - Second workshop on Security in Clouds, Lorient, 2016.

[9] Forrester Research,, "Developing a Framework to Improve Critical Infrastructure Cybersecurity," 2013. [Online]. Available: http://csrc.nist.gov/cyberframework/rfi_comments/04 0813_forrester_research.pdf>.. [Accessed 19 February 2018].

[10] A. Gontarczyk, P. McMillan and C. Pavlovski, "Blueprint for Cybersecurity Zone Modelling," IT in Industry, vol. 3, no. 2, 2015.

[11] B. Osborn, J. McWilliams, B. Beyer and M. Saltonstall, "BeyondCorp; Design to Deployment at Google," ;login:, vol. 41, no. 1, 2016.

[12] S. Jeuk, G. Salgueiro, F. Baker and S. Zhou, "Network Segmentation in the Cloud A Novel Architecture Based on UCC and IID," in IEEE 4th International Conference on Cloud Networking (CloudNet), 2015.

[13] S. Kandala, R. Sandhu and V. Bhamidipati, "An Attribute Based Framework for Risk-Adaptive Access Control Models," in RES '11 Proceedings of the 2011 Sixth International Conference on Availability, Reliability and Security, Washington, 2011.

[14] OASIS, "eXtensible Access Control Markup Language (XACML) Version 3.0," 2013. [Online]. Available: http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html. [Accessed 7 May 2017].

[15] International Electrotechnical Commission., "IEC 61131-7:2000 Programmable Controllers – Fuzzy Control Programming," 2000. [Online]. Available: <https://webstore.iec.ch/publication/4556?.

[16] R. McGraw, "Risk Adaptable Access Control,," 2009. [Online]. Available: http://csrc.nist.gov/news_events/privilege- management-workshop/radac-Paper0001.pdf. [Accessed 11 May 2017].

[17] OASIS, "Abbreviated Langauge for Authorisation , Version 1.0," 12 March 2015. [Online]. Available: https://www.oasis-open.org/committees/download.php/55228/alfa-for-xacml-v1.0-wd01.doc. [Accessed 27 February 2018].

[18] Apache, "Apache Velocity," August 2017. [Online]. Available: http://velocity.apache.org/. [Accessed Feb. 2018].

[19] Vmware, "Data Center Micro-Segmentation: A Software Defined Data Center Approach for a Zero Trust Security Strategy," VmWare, 2104.

[20] O. Mammel, J. Hiltunen, J. Suomalainen, K. Ahola, P. Mannersalo and J. Vehkaper, "Towards micro-segmentation in 5G network security," in European Conference on Networks and Communications (EuCNC), Athens, 2016.

[21] B. Zhang, E. Al-Shaer, R. Jagadeesan, J. Riely and C. Pitcher, "Specifications of A high-level conflict-free firewall policy language for multi-domain networks," in ACM Symposium on Access Control Models and Technologies, Monterey, 2007.

[22] J. Lobo, M. Marchi and A. Provetti, "Firewall Configuration Policies for the Specification and Implementation of Private Zones," in Proceedings IEEE International Workshop on Policies for Distributed Systems and Networks, 2012.

[23] Cisco Corpporation, "Zone-Based Policy Firewall Design and Application Guide," Mar 2010. [Online]. Available: http://www.cisco.com/c/en/us/support/docs/security/i os-firewall/98628-zone-design-guide.html#topic6. [Accessed 26 Feb 2018].

[24] P.C. Chen, P. Rohatgi, C. Keser, P.A. Kargr. "Fuzzy Multi–Level Security : An Experiment on Quantified Risk–Adaptive Access Control," IEEE Access, vol. 2, pp. 514-525, 2014.

[25] Q. Ni, E. Bertino and J. Lobo, "Risk-based Access Control Systems Built on Fuzzy Inferences," in ASIACCS'10 5th ACM Symposium on Information, Computer and Communications Security, Beijing, 2010.

[26] Keycloak, "Open Souce Identity Management," 2018. [Online]. Available: http://www.keycloak.org/. [Accessed March 2018].