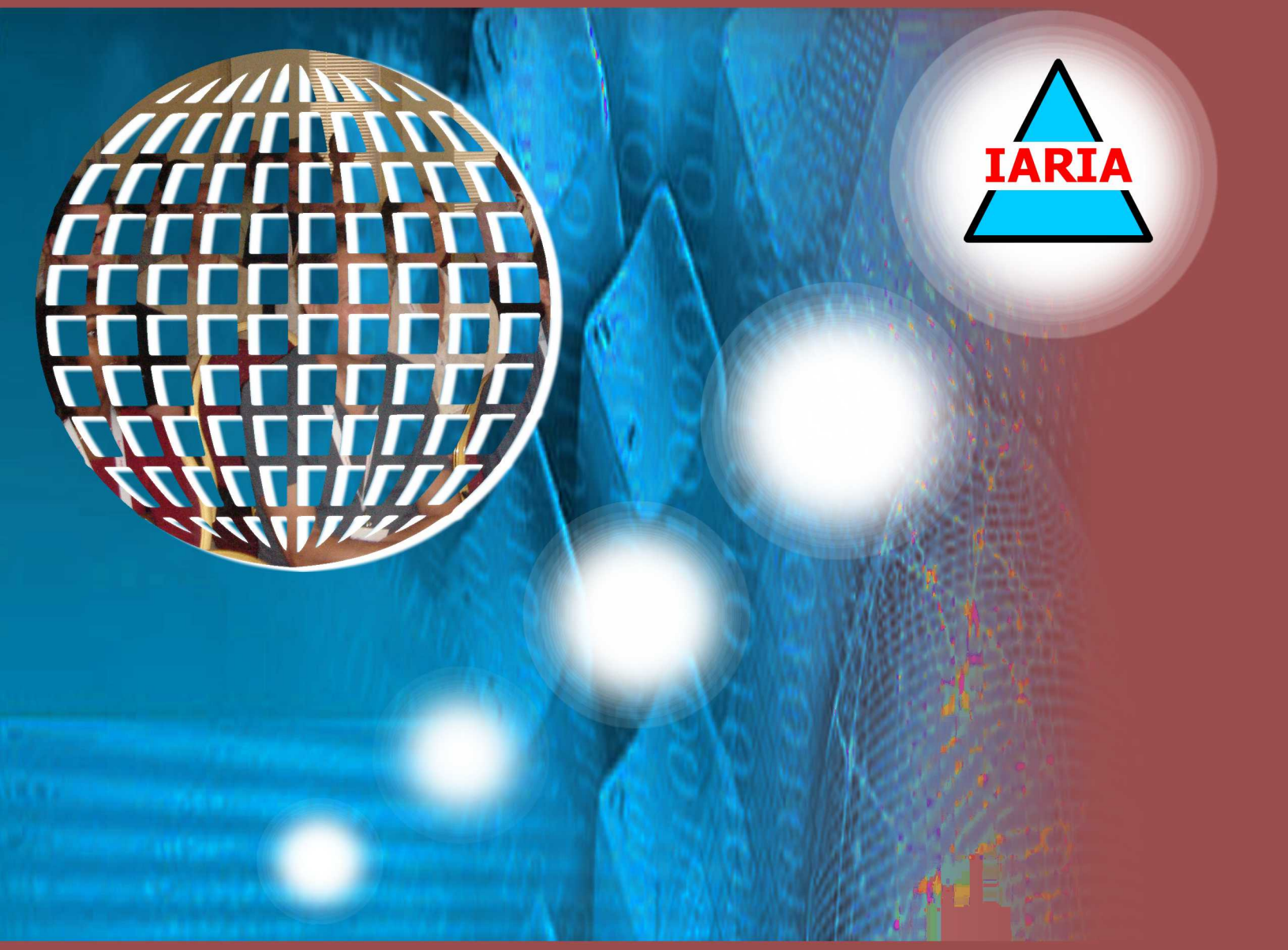
International Journal on Advances in Security



2013 vol. 6 nr. 3&4

The *International Journal on Advances in Security* is published by IARIA.
ISSN: 1942-2636
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Security
Volume 6, Number 3 & 4, 2013

Shao-Ming Fei, Capital Normal University, Beijing, China
Eduardo B. Fernandez, Florida Atlantic University, USA
Anders Fongen, Norwegian Defense Research Establishment, Norway
Somchart Fugkeaw, Thai Digital ID Co., Ltd., Thailand
Steven Furnell, University of Plymouth, UK
Clemente Galdi, Universita' di Napoli "Federico II", Italy
Emiliano Garcia-Palacios, ECIT Institute at Queens University Belfast - Belfast, UK
Marco Genovese, Italian Metrological Institute (INRIM) -Torino, Italy
Birgit F. S. Gersbeck-Schierholz, Leibniz Universität Hannover, Certification Authority University of Hannover (UH-CA), Germany
Manuel Gil Pérez, University of Murcia, Spain
Karl M. Goeschka, Vienna University of Technology, Austria
Stefanos Gritzalis, University of the Aegean, Greece
Michael Grottke, University of Erlangen-Nuremberg, Germany
Ehud Gudes, Ben-Gurion University - Beer-Sheva, Israel
Indira R. Guzman, Trident University International, USA
Huong Ha, University of Newcastle, Singapore
Petr Hanáček, Brno University of Technology, Czech Republic
Gerhard Hancke, Royal Holloway / University of London, UK
Sami Harari, Institut des Sciences de l'Ingénieur de Toulon et du Var / Université du Sud Toulon Var, France
Dan Harkins, Aruba Networks, Inc., USA
Ragib Hasan, University of Alabama at Birmingham, USA
Masahito Hayashi, Nagoya University, Japan
Michael Hobbs, Deakin University, Australia
Neminath Hubballi, Infosys Labs Bangalore, India
Mariusz Jakubowski, Microsoft Research, USA
Ángel Jesús Varela Vaca, University of Seville, Spain
Ravi Jhawar, Università degli Studi di Milano, Italy
Dan Jiang, Philips Research Asia Shanghai, China
Georgios Kambourakis, University of the Aegean, Greece
Florian Kammueller, Middlesex University - London, UK
Sokratis K. Katsikas, University of Piraeus, Greece
Seah Boon Keong, MIMOS Berhad, Malaysia
Sylvia Kierkegaard, IAITL-International Association of IT Lawyers, Denmark
Marc-Olivier Killijian, LAAS-CNRS, France
Hyunsung Kim, Kyungil University, Korea
Ah-Lian Kor, Leeds Metropolitan University, UK
Evangelos Kranakis, Carleton University - Ottawa, Canada
Lam-for Kwok, City University of Hong Kong, Hong Kong
Jean-Francois Lalande, ENSI de Bourges, France
Gyungho Lee, Korea University, South Korea
Clement Leung, Hong Kong Baptist University, Kowloon, Hong Kong
Diego Liberati, Italian National Research Council, Italy
Giovanni Livraga, Università degli Studi di Milano, Italy
Gui Lu Long, Tsinghua University, China
Jia-Ning Luo, Ming Chuan University, Taiwan

Thomas Margoni, University of Western Ontario, Canada
Rivalino Matias Jr ., Federal University of Uberlandia, Brazil
Manuel Mazzara, UNU-IIST, Macau / Newcastle University, UK
Carla Merkle Westphall, Federal University of Santa Catarina (UFSC), Brazil
Ajaz H. Mir, National Institute of Technology, Srinagar, India
Jose Manuel Moya, Technical University of Madrid, Spain
Leonardo Mostarda, Middlesex University, UK
Jogesh K. Muppala, The Hong Kong University of Science and Technology, Hong Kong
Syed Naqvi, CETIC (Centre d'Excellence en Technologies de l'Information et de la Communication),Belgium
Sarmistha Neogy, Jadavpur University, India
Mats Neovius, Åbo Akademi University, Finland
Jason R.C. Nurse, University of Oxford, UK
Peter Parycek, Donau-Universität Krems, Austria
Konstantinos Patsakis, Rovira i Virgili University, Spain
João Paulo Barraca, University of Aveiro, Portugal
Juan C Pelaez, Defense Information Systems Agency, USA
Sergio Pozo Hidalgo, University of Seville, Spain
Vladimir Privman, Clarkson University, USA
Yong Man Ro, KAIST (Korea advanced Institute of Science and Technology), Korea
Rodrigo Roman Castro, Institute for Infocomm Research (Member of A*STAR), Singapore
Heiko Roßnagel, Fraunhofer Institute for Industrial Engineering IAO, Germany
Claus-Peter Rückemann, Leibniz Universität Hannover / Westfälische Wilhelms-Universität Münster / North-German Supercomputing Alliance, Germany
Antonio Ruiz Martinez, University of Murcia, Spain
Paul Sant, University of Bedfordshire, UK
Reijo Savola, VTT Technical Research Centre of Finland, Finland
Peter Schartner, University of Klagenfurt, Austria
Alireza Shameli Sendi, Ecole Polytechnique de Montreal, Canada
Dimitrios Serpanos, Univ. of Patras and ISI/RC ATHENA, Greece
Pedro Sousa, University of Minho, Portugal
George Spanoudakis, City University London, UK
Lars Strand, Nofas, Norway
Young-Joo Suh, Pohang University of Science and Technology (POSTECH), Korea
Jani Suomalainen, VTT Technical Research Centre of Finland, Finland
Enrico Thomae, Ruhr-University Bochum, Germany
Tony Thomas, Indian Institute of Information Technology and Management - Kerala, India
Panagiotis Trimintzios, ENISA, EU
Peter Tröger, Hasso Plattner Institute, University of Potsdam, Germany
Simon Tsang, Applied Communication Sciences, USA
Marco Vallini, Politecnico di Torino, Italy
Bruno Vavala, Carnegie Mellon University, USA
Mthulisi Velempini, North-West University, South Africa
Miroslav Velev, Aries Design Automation, USA
Salvador E. Venegas-Andraca, Tecnológico de Monterrey / Texia, SA de CV, Mexico
Szu-Chi Wang, National Cheng Kung University, Tainan City, Taiwan R.O.C.
Piyi Yang, University of Shanghai for Science and Technology, P. R. China

International Journal on Advances in Security
Volume 6, Numbers 3 & 4, 2013

CONTENTS

Firewall Analysis by Symbolic Simulation: Advanced Optimizations

Arno Wagner
Consecom AG
Zurich, Switzerland
arno@wagner.name

***Abstract*—There are two primary tasks when doing a Layer 4 firewall security analysis. First, unifying a chain of firewalls on a given network path into a single one to efficiently determine what it allows to pass and what it drops, and second, comparing a firewall with a security policy. Both tasks are work-intensive and error-prone if performed manually and become infeasible in the presence of large firewall rule sets. To automate the process of unifying a chain of firewalls, we have created the Consecom Network Analyzer that uses symbolic simulation with an interval representation to generate a unified equivalent firewall in a normalized, simple and flat form. The unification process is also suitable to implement comparison with a policy, by representing the policy in a special way in the form of a firewall rule set. We show the suitability of this approach for firewalls with large configurations by giving benchmarks based on deployed rule sets. In addition, we demonstrate the effects of different optimization techniques on runtime and memory footprint, including the use of an advanced optimization technique that builds on ideas from geometrical search to reduce unnecessary rule applications by means of interval search trees. The Consecom Network Analyzer has been used successfully for a number of industrial security reviews.**

Keywords-Network Security; Firewall Analysis; Symbolic Simulation; Interval Search Trees.

I. Introduction

This work describes the *Consecom Network Analyzer* (CNA), which is the result of a collaboration between academia and industry. It is an invited extension of results previously published in [1]. The main improvement is the use of *Interval Search Trees* as additional optimization technique, as described in Section VII.

The CNA is a tool-set that greatly reduces the effort, and thereby cost, for practical firewall security analysis in the presence of firewalls with large rule sets. A firewall security analysis is one type of network security review. It is often done on network Layer 4, for example for TCP and UDP traffic. Figure 1 shows the basic scenario. The typical steps to be done include:

1) Normalize firewall configurations
2) Identify critical network paths
3) Identify firewalls along each critical path
4) Determine network reachability on each critical path
5) Compare reachability and security requirements
6) Identify non-compliant firewall rules

The primary motivation for creating the CNA lies in steps 4, 5 and 6. In step 4, the CNA calculates the reachability in a unified simple format. Each element of the combined reachability is annotated with the firewall rules that give raise to it. If a formalized or easy to formalize security policy is available, it can be compared automatically to the actual network reachability using the CNA. As such a security policy is often not available in practice, step 5 may still need to be done manually or can be only partially automatized.

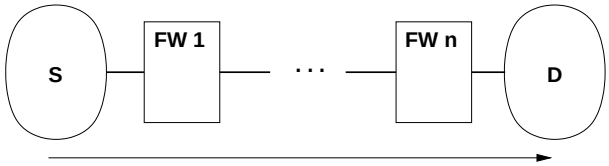


Fig. 1. Unidirectional reachability along a critical network path.

Figure 2 shows the typical data flow for a firewall analysis task. The Rule-Set Converter is not part of the core CNA system and has to be adapted for each different firewall description format. The CNA uses a normalized symbolic Layer 4 format internally that is based on intervals. As core contribution of this paper, we show this representation is suitable for calculating reachability even in the presence of large firewall configurations. To this end, we present benchmark calculations on deployed rule-sets. The CNA has been used successfully in several industrial firewall security reviews.

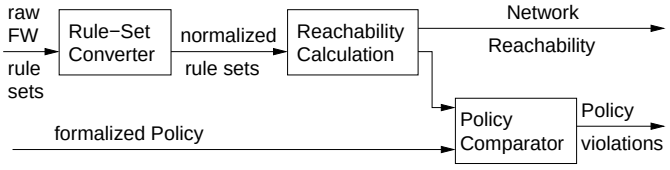


Fig. 2. Typical analysis data-flow with the CNA.

The rest of the paper is organized as follows: Section II introduces our network and firewall model, and the

symbolic representation used. Section III gives the operations used for single firewalls. Section IV explains how to calculate unidirectional reachability. A complexity analysis is sketched briefly in Section V. Section VI describes the implementation, while Section VI states benchmark results and the effects of different optimization techniques. Section VII explains how interval search trees can be used to speed up the CNA core loop and justifies their effectiveness with a separate set of benchmarks. Section VIII explains how to extend the approach to two-sided reachability and to automated comparison with a policy. The paper finishes with a discussion of related work in Section X and a conclusion in Section XI.

II. Approach

The reachability calculation process starts with a representation of the initial reachability (disregarding firewalls), which will often be unconstrained. This initial reachability is then successively reduced by applying firewall rules. The end-result is a flat, unified representation of the firewall-chain, restricted by the initial reachability.

A. Network Model

We are primarily interested in network reachability as restricted by firewalls. Given a source network S, sequence of firewalls FW_1, ... , FW_n and a destination network D (see also Figure 1), we say that D is *reachable* from S if there are network packets that can traverse FW_1, ..., FW_n without being dropped by any FW_i. Note that some attacks will need *two-sided reachability*. For example services used over TCP can usually only be attacked if response packets can traverse the firewall sequence in reverse order. See Section VIII-A for a discussion on how to check for two-sided reachability.

We restrict the packet information visible to firewalls to IP addresses and ports, which results in a Layer 4 model. Each protocol is treated separately, although it is possible to mix protocols, for example by doing a forward analysis with TCP and a backward analysis with ICMP in order to determine whether an ICMP response to a TCP packet would get through. This situation arises, for example, when determining whether a firewall configuration allows port scanning. Routing is out of scope for this work, as we do not see it as a security mechanism; see Section IV-A for a brief discussion.

B. Subspaces, Boxes and Intervals

Reachability is represented by subspaces of

$$M = \{\text{src IPs}\} \times \{\text{src ports}\} \times \{\text{dst IPs}\} \times \{\text{dst ports}\}$$

with the four fields representing the corresponding IP v4 layer 4 header address fields for TCP and UDP, and the port fields being misused to represent ICMP Type and Code for ICMP. Other layer 4 protocols that fit this scheme can also be represented.

We organize these subspaces into sets of axis-aligned hyperrectangles in M, also called *axis aligned boxes* [2], [3]. In this paper, boxes will always be axis-aligned, hence we will simply call them *boxes* for short.

Note that any non-empty subspace of M that has an interval for each of its 4 components trivially is a box. At the same time, *any* subspace of M can be represented as the union of a set of boxes. A subspace A of M can hence be represented by

$$\begin{aligned} A &\subseteq M \text{ and} \\ A &= \{b_1, \ldots, b_n\} \text{ with } b_i \in M \text{ and } b_i \text{ is a box.} \end{aligned}$$

The matching expressions of a firewall rule can be represented by a single box. Security policies can also be represented this way, by giving a set of boxes that specifies forbidden reachability. If the intersection between network reachability and a policy represented this way is non-empty, then the policy is violated. In the implementation, boxes can have attached information. In particular, trace information can be attached in order to document which firewall rules were applied to a box. Trace information is critical to determine why a specific box is in the final reachability or why it was dropped.

A box can be represented as a 4-tuple of intervals, which allows symbolic computations. As far as we know, Eronen and Zitting [4] were the first to use intervals in this context.

Box example:
$b = (10.0.0.0 - 10.0.0.255, 1024 - 65535, 10.1.1.1, 80)$

We use intervals with wrap-around, where IP and port number spaces are regarded as circles. This facilitates representing complements and reduces the number of elements in the complement of a box, see below. Figures 3 and 4 gives graphical examples of three boxes in two dimensions represented this way. Some textual examples for intervals with wrap-around are:

- Port interval $[81, 80)$ represents all ports except port 80, i.e., port 81-65535 and port 0-79. Without wrap-around this complemented interval would need to be represented as $[0, 80)$ and $[81, 65535)$
- IP interval $[127.0.0.256, 127.0.0.0)$ represents all IP addresses except $127.0.0.0 - 127.0.0.255$.

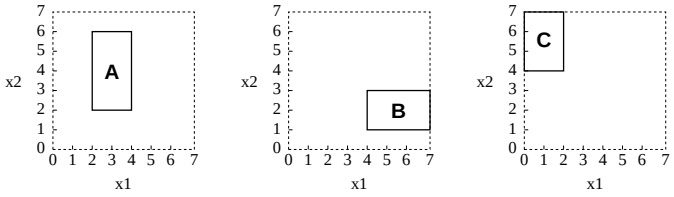


Fig. 3. Boxes in two dimensions.

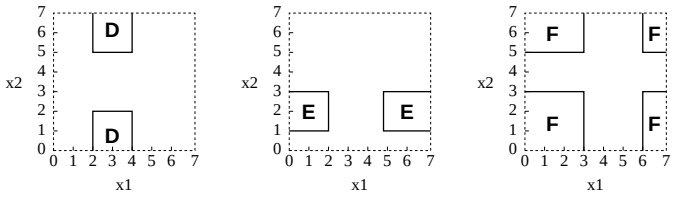


Fig. 4. Boxes with wrap-around in two dimensions.

C. Firewall Model

The CNA uses a simple firewall model, where each firewall consists of a linear sequence of rules r that each have a box describing their applicability and one of the target actions *accept* or *drop*, with a default *drop* at the end of sequence. This corresponds to the "simple" model used in [5].

D. Rule Application and Set Operations

In order to apply a firewall rule $r = (b, <\text{action}>)$ to a subspace $A = \{b_1, \ldots, b_n\} \subseteq M$, we intersect b with the different b_i in turn and apply the action to the result $A \cap \{b\} = \{b \cap b_1, \ldots, b \cap b_n\}$.

The usual set operations are defined on boxes and, by extension, on subspaces of M. Some deserve additional comments.

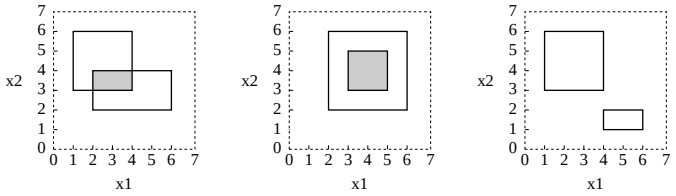


Fig. 5. Box intersections in two dimensions.

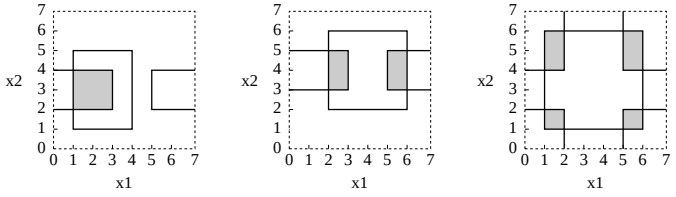


Fig. 6. Box intersection in two dimensions with wrap-around.

Intersection: Intersecting two boxes in d dimensions can have up to 2^d result boxes. Figures 5 and 6 illustrates this in two dimensions. For $b_1, b_2 \in M$, the intersection $b_1 \cap b_2$ may consist of up to 16 boxes as M has 4 dimensions.

Box complement: The complement of an interval is derived by adjusting the boundaries. The complement of a box is derived by complementing each interval in turn and setting all other intervals to full range. Hence, a 4-dimensional box has up to four boxes as its complement. Without wrap-around, the complement of a box could have up to 8 elements.

Subtraction: Calculating $a - b$ for boxes a and b is done by using the relation $a - b = a \cap \bar{b}$ from set calculus.

III. Restricting Reachability by a Single Firewall

The core operations used in determining reachability through a single firewall are `apply_firewall()` and `apply_rule()`, shown in Figure 7 in simplified form. The task of `apply_firewall()` is to take a given reachability description, stated as a set of boxes, called here a *Work Set* (WS) and, using the rules of the firewall, determine both an *Accept Set* (AS), which is the part of the WS that can pass the firewall, and a *Drop Set* (DS) that is the part of the WS that cannot pass the firewall. AS and DS are represented as sets of boxes. The function `apply_rule()` forms the basis of `apply_firewall()` and implements calculation of the intersection I between a given rule and WS. The intersection I is then added to the AS for an *accept* rule or to the DS for a *drop* rule.

```
apply_firewall(WS, FW):
  AS := ∅      /* Accept Set */
  DS := ∅      /* Drop Set   */
  for r ∈ in FW:     /* r: box of a rule */
    I := apply_rule(WS, r)
    WS := WS - I     /* reduce Work Set  */
    if r is accept:  AS := AS ∪ I
    if r is drop:    DS := DS ∪ I
  return(AS, DS)

apply_rule(WS, r):
  I := ∅
  for b ∈ WS:        /* b is a box */
    i := b ∩ r
    I := I ∪ i
  return(I)
```

Fig. 7. Pseudo-code for `apply_firewall()` and `apply_rule()` (simplified).

Building on these two operations, more complex operations can be constructed. Note that `apply_rule()` may attach trace information to boxes, for example to document rule application. If desired, the full history of each box can be recorded in the trace. This allows to determine the specific firewall rules that are responsible for a box being in the final reachability and represents information needed in any report about firewall configuration issues.

IV. Unidirectional Reachability Computation

Pseudo-code for the calculation of unidirection reachability through a *sequence* of firewalls is given in Figure 8.

We will typically choose the initial reachability as unrestricted. This is a sound practice, as network routing can usually not be regarded as a security feature and quite a few customers cannot specify source network `S` and destination network `D` with the required exactness. Starting with full, unconstrained reachability will ensure the final results only rely on the given firewall configurations. A more restricted initial reachability can still be used when appropriate. Ports are unconstrained in the initial reachability.

A. Comments on Routing

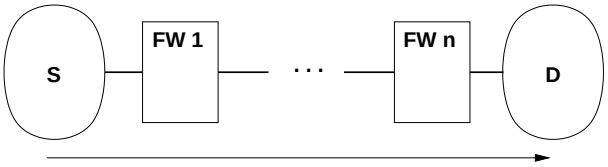
A frequent issue that crops up when doing a firewall security analysis in the field is that often routing is mixed with firewalling. This view gives flawed results. There are several reasons:

- The primary task of routing is to get packets to a specific destination, while the primary task of a firewall is to prevent packets reaching a specific destination. Routing configuration and firewall configuration hence have diametrically opposed primary tasks and this is reflected in procedures and mind-sets.
- Due to the different primary tasks, often the teams responsible for routing and for firewalls are different.
- While firewall configurations are handled securely and all updates are done with the security model in mind, routing configurations are typically changed with the network model in mind and handled in a less secure fashion. Routing is hence easier to compromise.
- Sometimes customers cannot even specify the IP ranges of `S` and `D` precisely, but have precise firewall information. This may sound surprising, but if routing delivers more to a physical target network than expected, this is not necessarily a problem. For firewalls, it is a critical error.
- Routing works on Layer 3, while firewalls work on Layer 4. Mixing the two complicates things and increases maintenance effort.
- Firewall configurations often do not include information about physical or virtual interfaces, but solely refer to layer 4 information. If routing were regarded as a security feature, interface information would be needed in addition and would be critical for security. This would also complicate firewall configuration and make network security critically dependent on the details of physical or virtual network cabling.

Overall, it is far more practical to separate routing and firewalls and to require that all restrictions on reachability must be implemented by firewalls placed into the critical network paths. This is especially true for customers with complex firewall configurations.

It should be noted that with this approach, the question arises whether a specific firewall actually is on the critical network paths it is supposed to be on. Answering this question requires a network topology analysis and is outside of the scope of this work.

It should also be noted that network scanning always takes routing into account and is restricted by it. This is a fundamental limitation of network scanning that is not present in firewall simulation approaches.




```
in:  S, D  /* Source, Destination networks */
     FW1, ..., FWn   /* firewalls          */
out: ASn             /* final reachability */
     DS1,...,DSn     /* Drop Sets          */

WS1 := S × <all> × D × <all>
(AS1, DS1) := apply_firewall(WS1, FW1)
WS2 := AS1
(AS2, DS2) := apply_firewall(WS2, FW2)
WS3 := AS2
...
(ASn, DSn) := apply_firewall(FWn − 1, WSn − 1)
```

Fig. 8. Pseudo-code for calculating unidirectional reachability with `apply_firewall()` for the scenario shown in Figure 1.

V. Algorithmic Complexity

We briefly sketch the complexity analysis idea. For a worst-case scenario, start with one box and a single firewall with n drop rules. Each drop rule can split (asymptotically) at most one element of the Work Set into a maximum of 2^d (with dimension $d = 4$) non-overlapping parts that are kept in the working set. Hence, each rule increases the size of the working set by a maximum of 16, giving an overall space complexity of the result of $16 * n \in O(n)$ for n firewall rules. As each successive rule application has to work on 16 more boxes, time complexity is $1 * 16 + 2 * 16 + \ldots + n * 16 = 16 * (1 + 2 + \ldots + n) = \frac{16}{2} n(n-1) \in O(n^2)$. A very similar argument applies to accept rules and mixed rule-sets.

In comparison, in [6], the authors need worst case effort $O(n^4)$ to build a Firewall Decision Diagram (FDD) for n firewall rules with the same firewall model as we use. It is reasonable to expect that this worst-case is extremely unlikely to happen in practice.

In [5], the authors claim a worst case complexity of $O(n)$ for processing a firewall with n rules in their "simple model". However, they wrongly assume constant effort for set operations on their accept (A) and drop (D) sets. While the BDDs used in [5] are often very efficient in practice, they do not have constant worst case effort for set

TABLE I
BENCHMARKS

No	Firewall or Firewall sequence	rule-set size raw	nor-malized	opt.	benchmark results Python baseline		input opt.		trace reduction		core-loop ported to C	
1	S	27	2'000	180	4:52min	12MB	3.9s	6MB	3.2s	6MB	0.07s	6MB
2	M	67	23'000	8300	1752 min	184MB	346min	84MB	148min	48MB	29s	18MB
3	L	170	27'000	3100	2392min	292MB	21:54min	26MB	12:45min	18MB	2.8s	10MB
4	S, M				64min	34MB	191s	14MB	156s	13MB	1.8s	13MB
5	M, S				1870min	186MB	347min	84MB	146min	48MB	29s	19MB
6	M, L				5000min	187MB	660min	77MB	250min	56MB	38s	21MB
7	S, M, L				205min	58MB	370s	16MB	305s	16MB	4s	16MB

operations and the stated complexity analysis is therefore incorrect.

VI. Implementation, Optimization and Benchmarks

The CNA is implemented in Python 3 [7] with C extensions. This allows a clean and flexible OO design and facilitates targeted optimization. IP addresses and port numbers are represented directly by Python integers. Boxes are represented as Python 8-tuples (representing 4 intervals) and encapsulated into class objects in order to allow attachment of traces, annotations and firewall rule actions. Subspaces are represented as Python lists. The pure-Python prototype is relatively slow and has high memory consumption, but can already be used for security reviews involving firewalls with small and medium-sized rule-sets.

First, note that in the absence of Network Address Translation (NAT), which is rarely deployed in security critical networks, firewalls can be arbitrarily reordered, as exactly those packets that make it through *all* of them are part of the final reachability space. In particular, a good selection of the first firewall to be processed can have significant performance benefits. Benchmarks must therefore always be seen together not only with the relevant firewall configurations, but also their processing order.

A. Benchmarks

In order to determine performance and to examine the performance impact of different optimizations, we give a selection of benchmark results[1] in Table I. Times are CPU times including input data parsing and result output. Memory sizes are the whole process memory footprint, excluding shared areas (libraries). The calculations were done using Linux (Debian Squeeze 32bit) on an AMD Phenom II X4 970 CPU with 3.5GHz, using only one CPU at a time. Memory was set to the 4GB memory model and the machine was running kernel 3.4.7 from kernel.org without any special optimizations. Python version used was 3.1.

Lines $1, 2, 3$ of Table I describe the firewall configurations used. These are firewall configurations deployed in the real world. They have a flat form (no sub-chains) and a default-drop policy.

Line 4 and following lines of Table I give benchmarks for different firewall combinations. The order of the firewalls is important as the first one has to be completely represented in memory, which causes effort $O(|\mathrm{FW}_1|^2)$ (where $|\mathrm{FW}_k|$ is the number of rules in firewall FW_k). The effort for each additional firewall in the chain is $O((|\mathrm{WS}_i|+|\mathrm{FW}_i|)\cdot|\mathrm{FW}_i|)$ and hence higher in the worst case. But when starting with a firewall with small rule-set, we observed that a later combination with a firewall with a large rule-set does often not increase the WS size significantly, as most rules of the larger firewall do not apply. For that case, the complexity goes effectively down to $O(|\mathrm{WS}_i| \cdot |\mathrm{FW}_i|)$, which is a lot smaller than $O(|\mathrm{FW}_i|^2)$ if $|\mathrm{FW}_i|$ is large but $|\mathrm{WS}_i|$ is small. If the firewall processed first has a much larger rule-set than the others, we have observed that processing it will often dominate the runtime.

The columns "rule-set size" give the number of rules in the raw input in vendor format (including groupings, lists, etc.), the normalized number of rules without optimization and the optimized rule-set size. Benchmarks are given only for TCP for brevity, UDP and ICMP analysis have comparable results. We do not have benchmarks for comparison against a policy, as we do not have a sufficiently formalized policy and hence looking directly at reachability was more efficient. Comparison with a policy would incur effort comparable to adding one more firewall configuration in the size of the negated policy specification. The idea is that nothing must be able to pass through the given firewall chain *and* an additional firewall representing the negated policy, with the negated policy representing all forbidden traffic.

As can be seen in Table I, each evaluated optimization step has significant impact on observed run-time. The final implementation with all optimizations included has very reasonable performance even in the presence of firewalls

[1] While in theory there is no difference between theory and practice, in practice there is and benchmark results are very much subject to this limitation. Hence the stated benchmark results only give a rough idea about runtime, memory footprint and effects of different optimizations.

with large rule-sets.

B. Firewall Evaluation Sequence Optimization

The benchmarks demonstrate that the selection of the first firewall to be processed has a huge impact on performance. For the first firewall, the Work Set can grow for each rule application as it has to be completely represented in memory, while for later firewalls only rules that have a non-empty intersection with the Work Set can increase Work Set size by splitting elements already contained in it.

If the first firewall contains a large number of rules that allow traffic through that is later dropped by the other firewalls, then all these irrelevant rules will cause significant load on the memory allocator that can be avoided with a different selection for the first firewall to be processed. Our experiences show that the most restrictive firewall configuration should be processed first. In many scenarios, this will be the smallest firewall configuration, measured in number of rules.

C. Rule-Set Representation Optimization

Firewall configurations in a vendor-format often allow more complex specifications, such as lists or groupings of multiple sources, destinations or services. Decomposing such input rules into rules using a single box each can results in a number of normalized rules that is a lot higher than needed. The reason is that many resulting rules will be overlapping or adjacent in such a way that they can be combined. The column "opt." under "rule-set size" in Table I states the reduced number of rules after optimization and the column "input opt." gives the improved run times and memory footprints. The runtime for the input optimization itself is small, as it only works with a focus of one raw input rule at a time.

Note that global box combination would be possible, but combining boxes from different raw rules has two problems: First, if both *accept* and *drop* rules are present, the combination algorithm has to take rule sequence into account. And second, in this approach a box cannot be labeled with the single raw firewall rule it originated from. This makes the identification of policy-violating rules in the end-result difficult.

D. Trace Reduction

While the original prototype retained traces for all operations that changed a box, it turns out these full traces are only beneficial for debugging. In a security analysis, only *accept* and *drop* actions are relevant and hence it is enough to add trace information to a box when it is added to the Accept Set or Drop Set. It is not necessary to trace when boxes are reduced or split in the Work Set. Hence, traces were reduced accordingly. This also means that there can be at most one trace entry per firewall in each box contained in the result. The column "trace reduction" in Table I states the additional performance gains. Note that trace reduction was benchmarked with input optimization applied as well.

E. Core-Loop Ported to C

In a last step, the core loop function `apply_rule()` was ported to C and embedded into the Python code. Contrary to Figure 7, WS, AS and DS are passed to `apply_rule()` and are manipulated in-place according to the rule action. This puts expensive operations, such as data-structure manipulations, into the C code. No other special optimizations were done for the C code and in particular the standard GNU libc memory allocator was used. The column "core-loop ported to C" in Table I states final performance figures. Note that trace reduction and rule-set representation optimization was applied as well.

In addition, we performed a benchmark calculation for deployed firewall configuration "XL". It has a normalized rule-set size of 2.8 million rules, which reduces to 300'000 rules after input optimization. Raw rule number is 95. Representing configuration XL in memory took 20h of CPU time and resulted in a memory footprint of 900MB. This shows that firewall configurations of this size can still be processed with the CNA with reasonable effort.

The C code can keep box description efficiently in structs and does not need any wrapping and unwrapping of tuple elements and can therefore speed up execution massively, while at the same time reducing memory footprint significantly. However, the unit tests written in Python can still be applied by exposing the interval and box operations implemented in C to Python via the class interface. This helped significantly in the optimization effort.

VII. Advanced Optimization

The algorithm described so far compares each working set element against each rule. This leads to effort linear in the size of the Work Set and linear in the size of the rule set. This is problematic for large inputs. At the same time, for typical firewall rule sets, most elements of any given Work Set do not intersect most rules and hence a large part of the effort is wasted. If it were possible determine a subset of the Work Set that has a higher likelihood of intersecting a given rule r efficiently, a significant speed-up could be obtained. One such possibility is represented by interval search trees.

A. Interval Search Trees

Different types of interval search trees are known. They include trees that support searching with a point, where the result consists of all intervals in a given set that include the point, and searching with an interval, where the result includes all intervals that intersect the given search interval. We need the second variant.

TABLE II
BENCHMARKS: WORK SET AS ARRAY VS. WORK SET AS INTERVAL SEARCH TREE

No	Firewall or Firewall sequence	rule-set size(s) (opt.)	array	interval search tree
1	A	100	0.06s, 8.5MB	0.06s, 8.5MB
2	B	7.5k	4.8s, 27MB	1.2s, 28MB
3	C	1.3M	163h, 15GB	96min, 15GB
4	A,B	100, 7.5k	1s, 22MB	1s, 22MB
5	B,C	7.5k, 1.3M	16:05min, 2.6GB	2:47min, 2.6GB
6	A,B,C	100, 7.5k, 1.3M	2:49min, 2.6GB	2:45min, 2.6GB

As we want to represent the Work Set in an interval search tree, we also need efficient insertion and deletion of intervals from an already constructed tree. Unfortunately, many interval search tree variant do not support these operations efficiently and to the best or our knowledge, no multi-dimensional interval search tree variant can support insertion, deletion and searching with an interval, efficiently.

Due to these restrictions, we selected the interval trees from [8], Section 14.3. These are one-dimensional interval search trees constructed from balanced trees and support all operations we need efficiently. In [8], they are constructed on top of red-black trees as they are claimed to be simpler to implement than alternatives. As an implementation using AVL trees generally gives a smaller tree-height, we adapted the idea from [8] to AVL trees and used them as basis for our implementation.

The complexity of performing an interval search on an interval search tree with n elements is $O(k \cdot log(n))$, with k the number of results. For large k, the overall effort is bound by n, as each tree element is at most inspected once. For example, when the search result includes the full set of tree elements, the effort is only $O(n)$ and not $O(n \cdot log(n))$.

As one-dimensional interval search trees can only handle one component of the 4 different dimensions represented in a box, the idea is to use the most selective dimension of the set of multi-dimensional sets in the interval search, and then iterate linearly over the results as before. For typical large firewall rule sets, the most selective interval is the destination IP address interval. It is possible to use a different dimension. It would also be possible to use several interval search trees for the different dimensions, and then, for a given rule, perform the interval search in each dimension and then continue processing with the smallest result. It should be noted that using one-dimensional interval trees does not decrease the theoretical worst-case complexity of the algorithm and hence effectiveness has to be demonstrated by benchmark calculations.

B. Adjusting the Implementation

The core loop modified to use an interval search tree is shown in Figure 9. The `WS`, `AS` and `DS` are now kept as elements of an interval search tree, different from the linear array that was used before. The key effort reduction lies in reducing the Work Set size in `apply_rule()` by performing an interval search on the complete Work Set with the destination IP interval of the rule r. Only elements of the `WS` that intersect this interval in their destination IP component are added to the `WS_reduced` and have the complex box intersection algorithm applied to them.

```
apply_firewall(WS, FW):
  AS := ∅      /* Accept Set */
  DS := ∅      /* Drop Set   */
  for r ∈ in FW:     /* r: box of a rule */
    I := apply_rule(WS, r)
    WS := WS - I     /* reduce Work Set  */
    if r is accept:  AS := AS ∪ I
    if r is drop:    DS := DS ∪ I
  return(AS, DS)

apply_rule(WS, r):
  I := ∅
  WS_reduced := interval_search(WS, r)
  for b ∈ WS_reduced:        /* b is a box */
    i := b ∩ r
    I := I ∪ i
  return(I)
```

Fig. 9. Pseudo code from Figure 7 modified for interval search trees

C. Rules in an Interval Search Tree

An alternative to putting the Work Set elements into an interval search tree is putting the rules into one. The core loop in `apply_rule()` of Figure 9 would then have to be changed to select an element of the Work Set and then apply all rules to it in turn. The set of all rules would first be restricted using the interval search tree to those rules that intersect, for example, the destination IP interval of the Work Set element being processed.

At a first glance, this looks attractive: the rule-set does not change and hence tree construction does only happen once and no additions or deletions are performed on the tree. Unfortunately, the use of the interval search tree for the rules changes the application order of the rules. Rule sets with accept and drop rules can change their semantics whenever an accept and a drop rule are switched with regard to application order.

This means that while it is possible to apply the idea of using interval search trees for the rule sets, it only works correctly for rule sets that are all accept or all drop rules, with a possible final drop or accept, respectively. While many rule sets observed in practice have this form, some of the largest ones we have encountered do not and hence we are unwilling to accept this limitation.

A second, less problematic, limitation is that if rule application order is changed, it becomes more difficult to determine which rule actually accepted or dropped a specific packet. This ambiguity arises when a specific packet could have been accepted (dropped) by a rule R1 or a rule R2, but rule order determines which one actually does it. This becomes meaningful if it is necessary to determine which rule exactly processed a packet, for example if the packet is to be tagged for policy-based routing or a similar application.

D. Benchmarks for the Interval Search Tree Optimization

As the CNA is subject to on-going optimization, the experimental setup and base-line have changed. In particular redundant element copying and inefficient handling of element traces has been eliminated, resulting in a different baseline than thr one given in Table I. At the same time an updated benchmark firewall set was used that is similar in nature to the older one used for Table I, but changes all firewalls to some degree and includes one much larger firewall rule set. To prevent accidental confusion of the benchmark rule sets in the two tables, the firewalls in Table II have been named differently.

The Benchmarks in Table II were performed on an AMD Phenom II core with 3.4GHz core clock and 32GB available memory. The benchmarks were compiled and run in 64 bit mode, using gcc 4.7.2, Python 3.1.3 on Linux kernel 3.10.11. The characteristics of this setup are very similar to the one used for Table I, except for the 64 bit memory model.

The second column of Table II gives the firewall or firewall sequence processed left-to-right. Single firewalls are given as the process of representing a single firewall in memory is the same as processing it as the first element of a chain. The 3rd column lists the optimized rule-set sizes, similar to the 5th column of Table I. For a sequence of firewalls, the individual sizes are stated. The 4th column of Table II gives the runtimes and memory footprint with the classical array-based Work Set representation. These numbers include the full process including input parsing and result output. Finally, the last column of Table II lists execution time and memory footprint with the Work Set placed into an interval search tree.

E. Discussion

As can be seen, for some benchmarks, the advantage of using interval search trees is significant. In particular for computations with large reachabilities and hence large Work Set sizes, a massive speed improvement can be observed.

For computations with small Work Sets, like the firewall sequences ABC or AB, the speed-up is small or non-existent. The main reason is that storing the Work Set in an interval search tree is slower than storing it in an array. At the same time we do not observe any measurable slow-down due to the use of interval search trees and the memory footprint remains nearly the same.

The benchmark results support the claim that representing the Work Set in an interval search tree is superior, as the overhead created by the tree is compensated by smaller box intersection effort even in cases where restrictive firewalls are processed first and small Work Sets ensue. Tests with a synthetic, tiny first firewall that generates a Work Set of only 4 elements combined with firewalls B and C from Table II confirm that even in this extreme case, use of interval search trees does not slow down the computation to any measurable degree. Hence there is no need to retain the old, array-based Work Set representation.

As the optimization using interval search tree retains the full flexibility and expressiveness of the original CNA implementation, and does not increase memory consumption or CPU load even in the worst cases examined, use of interval search trees represents a significant improvement in the usefulness of the CNA for the processing and analysis of large firewall rule sets.

VIII. Performing Advanced Analysis Tasks

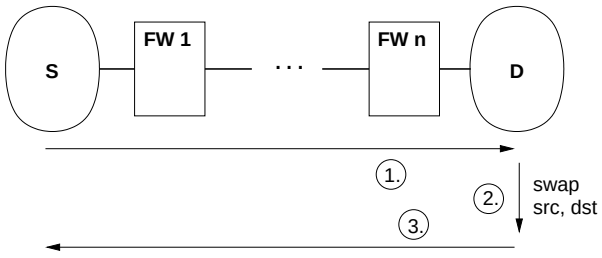
There are two common analysis tasks we have not yet described in detail. One is checking for presence or absence of bidirectional reachability. This answers the question whether a *connection* can be established through a series of firewalls. The second one is checking a chain of firewalls for compliance with a formalized policy. While we have anticipated this task earlier, we now describe how to perform it.

A. Computing Two-Sided Reachability

Two-sided Reachability allows determining whether an agent in the source network S can use a service offered in the destination network D that needs a *connection*, for example any service offered over TCP, or a *response*, as for example TCP port scanning, where TCP SYN packets are sent and potentially answered by ICMP packets. It allows limited comparison with scan results (for example from `nmap` [9]), which are sometimes used to verify a firewall deployment. Figure 10 gives the idea on how to obtain a two-sided reachability result.

B. Verifying Policy Compliance

Policies can be represented as an undesired reachability U, with the meaning that if anything in $U \subseteq M$ is actually reachable through the firewalls, then the policy is violated.




```
in: S, D, FW_0, ..., FW_n
out: AS2n (final reachability), DS0,...,DS2n

WS0 := S × <all> × D × <all>
(AS0, DS0) := apply_firewall(FW0, WS0)
WS1 := AS0
...
(ASn, DSn) := apply_firewall(FWn − 1, WSn − 1)
WSn + 1 := swap_src_and_dst(ASn)
(ASn + 1, DSn + 1) :=
          apply_firewall(FWn, WSn + 1)
WSn + 2 := ASn + 1
...
(AS2n, DS2n) :=
          apply_firewall(FW2n − 1, WS2n − 1)
```

Fig. 10. Calculating bidirectional reachability.

A firewall representing U is constructed by adding accept rules for all traffic components that are undesirable and a final drop rule that drops everything else. In a sense, this firewall acts as a filter that only leaves the undesired components of the actual reachability through a sequence of firewalls.

To test policy compliance, the actual network reachability A on each critical network path is calculated. Let V be the policy-violating reachability. Then $V = A \cap U$. If V is non-empty, all elements of V represent violations. The non-compliant firewall rules can be identified by looking at the trace information attached to elements of V, which they inherit from A.

A rarer compliance test is whether desired reachability is actually present. It can be used to determine which firewall of a firewall chain blocks desired traffic. Here, the desired reachability R is intersected with the subspace D that represents all dropped packets. If the intersection $V = D \cap R$ is nonempty, then parts of R will be dropped by some firewall drop rule and will not be part of the network reachability. As above, the problematic firewall rules can be identified from the traces attached to elements (boxes) of V, which they inherit from D.

Other compliance tests are possible and can be implemented when needed.

IX. Lessons Learned

Input Data: When converting firewall configuration data from customers, we found that significant effort may be needed to account for deviations from expected format convention and outright errors. We expect that for large firewall configurations some manual adaption may be hard to avoid. In seems that in their desire to accommodate customer requests, firewall vendors sometimes allow their customers to do things that are not advisable with regard to clean structuring and consistency, such as overlapping network groups, empty network groups and increasingly more action keywords in new versions. Some of these require manual intervention in order to map them to a unified firewall model. In addition, the right mapping may depend on the actual analysis task to be performed.

Software Engineering: Both, prototyping in Python and providing full, meaningful unit-tests provided hugely beneficial in creating a correctly working prototype and in making sure optimizations did not introduce additional errors. As the same time, keeping the Python-layer as "glue" on top of the implementation of the core loop in C allows for very efficient configuration and scripting of arbitrary analyses. The chosen implementation approach can be qualified as a success and is highly recommended for similar projects.

Performance: We found that run-time and memory footprint allow analysis of large and very large firewalls on standard hardware. This result is unexpected, as the underlying problems are algorithmically not efficient. We theorize that the reason lies in the fact that real-world firewall deployments only sparingly use most of the possibilities that firewalls offer (for example, mixing *accept* and *drop* rules excessively) as the firewall configuration still has to be created and maintainable by human beings.

X. Related Work

Reachability Analysis: One alternative to using the CNA is network scanning, for example with nmap [9]. It should be noted however that this suffers from the limitations that routing affects scanning and that normal scanning cannot find undesired *unidirectional* reachability.

Algorithmic Firewall Analysis: It is possible to formalize firewall functionality with a suitable logic and then use approaches from automated theorem proving to derive properties and check against violation of conditions. Work in this area includes FIREMAN [5] by Yuan, Mai, Su, Chen, Chuah and Mohapatra, which uses a BDD (Binary Decision Diagram) representation. The idea of using BDDs is developed further by Liu and Gouda [6], [10], with the introduction of Firewall Decision Diagrams (FDDs).

A different approach based on Decision Diagrams is described by Liu in [11]. It allows the checking of properties given a specific firewall rule set. The properties are formalized as firewall rules with wildcards, e.g., that no traffic must flow to or from IP address `1.2.*.*`. This formalization has a close relation to our policy checking approach where we formalize a policy as an additional firewall. Unfortunately, [11] only tested performance for

small real-world firewall rule-sets up to 661 rules and hence a meaningful performance comparison with our approach is not possible.

Firewall Models: Leporati and Ferretti [12] use Tissue-like P Systems to model connected sets of firewalls. In [13], Bourdier and Cirstea employ rewrite systems to model firewall filtering and translation rules. Bera, Dasgupta and Ghosh [14] is an example for the use of a Boolean SAT solver to verify firewall ruleset properties.

Firewall Redundancy Analysis: Firewall redundancy analysis is aimed at identifying and removing redundancies in a firewall ruleset, such as rules that have overlapping boxes. While a prolific theoretical field, its relevance to practice is minor. For example, [15], [16] and [17] deal with this aspect of firewall analysis.

Query Engines: The query-engine of Mayer, Wool and Ziskind [18] answers questions on whether a specific packet would traverse a set of firewalls by using a rule-based simulator. This is mostly useful to determine the impact of specific firewall configuration changes. Its value in a complete firewall security analysis is limited. The Margrave Tool [19] uses a similar approach.

Commercial Tools: A commercial firewall analyzer is offered by AlgoSec [20]. This tool seems to be targeted at maintenance and administration of large numbers (up to 1000) of firewalls. Commercial firewall maintenance tools with limited audit capabilities are also offered by Tufin [21] and FireMon [22].

XI. Conclusion and Future Work

We have designed and implemented the CNA (Consecom Network Analyzer), a tool that calculates network reachability through a series of firewalls given as a Layer 4 abstraction by symbolic simulation. The primary use is for real-world security audits that examine firewalls with large rule-sets. While using set operations to model firewalls is simple, to the best of our knowledge we are the first to demonstrate that an abstraction based on intervals is efficient enough to calculate reachability through large deployed firewall configurations in practically useful time and with moderate memory footprint, while at the same time retaining the capability to annotate each result sub-set with a full trace of the applied firewall rules. Automated result annotation is essential when analyzing firewall chains that include firewalls with a large number of rules.

We also have demonstrated the effect of a series of implementational and algorithmic optimizations on execution time and memory-footprint. The last step is the application of ideas from geometrical search to use one-dimensional interval search trees for reduction of ineffective rule applications to Work Set elements. The benchmarks given include performance on large firewall rule sets actually deployed in real applications.

One possible direction for future work is further investigation into how multi-dimensional geometric search structures could be used to improve efficiency even more. Primary issues are that most known multi-dimensional search structures do not handle updates (additions and deletions) efficiently. Using these structures for the CNA would mean finding design and implementation trade-offs that work well for real problems, even if their theoretical worst-case performance is bad.

A second possibility for future work is the adaption of the CNA *IPv6* addresses. With the current system, this can be done by swapping out 32 bit unsigned integers for 128 bit unsigned integers in the C code. Python already handles all integers as long-numbers and no change in the Python code would be needed. However, input-parsing and result output would have to be adapted. However, the larger memory footprint may have significant impact on the actual implementation and may require specific additional optimizations to retain efficiency.

Finally, the CNA could be extended to handle subchains in firewall rule sets. At this time, subchains can be handled by a preprocessing step. A native implementation of subchains into the CNA core code by adding suitable rule actions could speed up processing of subchains significantly.

Acknowledgments: We thank the Swiss KTI and Consecom AG for funding parts of this work and the anonymous reviwers for their helpful suggestions.

References

[1] A. Wagner and U. Fiedler, "Firewall Analysis by Symbolic Simulation," in *The Seventh International Conference on Internet Monitoring and Protection (ICIMP 2012)*, 2012, pp. 95–100.

[2] "Wikipedia: Hyperrectangle," http://en.wikipedia.org/wiki/Hyperrectangle, last visited December 2013.

[3] H. S. M. Coxeter, *Regular Polytopes*, 3rd ed. New York: Dover, 1973.

[4] P. Eronen and J. Zitting, "An Expert System for Analyzing Firewall Rules," in *Proc. 6th Nordic Worksh. Secure IT Systems*, 2001, pp. 100–107.

[5] L. Yuan, J. Mai, Z. Su, H. Chen, C.-N. Chuah, and P. Mohapatra, "FIREMAN: A Toolkit for FIREwall Modeling and ANalysis," in *IEEE Symposium on Security and Privacy*, 2006, pp. 199–213.

[6] A. X. Liu and M. G. Gouda, "Diverse Direwall Design," in *IEEE Transactions on Parallel and Distributed Systems, 19(8)*, August 2008.

[7] "The Python Homepage," http://python.org/, last visited December 2013.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Mit Press, 2009.

[9] "Nmap Security Scanner," http://nmap.org/, last visited December 2013.

[10] A. X. Liu and M. G. Gouda, "Firewall Policy Queries," in *IEEE Transactions on Parallel and Distributed Systems, 20(6)*, June 2009.

[11] A. X. Liu, "Formal Verification of Firewall Policies," in *2008 IEEE International Conference on Communications, ICC '08*, 2008, pp. 1494 – 1498.

[12] A. Leporati and C. Ferretti, "Modelling and Analysis of Firewalls by (Tissue-like) P Systems," in *Romanian Journal of Information Science and Technology, Vol. 13, No 2*, 2010, pp. 169–180.

[13] T. Bourdier and H. Cirstea, "Symbolic Analysis of Network Security Policies Using Rewrite Systems," in *Symposium on Principles and Practices of Declarative Programming*, 2011, pp. 77–88.

[14] P. Bera, P. Dasgupta, and S. Ghosh, "Formal Analysis of Security Policy Implementations in Enterprise Networks," in *International Journal of Computer Networks and Communications (IJCNC), Vol. 1, No. 2*, 2009, pp. 56–73.

[15] S. Pozo, A. Varela-Vaca, and R. Gasca, "A Quadratic, Complete, and Minimal Consistency Diagnosis Process for Firewall ACLs," in *24th IEEE International Conference on Advanced Information Networking and Applications*, 2010.

[16] K. Karoui, F. B. Ftima, and H. B. Ghezala, "Formal Specification, Verification and Correction of Security Policies Based on the Decision Tree Approach," in *International Journal of Data and Network Security 08/2013; 3(3):92-111*, 2013.

[17] P. Rajkhowa, S. M. Hazarika, and G. R. Simari, "An Application of Defeasible Logic Programming for Firewall Verification and Reconfiguration," in *Quality, Reliability, Security and Robustness in Heterogeneous Networks, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Volume 115*, 2013, pp. 526–542.

[18] A. J. Mayer, A. Wool, and E. Ziskind, "Offline firewall analysis," *Int. J. Inf. Sec.*, vol. 5, no. 3, pp. 125–144, 2006.

[19] T. Nelson, C. Barratt, D. J. Dougherty, K. Fisher, and S. Krishnamurthi, "The Margrave Tool for Firewall Analysis," in *USENIX Large Installation System Administration Conference (LISA)*, 2010.

[20] "Algosec Homepage," http://www.algosec.com/, last visited December 2013.

[21] "tufin Homepage," http://www.tufin.com/, last visited December 2013.

[22] "FireMon Homepage," http://www.firemon.com/, last visited December 2013.

A Diversified Set of Security Features for XMPP Communication Systems Useful in Cloud Computing Federation

Antonio Celesti, Massimo Villari, and Antonio Puliafito
DICIEAMA, University of Messina
Contrada di Dio, S. Agata, 98166 Messina, Italy.
e-mail: {acelesti, mvillari, apuliafito}@unime.it

***Abstract*—Nowadays, in the panorama of Cloud Computing, finding a right compromise between interactivity and security is not trivial at all. Currently, most of Cloud providers base their communication systems on the web service technology. The problem is that both Cloud architectures and services have started as simple but they are becoming increasingly complex. Consequently, web services are often inappropriate. Recently, many operators in both academia and industry are evaluating the eXtensible Messaging and Presence Protocol for the implementation of Cloud communication systems. In fact, the XMPP offers many advantages in term of real-time capabilities, efficient data distribution, service discovery, and inter-domain communication compared to web service technologies. Nevertheless, the protocol lacks of native security features. In this paper, we explore such security issues, discussing how they can be mitigated using both SAML/SASL Single Sign-On (SSO) and XEP 0027.**

Keywords-Cloud computing, federation, security, XMPP, SSO authentication, data encryption, digital signature.

I. Introduction

Nowadays, Cloud providers and their services are becoming more and more complex. Until now, the trend for Cloud Computing has been to base the communication systems on well-known web services such as the Representational State Transfer (REST) and the Simple Object Access Protocol (SOAP). This model has succeeded so far, however, due to the increasingly degree of complexity that Cloud architectures need for fulfilling the new emerging business scenarios, the achievement of both interactivity and security is not trivial at all. In fact, both REST and SOAP web services present the following disadvantages: they are based on request/response patterns, they do not provide any native asynchronous interaction, their polling does not scale and it is not real-time, they require a two-way data exchange. Consequently, web services make complicated *i)* the presence (availability) and discovery of software modules and services; *ii)* many-to-many distribution patterns; *iii)* asynchronous and multi-step calls to remote services; *iv)* federation with third-party providers and services especially behind firewalls.

For these reasons, most operators in both academia and industry fields are looking at alternative communication systems for Cloud providers. To this regards, a valuable solution consists in adopting the Extensible Messaging and Presence Protocol (XMPP), i.e., an open-standard communications protocol for message-oriented middleware based on the XML (Extensible Markup Language). On one hand, the XMPP is able to overcome the disadvantages of web services in terms of performance, but on the other hand it lacks of native security features for addressing the new emerging Cloud computing scenarios.

In this paper, we discuss how the XMPP can be adopted for the development of secure Cloud communication systems. In particular, we combine and generalize the assumptions made in our previous works respectively regarding how to secure inter-domain federation [1] and how to secure inter-module communication [2] with the objective to provider to the reader a guideline.

The US Department of NIST, is actively working for accelerating Standards to foster the Adoption of Cloud Computing [3]. *Interoperability, Portability and Security* are the main objectives to achieve. Considering the intrinsic nature of the XMPP, the first and the second objective can be easily achieved [4], instead the third one deserves further assumptions. More specifically, to achieve secure federation-enabled Cloud architectures over the XMPP, we will discuss how to carry out Single Sign On (SSO) authentication between providers belonging to different administrative domains, data integrity, confidentiality, and non-repudiation. In order to fulfill such goals, we will discuss an experimental integration of the XMPP with both a) *Security Assertion Markup Language* and *Simple Authentication - Security Layer* (SAML/SASL) for server-to-server SSO authentication and b) XEP-0027 extensions for secure message exchange.

The paper is organized as follows. Section II describes the state of the art. Section III discusses the advantages of using XMPP-based communication systems for complex federation-enabled Cloud architectures. In Section IV, we highlight the limits of the XMPP in term of security. Possible security integration to the XMPP are discussed in Section V. More specifically, we will discuss how the Internet-Draft entitled "A SASL Mechanism for SAML", defined by CISCO TF-Mobility Vienna can be adopted to achieve secure federation between Cloud providers belonging to different administrative domains and how the XEP-0027 Specification can be adopted to achieve message signing/encryption for the inter-module communication. Section VI

concludes the paper.

II. Related Works

In this section, we describe the current state-of-the-art on Security and Cloud computing.

More works treating security and privacy on Cloud computing exist in literature. Many of them provide theoretical discussions on different security aspects, but a few try to find concrete assessments and real implemented solutions [5], [6], [7]. In [8], the authors identify security challenges to manage multi-provider Inter-Cloud environments with dedicated communication infrastructures, security mechanisms, processes and policies. The existing security challenges in Collaborative Clouds are highlighted in [9]. The authors analyze different security initiatives, such as the FCAPS model (ISO 10164), that is helpful for describing security management functionalities, and the ISO/IEC 27001, that offers a methodology for managing security services. We agree with the authors in [10], who assert that an Information Technology (IT) auditing mechanisms and framework can play an important role in compliance of Cloud IT security policies. In particular, a Third Party Auditor (TPA), introduced on behalf of Cloud users, has resources and experience that users do not have and it can be emplaced to audit the integrity of large data stored on Clouds. Their survey highlights the possibility to theoretically use recent technologies for enforcing security from different point of views, as well as SAML, OAuth, HMACs, homomorphic linear authenticators (HLAs), identity and access management as a service (IDaaS). As the architecture of IDS (Intrusion Detecting System), aimed at Nodes NIDS and Hosts HIDS. A collaboration-based Cloud computing security management framework is presented in [11]. The alignment of NIST-FISMA standard with the Cloud computing model is reported in a table form. The table reports the responsibilities of Service Providers (SPs), Cloud Customers (CCs) and Cloud Providers (CPs) in managing security assets. The work that authors described is interesting, because it offers a web portal along with a database where to track Cloud resources utilization and their security implications. Risks Management, where risk probabilities and vulnerability descriptions along with standards are reported. The authors in [12] describe a dynamic hierarchical role-based access control model, useful in Cloud service aimed at Mobile Internet. In particular, they introduce an interesting security model with self-adaptive features. Their self-adaptive schema enables their system to automatically meet the environment parameters, hence offering the corresponding protections. Another work that presents possible threats in Cloud is [13]. The authors highlight that when organizations migrate their data or services into the Cloud, they are not aware of their locations. Organizations lose control over their data and are not aware of any security mechanisms put in place by the Cloud provider. In this situation, the main three security concerns are: *Loss Of Control*, *Compliance Implications* and *Confidentiality and Auditing*. Our work focuses the attention on these issues.

III. Cloud Computing and XMPP

A valuable solution for the design of a performance and secure Cloud middleware is to conceive its communication system adopting an instant message-oriented approach. To this regard the XMPP (RFC 3920 and RFC 3921), also called Jabber, is becoming more and more popular due to its flexibility to suit different scenarios where a high level of re-activeness is strongly required. Despite it was born for human interaction via chat room it can be used to develop the communication of whatever distributed system well fitting the requirements of Cloud computing. XMPP is an XML-based protocol used for near-real-time, extensible instant messaging and presence information. XMPP remains the core protocol of the Jabber Instant Messaging and Presence technology. The "Jabber" technology leverages open standards to provide a highly scalable architecture that supports the aggregation of presence information across different devices, users and applications. Like email, anyone who has a domain name and an Internet connection can run the Jabber server and chat with other users. The Internet Engineering Task Force has formalized XMPP as an approved instant messaging and presence technology, and the specifications have been published as RFC 3920 and RFC 3921. Born for

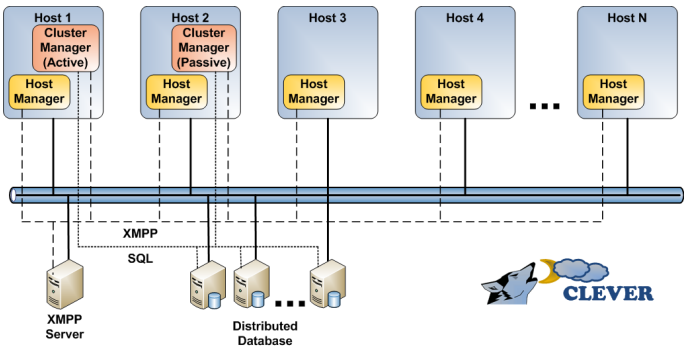


Figure 1. CLEVER architecture.

human interaction via chat the XMPP offers many advantages for the design of communication system of complex distributed system In the panorama of Cloud computing the XMPP represents a flexible solution because custom functionality can be built on top of XMPP, and common extensions are managed by the XMPP Software Foundation. The XMPP provides a technology for asynchronous end-to-end exchange of structured data. Considering a distributed system, the protocol allows to build one or more overlay networks having: global addressing (JIDs), network availability (presence), concurrent information transactions, distributed federated networks, structured data with XML payload.

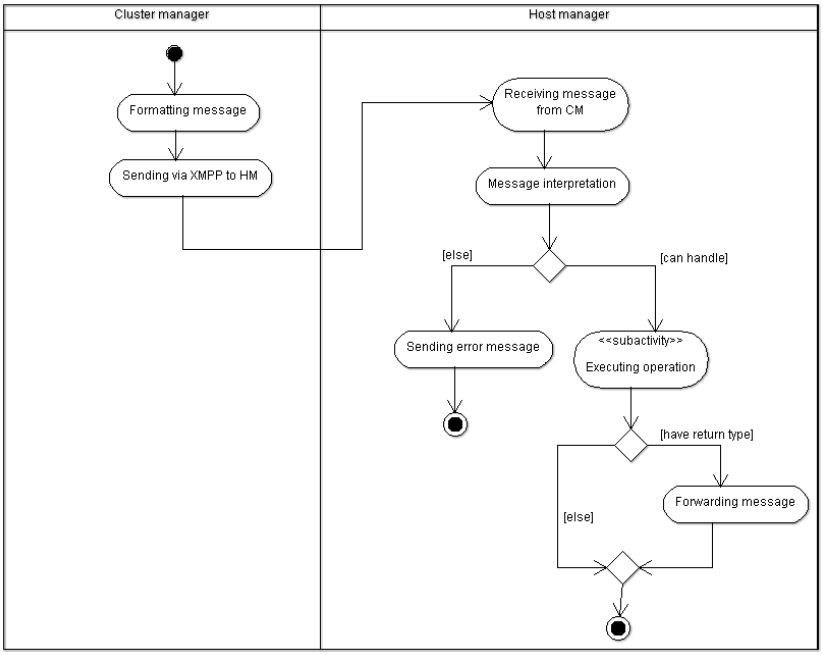


Figure 2. Activity diagram of the external communication.

The architecture is similar to the email network, but it introduces several added value features to facilitate near-real-time communications. The end-to-end communication in XMPP is logically peer-to-peer but logically client-to-server-to-server-to-client. If we assume that each server can manage a domain, a server-to-server connection can enable inter-domain federation. In the XMPP, data are sent over persistent XML steams. XMPP clients (i.e., human or software modules) are connected over a room that represents a sort of broadcast domain.

Summarizing, the XMPP presents several advantages compared to the HTTP-based web services including

- End-to-end communication
- It offers real-time capabilities such as heartbeat, alarms, and any kind of asynchronous communication.
- Efficient distribution of data with public/subscribe and direct-push approaches (e.g., configuration distribution, push RSS/Atom, data collection, log processing, fast results delivery software modules and clients.).
- Advance service discovery.
- Federation. Most firewalls allow users to fetch and post messages without hindrance. Thus, if the TCP port used by XMPP is blocked, a server can listen on the normal HTTP port and the traffic should pass without problems.

In order to describe how an XMPP-based communication system of a federation-enabled Cloud architecture works, in the following we will consider as model CLEVER [14], an Infrastructure as a Service (IaaS) middleware. The CLEVER middleware is based on the architecture schema depicted in Figure 1, which shows a cluster of n nodes (also an inter-connection of clusters could be analyzed) each containing a host level management module (Host Manager). A single node may also include a cluster level management module (Cluster Manager). All the entities interact exchanging information by mean of the Communication System based on the XMPP. The set of data necessary to enable the middleware functioning is stored within a specific Database deployed in a distributed fashion.

Figure 1 shows the main components of the CLEVER architecture, which can be split into two logical categories: the software agents (typical of the architecture itself) and the tools they exploit. To the former set belong both the Host Manager and the Cluster Manager:

- The Host manager (HM) performs the operations needed to monitor the physical resources and the instantiated VMs. It runs the VMs on the physical hosts and performs the migration of VMs.
- The Cluster Manager (CM) acts as an interface between the clients (software entities, which can exploit the Cloud) and the HM agents. It performs the management of VM images (uploading, discovering, etc.) and the monitoring of the overall state of the cluster (resource usage, VMs state, etc.).

Regarding the tools such middleware components exploit, we can identify the Distributed Database and the XMPP Server. The XMPP-based communication system allows to conceive more flexible inter-module communication and

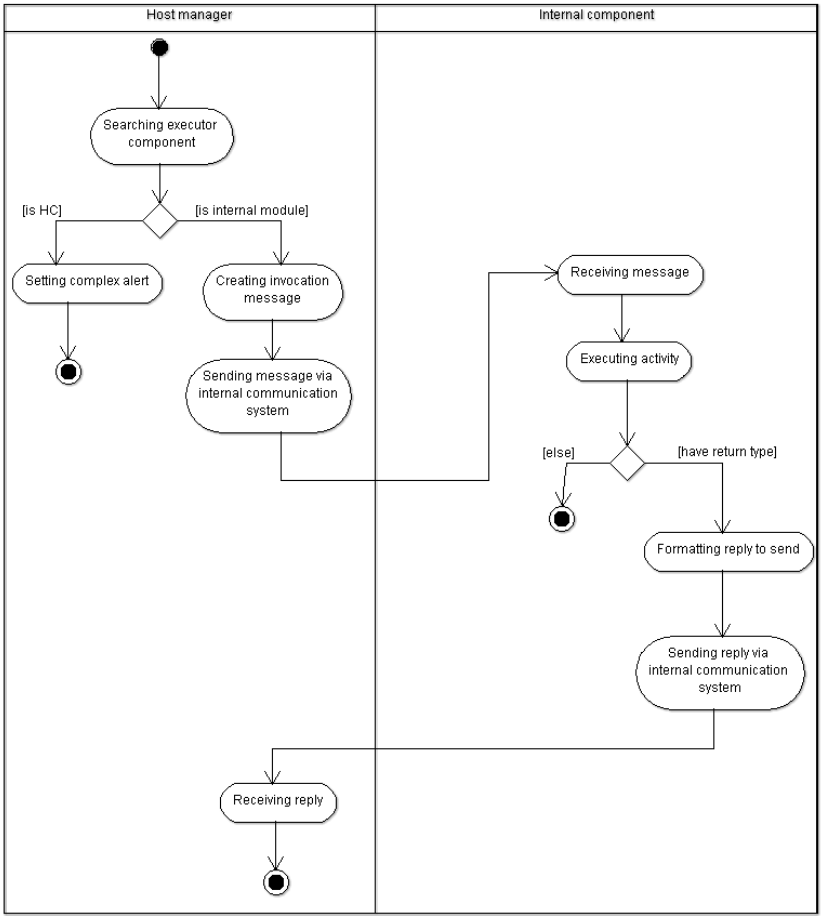


Figure 3. Activity Diagram of the sub-activity Executing Operation.

inter-domain federation capabilities compared to the HTTP-base web services.

A. *Inter-Module Communication*

When two different hosts have to interact each other, the inter-module communication has to be exploited. The typical use cases refer to:

- Communication between CM and HM for exchanging information on the cluster state and sending specific commands;
- Communication between the administrators and CM using the ad-hoc client interface.

As previously discussed, in order to implement the inter-module communication mechanism, an XMPP server must exist within the CLEVER domain and all its entities must be connected to the same XMPP room. When a message has to be transmitted from the CM to an HM, as represented in Figure 2, it is formatted and then sent using the XMPP. Once received, the message is checked from the HM, for verifying if the requested operation can be performed. As the figure shows, two different situations could lay before: if the request can be handled, it is performed sending eventually an answer to the CM (if a return value is expected), otherwise an error message will be sent specifying an error code. The "Execution Operation" is a sub-activity whose description is pointed out in Figure 3. When the sub-activity is performed, if any return value is expected the procedure terminates, else this value has to be forwarded to the CM in the same way has been done previously with the request. The sequence of steps involved in the sub-activity is represented in Figure 3. If the operation that has to be executed involves a component different from the Host Coordinator, the already described intra-module communication has to be employed. Once the selected component receives the message using this mechanism, if no problem occurs, the associated activity will be performed, else an error will be generated. If the operation is executed correctly and a return value has to be generated, the component will be responsible of generating the response message that will be forwarded to the HM, and thus, to the CM.

B. Inter-Domain Federation

CLEVER has been designed with an eye toward federation. In fact, the choice of using XMPP for the CLEVER module communication (i.e., external communication XMPP room) has been made thinking about the possibility to support in the future also interdomain communication between different CLEVER administrative domains. Federation allows Clouds to "lend" and "borrow" computing and storage resources to/from other Clouds. In the case of CLEVER, this means that a CM of an administrative domain is able to control one or more HMs belonging other administrative domains. For example, if a CLEVER domain A runs out of resources of its own HMs, it can establish a federation with a CLEVER domain B, in order to allow the CM of the domain A to use one or more HMs of the domain B. This enables the CM of domain A to allocate VMs both in its own HMs and in the rented HMs of domain B. In this way, on one hand the CLEVER Cloud of domain A can continue to allocate services for its clients (e.g., IT companies, organization, desktop end-users, etcetera), whereas on the other hand the CLEVER Cloud of domain A earns money from the CLEVER Cloud of domain B for the renting of its HMs.

As anyone may run its own XMPP server on its own domain, it is the interconnection among these servers that exploits the interdomain communication. Usually, every user on the XMPP network has an unique Jabber ID (JID). To avoid requiring a central server to maintain a list of IDs, the JID is structured similarly to an e-mail address with an user name and a domain name for the server where that user resides, separated by an at (@) sign. For example, considering the CLEVER scenario, a CM could be identified by a JID bach@domainB.net, whereas a HM could be identified by a JID liszt@domainA.net: bach and liszt respectively represent the host names of the CM and the HM, instead domainB.net and domainA.net represent respectively the domains of the Cloud that "borrows" its HMs and of the Cloud that "lends" HMs. Let us suppose that bach@domainB.net wants to communicate with liszt@domainA.net, bach and liszt, each respectively, have accounts on domainB.net and domain A XMPP servers.

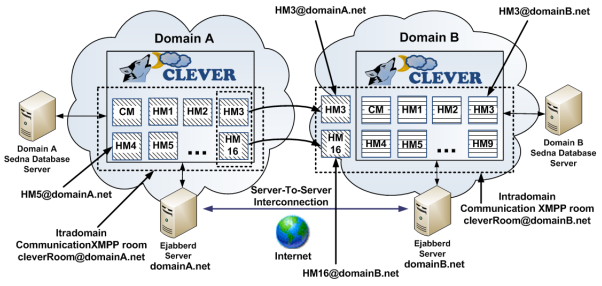


Figure 4. Example of federation between two CLEVER Clouds.

The idea of CLEVER federation is straightforward by means of the built-in XMPP features. Figure 4 depicts an example of interdomain communication between two CLEVER administrative domains for the renting of two HMs from a domain A to domain B. Considering the aforementioned domains, i.e., domainA.net and domainB.net, in scenarios without federation, they respectively include different XMPP rooms for intradomain communication (i.e., cleverRoom@domainA.net and cleverRoom@domainB.net) on which a single CM, responsible for the administration of the domain, communicates with several HMs, typically placed within the physical cluster of the CLEVER domain. Considering a federation scenario between the two domains, if the CM the domainB.net domain needs of external resources, after a priori agreements, it can invite within its cleverRoom@domainB.net room one or more HMs of the domainA.net domain. For example, as depicted in Figure 4, the CLEVER Cloud of domainB.net rents from the CLEVER Cloud of domainA.net, HM6 and HM16. Thus, the two rented HMs will be physically placed in domainA.net, but they will be logically included in domainB.net. As previously stated, in order to accomplish such a task a trust relationship between the domainA.net and the domainB.net XMPP servers has to be established in order to enable a Server-to-Server communication allowing to HMs of domain A to join the external communication XMPP room of domain B.

IV. Security Issues in XMPP-based Clouds

According to the Domains and sub-topics investigated by the CSA, we worked on partial security needs for making a concrete solution aimed at IaaS level. The elements we identified in this assessment that are useful in the IaaS context are presented in the CSA guidance [15] and summarized as following:

1) In the Governance and Enterprise Risk Management, there is the need to *"divulge policies, procedures and processes comprising the Cloud providers' Information Security Management System (ISMS)"*, knowing who makes what.
2) Whereas in the Information Management and Data Security, it is necessary to *"assure that Cloud provider personnel controls are in place to provide a logical segregation of duties."*
3) In the Traditional Security, Business Continuity and Disaster Recovery, *"Customers should perform onsite inspections of Cloud provider facilities whenever possible."*
4) Data Center Operation, *"Cloud providers must all be able to demonstrate comprehensive compartmentalization of systems, networks, management, provisioning and personnel."*
5) In the Incident Response, Notification and Remediation, *"Cloud providers should construct a registry of*

application owners by application interface (URL, SOA service, etc.)".

6) Encryption and Key Management, where *"segregate the key management from the Cloud provider hosting the data, creating a chain of separation"*.

Even though the XMPP support the SASL and TLS technologies for the authentication and encryption of the communication channels between different eJabberd servers, it presents some security limitations due to the decentralized nature of the protocol that demands the accomplishment of specific security mechanisms to the different implementations. On the other hand, the flexible and extensible nature of the protocol allows to integrate basic security mechanisms, improving the level of the security in communication. In particular, considering federation-enabled Clouds, the XMPP does not allow to natively develop the following security mechanisms

- **Data Confidentiality, Integrity, and Non Repudiation for Inter-Module Communication**. As previously discussed, the different software modules can communicate over one or more chat-rooms. Chat-rooms allow to isolate the communication of the involved software modules also providing a way to control which module can join a chat-room by means of a username/password authentication. This level of security is particularly weak considering Cloud architectures. Considering software modules A and B of a Cloud system i) module A and B have to perform a mutual authentication before communicating through X.509 certificates in order to avoid identity-thief attacks; ii) message exchanged between two software modules have to be confidential and not corrupted in order to avoid man-in-the-middle attacks; iii) if software module A sends a message to B, module A cannot deny of having done so. In order to guarantee Data Confidentiality, Integrity, and Non Repudiation for Message Exchange further security mechanisms have to be integrated in the XMPP.
- **SSO Authentication for Inter-Domain Federation**. Federation between Cloud providers implies the establishment of a secure inter-domain communication between the XMPP servers of the involved Clouds. This raises several issues regarding the management of authentication between the XMPP servers of different Clouds. In fact, considering a scalable scenario including n Clouds in order to perform an inter-domain federation the XMPP server of each Cloud should perform $n-1$ authentication on the other $n-1$, hence managing $n-1$ different credentials for accessing the federated Clouds. Considering the whole Cloud federation ecosystem it is required to manage $n(n-1)$ different credentials. The IdP/SP scheme allows to address this problem introducing a trusted third-party, the IdP, so that a cloud that wants to perform a federation with the other $n-1$ Clouds has to perform the authentication once, gaining the access on the other $n-1$ Clouds, which will be trusted with the IdP. This form of federation is called SSO. Unfortunately, at the moment of the writing of this paper, the SASL/TLS on which the XMPP is based does not support any standard form of SSO Authentication for server-to-server federation.

In the following, we will deepen the previously introduced security limitations considering a federated Cloud computing scenario based on the XMPP.

A. XMPP Concerns Regarding Data Confidentiality, Integrity, and Non Repudiation for Inter-Module Communication

Let us consider a message oriented Cloud system including several distributed software modules or components and whose inter-module communication takes place by means of an instant messaging protocol. The question is: which are the security requirements of the involved communication system? Definitely it should ensure: *data confidentiality*, *data integrity*, and *data non-repudiation of the sender/receiver*. Let us assume that in order to achieve a totally secure communication system each message has to be signed and encrypted by each component. From now on, for simplicity,

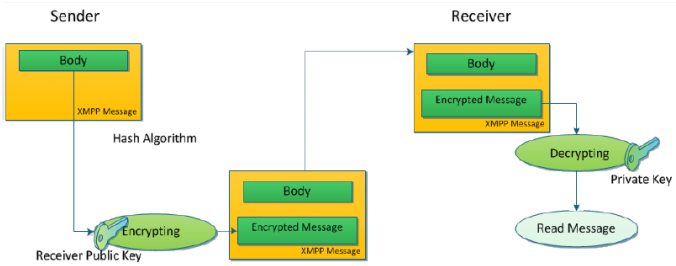


Figure 5. XMPP messages encryption in CLEVER.

we will focus our analysis on the XMPP as instant messaging protocol. Considering the aforementioned security requirements, XMPP as has to be properly extended. In our opinion, considering a Public Key Infrastructure (PKI), the XMMP-based communication of a message oriented Cloud system should support the following functionalities:

- **Digital identity management**. Each component during the in-band registration (i.e., an automatic enrollement of a client on the XMPP server) with the XMPP server requires a digital certificate to a trusted Certification Authority (CA) through the Simple Certificate Enrollment Protocol (SCEP).
- **Signed message exchange**. Each component should be able to sign a message sent to another one.

- **Encrypted message exchange**. Each component should be able to perform a total or partial encryption of a message.
- **Private chat rooms**. The communication system should allow the management of private chat room with restricted access to authorized components.
- **Encrypted chat rooms.** The communication system should allow the management of private and encrypted chat rooms. The key exchange between the communicating components should take place according to a PKI schema. The component that play the role of "moderator" instantiate a new chat room associating a session key. When a new component wants to join the communication, the "moderator" component sends the session key encrypted with the public key of the new component itself.

B. XMPP Concerns Regarding SSO Authentication for Inter-Domain Federation

Considering that the communication in each CLEVER Cloud is achieved through XMPP or Jabber messages by means of an Ejabberd server (i.e., an instant message server), the federation establishment between two or more CLEVER Clouds implies a secure server-to-server inter-domain communication between their respective Ejabberd servers. In fact, in the XMPP terminology, the term "federation" is commonly used to describe communication between two servers.

Thus, in CLEVER, each Cloud belongs to a domain managed by an XMPP server. In CLEVER, the way to federate two Clouds is to establish a server-to-server inter-domain communication between the XMPP servers to the involved Clouds.

Cloud federation raises many issues especially in the field of security and privacy. Single Sign On (SSO) authentication is fundamental for achieving security in a scalable scenario such as Cloud federation. However, the Simple Authentication and Security Layer (SASL) [16], i.e., a framework for authentication and data security in Internet protocols, supported by XMPP does not support any SSO authentication mechanism.

The public-subscribe technology is reemerging for enabling real-time communication within Cloud infrastructure, nevertheless its major protocol XMPP is somewhat dated from the point of view of security.

In order to enable federation between servers, it is needed to carry out a strong security to ensure both authentication and confidentiality thanks to encryption. According to the IETF 6120, compliant implementations of servers should support Dialback or SASL EXTERNAL protocol for authentication and the TLS protocol for encryption.

The basic idea behind Server Dialback [17] is that a receiving server does not accept XMPP traffic from a sending server until it has (i) "called back" the authoritative server for the domain asserted by the sending server and (ii) verified that the sending server is truly authorized to generate XMPP traffic for that domain. The basic flow of events in Server Dialback consists of the following four steps:

1) The Originating Server generates a Dialback key and sends that value over its XML stream with the Receiving Server. (If the Originating Server does not yet have an XML stream to the Receiving Server, it will first need to perform a DNS lookup on the Target Domain and thus discover the Receiving Server, open a TCP connection to the discovered IP address and port, and establish an XML stream with the Receiving Server.)
2) Instead of immediately accepting XML stanzas on the connection from the Originating Server, the Receiving Server sends the same Dialback key over its XML stream with the Authoritative Server for verification. (If the Receiving Server does not yet have an XML stream to the Authoritative Server, it will first need to perform a DNS lookup on the Sender Domain and thus discover the Authoritative Server, open a TCP connection to the discovered IP address and port, and establish an XML stream with the Authoritative Server).
3) The Authoritative Server informs the Receiving Server whether the key is valid or invalid.
4) The Receiving Server informs the Originating Server whether its identity has been verified or not.

SASL is a framework for providing authentication and data security services in connection-oriented protocols via replaceable mechanisms. It provides a structured interface between protocols and mechanisms. The resulting framework allows new protocols to reuse existing mechanisms and allows old protocols to make use of new mechanisms. SASL is used in various application protocols (e.g., XMPP, IMAP, LDAP, SMTP, POP, etc.) and support many mechanisms including:

- **PLAIN**, a simple clear text password mechanism. PLAIN obsoleted the LOGIN mechanism.
- **SKEY**, an S/KEY mechanism.
- **CRAM-MD5**, a simple challenge-response scheme based on HMAC-MD5.
- **DIGEST-MD5**, HTTP Digest compatible challenge-response scheme based upon MD5. DIGEST-MD5 offers a data security layer.
- **GSSAPI**, for Kerberos V5 authentication via the GSS-API. GSSAPI offers a data-security layer.
- **GateKeeper**, a challenge-response mechanism developed by Microsoft for MSN Chat

At the time of writing of the IETF 6120, in March 2011, most server implementations still use the Dialback protocol to provide weak identity verification instead of using SASL to provide strong authentication, especially in cases where SASL negotiation would not result in strong authentication anyway (e.g., because TLS negotiation was not mandated

by the peer server, or because the PKIX certificate presented by the peer server during TLS negotiation is self-signed and has not been previously accepted). The solutions is to offer a significantly stronger level of security through SASL and TLS.

V. Securing XMPP-Based Communication System for Federation-Enabled Clouds

A. A Solution for Secure Message Exchange in Inter-Module Communication

Custom functionality can be built on top of XMPP, and common extensions are managed by the XMPP Software Foundation. Regarding the security, even though the XMPP specification support the SASL and TLS technologies for the authentication and encryption of the communication channels, it presents some limitations due to the decentralized nature of the protocol that demands the accomplishment of specific security mechanisms to the different implementations. On the other hand, the flexible and extensible nature of the protocol allows to integrate basic security mechanisms, improving the level of the security in the communications.

As previously discussed, in order to guarantee data confidentiality, integrity, and non repudiation in the XMPP-based communication system of a federation-enabled Cloud Architecture specific security extensions are required. The *XEP 0027* [18] specification describes the use of Jabber with the Open Pretty Good Privacy (OpenPGP - RFC 4880 - [19]). OpenPGP is an interoperable specification that provides cryptographic privacy and authentication for data communications. As highlighted by the *internet draft*, the XEP 0027 does not represent a standard, although it could be in the future, but it describes a possible solution for authentication and data encryption in end-to-end XMPP communication. XEP 0027 allows the addition of specific XML tags in the XMPP message, each one defined by a specific *namespace*: for example *"jabber:x:signed"* and *"jabber:x:encrypted"*. Such tags, indicate to the system how to process the information contained within them. As suggested in the specification, it is possible to apply the digital sign of a sender to the message, for example calculating a *message digest* and signing it with his/her private key. The specification, also allows to sign the presence message in a chat room. In this case, it is possible to sign the status of the sender. In the following, it is shown an example of XMPP message sent from the *Alice* to *Bob* user.

```
<presence from='alice@example.com'
        to='bob@example2.com'>
  <status>Online</status>
  <x xmlns='jabber:x:signed'>
     iQA/AwUBOjU5dnol3d88qZ77EQI2JAC
     fRngLJ045brNnaCX78ykKNUZaTIoAoP
     HI2uJxPMGR73EBIvEpcv0LRSy+=45f8
  </x>
</presence>
```

The status of *alice* is signed with her private key, so that *bob* can verify by means of the *Alice*'s public key that she is really online. In the same way, it is possible to encrypt the content of the tag body using the public key of the receiver in order to achieve confidentiality. In the following, it is shown a message sent from *Alice* to *Bob* whose content has been encrypted with the public key of *Bob*.

```
<message to='alice@example1.com'
        from='bob@example2.com'>
  <body>This message is encrypted.</body>
    <x xmlns='jabber:x:encrypted'>
      qANQR1DBwU4DX7jmYZnncmUQB/9KuKBdd
      zQH+tZ1ZywKK0yHKnq57kWq+RFtQdCJWp
      dWpR0uQsuJe7+vh3NWn59/gTc5MDlX8dS
      9p0ovStmNcyLhxVgmqS8ZKhsblVeuIpQ0
      JgavABqibJolc3BKrVtVV1igKiX/N7Pi8
      RtY1K18toaMDhdEfhBRzO/XB0+P
    </x>
  </body>
</message>
```

The specification does not define the exchange of public keys that is demanded to OpenPGP. Even though the chat messaging is something that purely seem regarding the human interaction, the same approach can be applied to Cloud computing systems in which different distributed software components need to interact each others in both real time and in secure way.

In order to secure the inter-module communication, it is needed to integrate PKI, SCEP, CA, and LDAP mechanism. In order to achieve a secure inter-module communication, it is mandatory to have a digital identity for each element. For this reason, during the initialization of each entity (e.g., administration client, CM, or HM module), it is needed to setup the corresponding digital identity. Each entity obtains through the SCEP a private/public key pair from the CA. After that, it creates a KeyStore local object, in which each requesting entity can find, protected by password, its private key and the digital certificate in PKCS# format. After that the certificate is published on the LDAP server acting as "publisher" of the digital certificates associated to the various entities.

When a module has obtained its own digital identity and it can access the LDAP server storing the public keys of the other entities, it is able to establish a secure inter-module communication with other modules. Thus, each module will be able to sign a message with its private key and to encrypt target message contents. In the first case, the receiver module will verify the digital sign of the sender by means of the corresponding public key read from the LDAP server. In the second case, a module will be able to use the PKI infrastructure in order to negotiate a shared key in order to encrypt the date according to a symmetric cryptography scheme (we remark that the symmetric encryption is more performing than an asymmetric one from a computational point of view). Figure 6 and Figure 7 show the activity diagrams of the inter-module communication respectively

in plain text and with authentication/encryption. The basic difference from the two activity diagrams consists in an encoding task before the message sending and in a decoding task after the reception of the message.

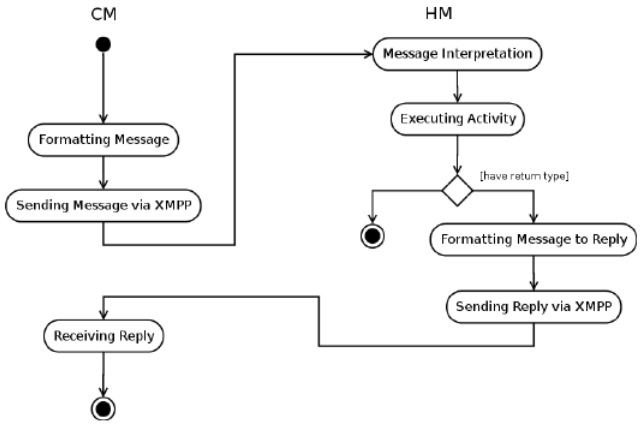


Figure 6. Inter-module communication without security.

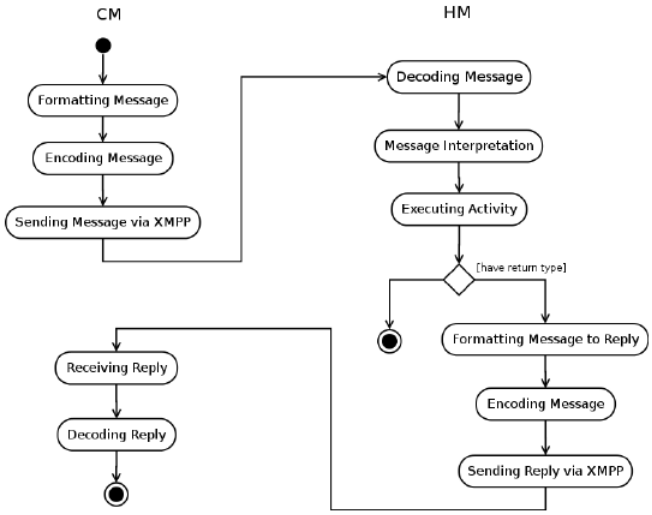


Figure 7. Inter-module communication with security.

In order to guarantee data confidentiality, integrity, and non repudiation in the XMPP-based communication system of a federation-enabled Cloud Architecture, four basic extensions are required in the XMPP messages:

- **Signed**. It allows to attach to the message body a digest signed with the private key of the sender component. The signed extension is identified by the XML name space jabber:x:signed (*<x xmlns='jabber:x:signed'>*) known by all the components. When the message arrives to the receiver component, it detects the signed extension and it queries the LDAP publisher server if an X509 certificate exists for the sender. If it exists, the receiver validates the sign and verifies the message digest according to a shared algorithm.
- **Encrypted**. It allows to attach to the message body a content encrypted with the public key of the receiver component. When a component wants to send an encrypted message, it requests to the LDAP publisher server the X509 certificate of the receiver component. Thus using the public key of the receiver, the sender encrypts the message and it includes the encryption extension identified by the "jabber:x:encrypted" name space (*<x xmlns='jabber:x:encrypted'>*). When the message arrives to destination, the receiver component decrypts it with its private key. This process is schematized in Figure 5.
- **Session Key**. It allows to attach the message a session key. It is used to support an hybrid encryption scheme: the unique share key or the session key is used to encrypt/decrypt the messages by both sender and receiver according to a symmetric encryption scheme (already used in the SSL/TLS protocol), but the session key is exchanged between the two parties according to a public key or asymmetric schema. The advantages of such a hybrid cryptographic scheme is twofold: session key secrecy and faster processing during the encryption of the message.
- **Timestamp**. It allows to attach to the message a signed timestamp, in order to make the message oriented Cloud system normative compliant.

B. A Solution for SSO Authentication in Inter-Domain Federation

In a scalable scenario of federation, each Cloud can require to frequently establish/break partnerships with other Clouds. This implies that each Cloud should manage a huge number of credentials in order to authenticate itself in other Clouds. In a federated environment, this means that the XMPP server of the Cloud requiring federation has to be authenticated by the XMPP server of the Cloud accepting the federation request. If we consider thousand of Clouds, each Cloud should manage one credential for accessing to each federated Cloud. This is problem is commonly known as Single-Sign-One (SSO), i.e., considering an inter-domain environment, performing the authentication once, gaining the access to the resources supplied by different Service Provider, each one belonging to a specific domain. A model addressing the SSO problem is the Identity Provider/Service Provider Model (IdP/SP). Typically, a client who wants to access to the resources provided by a SP performs the authentication once on the IdP (asserting party), which asserts to the SP (relaying party) the validity of the authentication of the client. Considering many SPs relaying on the IdP if the client wants to access another SP, as this latter will be trusted with the IdP, no further authentication will be required. This

model is widely known on the Web with the term "Web Browser SSO", in which the client is commonly an user who perform an authentication fill in an HTML form with his user name and password. Nowadays, the major standard implementing defining the IdP/SP model is the Security Assertion Markup Language (SAML) [20], developed by OASIS.

The scenario of federation is quite similar. In this case, the client who wants to perform the authentication is the XMPP server of the Cloud requiring federation, instead the role of the SP is played by the XMPP server of the Cloud accepting the federation request. As the XMPP server support authentication through SASL a concern raises: the RFC 4422 does not support any security mechanism implementing the IdP/SP model. Therefore, in order to achieve such a scenario, we followed the Internet-Draft entitled "A SASL Mechanism for SAML", defined by CISCO TF-Mobility Vienna, describing the applicability and integration between the two protocols for non-HTTP use cases. According to such a draft, the authentication should occur as follows:

1) The server MAY advertise the SAML20 capability.
2) The client initiates a SASL authentication with SAML20
3) The server sends the client one of two responses:
   a) a redirect to an IdP discovery service; or
   b) a redirect to the IdP with a complete authentication request.
4) In either case, the client MUST send an empty response.
5) The SASL client hands the redirect to either a browser or an appropriate handler (either external or internal to the client),and the SAML authentication proceeds externally and opaquely from the SASL process.
6) The SASL Server indicates success or failure, along with an optional list of attributes

In this way, thanks to SASL and SAML, for each Cloud it is possible to perform the authentication once gaining the access to all the other Clouds relying on the IdP, thence, lending and/or borrowing HMs according to agreements.

In a Cloud Federated scenario, in which the communication system of each involved Cloud is based on the XMPP, the most convenient and easy way for the establishment of a federation is the employment of the federation features made available by the XMPP protocol itself. This latter assumes a XMPP server can be configured for accepting external connections from other servers for creating server-to-server interactions (server federation).

According to the XMPP specifications, this mechanism is quite easy to implement and the result will be the ability for two XMPP servers in different domains to exchange XML stanzas. There are different levels of federation:

- Permissive Federation, a server accepts a connection from any other peer on the network, even without verifying the identity of the peer based on DNS lookups.
- Verified Federation, a server accepts a connection from a peer only after the identity of the peer has been weakly verified via Server Dialback, based on information obtained via the Domain Name System (DNS) and verification keys exchanged in-band over XMPP.
- Encrypted Federation, a server accepts a connection from a peer only if the peer supports Transport Layer Security (TLS) and the client authenticates itself using a SASL mechanisms.

On one hand, Permissive and Verified Federation are the simplest federation approaches: as discussed in the previous section, they lack some security aspects since they are not based on any password exchange procedure and, in order to implement domain filtering (in the second case), a list of allowed sites has to be compiled preemptively. On the other hand, the Encrypted Federation level relies on a more secure way to perform the authentication, based on challenge-response authentication protocols relaying on passphrase.

This standard authentication mechanisms are enough when you want to enable the communication among a limited endpoint number but, in a scenario where several XMPP servers might exist, it could be a difficult task to statically pre-configure the binding among all the involved entities and manage credentials for authenticating a given server to each other. Our idea aims to address these issues and propose the integration of a new SASL security mechanism for allowing a more scalable management of the authentication process exploiting the well-known concept of SSO. The integration we are talking about refers to the use of SAML 2.0.

In order to discuss how to achieve the above mentioned scenario, we two hypothetical Cloud providers each one relying on its own Ejabberd XMPP server [21] to allow communication within its domain. Generally, the server-to-server federation is accomplished by an Ejabberd module that manages incoming and out-coming connections from/to external servers. According to the XMPP core specification, this module is able to establish server federation according to the three different levels pointed out above. In order to enable Ejabberd servers to perform SSO authentication it is required to consider the Encrypted Federation case and extending the Ejabberd module performing SASL to add in the list of the supported security mechanism also SAML 2.0.

A possible way for the achievement of this environment is the implementation of the Internet-Draft entitled "A SASL Mechanism for SAML" defined by CISCO TF-Mobility Vienna relying on an external software module based on Shibboleth, we named Authentication Agent (AA). The AA acts as user when it is contacted from the Source Ejabberd Server for starting the Federation, whereas represents the Relying Party when it is contacted from the Destination Ejabberd Server.

In the following, we present the sequence of steps performed by two servers (for simplicity Source Server and

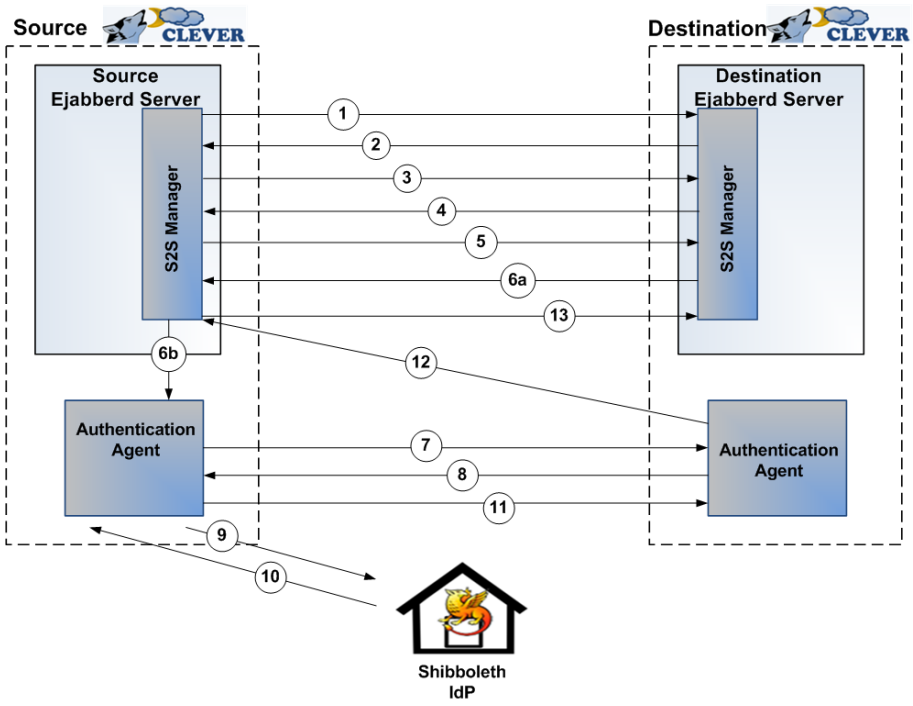


Figure 8. Step performed by two XMPP servers aiming to build Federation: the authentication process is executed using SAML 2.0 as external SASL mechanism.

Destination Server) that aim to build the federation. As Figure 8 depicts, the involved actors in the process are the s2s_Manager(s) of both the Ejabberd servers, the two authentication agents acting as User and Relying Party (User, the one interacting with the Source Server; Relying Party the one interacting with the Destination Server) and the Identity Provider (also implemented using Shibboleth).

- Step 1: s2s_Manager of Source Server initiates stream to the s2s_Manager of the Destination server.
- Step 2: s2s_Manager of the Destination Server responds with a stream tag sent to the s2s_Manager of the Source Server.
- Step 3: s2s_Manager of the Destination Server informs the s2s_Manager of the Source Server of available authentication mechanisms.
- Step 4: s2s_Manager of the Source Server selects SAML as an authentication mechanism.
- Step 5: s2s_Manager of Destination Server sends a BASE64 encoded challenge to the s2s_Manager of the Source Server in the form of an HTTP Redirect to the Destination AA (acting as Relying Party).
- Step 6: a) s2s_Manager of Source Server sends a BASE64 encoded empty response to the challenge and b) forward to the Source AA the URL of the Relying Party.
- Step 7: The Source AA (User) engages the SAML authentication flow (external to SASL) contacting the Destination AA (Relying Party).
- Step 8: Destination AA redirect Source AA to the IdP.
- Step 9: Source AA contacts IdP and performs Authentication
- Step 10: IdP responds with Authentication Assertion
- Step 11: Source AA contacts Destination AA for gaining access to the resource.
- Step 12: Destination AA contacts the s2s_Manager of the Destination Server informing it about the authentication result.
- Step 13: if the authentication is successful the s2s_Manager of the Source Server initiates a new stream to the s2s_Manager of Destination Server.

The advantage of performing the authentication among servers in such a way mainly consists in the higher security level achieved than the traditional Dialback/SASL mechanisms and in the possibility of exploiting the SSO authentication. Looking at Figure 8, after that the federation has been achieved with the depicted server, if the same Source Cloud aims to perform server-to-server federation with a new XMPP server that relies on the same IdP as trusted third-party, such a process would be straightforward. Since the Source Server already has an established security context with the IdP, once the SASL process starts and the SAML mechanism is selected, no further authentication will be required.

VI. Conclusion

Currently, the major Cloud solutions base their communication systems on HTTP-based web services that do not well suit the requirements of the new emerging Cloud architectures and services. The XMPP presents several advantages for designing the communication system of a federation-enabled Cloud architecture. On one hand, the XMPP well suits the requirement of interactivity, but on the other hand it lacks of native security features. Both the inter-module communication and the inter-domain federation require appropriate security mechanisms. In this paper, we discussed two possible solutions to achieve such goals. Regarding the inter-module communication, we discussed how to extend the XMPP for enabling secure message exchange according to the XEP-0027 specification, whereas, considering the inter-domain federation, we proposed an approach based on SSO authentication for server to server federation according to the Internet-Draft entitled "A SASL Mechanism for SAML", defined by CISCO TF-Mobility Vienna. We hope, we contributed to alleviate the gap in security for the adoption of the XMPP in federation-enabled Cloud architectures.

References

[1] A. Celesti, F. Tusa, M. Villari, and A. Puliafito, "Federation between clever clouds through sasl/shibboleth authentication," in *INTERNET 2012, The Fourth International Conference on Evolving Internet*, pp. 77–84, 2012.

[2] A. Celesti, M. Fazio, and M. Villari, "Se clever: A secure message oriented middleware for cloud federation," in *IEEE Symposium on Computers and Communications (ISCC)*, July 2013.

[3] National Institute of Science and Technology. Standards Acceleration to Jumpstart Adoption of Cloud Computing; http://csrc.nist.gov/groups/SNS/cloud-computing/ July 2011.

[4] P. Saint-Andre, "Xmpp: lessons learned from ten years of xml messaging," *Communications Magazine, IEEE*, vol. 47, no. 4, pp. 92–96, 2009.

[5] W. Liu, "Research on cloud computing security problem and strategy," in *Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on*, pp. 1216 –1219, April 2012.

[6] A. Behl and K. Behl, "Security paradigms for cloud computing," in *Computational Intelligence, Communication Systems and Networks (CICSyN), 2012 Fourth International Conference on*, pp. 200–205, July 2013.

[7] A. Celesti, N. Peditto, F. Verboso, M. Villari, and A. Puliafito, "Draco paas: A distributed resilient adaptable cloud oriented platform," in *IEEE 27th International Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW)*, pp. 1490–1497, 2013.

[8] M. Kretzschmar, M. Golling, and S. Hanigk, "Security management areas in the inter-cloud," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pp. 762 –763, July 2011.

[9] M. Kretzschmar and S. Hanigk, "Security management interoperability challenges for collaborative clouds," in *Systems and Virtualization Management (SVM), 2010 4th International DMTF Academic Alliance Workshop on*, pp. 43 –49, oct. 2010.

[10] I. Gul, A. ur Rehman, and M. Islam, "Cloud computing security auditing," in *Next Generation Information Technology (ICNIT), 2011 The 2nd International Conference on*, pp. 143–148, June.

[11] M. Almorsy, J. Grundy, and A. Ibrahim, "Collaboration-based cloud computing security management framework," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pp. 364 –371, July 2011.

[12] Z. Lian-chi and X. Chun-di, "Cloud security service providing schemes based on mobile internet framework," in *Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on*, vol. 3, pp. 307 –311, March 2012.

[13] A. Behl, "Emerging security challenges in cloud computing: An insight to cloud security challenges and their mitigation," in *Information and Communication Technologies (WICT), 2011 World Congress on*, pp. 217–222, December 2011.

[14] A. Celesti, F. Tusa, M. Villari, and A. Puliafito., "Integration of CLEVER Clouds with Third Party Software Systems Through a REST Web Service Interface," in *17th IEEE Symposium on Computers and Communication (ISCC'12)*, 1-4 July 2012.

[15] Cloud Security Alliance, Security Guidance for Critical Areas of Focus in Cloud Computing v3.0, https://cloudsecurityalliance.org/guidance/csaguide.v3.0.pdf, 2011.

[16] RFC 4422, Simple Authentication and Security Layer (SASL), http://www.ietf.org/rfc/rfc4422, Jun 2013.

[17] XEP-0220: Server Dialback, http://xmpp.org/extensions/xep-0220.html, Jun 2013.

[18] XEP-0027: Current Jabber OpenPGP Usage, http://xmpp.org/extensions/xep-0027.html, 2006.

[19] OpenPGP Message Format, http://www.rfc-editor.org/info/rfc4880, 2007.

[20] SAML V2.0 Technical Overview, OASIS, http://www.oasis-open.org/specs/index.php#saml, Jun 2013.

[21] Ejabberd, the Erlang Jabber/XMPP daemon: http://www.ejabberd.im/, Jun 2013.

Security Considerations for Multicast Communication in Power Systems

Rainer Falk and Steffen Fries
Corporate Technology
Siemens AG
Munich, Germany
e-mail: {rainer.falk|steffen.fries}@siemens.com

***Abstract*—Information security is gaining increasingly more importance for real-time automation networks. Multicast communication is used widely especially on field and process level to cope with performance requirements and to ease the handling of communication peers as the destinations need not to be known by the sender. A security design must not interfere with these communication types. This paper investigates into different approaches to achieve multicast security focusing on energy automation networks. Here, domain-specific protocols like GOOSE are used within substations to distribute measurement and status information between IEDs using plain Ethernet superseding classical copper wire connections. Hence, they have to cope with high performance requirements in terms of very low latency and transfer time. For these reasons, a solution is required allowing to perform efficient authentication of field-level multicast communication. Moreover, this multicast authentication may also be applicable in WAN communication, as the substation protocol GOOSE is meanwhile also being applied to exchange synchrophasor data.**

Keywords–security; device authentication; multicast; real-time; network access authentication; firewall; substation automation; wide area condition monitoring

I. INTRODUCTION

Decentralized energy generation, e.g., through renewable energy sources like solar cells or wind power, is becoming increasingly important to generate environmentally sustainable energy and thus to reduce greenhouse gases leading to global warming. Introducing decentralized energy generators into the current energy distribution network poses great challenges for energy automation (EA) in a smart grid scenario as decentralized energy generation needs to be monitored and controlled to a similar level as centralized energy generation in power plants while requiring widely distributed communication networks. Distributed energy generators may also be aggregated on a higher level to form a virtual power plant. Such a virtual power plant may be viewed from the outside in a similar way as a common power plant with respect to energy generation. But due to its decentralized nature, the demands on communication necessary to control the virtual power plant are much more challenging. Moreover, these decentralized energy resources may also be used in an autonomous island mode, without any connection to a backend system.

Furthermore, the introduction of controllable loads on residential level requires enhancements to the energy automation communication infrastructure as used today. Clearly, secure communication between a control station and equipment of users (e.g., decentralized energy generators) as well as with decentralized field equipment must be addressed. Standard communication technologies as Ethernet and IP are increasingly used in energy automation environments down to the field level. Guaranteed real-time communication plays an essential role for many industrial control applications (see also [1], [2]).

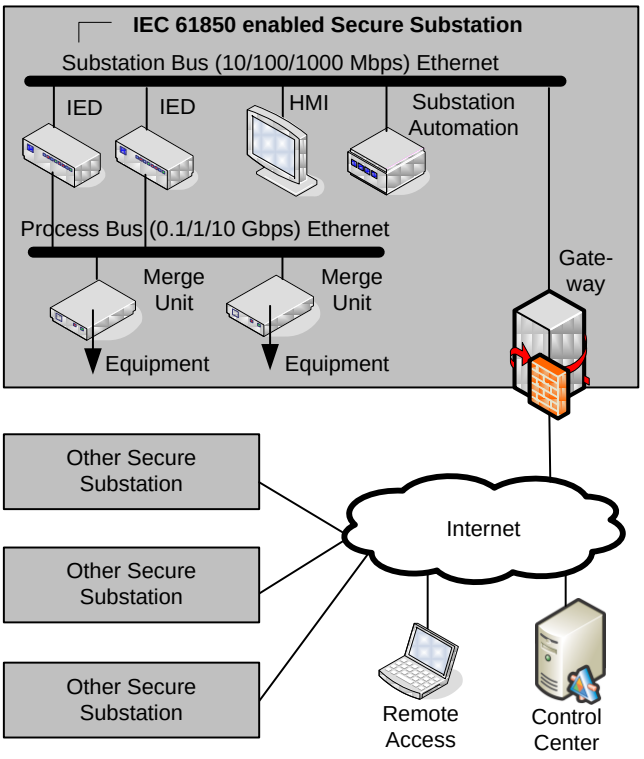


Figure 1. Typical IEC 61850 Scenario

IEC 61850 is a popular standard for communication in the domain of energy automation. It is envisaged to be the successor of the currently used standards IEC 60870-4-104 and DNP3 also for the North American region. IEC 61850 enables interoperability between devices used in energy automation, i.e., two IEC 61850 enabled devices of different manufacturers can exchange a set of clearly defined data and the devices can interpret and use these data to achieve the functionality required by the application due to a standardized data model. In particular, IEC 61850 enables continuous communication from a control station to

decentralized energy generators by using a standardized data format.

Today, IEC 61850 is mainly used for reporting status and sampled value information from Intelligent Electronic Devices (IED) to Substation automation controller as well as for command transport from Substation automation controller to IEDs. It also addresses the communication directly between IEDs using the Generic Object Oriented Substation Event (GOOSE) instead of dedicated wires. Necessary tasks comprise also configuration of equipment as well as control of circuit breakers. Figure 1 shows a typical example scenario in which IEC 61850 can provide a clear benefit (see also [3], [4]).

IT security is increasingly important in energy automation as on part of the Smart Grid. Here, IEC 62351 kicks in, defining security services for IEC 61850 based communication covering different deployment scenarios using serial communication, IP-based communication, and also Ethernet communication. The latter one is used locally with a substation to cope with the high real-time requirements. While these messages may not need to be encrypted to protect confidentiality, they need to be protected against manipulation and to allow for source authentication. Note that besides pure communication security, there is also the need to address security in the physical environment and also in the organizational processes. This is typically addressed in the context of IEC 27001 [5] describing the Information Security Management Standard (ISMS). While this standard targets general applicability, there exist domain specific mappings of the related ISO 27002 [6] best practice guidelines which are applicable for the automation domain. Relevant are in particular ISO TR 27019 [7] for energy automation and IEC 62443-2-1 (ISA 99.2.1) [8] for industrial automation. Both standards are mentioned here to underline that security is not restricted to the field communication, but applies to the embedding environment as well. However, this paper concentrates on the specific problem of multicast authentication on field level.

The remainder of this paper is structured as follows: Section II provides an overview on real-time control networks with the example of the GOOSE substation automation protocol. Section III maps GOOSE to wide area monitoring. Section IV describes the problem statement and the existing security solution as defined in the standard. Section V gives an overview about multicast authentication schemes in general. This is used later on in Section VI and Section VII by applying them to substation automation protocols. Section VIII concludes the paper and provides an outlook.

II. Substation Automation Communication

Real-time systems typically consist of hardware and software that are subject to time constraints regarding execution of commands. This comprises the initiation of a command, the execution itself and the acknowledgement of the execution. Real-time in the context of this paper refers to systems with a deterministic behavior, resulting in a predictable maximum response time. These systems will handle all events at appropriate (context-dependent) speed, without loss of events.

Automation networks are typically shared networks connected in a ring, star, or bus topology or a mixture of these. Most often, the time critical part is realized on a dedicated network segment, while the rest of the communication supporting the automation systems is performed on networks with lower performance requirements.

An example for energy automation is the communication within a substation. A substation typically transforms voltage levels, and includes power monitoring and protection functions. In the example shown in Figure 2, the communication of the protection devices is separated from the historian data (stored in the historian device, see Figure 2 in below) in a separate network zone of the substation. The historian data may even be sent to a SCADA (supervisory control and data acquisition) or office network. The historian is a device for archiving measurements, events, and alarms of the substation.

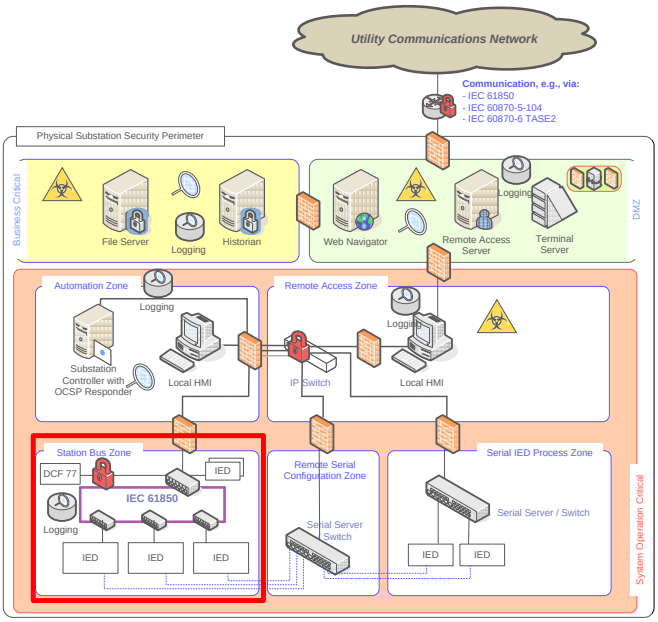


Figure 2. Substation – Functional Split into Zones

As depicted in Figure 2, the substation bus can be realized as ring, connecting the protection relays, acting in real-time. There is a connection to other zones within the substation, separated from the real-time part using Firewalls. Examples are the automation zone or the remote access zone. Another example is the zone storing the historian information also interacting with a backend SCADA system. Figure 2 already shows security elements deployed within a substation, like Firewalls, virus checking tools, or access control means to components or data.

Figure 3 shows a ring topology used to connect field devices in the process bus zone. Besides the field devices, which may be protection devices exchanging information about the current state of the measured values with respect to voltage or current, also controllers are likely to be available. These controllers provide the connectivity to other bays in a

substation or to a control center, relying on the operation of the protection devices but also on the measurement data to counter certain electric effects.

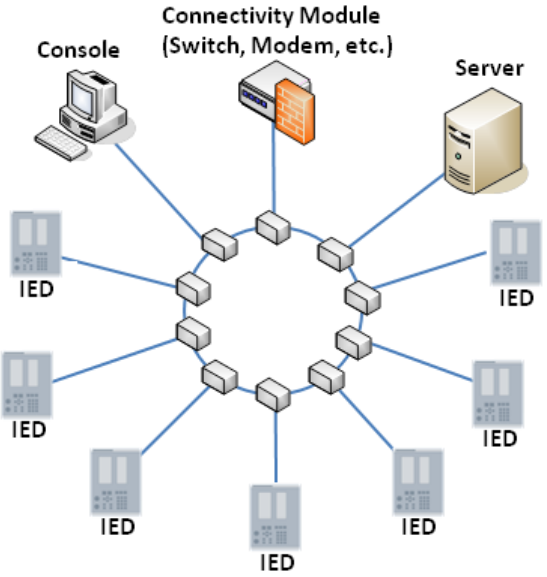


Figure 3. Ring topology in a substation

One of the protocol sets used in substation automation is IEC 61850, which provides Generic Object Oriented Substation Events (GOOSE) on process bus level. It is a control model mechanism in which any format of data (status, value) is grouped into a data set and transmitted as set of substation events, such as commands, alarms, or indications. It aims to replace the conventional hardwired logic necessary for intra-IED (Intelligent Electronic Device) coordination with station bus communications. Upon detecting an event, field devices use a multi-cast transmission to notify those devices that have registered (subscribed) to receive the data (see also Figure 4). GOOSE messages or Sampled Values (SV) are re-transmitted multiple times by each field device. The reaction of each receiver depends on its configuration and functionality. Note that the registration to events is purely device local at the receiver side. This results in the fact that the sender does not know the receiver of its GOOSE message sent.

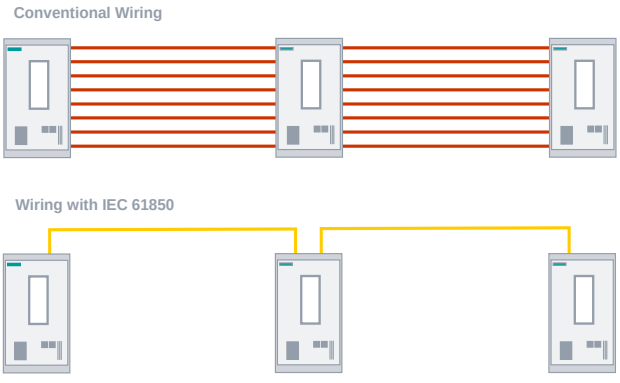


Figure 4. Advantage of using IEC 61850 GOOSE

Following mechanisms are used to ensure the required transmission speed and reliability:

- GOOSE data is directly embedded into Ethernet data packets and works on publisher-subscriber mechanism on multicast or broadcast MAC addresses.
- GOOSE uses VLAN and priority tagging as per IEEE 802.1Q to have a separate virtual network within the same physical network and to set an appropriate message priority level.
- Enhanced retransmission mechanisms – the same GOOSE message is retransmitted with varying and increasing re-transmission intervals. A new event occurring within any GOOSE dataset element will result in the existing GOOSE retransmission message being stopped. A state number within the GOOSE protocol identifies whether a GOOSE message is a new message or a retransmitted message.

IEC 61850-5 [3] defines message types and their performance classes. The following performance classes are supported:

- P1 typically applies to a distribution bay (or where low requirements can be accepted),
- P2 typically applies to a transmission bay (or if not otherwise specified by the customer),
- P3 typically applies to a top performance transmission bay.

Table I below shows the different message types and their timing requirements based on IEC 61850-5 [3].

TABLE I. GOOSE TRANFER TIMES

Type	Definition	Timing Requirements
1	**Fast messages** contain a simple binary code containing data, command or simple message, examples are: "Trip", "Close", etc.	See Type 1a and 1 b below
1A	**TRIP** – most important message	– P1: transfer time shall be in the order of half a cycle. → 10 ms – P2/3: transfer time shall be below the order of a quarter of a cycle. → 3 ms
1B	**OTHER** – Important for the interaction of the automation system with the process but have less demanding requirements than trip.	– P1: transfer time < 100ms – P2/3: transfer time shall be below the order of one cycle. → 20 ms
2	**Medium speed messages** are messages where the time at which the message originated is important but where the transmission time is less critical.	– Transfer time < 100ms
3	**Low speed messages** are used for slow speed auto-control functions, transmission of event records, reading or changing set-point values and general presentation of system data.	– Transfer time < 500ms

The definition of transfer time, according to IEC 61850-5, is shown in Figure 5 below. The transfer time includes the complete transmission of a message including necessary handling at both ends. The time counts from the moment the sender feeds the data content into transmission stack till the moment the receiver extracts the data from its transmission stack. As shown in Table I, the transfer time of GOOSE messaging for a TRIP command shall be such that the command should arrive at the destination IED within 3ms.

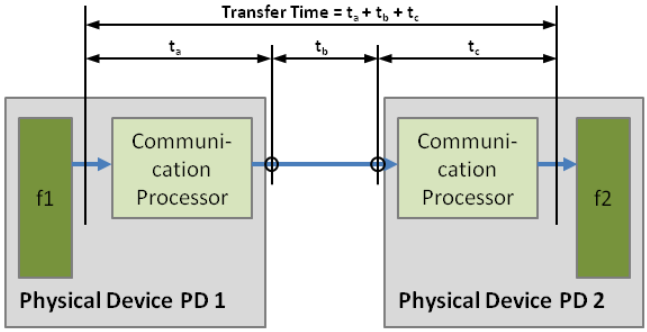


Figure 5. Transfer Time [3]

For a single IED, by assuming the time for the publishing process t_a and the subscribing process t_c are approximately equal and if t_b (network transfer time) can practically be ignored, then at least half of the defined time is needed for the IEDs to process the message (i.e., 1.5ms for a TRIP message). As shown in Figure 6, if a signal as, e.g., the pick-up "Overcurrent *I*>picked up", is configured in a GOOSE message, the IED sends this message cyclically every 0.5 seconds as a telegram with high priority over the Ethernet network. The content of this telegram communicates the state of pick-up ("not picked up" or "picked up") to the subscribers of the GOOSE message. The cyclic transmission enables each of the subscribers to detect a failure using a logic block when a transmitter has failed or a communications channel has been interrupted.

This approach provides constant monitoring of the transmission line because the subscriber expects to receive a telegram at several-second intervals. This can be compared with pilot-wire monitoring in conventional wiring. On a pick-up, i.e., a signal change, a GOOSE telegram is transmitted spontaneously and is repeated after 1 ms, 2 ms, 4 ms etc. before returning to cyclic operation.

Typical examples for GOOSE application in substation automation comprise:

- Tripping of switchgear
- Starting of disturbance recorder ("Störschrieb")
- Providing position status of interlocking

Security requirements and solutions for GOOSE communication have already been specified. They are discussed as part of Section IV.

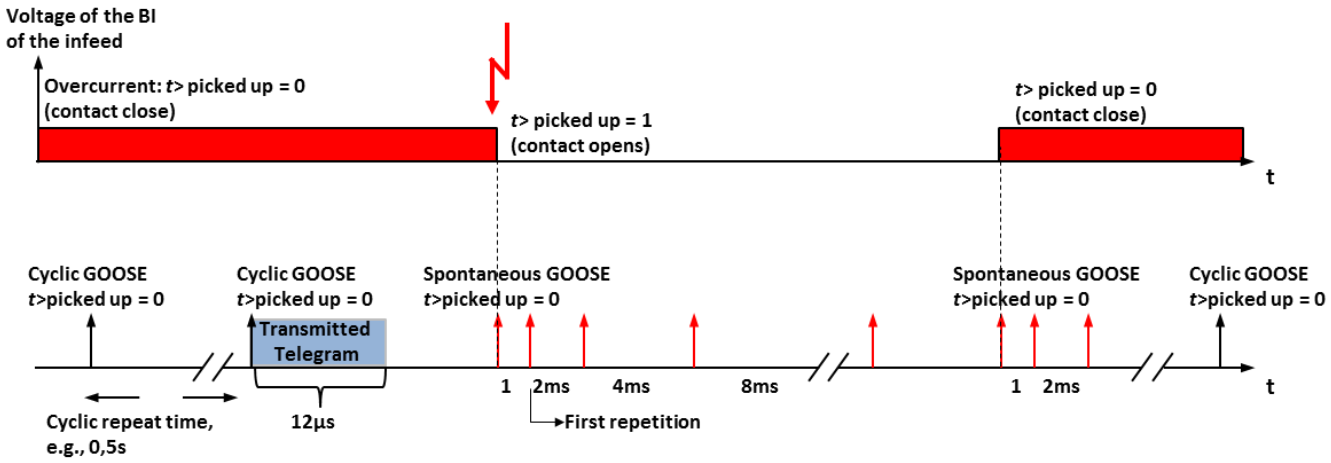


Figure 6. Transmission of binary states with GOOSE messages

III. WIDE AREA STATUS MONITORING

Besides the application of GOOSE and SV communication within a substation, there is also the need to transmit status and power measurements as synchronized status information over wide area networks. One driver of this is the request to be able to detect frequency deviations from widely dispersed areas very early and act accordingly to prevent blackouts. Examples are:

- The blackout in the North America northeastern in 2003, affecting more than 50 million people in the US and Canada [9].
- The blackout in India in 2012 was the biggest blackout so far. The power outage affected more than 620 million people [10].

Further information about major blackouts can be found in [11]. It is clear that not all of these blackouts can be prevented, but supporting wide area measurement and protection and control (WAMPAC) may certainly be used to identify the risk of a blackout. This information in turn can then be used to apply proper counter measures in time and to reduce of spreading of the blackout.

This is addressed in the technical report IEC 61850-90-5 [12] describing the use of GOOSE and SV over wide area networks. Note that Ethernet will not be the base for communication in these scenarios but UDP/IP, which also allows for multicast, e.g., of synchrophasor measurement unit data.

The security approach described for wide area usage of GOOSE and SV will also be fed into the further enhancement of IEC 62351. Specifically, the Internet group key management protocol GDOI [13] will be the bases for the key management standard IEC 62351-9, while the application of the group key will be described in an edition 2 of IEC 62351-6. Both documents are currently work in progress.

IV. SECURITY FOR SUBSTATION AUTOMATION MULTICAST MESSAGES

Security is a basic requirement for protecting substation automation communication. The main security requirements especially for GOOSE and SV communication have been determined as message integrity and source authentication.

Within the standard IEC 62351-6, a security solution is provided that exactly addresses these requirements for the transfer of GOOSE and SV messages in multicast Ethernet networks. The basic approach builds on digital signatures. They are used to calculate a cryptographic checksum over the payload of the Ethernet PDU (Protocol Data Unit). The transport of the security related part is defined as an extension to the existing definition of the GOOSE or SV PDU. Digital signature calculation requires a high computational load to the IED, especially if retransmissions are taken into account. Retransmissions require a new signing operation to avoid potential replay attacks by simply repeatedly sending signed packets. Moreover, at a sample rate of 80 samples per power cycle, up to 4000 packets per second have to be signed for the common power frequency of 50 Hz. If each of those messages is protected by a digital signature, a high computational burden is placed on the sender by the generation of the digital signatures, and also on the receiver for verifying the signature. IEDs are typically not built to handle this type of operation at that speed. This has been verified by prototypes running on FPGAs [14]. Therefore, there exists a demand for an alternative solution to address the security requirements for protected communication more efficiently [15].

As stated in the previous section, there exists also a demand to transmit Phasor Measurement information in distributed environments over wide area networks. A new requirement arising here is the confidentiality of the data. This requirement stems from the fact that the synchrophasor information may be misused by an eavesdropper to determine the current load and stability of a dedicated electricity network. While this information is protected in a substation by physical means, it needs to be protected when communication over wide area networks based on sound cryptographic methods. Note that the discussion of confidentiality is not part of this paper.

To better cope with the required performance, IEC 61850-90-5 proposes to rely on integrity check values (ICV), which are calculated using HMAC-SHA256 or AES-GMAC involving a shared key, rather than using digital signatures. This shared key is supposed to be a group based key, shared among the configured participants of a group. A key distribution center is responsible for authenticating the group participants and generating and distributing the shared group key to authenticated peers.

The underlying key distribution protocols is Group Domain of Interpretation GDOI, [13]. It has already proven its practical feasibility in many IP router implementations to distribute group keys for multicast services in the Internet. The integrity check is applied in the processing in a similar way as the digital signature. The sender creates the ICV, while the receiver checks the ICV upon receiving the message, before executing a command.

The following subsections discuss multicast authentication options in general and propose the application of authentication schemes for dedicated messages that allow for the delayed verification of message integrity of already received messages.

V. EXISTING APPROACHES FOR MULTICAST AUTHENTICATION

Many widely used security protocols as IPSec [4] and SSL/TLS [17] are designed mainly for point-to-point communication. However, the communication type of multicast requires specific handling. The objective of security within substation automation is to ensure the integrity and authenticity of messages. Protecting the confidentiality is not required, however.

Figure 7 shows the basic set-up. A sender sends a message containing data protected with a message authentication code MAC. Several receivers verify the received message. Cryptographic authentication of multicast communication comprises to main parts:

- Message protection: A data packet or frame has to be protected (encryption and/or message authentication). A cryptographic checksum (message authentication code) is applied to a message that is verified by the receivers.
- Multicast Key management: The cryptographic keys required by the sender and by the receiver have to be established.

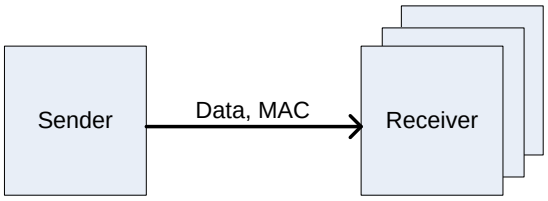


Figure 7. Broadcast/Multicast Sender Authentication

Conceptually, the problem would be solved by applying a digital signature scheme, e.g., PKCS#7 [18], based on public key cryptography, e.g., RSA [19], DSA [19], or ECC-based signatures [19]. However, the computational requirements of these algorithms render them inadequate for the targeted field level devices as already discussed in Section IV above. So a message level protection based on symmetric algorithms as AES-CBC-MAC, AES-GMAC, or HMAC-SHA256 [19] is used. The sender and the receiving nodes apply the same secret key for creating and for verifying the cryptographic checksum.

The following subsections discuss potential approaches for message protection as well as options for key management.

A. Delayed Authentication of Multicast Messages

The Timed Efficient Stream Loss-tolerant Authentication protocol (TESLA) [20] provides sender authentication. TESLA is based on loose time synchronization between the sender and the receivers. Source authentication is realized in TESLA by using Message Authentication Code (MAC) [19] using a symmetric key of a one-way key chain.

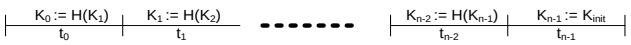


Figure 8. Hash Key Chain

Figure 8 illustrates the concept of a hash key chain. The hash key chain of length n is determined by the sender starting with a randomly chosen key K_{init} that is valid during a time period tn-1. The sender computes the keys K_i using a cryptographic hash function H as the hash of the key K_{i+1}, i.e., $K_i := H(K_{i+1})$. The key Ki is valid for sending messages only during the time period t_i. But the sender releases the key K_i only after the time period t_i has already passed, i.e., when the key is not valid for sending anymore. A receiver can verify messages received during the time period t_i only after t_i has passed, i.e., after having received the key. However, a malicious receiver cannot forge messages on behalf of the sender as the key is already invalid.

The sender provides the first key K_0 to receivers in a secure way (i.e., protected by a digital signature or provided over a protected communication channel). Each receiving node stores the key K_0. Further keys K_{i+1} are released by the sender in clear as a receiver can verify the authenticity of the released key efficiently by computing its hash value. Due to the one-way property of the hash function H, a receiver cannot practically determine a key K_{i+1} from a known K_i.

The important property of the one-way key chain is that once the receiver has obtained a single authenticated key of the chain, subsequent keys of the chain are self-authenticating. This means that the receiver can easily and efficiently authenticate subsequent keys of the one-way key chain using the one authenticated key. The initially distributed message is protected using a well-known digital signature.

µTESLA addresses sensor network scenarios and optimizes the TESLA protocol for this use case [15]. The general setup assumes a base station, which has an authenticated connection to sensor nodes based on a shared secret. As the digital signature for the initial message protection in TESLA is too costly for sensor nodes, µTESLA addresses this by using the node-to-base-station authenticated channel to bootstrap the authenticated broadcast. The remainder of the protocol is similar to the original TESLA approach.

B. Group based Key Management

Various protocols have been designed for group key management, e.g., the Group Key Management Protocol (GKMP) [21] and Scalable Multicast Key Distribution [22]. Group Secure Association Key Management Protocol (GSAKMP) [23]. A survey [24] of group key management protocols describes different options for group key management in centralized environments. Also common wireless communication standards support secure multicast/broadcast communication, e.g., IEEE 802.11 WLAN [25] and 3GPP Multimedia Broadcast/Multicast Service [26].

The basic design idea is to rely on a group key management server that authenticates group members and establishes group keys for protecting communication within the group. There exist also decentralized approaches for group key establishment that do not require a group key server, e.g., Group Diffie–Hellman Key Exchange [27].

All these approaches result in a symmetric group key shared between the members of the group. So each node can send and verify protected group messages. No authentication of the sending node is achieved, as each group member knows the group key that can be used for both sending and receiving messages. In contrast to group based key management in volatile environments like video conferences or similar, a join and leave policy is likely not be needed in energy environments. This join and leave policy typically ensures that whenever a group member joins or leaves a group a fresh key is distributed. This is being done to avoid that even a regular group member can eavesdrop the communication of his associated group when he is not participating in a group session. This requirement is not obvious in energy automation as the networks are rather static and engineered at a certain point in time, according to a fixed required functionality.

A specific key management based on key chains can be used to achieve sender authentication with symmetric cryptography. An element of the key chain is valid for sending only during a limited, defined time period. During that time period, it is known only by the sender. Only after the time validity has passed, the key is revealed to receiving nodes. To verify a received message, a receiving node has to store the received message until it has received the corresponding key. Only after receiving also the key, the receiver can verify the received messages. This leads to a delay in processing of the messages. The approach in general has been described in subsection V.A by using TESLA as one example. The following subsection elaborate more on selected group key management protocols frameworks.

1) Group Domain of Interpretation (GDOI)

GDOI is the result of the IETF multicast security working group and is defined in RFC 6407 [13]. It defines an architecture were a group controller manages the key material and the connected policies for a defined group. The group members typically authenticate towards the group controller before they are allowed to participate in the group. GDOI allows for pull and push distribution of the group key material and also allows for the update of this information. The difference between the two modes push and pull is mainly who initiates the key distribution, the group controller or the client. For application within IEC 61850-90-5 the focus is placed on the pull mechanism. Also, a key update is performed by simply reauthenticating towards the group controller.

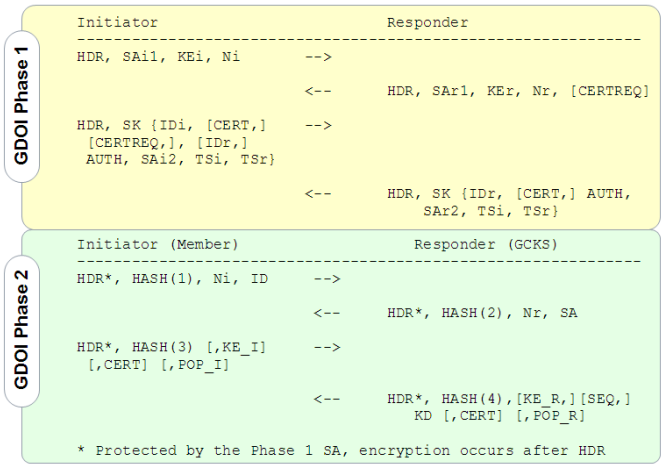


Figure 9. GDOI Call Flow

Figure 9 shows the messages and their content for the general call flow of GDOI. While the first phase basically resembles an Internet Security Association and Key Management Protocol (ISAKMP) phase 1 key exchange to authenticate both peers and establish security associations (SA), the second phase is used to realize the GDOI PULL registration or PUSH rekey exchange. Within its application in IEC 61850 environments, always the PULL method is used. Even in the case of rekeying, the client connects to the group key server and authenticates and then receives the new group key. This approach has been chosen to ensure that clients in the network authenticate towards the group key server. The "join-and-leave" behavior, e.g., in multimedia communication is not pertained as the configuration of groups in the energy environment is rather static. Joining and leaving of members of a group leads to updates of the group key otherwise, to ensure that a new participant gets now information about previous exchanges and leaving participants cannot eavesdrop the ongoing discussion.

Note that GDOI has been successfully implemented to negotiate key material for protecting router communication using IPSec.

2) Multimedia Internet Keying (MIKEY)

MIKEY has been defined in the IETF within RFC3830 [28]. It defines an authentication and key management framework that can be used for real-time applications (both for peer-to-peer communication and group communication). In particular, RFC3830 is defined in a way to support SRTP in the first place but is open to enhancements to be used for other purposes too. MIKEY has been designed to meet the requirements of initiation of secure multimedia sessions. Such requirements are for instance the establishment of the security parameters for the multimedia protocol within one round trip.

Another requirement is the provision of end-to-end keying material, and also independence from any specific security functionality of the underlying transport layers.

MIKEY defines several options for the user authentication and negotiation of the master keys all as maximum as 2 way-handshakes as there are:

- Symmetric key distribution (pre-shared keys, Message Authentication Codes (MAC) for integrity protection; may proceed in a one-way handshake)
- Asymmetric key distribution (based on asymmetric encryption; may proceed in a one-way handshake)
- Diffie Hellman key agreement protected by digital signatures (two-way handshake).

Unprotected key distribution, i.e., without authentication, integrity, or encryption, is also possible, but not recommended without any underlying security like TLS or similar. This use case is comparable with the security description approach described below (see the following section).

VI. Enhancements for Substation Automation Multicast Security

In this paper, we propose a new solution for the authentication and integrity protection of broadcast/multicast control messages. It combines hash key chains with digital signatures. This solution can be applied in particular to a field-level energy control protocol (e.g., a substation controller).

To avoid a centralized node as single point of failure each sending node manages its own key chain. As in TESLA, the initialization information of a hash key chain is protected by the sender using a digital signature.

Synchronized time is already available in energy automation using Network Time Protocol (NTP) [29] per substation. A GPS receiver is attached to the substation controller to provide the reference time for all connected components. If a GPS device is not available, the time information may also be received from a hierarchically higher system component like a control center over other signaling channels. Here, NTP may be used to synchronize to a time source in the associated control center.

Known enhancements to the basic TESLA scheme support immediate authentication by using buffering by the sender [30]. However, this requires that the sending node has to already have the information about the contents of future packets. This makes it unsuited for real-time control applications where the future changes in the physical world are not known in advance. Furthermore, the usage of multiple key chains has been proposed where a sending node manages multiple hash chains for receivers observing different network delays.

The following subsections describe new enhancements to TESLA to cope with the specific requirements of a real-time control network.

A. Multiple Message-class specific Hash Chains

A sending node manages multiple hash key chains. A hash key chain message is bound to a certain class of control messages. The class of control messages is specified by the sender as part of the hash chain's initialization information. This allows a receiver to determine whether an announced hash chain includes potentially control commands relevant for the receiver. Only if this is the case, the receiver has to store the initialization information. A receiver may also verify that a received control message is in fact of the class as announced in the hash chain initialization information.

B. Hierarchical Hash Key Chains

In TESLA, each hash key chain initialization information is protected by a separate digital signature. It is proposed to establish a first hash key chain that is used to protect initialization information of further hash key chains. This is in particular advantageous if several hash key chains are established for different message classes. Also, hash chains which have to be established frequently as they may have a short time delta between hash chain values can be established efficiently.

C. Early control command execution

When using a hash chain, a receiver can verify the cryptographic checksum of a received control message only after a certain delay (when the next element of the hash chain is disclosed by the sender). This leads to a non-negligible delay. It is therefore proposed that for some classes of commands the receiver performs the control action immediately after receiving the message, i.e., before verifying the command's cryptographic checksum. However, roll-back information is stored by the receiver. Should the checksum be invalid (once it is verified later), an inverse control operation is performed, neutralizing the effect of the invalid control command. If the checksum is valid, the roll-back information is deleted to free occupied memory. In an enhancement, this early command execution is performed only for certain control commands, e.g., for which parameter values have passed a plausibility check. The distinguished message handling, based on the type of the control command, allows a receiver to be also more resistant to denial of service attacks, as only dedicated commands are checked immediately. It is also obvious, that for better denial of service protection, additional means are to be provided in the network, to shift load from the IEDs. These means may comprise IDS (Intrusion Detection Systems) or IPS (Intrusion Prevention Systems). The interworking with these systems is outside the scope of this paper.

D. Comparison

The properties of the proposed enhancements are evaluated regarding their impact on the field devices. Performance requirements on field level devices are reduced even further as a device processing only data with low rate or with low real time requirements has to process only messages of a corresponding hash key chain. The number of digital signature verifications is kept low as the hash key chain initialization information of the multiple key chains is protected by a hash key chain itself. The design fits with the existing solutions, supporting publish/subscribe communication, and avoiding any central controller. It is one option that can be used in combination with currently defined options.

However, still support for digital signatures is required. This may be avoided by using the µTESLA approach in such cases where a substation controller is available to distribute the initial group key in an authenticated way. Also the time delay caused by the period of uncertainty between reception and verification of a message is still occurring, making it inappropriate for control traffic requiring a very short reaction time (e.g., an emergency power switch off in case of overload). So, there is basically a trade-off whether immediate reaction to a control command is more important than sender authentication. The described approach of defining different security solutions for different message classes allows addressing application-specific side conditions by the security solution. For example, it is possible that a power on command is accepted only with sender authentication, while emergency power off is performed using normal group membership authentication. The susceptibility to denial-of-service attacks is not necessarily increased as control equipment could also provide wrong, manipulated measurements or control command by themselves (independent of any cryptographic authentication scheme).

VII. Integration Into Substation Automation Protocols

The described approach for multicast sender authentication can be integrated in existing field level energy automation protocols transmitting GOOSE or SV information. This is shown in Figure 10 by depicting the initial key chain generation and delayed key distribution.

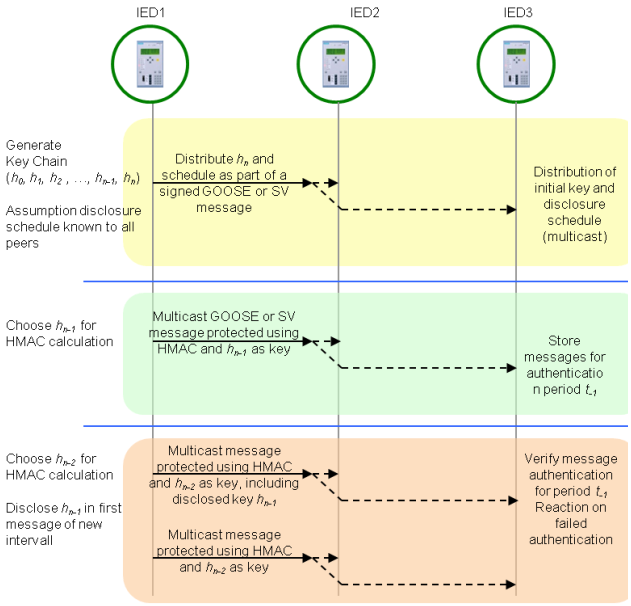


Figure 10. Broadcast/Multicast Control Message Sender Authentication in Field Level Energy Automation

This has the following implications on field level devices:

- Each field device requires a public private key pair to protect the initialization information. The public key is certified and available for other field devices.
- A disclosure schedule is known to all entities upfront, e.g., fixed or defined during engineering.
- The field device has to generate a hash key chain of determined length *n* (h_0, h_1, h_2 , ..., h_{n-1}). The length is determined by the time interval t_A that shall be covered by the overall hash key chain. Other factors are the storage requirements of messages at the receiver side. This time interval t_A is then divided into subintervals t_I. Each subinterval is associated with a key from the hash chain (t_0, h_n, t_1, h_{n-1} ... t_n, h_0).

The operation proceeds as follows:

- Step 1: Initialization of the Hash Chain by an IED.

  The field device sending GOOSE or SV broadcast/multicast messages provides the last value of the hash chain as part of a GOOSE or SV message and protects this message before sending it. The field level device uses a digital signature, or a higher-hierarchy hash key chain. The field device includes a description (manifest) of the message type protected with this hash chain. All subscribers will receive the message, and upon successful verification they will store the hash value together with an identifier of the sender. This identifier may be a MAC address, a serial number or similar.
- Step 2: Sending protected broadcast/multicast messages by a field device.

  After step 1, the time interval t_1, starts that is associated with the hash value h_{n-1}. The field device now uses a keyed hash for this time interval to protect the integrity of the GOOSE or SV values. The receiver has to store the messages until the sender has released the hash value h_{n-1}. This value can be released after the time interval has ended. The value can be released in clear. The receiver can now calculate the integrity check value of the stored message to achieve a delayed authentication of these messages.

An advanced variant of the key disclosure schedule may alternatively depend on the number of messages sent. Another advanced variant of the key disclosure schedule may alternatively depend on the priority (e.g., depending on the performance class) of the message sent.

As shown before, the general approach for protection of the distribution of the initial group key can be followed, allowing for authentication based on digital signatures (as in TESLA or as in IEC 61850-90-5) while the handling of the actual messages is protected using symmetric key application.

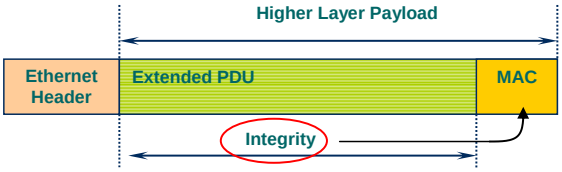


Figure 11. Application of a group key

Figure 11 shows the application of a group based key to provide integrity protection of the higher layer protocol.

VIII. Conclusions and Outlook

This paper described energy automation environments like substation communication where multicast authentication is used. Commands or sampled values are sent via GOOSE as defined in IEC 61850. As shown, the currently specified security mechanisms in IEC 62351-6 to ensure source authentication and message integrity provides for very good security. The flipside is that the application of this approach is hindered by the typical hardware used in IEDs. This hardware is limited anddoes not cope with performance requirements of the implied cryptographic operations (digital signatures) while matching the time restrictions of the deployment environment.

This paper analyzed various multicast authentication schemes as alternative solutions for the intended use case like digital signatures, GDOI for group key establishment in cooperation with a keyed hash for integrity protection, and TESLA. It investigates specifically the application of TESLA, and mapped the protocol to the substation automation use case. TESLA provides a solution for delayed authentication allowing an IED to perform a dedicated action in real-time and to perform the associated security check later on. It is obvious that there is a period of uncertainty between reception and verification of a message, making it inappropriate for control traffic requiring a very short

reaction time (e.g., an emergency power switch off in case of overload) for actions, which may not be reversible. So, there is basically a trade-off whether immediate reaction to a control command is more important than sender authentication. It is also possible to support different multicast authentication schemes within one technical solution and to use the described approach only for timely critical messages, while other messages may use the typical approach verifying a message, before operating on the content. Additionally, combining solutions allows for in-time authentication as a group member, while the delayed authentication can be used to identify an individual sender.

The described approach has not been implemented, yet. Hence, performance numbers and especially performance comparisons of the different approaches cannot be delivered at this time.

REFERENCES

[1] S.Fries and R.Falk, "Efficient Multicast Authentication in Energy Environments", Proc. IARIA Energy 2013, March 2013, ISBN 978-1-61208-259-2, pp. 65-71, http://www.thinkmind.org/download.php?articleid=energy_2013_3_30_40056 [retrieved July 2013]

[2] M. Felser, "Real-time Ethernet – industry prospective," Proc. IEEE, vol. 93, no.6, June 2005, pp. 1118-1128, http://www.felser.ch/download/FE-TR-0507.pdf [retrieved: Oct. 2012]

[3] IEC 61850-5 – "Communication requirements for functions and device models", July 2003.

[4] "Efficient Energy Automation with the IEC 61850 Standard Application Examples", Siemens AG, December 2010, http://www.energy.siemens.com/mx/pool/hq/energy-topics/standards/iec-61850/Application_examples_en.pdf [retrieved Oct. 2012].

[5] ISO 27001, ISO/IEC 27001:2005 Information technology – Security techniques – Information Security Management Systems – Requirements, http://www.iso27001security.com/html/27001.html [retrieved: Aug. 2013].

[6] ISO 27002, ISO/IEC 27002: 2005 Information technology – Security techniques – Information Security Management Systems – Code of practice for information security management, http://www.iso27001security.com/html/27002.html [retrieved: Aug. 2013].

[7] ISO 27019, ISO/IEC 27019: 2013 Information technology – Security techniques – Information Security Management Systems Information security management guidelines based on ISO/IEC 27002 for process control systems specific to the energy industry http://www.iso27001security.com/html/27019.html [retrieved: Aug. 2013].

[8] ISO/IEC62443-2-1 (99.02.01): Security for industrial automation and control systems Part 2-1: Industrial automation and control system security management system, Draft 6, Nov 2012.

[9] Northeast blackout of 2003, http://en.wikipedia.org/wiki/Northeast_blackout_of_2003 [retrieved: Aug. 2013].

[10] 2012 India blackouts, http://en.wikipedia.org/wiki/2012_India_blackouts [retrieved: Aug. 2013].

[11] List of major power outages, http://en.wikipedia.org/wiki/List_of_major_power_outages [retrieved: Aug. 2013].

[12] IEC 61850-90-5 – "Use of IEC 61850 to transmit synchrophasor information according to IEEE C37.118", December 2011

[13] B. Weis, S. Rowles, and T. Hardjono, "The Group Domain of Interpretation", RFC 6407, July 2012, http://tools.ietf.org/html/rfc6407 [retrieved: Aug. 2013]

[14] F. Hohlbaum, M. Braendle, and F. Alvarez, "Cyber Security – Practical considerations for implementing IEC 62351", May 2010, http://www05.abb.com/global/scot/scot387.nsf/veritydisplay/b3427a5374a35468c1257a93002d8df5/$file/1MRG006973_en_Cyber_Security_-_Practical_considerations_for_implementing_IEC_62351.pdf, [retrieved: Aug. 2013].

[15] T.S. Sidhu, M.G. Kanabar, and P. Palak, "Implementation issues with IEC 61850 based substation automation systems," Proc. Fifteenth National Power Systems Conference (NPSC), Dec. 2008, http://romvchvlcomm.pbworks.com/f/p274.pdf [retrieved: Oct. 2012].

[16] A. Perrig, R. Szewczyk, D. Tygar, V. Wen, and D. Culler, "SPINS: Security Protocols for Sensor Networks", Proceedings of the 8th Wireless Networks, pp 521-534, July 2002, http://www.csee.umbc.edu/courses/graduate/CMSC691A/Spring04/papers/spins-wine-journal.pdf [retrieved: Oct. 2012].

[17] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", Internet RFC 5246, Aug. 2008, http://tools.ietf.org/html/rfc5246 [retrieved: Oct. 2012].

[18] B. Kaliski, "PKCS#7 Cryptographic Message Syntax Version 1.5, Intenret RFC2315, March 1998, http://tools.ietf.org/html/rfc2315 [retrieved: Oct. 2012].

[19] C. Paar and J. Pelzl, "Understanding Cryptography", Springer, 2010.

[20] A. Perrig, D. Song, R. Canetti, J.D. Tygar, and B. Briscoe, "Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction", Internet RFC 4082, June 2005, http://tools.ietf.org/html/rfc4082 [retrieved: Oct. 2012].

[21] H. Harney and C. Muckenhirn, "Group Key Management (GPMP) Architecture", Internet RFC 2094, July 1997, http://tools.ietf.org/html/rfc2094 [retrieved: Oct. 2012].

[22] A. Ballardie, "Scalable Multicast Key Distribution", Internet RFC 1949, May 1996, http://tools.ietf.org/html/rfc1949 [retrieved: Oct. 2012].

[23] H. Harney, U. Meth, and A. Colegrove, "GSAKMP Group Secure Association Key Management Protocol", Internet RFC 4535, June 2006, http://tools.ietf.org/html/rfc4535 [retrieved: Oct. 2012].

[24] S. Rafaeli and D. Hutchison, "A Survey of Key Management for Secure Group Communication", ACM Computing Surveys, Vol. 35, No. 3, pp. 309-329, Sep. 2003, http://merlot.usc.edu/cs530-s08/papers/Rafaeli03a.pdf [retrieved: Oct. 2012].

[25] IEEE 802.11 "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Part 11", 2007.

[26] 3GPP TS33.246, "3G Security; Security of Multimedia Broadcast/Multicast Service (MBMS)", 2012, http://www.3gpp.org/ftp/Specs/html-info/33246.htm [retrieved: Oct. 2012].

[27] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman Key Distribution Extended to Group Communication", Proceedings of the 3rd ACM conference on Computer and communications security, pp. 31 – 37, ACM CCS96, 1996, http://corsi.dei.polimi.it/distsys/2007-2008/pub/p31-steiner.pdf [retrieved: Oct. 2012].

[28] J. Arkko, E. Carrara, F. Lindholm, M. Naslund, and K. Norrman, "MIKEY: Multimedia Internet KEYing", RFC 3830, August 2004

[29] D. Mills, U. Delaware, J. Martin, and W. Kasch, "Network Time Protocol 4: Protocol and Algorithms Specification", Internet RFC 5905, June 2010, http://tools.ietf.org/html/rfc5905 [retrieved: Oct. 2012].

[30] A. Perrig, R. Canetti, D. Song, and J.D. Tygar, "Efficient and Secure Source Authentication for Multicast", Network and Distributed System Security Symposium, NDSS '01, 2011, http://users.ece.cmu.edu/~adrian/projects/tesla-ndss/ndss.pdf [retrieved: Oct. 2012].

www.iariajournals.org

International Journal On Advances in Intelligent Systems

ICAS, ACHI, ICCGI, UBICOMM, ADVCOMP, CENTRIC, GEOProcessing, SEMAPRO, BIOSYSCOM, BIOINFO, BIOTECHNO, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS, CLOUD COMPUTING, COMPUTATION TOOLS, ENERGY, COLLA, IMMM, INTELLI, SMART, DATA ANALYTICS

issn: 1942-2679

International Journal On Advances in Internet Technology

ICDS, ICIW, CTRQ, UBICOMM, ICSNC, AFIN, INTERNET, AP2PS, EMERGING, MOBILITY, WEB

issn: 1942-2652

International Journal On Advances in Life Sciences

eTELEMED, eKNOW, eL&mL, BIODIV, BIOENVIRONMENT, BIOGREEN, BIOSYSCOM, BIOINFO, BIOTECHNO, SOTICS, GLOBAL HEALTH

issn: 1942-2660

International Journal On Advances in Networks and Services

ICN, ICNS, ICIW, ICWMC, SENSORCOMM, MESH, CENTRIC, MMEDIA, SERVICE COMPUTATION, VEHICULAR, INNOV

issn: 1942-2644

International Journal On Advances in Security

ICQNM, SECURWARE, MESH, DEPEND, INTERNET, CYBERLAWS

issn: 1942-2636

International Journal On Advances in Software

ICSEA, ICCGI, ADVCOMP, GEOProcessing, DBKDA, INTENSIVE, VALID, SIMUL, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS, CLOUD COMPUTING, COMPUTATION TOOLS, IMMM, MOBILITY, VEHICULAR, DATA ANALYTICS

issn: 1942-2628

International Journal On Advances in Systems and Measurements

ICQNM, ICONS, ICIMP, SENSORCOMM, CENICS, VALID, SIMUL, INFOCOMP

issn: 1942-261x

International Journal On Advances in Telecommunications

AICT, ICDT, ICWMC, ICSNC, CTRQ, SPACOMM, MMEDIA, COCORA, PESARO, INNOV

issn: 1942-2601