PATTERNS 2019

The Eleventh International Conferences on Pervasive Patterns and Applications

ISBN: 978-1-61208-708-5

May 5 - 9, 2019

Venice, Italy

PATTERNS 2019 Editors

Herwig Mannaert, University of Antwerp, Belgium

Alexander Mirnig, University of Salzburg, Austria

PATTERNS 2019

Forward

The Eleventh International Conferences on Pervasive Patterns and Applications (PATTERNS 2019), held between May 5 - 9, 2019 - Venice, Italy, continued a series of events targeting the application of advanced patterns, at-large. In addition to support for patterns and pattern processing, special categories of patterns covering ubiquity, software, security, communications, discovery and decision were considered. It is believed that patterns play an important role on cognition, automation, and service computation and orchestration areas. Antipatterns come as a normal output as needed lessons learned.

The conference had the following tracks:

- Patterns basics
- Patterns at work
- Discovery and decision patterns

Similar to the previous edition, this event attracted excellent contributions and active participation from all over the world. We were very pleased to receive top quality contributions.

We take here the opportunity to warmly thank all the members of the PATTERNS 2019 technical program committee, as well as the numerous reviewers. The creation of such a high quality conference program would not have been possible without their involvement. We also kindly thank all the authors that dedicated much of their time and effort to contribute to PATTERNS 2019. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations and sponsors. We also gratefully thank the members of the PATTERNS 2019 organizing committee for their help in handling the logistics and for their work that made this professional meeting a success.

We hope PATTERNS 2019 was a successful international forum for the exchange of ideas and results between academia and industry and to promote further progress in the area of pervasive patterns and applications. We also hope that Venice provided a pleasant environment during the conference and everyone saved some time for exploring this beautiful city.

PATTERNS 2019 Chairs

PATTERNS 2019 Steering Committee

Herwig Manaert, University of Antwerp, Belgium
Claudia Raibulet, University of Milano-Bicocca, Italy
Sneha Chaudhari, Carnegie Mellon University, Pittsburgh, USA
Valerie Gouet-Brunet, IGN, LaSTIG, MATIS, France
Wladyslaw Homenda, Warsaw University of Technology, Poland
Patrick Siarry, Université Paris-Est Créteil, France
Yuji Iwahori, Chubu University, Japan
Alexander Mirnig, University of Salzburg, Austria
Markus Goldstein, Ulm University of Applied Sciences, Germany
Stan Z. Li, Institute of Automation - Chinese Academy of Sciences, China
Reinhard Klette, Auckland University of Technology, New Zealand
Jacqueline Daykin, King's College London, UK / Aberystwyth University, Wales & Mauritius

PATTERNS 2019 Industry/Research Advisory Committee

Charles Perez, Paris School of Business, France
Christian Kohls, TH Köln, Germany
Krzysztof Okarma, West Pomeranian University of Technology, Szczecin, Poland
George A. Papakostas, Eastern Macedonia and Thrace Institute of Technology, Greece
Adel Al-Jumaily, University of Technology, Australia

PATTERNS 2019

Committee

PATTERNS Steering Committee

Herwig Manaert, University of Antwerp, Belgium
Claudia Raibulet, University of Milano-Bicocca, Italy
Sneha Chaudhari, Carnegie Mellon University, Pittsburgh, USA
Valerie Gouet-Brunet, IGN, LaSTIG, MATIS, France
Wladyslaw Homenda, Warsaw University of Technology, Poland
Patrick Siarry, Université Paris-Est Créteil, France
Yuji Iwahori, Chubu University, Japan
Alexander Mirnig, University of Salzburg, Austria
Markus Goldstein, Ulm University of Applied Sciences, Germany
Stan Z. Li, Institute of Automation - Chinese Academy of Sciences, China
Reinhard Klette, Auckland University of Technology, New Zealand
Jacqueline Daykin, King's College London, UK / Aberystwyth University, Wales & Mauritius

PATTERNS Industry/Research Advisory Committee

Charles Perez, Paris School of Business, France
Christian Kohls, TH Köln, Germany
Krzysztof Okarma, West Pomeranian University of Technology, Szczecin, Poland
George A. Papakostas, Eastern Macedonia and Thrace Institute of Technology, Greece
Adel Al-Jumaily, University of Technology, Australia

PATTERNS 2019 Technical Program Committee

Andrea F. Abate, University of Salerno, Italy
Mourad Abbas, STRCDAL, Algeria
Adel Al-Jumaily, University of Technology, Sydney, Australia
Adel M. Alimi, University of Sfax, Tunisia
Danilo Avola, Sapienza University of Rome, Italy
Berkay Aydin, Georgia State University, USA
Johanna Barzen, Universität Stuttgart, Germany
Abdel Belaïd, LORIA, France
Boulbaba Ben Amor, IMT Lille Douai / CRIStAL, France
Hatem Ben Sta, Université de Tunis - El Manar, Tunisia
Nadjia Benblidia, Saad Dahlab University - Blida1, Algeria
M.K. Bhuyan, IIT Guwahati, Assam, India
Silvia Biasotti, CNR – IMATI, Italy
Julien Broisin, University of Toulouse - Institut de Recherche en Informatique de Toulouse (IRIT), France
Michaela Bunke, Universität Bremen, Germany
Jocelyn Chanussot, Université Grenoble Alpes, France
Amitava Chatterjee, Jadavpur University, Kolkata, India

Sneha Chaudhari, Carnegie Mellon University, Pittsburgh, USA
David Daqing Chen, London South Bank University, UK
Sergio Cruces, University of Seville, Spain
Mohamed Dahchour, National Institute of Posts and Telecommunications, Rabat, Morocco
Mohamed Daoudi, Institut Mines-Telecom / Telecom Lille, France
Kaushik Das Sharma, University of Calcutta, Kolkata, India
Jacqueline Daykin, King's College London, UK / Aberystwyth University, Wales & Mauritius
Peter De Bruyn, University of Antwerp, Belgium
Danielly Cristina de Souza Costa Holmes, Federal Institute of Rio Grande do Sul, Brazil
Claudio De Stefano, University of Cassino and Southern Lazio, Italy
Vincenzo Deufemia, University of Salerno, Italy
Moussa Diaf, Mouloud MAMMERI University, Algeria
Susana C. Esquivel, Universidad Nacional de San Luis, Argentina
Christine Fernandez-Maloigne, University of Poitiers, France
Francesco Fontanella, Università di Cassino e del Lazio Meridionale, Italy
Christos Gatzidis, Bournemouth University, UK
Markus Goldstein, Ulm University of Applied Sciences, Germany
Valerie Gouet-Brunet, IGN, LaSTIG, MATIS, France
Christos Grecos, Central Washington University, USA
Jean Hennebert, University of Fribourg, Switzerland
Mannaert Herwig, University of Antwerp, Belgium
Wladyslaw Homenda, Warsaw University of Technology, Poland
Tzung-Pei Hong, National University of Kaohsiung, Taiwan
Wei-Chiang Hong, School of Education Intelligent Technology - Jiangsu Normal University, China
Chih-Cheng Hung, Kennesaw State University, USA
Karen Hunsdale, University of Winchester, UK
Shareeful Islam, University of East London, UK
Biju Issac, Northumbria University, Newcastle, UK
Yuji Iwahori, Chubu University, Japan
Agnieszka Jastrzebska, Warsaw University of Technology, Poland
Maria João Ferreira, Universidade Portucalense, Portugal
Hassan A. Karimi, University of Pittsburgh, USA
Yasmin M. Kassim, University of Missouri - Columbia, USA
Lazhar Khelifi, University of Montreal, Canada
Reinhard Klette, Auckland University of Technology, New Zealand
Christian Kohls, TH Köln, Germany
Sylwia Kopczynska, Poznan University of Technology, Poland
Sotiris Kotsiantis, University of Patras, Greece
Adam Krzyzak, Concordia University, Canada
Robert S. Laramee, Swansea University, UK
Fritz Laux, Reutlingen University, Germany
Gyu Myoung Lee, Liverpool John Moores University, UK
Alex Po Leung, Macau University of Science and Technology, Macau
Haim Levkowitz, UMass Lowell, USA
Stan Z. Li, Institute of Automation - Chinese Academy of Sciences, China
Josep Lladós, Universitat Autònoma de Barcelona, Spain
Khoa Luu, Carnegie Mellon University, USA
Pierre-Francois Marteau, IRISA / Université Bretagne Sud, France

Alexander Mirnig, University of Salzburg, Austria
Hongwei Mo, Harbin Engineering University, China
Fernando Moreira, Universidade Portucalense, Portugal
Moin Nabi, SAP Innovation Center Network, Berlin, Germany
Serena Nicolazzo, University Mediterranea of Reggio Calabria, Italy
Antonino Nocera, University Mediterranea of Reggio Calabria, Italy
Jean-Marc Ogier, Université de La Rochelle, France
Krzysztof Okarma, West Pomeranian University of Technology, Szczecin, Poland
Hélder Oliveira, INESC TEC, Portugal
Alessandro Ortis, University of Catania, Italy
Mrutyunjaya Panda, Utkal University, India
George A. Papakostas, Eastern Macedonia and Thrace Institute of Technology, Greece
Giuseppe Patane', CNR-IMATI, Italy
Christian Percebois, IRIT - University of Toulouse, France
Charles Perez, Paris School of Business, France
Agostino Poggi, DII - University of Parma, Italy
Giovanni Puglisi, University of Cagliari, Italy
Rafael Radkowski, Iowa State University, USA
Claudia Raibulet, University of Milano-Bicocca, Italy
Giuliana Ramella, CNR - National Research Council, Italy
Aurora Ramirez, University of Córdoba, Spain
Theresa-Marie Rhyne, Independent Visualization Consultant, USA
Alessandro Rizzi, Università degli Studi di Milano, Italy
Gustavo Rossi, UNLP - La Plata, Argentina
Antonio-Jose Sanchez-Salmeron, Universitat Politecnica de Valencia, Spain
María-Isabel Sanchez-Segura, Carlos III University of Madrid, Spain
José Santos Reyes, University of A Coruña, Spain
Muhammad Sarfraz, Kuwait University, Kuwait
Alessandra Scotto di Freca, University of Cassino and Southern Lazio, Cassino, Italy
Giuseppe Serra, University of Udine, Italy
Isabel Seruca, Portucalense University, Portugal
Patrick Siarry, Université Paris-Est Créteil, France
Mu-Chun Su, National Central University, Taiwan
Marek Suchánek, CTU in Prague / Centre for Conceptual Modelling and Implementation, Czech Republic
Yusuke Takezawa, Tokyo Denki University, Japan
Shanyu Tang, University of West London, UK
J. A. Tenreiro Machado, Polytechnic of Porto, Portugal
Daniel Thalmann, Institute for Media Innovation (IMI) - Nanyang Technological University, Singapore
Mati Tombak, University of Tartu, Estonia
Andrea Torsello, Universita' Ca' Foscari Venezia, Italy
Hiroyasu Usami, Chubu University, Japan
Michael Vassilakopoulos, University of Thessaly, Greece
Nicole Vincent, Université Paris Descartes, France
Panagiotis Vlamos, Ionian University, Greece
Krzysztof Walczak, Poznan University of Economics, Poland
Patrick Wang, Northeastern University, USA
Laurent Wendling, Paris Descartes University (Paris 5), France
Reuven Yagel, The Jerusalem College of Engineering, Israel

Hongchuan Yu, Bournemouth University, UK
Pong C. Yuen, Hong Kong Baptist University, Hong Kong

Copyright Information

Table of Contents

Evolvability Evaluation of Conceptual-Level Inheritance Implementation Patterns

Marek Suchánek and Robert Pergl

Faculty of Information Technology
Czech Technical University in Prague
Prague, Czech Republic
Email: marek.suchanek,robert.pergl@fit.cvut.cz

***Abstract*—Inheritance is a well-known construct in conceptual modelling, as well as in the object-oriented programming, where it is often used to enable reusability and to modularize complex applications. While it helps in conceptual modelling and understanding of complex domains, it usually results in evolvability issues in software implementations. This paper discusses problems caused by single and multiple inheritance with respect to increasing accidental complexity of a model and evaluates various patterns that can be used to transform conceptual-level inheritance into implementation with respect to code evolvability. The points are illustrated on the transformation of an example ontological conceptual model in OntoUML into various software implementation models.**

Keywords–Inheritance; Generalization; UML; OntoUML; Evolvability; Normalized Systems.

I. Introduction

Software engineering uses conceptual notions of inheritance, generalization, or subtyping for decades in an analysis of a problem domain as well as in design and implementation of applications [1]. It captures a very essential and natural property of something being a special case of something more generic. Inheritance is one of the abstractions that we can use to describe and model real-world concepts based on our cognition principles. On the other hand, in software implementations, inheritance is often abused and used inappropriately causing ripple effects and thus negatively affecting software evolvability [2]. This worsens obviously when it comes to multiple inheritance, which is, however, again natural in the real world but complicated and sometimes even unsupported in software implementations.

Evolvability problems caused by inheritance in Object-Oriented Programming (OOP) are well-known and it is often suggested to follow composite reuse principle clause "composition over inheritance" [3]. Majority of OOP design patterns use composition together with generic interfaces instead of class generalizations. Our goal here is to explore, describe, and evaluate patterns for translating conceptual-level inheritance into combinatorial effect-free (CE-free) implementation and discuss its suitable use cases. The result should help to reduce the *accidental complexity* of models that is introduced due to technical reasons – as opposed to domain-given and irreducible *essential complexity*.

In Section III, we summarize the related terminology and the current state of the art in solving the problem of inheritance in the software engineering domain. Then, in Section IV we use this knowledge to discuss and evaluate the evolvability of various solutions in the form of patterns. Evaluation is summarized and all patterns are compared in Subsection IV-E. Finally, we propose possible next steps in this work and describe related research questions in Section V.

II. Methodology

For the conceptual ontological models, we use the OntoUML language and for the object-oriented models, we use UML. OntoUML is an ontology-driven conceptual modelling language based on terms from Unified Foundational Ontology (UFO) [4] focused on producing expressive, highly ontologically-relevant models. The Unified Modeling Language (UML) by OMG [5] is probably not necessary to introduce. Its difference to OntoUML in our context is mainly its historical focus on depicting object-oriented programmes rather than ontological committment.

The core of this paper are proposals of model transformation patterns. As such, we suppose generation of code from the models and then customizations in the style of model-driven development [6] or Normalized Systems Expanders [7]. In descriptions of models, we strictly distinguish between an *entity* on the conceptual level and a *class* in UML and a related OOP implementation since there is a significant semantic difference between these two – a (UML) class is an OOP representation of (possibly more) OntoUML entities, as we show later. To avoid misunderstanding we also respectively use *instance* of entity or class instead of word *object*.

We discuss the evolvability issues with the respect to a "user", being an end user of a software application implementing CRUD operations related to a discussed model and also with the respect to "programmer" being a programmer developing customizations of the generated code. These customizations are considered to be any piece of code that enhances or otherwise changes the functionality of the generated code.

III. Related Work and Terminology

In this section, we very briefly review necessary terminology for understanding various notions and use cases of inheritance on conceptual-level and its counterparts in implementation. Then, we also shortly describe existing and related solutions that affect our evaluation.

A. Generalization Typology

The notion of inheritance may mean semantically different things and such differences are unfortunately often ignored; a thorough discussion of various types of inheritance in software engineering is provided by Taivalsaari in [1]. When we talk about conceptual-level inheritance, we speak about *generalization* and *specialization*, or *is-a* relationship. It reflects our cognitive resolution of an entity being related to a different entity in a way that all instances of the first are also instances of the second, but in some specific aspects differ.

For inheritance in implementation, the terminology is a bit more complicated. The term *subclassing* describes using inheritance to reuse code, which is also called “accidental” use of inheritance. This sort of inheritance causes misunderstanding because it is used mainly for maintaining Do-not-Repeat-Yourself (DRY) code even though it creates combinatorial effects and there is no semantic foundation of such relation. The designation “essential” use of inheritance is used for *subtyping* that ensures substitutability in terms of the Liskov substitution principle [8]. [1] states that *subtyping* ensures *specialization*, but of course it can be again used inappropriately without alignment to the real world.

B. Is-a Hierarchies

The simplest way of modelling inheritance is the so-called *is-a hierarchy*, where two concepts are connected with is-a relationship as a subclass and a superclass of subsumption, e.g., *Student is a Person*. This started to appear widely in Extended Entity-Relationship (EER) models that can be considered as the predecessor of today’s widely used object-oriented modelling languages, such as UML or Object-role modeling (ORM) [9]. Although such representation of inheritance is simplified, it is well-defined and allows straightforward mathematical reasoning [10].

C. OntoUML Conceptual-Level Inheritance

Generalization and specialization are essential in OntoUML; we use it to demonstrate conceptual-level inheritance. An instance in OntoUML can be an instance of multiple entities, however it must obey the single identity principle (given by `<<kind>>` or `<<subkind>>`), as depicted in Figure 1. This notion of multi-class instances is natural at the ontological level, however, as we discuss further, it is not typical in OOP and as such in standard UML. To bridge the ontological level with implementation, a research has been performed of transformation of OntoUML into UML and relational model [11]–[15], summarized in [16]. We evaluate here some of the patterns described in this work.

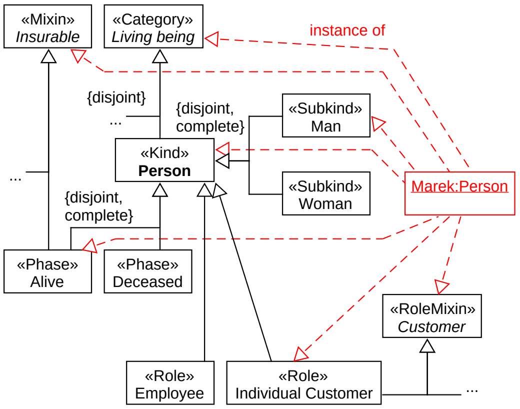


Figure 1. An excerpt of an OntoUML diagram showing inheritance and instantiation in OntoUML

D. Generalization Sets

Unified Modelling Language (UML) [5] defines a *generalization set* as an element whose instances define collections of subsets of generalization relationships. The purpose of such elements is to tell more about multiple generalizations via meta attributes. For example, a generalization set can be used to express that an instance of entity *Person* is either an instance of entity *Alive Person* or *Deceased Person* with meta attributes *disjoint* and *complete*.

E. Object-Oriented Inheritance

In OOP, inheritance is very often used as an enabler of well-known DRY and the mentioned Liskov substitution principle, one from SOLID principles recommended by Robert C. Martin [17]. A subclass can easily extend its superclass(es) by adding or changing attributes and methods. Deletion or cancellation of methods or attributes is often possible only by throwing an exception if called on a subclass. We take into further considerations only OOP where an object is an instance of exactly one class since the most widely used languages (Java, C#, C++, Python, etc.) do not support making instances of several classes [8]. Other languages that allow duck-typing and prototyping deal with some inheritance problems but creating other, usually leading to runtime errors caused by weak type system [18].

F. Inheritance in Normalized Systems

The Normalized System (NS) theory [2] deals with evolvability of systems in general and declares fine-grade modularization based on four main principles: Separation of Concerns, Separation of States, Data Version Transparency, and Action Version Transparency. Using those principles leads to a system with eliminated or at least controlled combinatorial effects (CE) – effectively being a measure of evolvability over time. Software systems are the core domain of NS theory application in practice [7].

In the NS theory, inheritance is criticized for causing CEs in software systems [2]. It is important to stress that CEs are caused by inheritance in OOP, not inheritance at the conceptual-level, as it may be transformed into a CE-free implementation, as will be discussed further. For the transformation of models in our work, the Separation of Concerns principle is crucial. Nevertheless, the remaining three principles are relevant for the implementation even though they are not applied directly in the transformation.

IV. Evaluation of Inheritance Patterns

The main part of this work is the description, evaluation, and summarization of various ways of implementing a conceptual-level inheritance in various use cases. For a demonstration of different patterns, we use Figure 1 as a conceptual model that should be implemented. We first introduce a simple approach with traditional inheritance implementation and discuss its problems. Then, we explore three patterns from the least to the most complex, their possible combinations and summarize their comparison. Those patterns are based on [9] [16], and can be found in many other software engineering resources. Our focus is not just directed on a generation of CE-free implementation of conceptual-level inheritance, but also on its impact on usability for both the user and the programmer.

We observe and discuss mainly the following measures that we quantify if possible to support our evaluation of the patterns:

- Total number of classes needed for implementation, composed of conceptual-level entity and additional classes.
- How many steps it takes to navigate to superclass or subclass.
- Change boundary, i.e., how many classes need to be changed when one changes in the conceptual model.

A. Traditional OOP Inheritance

A straightforward way how to implement conceptual-level inheritance is using *subtyping*. First, inheritance from conceptual model remains as is; the only change is that OntoUML's stereotypes are stripped off and some of the classes are made abstract to forbid their instantiation, as can be seen in Figure 2. Then, for each possible and instantiatable combination of conceptual entities, an empty concrete class is generated if needed using multiple inheritance depicted in gray. A generated concrete class is a subclass of appropriate abstract classes that implement entities. If there is a case that single inheritance is used (a class matches an entity $1:1$), such generated class can be removed and the superclass turned into a concrete one; in other cases, the accidental complexity grows and we must generate 1 class mixing $n > 1$ entities. In our model, instead of generating trivial classes with single inheritance *Man* and *Woman*, we just turn them from abstract to concrete resulting into 6 generated classes instead of 8.

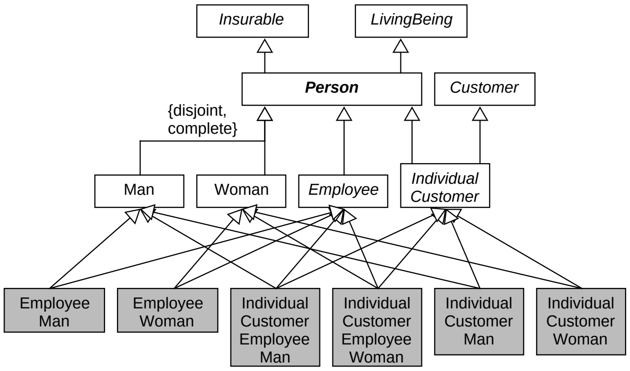


Figure 2. Traditional (multiple) inheritance pattern

Problem with this approach is that a language that supports multiple inheritance is needed, but more importantly, the introduced combinatorial effect is huge. A class instance implements a single or multiple conceptual entity instances. Therefore, there will be $\binom{n}{0} + \binom{n}{1} + \ldots + \binom{n}{n} = 2^n$ generated classes, where $n + 1$ is the number of entities that can be instantiated (if there is just one entity, $n = 0$). For example, if we want to incorporate subclasses *Alive* and *Deceased* as in Figure 1, we need to completely change everything. There would be 16 generated classes in total and none of them would be the same as in the previous situation with 8 classes, which means a potential breaking change.

Another problem is caused by the rigidity of this design. When you want to turn an instance of class `AliveEmployeeWoman` into instance of `DeceasedEmployeeWoman`, a transformation procedure must be implemented – making system more complex internally, as well as for the user, as new forms are required. Last, programming customizations would be very difficult since, as shown, adding or removing classes at the conceptual level leads into significant changes in code that can make customizations invalid. Imagine that there is a customization for `CustomerWoman`, but then there is no such class only two `AliveCustomerWoman` and `DeceasedCustomerWoman` – a decision how to apply the customization on those would need to be made. As result, the only advantage is direct inheritance without any necessary navigation steps.

B. The Union Pattern

This new pattern we introduce here is very close to *Single Table Inheritance* (in some frameworks also referred as *Table-Per-Hierarchy*) that is used widely in relational databases. Basically, it is a class that unions all properties from the class and its subclasses – thus we call it *union pattern*. For each core class (identity provider entity in OntoUML) and its inheritance tree including all abstract superclasses and all subclasses, a union class is made as in Figure 3. Every subclass decision is covered by so-called *discriminator* that can be generated as an attribute and it tells which subclass(es) subset of every generalization sets is picked (e.g., `dManOrWoman`) or which single subtype is used (e.g., `dEmployee`).

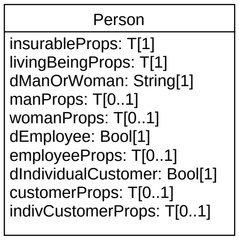


Figure 3. Union pattern

A significant disadvantage can be seen in the massive complexity of a single class and violating the Separation of Concerns principle. Adding a new subclass results in a definition of a new discriminator value and new fields in an existing class. It acts as a single complex data element with optional properties based on some other attributes. With this pattern, checks of actual subtypes must be implemented manually if needed, for instance, if there is a need to create a relationship only with *Employee*. Even more problems emerge where the hierarchy of subclasses is deeper. However, a user can be fully shielded from this implementation by having a single form with picking what is needed. For a programmer, this may result in tedious work (meaning time-consuming and error-prone) to handle all cases of complex class with multiple discriminators and many optional properties.

From the perspective of observed measures, there are precisely 0 of additional classes and one whole hierarchy from conceptual model is merged into one class. For navigation to subclass a single step of discriminator comparison has to be made (when relying on the integrity checks). Although this pattern is simple and seems to solve evolvability issues, it is practically a conceptual anti-pattern.

C. Composition Pattern

We call this pattern after the already-mentioned advice *composition over inheritance*. It is similar to the previous Union Pattern, but instead of merging all properties into one class, a composition is used. Thanks to that, no *discriminator* is needed and also no *subclassing* is needed as shown in Figure 4. All generalization relations are replaced by bidirectional *is-a* associations that are always mandatory for a subclass, but can be optional for superclass if the used generalization set is not *complete*. For *disjoint* sets or more complex constraints, additional checks, such as *xor*, must be implemented and a mere multiplicity setting is not sufficient.

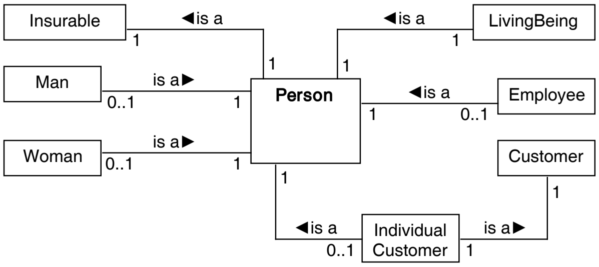


Figure 4. Composition pattern

An important change against the original conceptual model is that every abstract class is turned into concrete in order to be instantiated and such instance must be related (transitively) to exactly one instance of a core class. Transformation of generalizations into *is-a* associations can be also done fully automatically including integrity checks (for instance, that *Person* is *Woman* or *Man*, but never both). It is possible again to fully shield the implementation from the user as it works from this point of view similar to union pattern, just with special associations. Instead of additional comparison of discriminator value for navigation to subclass, the relation is checked and used if is realised. There is the same number of classes as in the conceptual model.

A programmer should have easier work than in the previously described patterns thanks to Separation of Concerns and possibility to use the power of a type system when passing subclass or superclass instances. Implementation and support mechanisms of *is-a* should allow a programmer to easily navigate to superclass instances, check if subclass instances are set and if so, navigate to it. When the inheritance hierarchy is changed within the model, some of the previous *is-a* links can be removed or edited causing incompatibility of customization. However, it is the same sort of ripple effect as for the union pattern, only the change is not encapsulated within a single (but huge) class but in a limited number of classes within a single hierarchy.

D. Generalization Set Pattern

The main weakness of the composition pattern is the need of handling generalization set constraints by implementing or generating constraints, and it can still make a problem when accessing instances of subclasses. We introduce this pattern to solve this problem and also to avoid having multiple *is-a* links attached directly to the instance together with all integrity checks – we want to separate this concern out. As depicted in Figure 5, for each class that has subclass(es), a special class for generalization sets (marked by «GS» stereotype) is generated. If there are other specialized generalization sets for an entity in the conceptual model (such as the one for *Man* and *Woman*), an extra generalization set class is generated and linked from the core one.

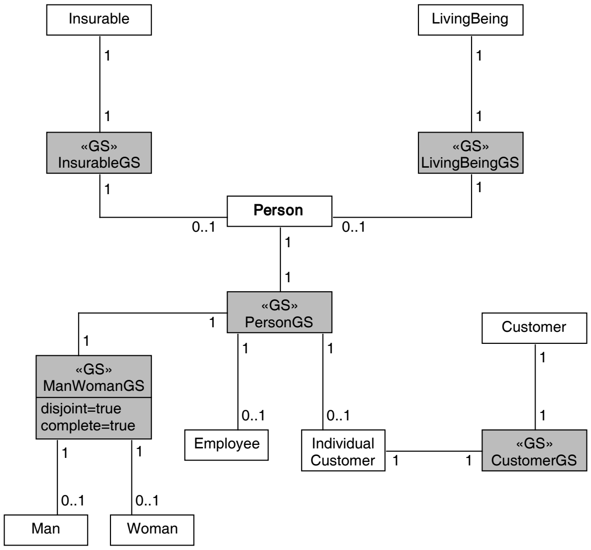


Figure 5. Generalization set pattern

Each «GS» class can be defined with class attributes such as *complete* and *disjoint* that can adjust the behaviour of generalization set including integrity checks or default form generation. Moreover, new attributes can be introduced to enhance functionality in the future without breaking the existing model. For a programmer, the work is simplified by cleaner data classes and «GS» encapsulating necessary utility for implementing the inheritance in exchange for extra navigation per a single generalization set ($A \leftrightarrow GS \leftrightarrow B$ instead of directly $A \leftrightarrow B$). An end user is again shielded and thanks to the generated «GS», the user interface can be generated easily and universally thanks to straightforward reuse and composition of simpler parts.

The significant negative issue with this pattern comes in form of overhead caused by a number of extra classes and extension of the generation tool by a new and non-trivial concept. Sometimes the «GS» class is unnecessary, for example, if there is no other *LivingBeing* than *Person* in the model. In such cases, extra «GS» class could be omitted but then a customization would use direct navigation. However, such a simplification leads to an increased impact of changes when a new subclass is added and generalization set is additionally created.

E. Comparison of Inheritance Patterns

We showed benefits of the discussed three patterns above in terms of evolvability compared to traditional OOP inheritance. We deliberately introduced the patterns in order so that the following removes the most significant problems of the previous. Unfortunately but naturally, it is never “for free” and each solution has its own problems, mainly in terms of additional complexity, which is, however, not an issue when we talk about automatically generated code. A summary of the discussion is provided in Table I.

We evaluated the generalization set pattern as generally the most suitable from those covered in this paper, however, there are use cases where some of the others may be better applicable. Below, we make such discussion from the conceptual point. In cases where time and/or space complexity needs to be optimised, conclusions about patterns suitability may differ.

TABLE I. COMPARISON OF INHERITANCE PATTERNS EVOLVABILITY

Evolvablity aspect	Traditional inheritance	Union pattern	Composition pattern	GS pattern
Number of additional classes	2^n	0	0	1 per GS
Multiple-inheritance support	yes (if the PL supports)	causing CE	with constraints	yes
Customizations evolvability	breaking changes	SoC violated	OK	OK
Accessing superclass	by the PL used	discriminator	association	associations
UI uniformity obstacles	migrations, repetition	repetition	constraints	no
Hierarchy change impact	whole hierarchy	class	association(s)	association(s), GS class(es)

The union pattern, when compared to the generalization set pattern, is suitable when the inheritance hierarchy is not deep but shallow with a single or very low number of generalization sets. In such case, multi-valued discriminator and optional fields will be used and handling a single instance will be very simple. On the other hand, if there are more levels of inheritance, it is too complex to use. The first option is that a superclass is merged into subclass(es) resulting in repetition (e.g., *Insurable* for *Person* but also *Building*). The second option is to merge at the top level, which causes a serious problem of a too complex class including non-related properties. The last serious problem is that this pattern cannot be used without repetition for multiple inheritance.

The composition pattern is very similar to the generalization set pattern and the only advantage is a reduced complexity by leaving out special «GS» classes. It allows multiple inheritance with simple reuse thanks to creating linked instances and it has no problem with deeper inheritance hierarchies compared to the union pattern. The key problem is with handling inheritance and generalization set constraints inside the superclass and subclass. That is not a problem for simple unconstrained generalizations, as already mentioned. The violation of Separation of Concerns principle is removed in the generalization set pattern, where it can be hidden using various OOP design patterns (such as Proxy) even for programmers.

F. Combinations of the Previous Patterns

For a transformation of a complex OntoUML model, it may also be a possibility to select different patterns for different parts of the model according to the discussion above. However, new concerns arise:

- The uniformity of user interface (UI) and experience user (UX) should be maintained.
- Using different patterns will necessarily result in different data accessing strategies (especially navigation of subclass/superclass), which complicates the job of the programmer and requires extra knowledge of the various patterns and their characteristics, also during the transformation phase.

It might seem a good idea to go for the Union pattern in case of a simple inheritance with multiple subclasses in one generalization set, which leads to one *discriminator*, or to remove «GS» class when it is not needed, but after considering the listed negative effects or additional needs, sticking with the most powerful and flexible generalization set pattern is generally recommended in spite of its higher internal complexity.

V. FUTURE RESEARCH AND RELATED PROBLEMS

The obvious future work is implementation of the transformations in an MDA-enabled CASE tool and design the necessary transformations and code generation. The proposed transformations should be now encoded and verified by real-world practice on larger code bases. There are also topics for more theoretical research on the topic.

A. Interfaces Evolvability

In this paper, we evaluated the inheritance of OOP in terms of subclassing. Some object-oriented programming languages, for instance, Java, comes with a construct called *interface* and a relation *implements*. Implementing an interface is in some aspects similar to extending a class but it is more constrained. Although this construct is not generally used at the conceptual-level and is tied to implementation, possible future research could evaluate those differences and its relation to evolvability. Similarly, there are additional constructs such as traits (e.g., in Pharo [19]) or mixins (e.g., in Ruby [20]) that call for similar analysis. Their potential to serve similarly as multiple inheritance in some respects make them interesting for this purpose. However, as they seem not the mainstream of today's OOP, we did not deal with them in the first case.

B. Automatic Transformation with Pattern Selection

In Section IV-F, we sketched an idea of patterns combination. If the selection is implemented in an automatic way and moreover, the API of the generated model is further abstracted so that it provides a unified approach for all the patterns, benefits of the patterns may be used without the pointed drawbacks. However, we are aware that this is a quite challenging task.

VI. CONCLUSION

In this paper, we briefly described and summarized terminology of inheritance in software engineering and the relation between inheritance on the conceptual level and its implementation in the source code. The possible patterns of implementing conceptual-level inheritance in OOP were introduced, evaluated, and compared in multiple aspects of evolvability. The implementation itself is shown to be non-trivial and there are more options suitable for different cases. Further verification of our contribution on real-world use cases is desirable as follow up in this research. During the work, we have encountered multiple related important questions and problems and suggested future research.

ACKNOWLEDGEMENTS

This research was supported by the grant of Czech Technical University in Prague No. SGS17/211/OHK3/3T/18. Initial steps in this research were done during the Normalized Systems Summer School 2018 organized by the University of Antwerp.

References

[1] A. Taivalsaari, “On the Notion of Inheritance,” ACM Computing Surveys, vol. 28, no. 3, September 1996, pp. 438–479.

[2] H. Mannaert, J. Verelst, and P. De Bruyn, Normalized Systems Theory: From Foundations for Evolvable Software Toward a General Theory for Evolvable Design. Kermt (Belgium): Koppa, 2016.

[3] E. Gamma, R. Helm, R. E. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, March 1995.

[4] G. Guizzardi, Ontological Foundations for Structural Conceptual Models. CTIT, Centre for Telematics and Information Technology, 2005.

[5] B. Selic et al., “OMG Unified Modeling Language (Version 2.5),” March 2015.

[6] A. G. Kleppe, J. Warmer, J. B. Warmer, and W. Bast, MDA Explained: The Model Driven Architecture – Practice and Promise. Addison-Wesley Professional, 2003.

[7] G. Oorts, P. Huysmans, P. D. Bruyn, H. Mannaert, J. Verelst, and A. Oost, “Building Evolvable Software Using Normalized Systems Theory: A Case Study,” in 2014 47th Hawaii International Conference on System Sciences(HICSS), January 2014, pp. 4760–4769.

[8] R. W. Sebesta, Concepts of Programming Languages, 10th ed. Pearson, 2012.

[9] R. Elmasri and S. Navathe, Fundamentals of Database Systems. London: Pearson, 2016.

[10] A. Artale, D. Calvanese, R. Kontchakov, V. Ryzhikov, and M. Zakharyaschev, “Reasoning over Extended ER Models,” in International Conference on Conceptual Modeling. Springer, 2007, pp. 277–292.

[11] Z. Rybola and R. Pergl, “Towards OntoUML for Software Engineering: Introduction to the Transformation of OntoUML into Relational Databases,” in Enterprise and Organizational Modeling and Simulation, ser. LNBIP. Ljubljana, Slovenia: Springer, June 2016.

[12] ——, “Towards OntoUML for Software Engineering: Transformation of Rigid Sortal Types into Relational Databases,” in Proceedings of {FedCSIS} 2016, ser. ACSIS, vol. 8. Gdańsk, Poland: IEEE, September 2016, pp. 1581–1591.

[13] ——, “Towards OntoUML for Software Engineering: Transformation of Anti-Rigid Sortal Types into Relational Databases,” in Model and Data Engineering, ser. LNCS. Aguadulce, Almería, Spain: Springer, September 2016, pp. 1–15.

[14] ——, “Towards OntoUML for Software Engineering: Transformation of Kinds and Subkinds into Relational Databases,” Computer Science and Information Systems, vol. 14, no. 3, 2017, pp. 913–937.

[15] ——, “Towards OntoUML for Software Engineering: Optimizing Kinds and Subkinds Transformed into Relational Databases,” in Enterprise and Organizational Modeling and Simulation, ser. LNBIP. Tallinn, Estonia: Springer, Cham, November 2018, pp. 31–45.

[16] Z. Rybola, “Towards OntoUML for Software Engineering: Transformation of OntoUML into Relational Databases,” Ph.D. thesis, Czech Technical University in Prague, Prague, Czech Republic, August 2017, [retrieved: Apr, 2019]. [Online]. Available: https://www.fit.cvut.cz/sites/default/files/PhDThesis-Rybola.pdf

[17] R. C. Martin, “Design Principles and Design Patterns,” Object Mentor, vol. 1, no. 34, 2000, p. 597.

[18] B. C. Pierce and C. Benjamin, Types and Programming Languages. MIT press, 2002.

[19] A. Bergel, D. Cassou, S. Ducasse, J. Laval, and J. Bergel, Deep into Pharo. Square Bracket, 2013.

[20] D. Thomas et al., Programming Ruby: The Pragmatic Programmers’ Guide. Raleigh, NC: Pragmatic Bookshelf, 2005.

Using Normalized Systems to Explore the Possibility of Creating an Evolvable Firewall Rule Base

Geert Haerens

Faculty of Business and Economics
University of Antwerp, Belgium
and Engie IT — Dir. Architecture
Email: geert.haerens@engie.be

Peter De Bruyn

Department of Management Information Systems
Faculty of Business and Economics
University of Antwerp, Belgium
Email: peter.debruyn@uantwerp.be

***Abstract*—A firewall is an essential network security component. The firewall rule base, the list of filters to be applied on network traffic, can have significant evolvability issues in a context where companies consider their firewall as complex. Whereas sufficient literature exists on how to analyze a rule base which is running out of control, little research is available on how to properly construct a rule base upfront, preventing the evolvability issues to occur. Normalized Systems (NS) theory provides proven guidance on how to create evolvable systems. In this paper, NS is used to study the combinatorics involved when creating a firewall rule base. Based on those combinatorics, an artifact (method) is proposed to create a firewall rule base which has evolvability in its design.**

Keywords–Normalized Systems; Firewall; Rule base

I. INTRODUCTION

Firewalls are an essential component of network security. They have been protecting network connected resources for over 25 years and will continue to do so for the next decades [14] [15]. Initially, firewalls were used to protect a company against threats coming from the outside (i.e., the “evil Internet”). This is referred to as filtering North-South traffic [20]. But security breaches are not only caused by access through the Internet. A significant portion of security breaches are caused from within the company network [18] where hacks have become more sophisticated. Getting a foothold on one resource on the internal network and from there on hopping between resources, is a known hacking strategy against which filtering North-South traffic offers no protection. For this reason, protecting the network connected resources from internal traffic, referred to as East-West traffic [20], is gaining ground.

Networks are becoming more and more complex: they often contain multiple firewalls, which protect multiple network segments. The rule base of those firewalls (i.e., the definitions of which traffic is allowed or not) is becoming equally complex, up to the point where it becomes almost unmanageable. In a survey organized by Firemon [13], 73 % of survey participants stated that their firewall ranges from “somewhat complex” to “out of control”. Further, complexity is the challenge which was ranked the highest for firewall management [14] [15].

The firewall rule base is a classic example of a system that needs to evolve over time. It starts with one firewall, and two network segments and filtering rules between them. As the network grows, the number of resources connected to the network grows, the number of services offered on the network grows and the number of security threats grows. The resulting firewall rule base will enlarge dramatically. This evolution will at some point result in a rule base where normal changes (i.e., the addition of a rule or the removal of a rule) result in unforeseen side effects. Those effects are proportional to the size of the rule base: the bigger the system (rule base), the worse it gets [14].

Normalized Systems (NS) theory [23]–[27] studies combinatorics in modular systems and provides a set of theorems to design modular systems exhibiting ex-ante proven evolvability. Here, the goal is to avoid so-called combinatorial effects (CE). CE’s are impacts which are proportional to the type of change as well as the size of the system to which the change is applied. When all modules of a system respect the NS theorems, the system will be free of such CE’s. At that point, the system can be considered stable under change with respect to a set of anticipated changes (such as adding and removing components from the system).

There are multiple vendors which sell tools to analyze a firewall rule base and can even be used to simplify it (e.g., Firemon, Tufin, Algosec). Some academic research on such analyses is available as well. Both industry and academics seem to focus on improving existing rule bases. However, a more ambitious objective would be to avoid this type of problems upfront through the deliberate design of the rule base and incorporate evolvability by design.

This paper will study the combinatorics involved in the firewall rule base. We will propose an artifact (a method), which translates the general NS theorems into a set of firewall rule base principles. When applied, this will result in an ex-ante proven evolvable (free of CE) rule base with respect to the addition and removal of rules to the firewall rule base.

The paper uses a Design Science approach [28] [29]. Therefore, Section 2 starts with explaining some firewall basics and explains the evolvability issues of a firewall rule base. In Section 3, the artifact goals and design are described. Different changes will be applied on a rule base created with the artifact to demonstrate the evolvability in Section 4. In Section 5, the artifact is evaluated based on the demonstration. Section 6 includes a literature review to link the newly created artifact with existing work, and points out the weaknesses of the artifact which also includes some suggestions for future research. Finally, our conclusions are offered in Section 7.

This paper elaborates on earlier research [27], where the applicability of NS for IT infrastructure systems was being explored. The current paper focuses on a practical case where NS and domain specific knowledge on firewalls are combined resulting in a design strategy for an evolvable firewall rule base.

II. General background and problem description

In this section, some fundamental concepts about firewalls will be explained, followed by a summary of the issues regarding the evolvability of a firewall rule base. The section continues with explaining the notion of firewall group objects, their value and related issues, and finishes with a brief explanation of the “Zero Trust” concept, which is one of the design objectives of the envisioned artifact.

A. Firewall concepts

An IPV4 TCP/IP based firewall can filter traffic between TCP/IP network connected resources based on Layer 3 (IP addresses) and 4 (TCP/UDP ports) information of those resources [21] [22]. The firewall must be in the network path between the resources. Filtering happens by making use of rules. A rule is a tuple containing the following elements: <Source IP, Destination IP, Destination Port, Protocol, Action>. IP stands for IP address and is a 32 bit number which uniquely identifies a networked resource on a TCP/IP based network. The rule is evaluated by the firewall, meaning that when it sees traffic coming from a resource with IP address =<Source IP>, going to resource =<Destination IP>, addressing a service listening on Port = <Destination port>, using Protocol = <Protocol>, then the firewall will perform an action = <Action>. The action can be “Allow” or “Deny”.

A firewall rule base is a collection of order-sensitive rules. The firewall will evaluate all inbound traffic against the ordered rule base, starting at the top of the rule base untill it encounters the first rule that matches the criteria (Source, Destination, Destination Port, Protocol) of the traffic, and perform the action as specified in the rule it hits. In a firewall rule, <Source IP>, <Destination IP>, <Destination Port> and <Protocol> can be one value or a range of values. Protocol can be TCP or UDP. In the remainder of this document, the notion of protocol is omitted as it can be included in the Port variable (for example TCP port 58 or UDP port 58).

B. Firewall evolvability issues

A typical firewall rule base might become complex over time and consist of many different rules that can have different types of relations with regard to each other. In [2], the following relations are defined between rules:

- **Disjoint:** Two rules **R1** and **R2** are disjoint, if they have at least one criterion (source, destination, port, action) for which they have completely disjoint values (= no overlap or match).
- **Exactly Matching:** Two rules **R1** and **R2** are exactly matched, if each criterion (source, destination, port, action) of the rules match exactly.
- **Inclusively Matching:** A rule **R1** is a subset, or inclusively matched to another rule **R2**, if there exists at least one criterion (source, destination, port, action) for which **R1**’s value is a subset of **R2**’s value and for the remaining attributes, **R1**’s value is equal to **R2**’s value
- Correlated: Two rules **R1** and **R2** are correlated, if **R1** and **R2** are not disjoint, but neither is a subset of the other.

Exactly matching, inclusively matching and correlated rules can result in the following firewall anomalies [2]:

- *Shadowing Anomaly*: A rule **R1** is shadowed by another rule **R2** if **R2** precedes **R1** in the policy, and **R2** can match all the packets matched by **R1**. The result is that **R1** is never activated.
- *Correlation Anomaly*: Two rules **R1** and **R2** are correlated if they have different filtering actions and **R1** matches some packets that match **R2** and **R2** matches some packets that **R1** matches.
- *Redundancy Anomaly*: A redundant rule **R1** performs the same action on the same packets as another rule **R2** so that if **R1** is removed the security policy will not be affected.

A fully consistent rule base should only contain disjoint rules, as in that case, the order of the rules in the rule base is of no importance and the anomalies described above will not occur [2] [8]–[12]). However, due to a number of reasons such as unclear requirements, a faulty change process, lack of organization, manual interventions and system complexity [13], the rule base will include correlated, exactly matching and inclusively matching rules. Combined with the order-sensitivity of the rule base, changes to the rule base (the addition or removal of a rule) can result in unforeseen side effects. To be confident that a change will not introduce unforeseen side effects, the whole rule base needs to be analyzed. Therefore, the effect of the change is proportional to the change and the size of the system, being the complete rule base. According to NS, this is a CE. As a result, a firewall rule base containing rules other than disjoint rules, is unstable under change.

C. Firewall group objects

A rule base which is made up of IP’s as Source/Destination and port numbers is difficult to interpret by humans. It is just a bunch of numbers. Modern firewalls allow the usage of firewall objects, called groups, to give a logical name to a source, a destination or a port, which is more human friendly. Groups are then populated with IP addresses or ports. Groups can be nested.

Although it should improve the manageability of the firewall, using groups can easily result in the introduction of exactly matching, inclusively matching or correlated rules.

Example:

“Group_Windows_APP” and “Group_Windows_APPS” could be two groups with each contain the IP addresses of all Windows Application Servers. The second group may have been created without knowledge of the existence of the other [13], introducing exactly matching rules. The group memberships may start to deviate from each other, introducing correlated or inclusively matching rules, which could lead to anomalies in the rule base. The group structure must be well designed to avoid this.

D. Zero Trust

In [18] [19] [20] Forrester advocates the usage of a Zero Trust model:

- Ensure all resources are accessed securely, regardless of location and hosting model,

- Adapt a “least privilege” strategy and strictly enforce access control,
- Inspect and log all traffic for suspicious activity.

The working assumption in the case of protecting network connected resources, is that all traffic towards those resources is considered a threat and must be inspected and secured. A network connected resource should only expose those services via the network which are minimally required and each network connected resource should only be allowed access to what it really needs.

III. Creating an Artifact for an Evolvable Rule Base

Based on the analysis of the problem space in the previous section, the objective is:

- To create a rule base compliant with the “Zero Trust” concept.
- To create a rule base which only contains disjoint rules.
- To create a rule base making use of firewall group objects to improve readability and manageability.
- To create a rule base which is evolvable with respect to the following anticipated changes: the addition and removal of rules.

NS will be used to structure this evolvable rule base.

In order to obtain this goal, this section starts with the investigation of the modular structure of a firewall rule base and is followed by a discussion of the issues which surface when the modular structure is instantiated. The section continues with a set of formal definitions of the firewall rule base components, from which the combinatorics are derived when creating a firewall rule base. Based on these combinatorics, the design rules for the evolvable rule base are distilled and translated into the actual artifact.

A. *Modular structure of the rule base*

A rule base is the aggregation of rules. A rule is the aggregation of: Source, Destination, Service and Action. Source is the aggregation of Clients requiring services. Destination is the aggregation of Hosts offering services. Service is the aggregation of Ports (combination of port number and protocol) which compose a service. Figure 1 represents the implicit modular data structure of a rule base in a firewall. Implicit because firewall vendors do not publish the internal data structure they use, but based on the input one needs to enter a rule in the rule base. Therefore, we assume that the model is a sufficient representation of a firewall rule base. In NS terms, the modular structure would be considered as evolvable when “Separation of Concern” is respected, as the theorems “Separation of State” and “Data and Action Version Transparency” are not relevant. As each of the mentioned modules seems to be focused on one concern, one tends to conclude that the design of a rule base can be considered as stable under change.

B. *Module instantiation*

If the modular structure of the rule base seems to be stable under change, then where does the problem of non-evolvable rule bases come from? In this respect, it is important to be aware of the fact that a firewall rule base is an order-sensitive system. More specifically, each instantiation of a rule must be given the correct place in the rule base or the rule will have an impact on existing rules (see Section II). Due to this, there are evolvability issues at the level of the instantiations of the modular structure. Indeed, it seems that —in some specific situations— certain evolvability issues of a modular structure only show up at runtime. Therefore, it is interesting to look at the application of the NS theorems at this instantiation level as well. In the context of this research, this would mean that we need to look whether the addition or removal of instantiations (of rules) can result in CE’s and thus evolvability issues making an operational system unmanageable. Here, eliminating the order-sensitivity of the rule base is the key to the problem. In order to do this, the rule base can only contain disjoint rules. Disjoint rules have no coupling with other rules, and are thus compliant with the “Separation of Concern” theorem of NS.

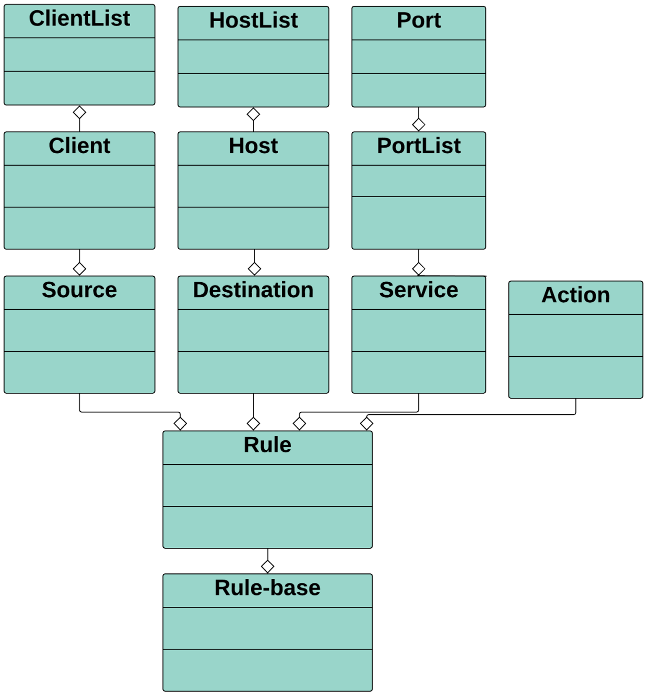


Figure 1. Modular Structure of a rule base

C. *Formal definitions of rule base components*

Let **N** represent a Layer 4 TCP/IP based network, in which 2 groups of network connected resources can be defined:

- The hosts, providing network services via TCP/IP ports.
- The clients, requiring access to the services offered by the host.

The network contains a firewall with configuration **F**, which is configured in a way that only certain clients have access to certain services on certain hosts. The “Zero Trust” principle should be applied, meaning that clients have only access to those services on hosts to which they have been given explicit access.

Let **Port** represent a Layer 4 TCP/IP defined port.

- Port.name = the name of the port.
- Port.protocol = the layer 4 TCP/IP protocol, being one of the following two values: TCP or UDP.
- Port.number = the number of the port, represented as an integer ranging from 1 to 2^{16}.

Let **P** represent the list of **Ports**, of length = pj .

- **P**[1] ... **P**[pj].
- **P**[j] contains a **Port**.
- $1 \leq j \leq$ pj.

Let **Service** represent a network service accessible via a list of layer 4 TCP/IP ports.

- **Service**.name = name of the service.
- **Service**.ports = list of ports = **P**.

Let **S** represent a list of **Services**, of length = sj.

- **S**[1] ... **S**[sj].
- **S**[i] contains a **Service**.
- $1 \leq i \leq$ sj.

Let **Host** represent a network host which provides services.

- **Host**.name = the Fully Qualified Domain Name (FQDN) of the network host.
- **Host**.IP = the IP address of the network host.

Let **H** represent a list of **Hosts**, of length = hj. The length of **H** is a function of the network **N**.

- **H**[1] ... **H**[hj].
- **H**[k] contains a **Host**.
- $1 \leq k \leq$ hj.
- hj = f_h(**N**)

Let **Client** represent a network client that requires access to hosted services.

- **Client**.name = the FQDN of the network client.
- **Client**.IP = the IP address of the network client.

Let **C** represent a list of **Clients**, of length = cj. The length of **C** is a function of the network **N**.

- **C**[1] ... **C**[cj].
- **C**[l] contains a **Client**.
- $1 \leq l \leq$ cj.
- cj = f_c(**N**)

Let **R** represent a firewall rule.

- **R**.Source = a list of Clients **Cs** of length = csj, where
  - $1 \leq$ csj$\leq$ cj
  - **Cs** $\subset$ **C**
- **R**.Destination = a list of Hosts **Hd** of length = hdj, where
  - $1 \leq$ hdj$\leq$ hj.
  - **Hd** $\subset$ **H**.
- **R**.Ports = a list of Ports = a Service **Sp**
  - where **Sp** $\in$ **S**$[sj]$.
- **R**.Action = either “Allow” of “Deny”.

Let **F**, representing a list of rules **R** of length = fj, be the ordered firewall rule base **F**

- **F**[1] ... **F**[fj]
- **F**[m] contains a firewall rule **R**
- $1 \leq m \leq$ fj

F is order-sensitive. If **R**x is a firewall rule at location y in **F**, then the behavior of the firewall can be different if **Rx** is located at position z instead of y, where z:1$\rightarrow$ fj and z $\neq$ y. Whether or not the behavior is different depends on the relation **R**x has with the other rules of **F**.

D. Combinatorics

1) Ports: Port numbers are represented by 16 bit binary number and thus go from 1 to 2^{16}. Assuming that only TCP and UDP protocols are considered for OSI Layer 4 filtering, the possible number of values for Ports is equal to $2.2^{16} = 2^{17}$.

2) Services: **S** is the list of all possible services delivered via all ports exposed on the network **N**.
S$_{\text{max}}$ is the largest possible list of services, with length = sj$_{\text{max}}$, in which all possible combinations of possible **Ports** are being used, where

$$sj_{\max} = \sum_{k=1}^{2^{17}} \binom{2^{17}}{k} \tag{1}$$

3) Hosts: The size of the list **H**, hj, is function of the network **N** and expressed as hj = f_h(**N**).
H$_{\text{max}}$ is the list of all possible lists of hosts which are part of **H**. The length of this list is hj$_{\text{max}}$, where

$$hj_{\max} = \sum_{a=1}^{hj} \binom{hj}{a} \tag{2}$$

and where hj = f_h(**N**).

4) Services on Host: The maximum number of Hosts/Services combinations = hj$_{\text{max}}$.sj$_{\text{max}}$ =

$$hj_{\max}.sj_{\max} = \left(\sum_{a=1}^{hj} \binom{hj}{a}\right) \cdot \left(\sum_{k=1}^{2^{17}} \binom{2^{17}}{k}\right) \tag{3}$$

where hj = f_h(**N**).

5) Clients: The size of the list **C**, cj, is a function of the network **N**. and expressed as cj = f_c(**N**).
C$_{\text{max}}$ is the list of all possible lists of clients which are part of **C**. The length of this list is cj$_{\text{max}}$ where

$$cj_{\max} = \sum_{a=1}^{cj} \binom{cj}{a} \tag{4}$$

where cj = f_c(**N**).

6) Rules and rule base: In a rule **R**,

- **R**.Source can contain any element of **C$_{\text{max}}$**.
- **R**.Destination can contain any element of **H$_{\text{max}}$**.
- **R**.Ports can contain any element of **S$_{\text{max}}$**.
- **R**.Action is the maximum number of action combinations, which is 2 (“Allow” or “Deny”)

The firewall rule base **F$_{\text{max}}$** contains all possible rules which can be made with **C$_{\text{max}}$**, **H$_{\text{max}}$** and **S$_{\text{max}}$**

$$fj_{\max} = cj_{\max}.hj_{\max}.sj_{\max}.2 \tag{5}$$

$$fj_{\max} = 2.\left(\sum_{a=1}^{cj} \binom{cj}{a}\right) \cdot \left(\sum_{a=1}^{hj} \binom{hj}{a}\right) \cdot \left(\sum_{k=1}^{2^{17}} \binom{2^{17}}{k}\right) \tag{6}$$

where cj = f_c(**N**) and hj = f_h(**N**)

The possible design space for a rule base is phenomenal. Multiple rules can deliver one particular required functionality. Choosing the right rule is a real challenge. As the network grows and f_c(**N**) and f_h(**N**) grow, choosing the right firewall rule from the design space becomes even more challenging. To gain control over the design space, it needs to be consciously reduced.

E. Designing an evolvable rule base

A rule will be made up of:

- **Cs** representing the Source, where $\mathbf{Cs} \subset \mathbf{C}$.
- **Hd** representing the Destination, where $\mathbf{Hd} \subset \mathbf{H}$.
- **Sp** representing the Ports, where $\mathbf{Sp} \in \mathbf{S}$.
- Action is to be “Allow” as each rule in the rule base explicitly provides access to allowed services on allowed hosts.
- **R** = (**Cs**, **Hd**, **Sp**, “Allow’)

Note that the last rule in the rule base **F**, **F**[fj] has to be the default deny rule ($\mathbf{R}_{\text{default\_deny}}$) as, when no rule explicitly provides access to a service on a host, the traffic needs to be explicitly blocked.

- $\mathbf{R}_{\text{default\_deny}}$.Source = ANY,
- $\mathbf{R}_{\text{default\_deny}}$.Destination=ANY,
- $\mathbf{R}_{\text{default\_deny}}$.Port= ANY,
- $\mathbf{R}_{\text{default\_deny}}$.Action = “Deny”.

From Section II-B, it is known that:

- A Firewall rule base is order-sensitive.
- Different types of relations/coupling can exist between rules.
- If all rules are disjoint from each other, there is no coupling between the rules.
- If all rules are disjoint, the rule base is no longer order-sensitive.
- If a new rule which is added to the rule base is disjoint with all existing rules, the location of the rule in the rule base is not important.

If the whole firewall rule base needs to be checked to see if the a rule is disjoint to all existing rules, a CE is being introduced. Introducing a new rule to, or removing a rule from the system should result in work which is proportional to the newly required functionality and not into work which has no logical link to the required functionality and which requires searching throughout the whole system (being the entire rule base). Or as NS formulates it: the impact of the change should be proportional to the nature of the change itself, and not proportional to the system to which the change is applied.

Disjoint rules have no overlap in source or destination or ports. This gives the following combinations:

- No overlap in sources - don’t care about destination and port overlaps.
- No overlap in destinations - don’t care about source and port overlaps.
- No overlap in ports - don’t care about source and destination overlap.
- No overlap in source-destination combination, don’t care about ports.
- No overlap in source-ports combinations, don’t care about destinations.
- No overlap in destination-ports combinations, don’t care about sources.
- No overlap in source-destination-port combination.

Cs is $f_c(\mathbf{N})$ and **Hd** is $f_h(\mathbf{N})$. The network is an uncontrollable variable. Trying to find a way to structure **Cs** and **Hd** to allow for disjoint rules starting from this variable, will not yield to anything useful. On the other hand, **S** represents the ports and is bound: the nature of TCP/IP limits the amount of possible ports and thus all port combinations. It thus makes sense to look for a way to guarantee that there is no overlap at port/service level.

Let us consciously restrict **S** to **Su**, so that **Su** only contains unique values.

$\exists!\mathbf{Su}[m]$ in **Su**.

$\mathbf{Su}[u] \cap \mathbf{Su}[v] = \emptyset$, where u, v:1→suj, and $u \neq v$

If each service is represented by 1 port, **Su** will contain 2^{17} elements, which is the max size of **Su** in this restricted case.

The service **Su**[m] can be delivered by many hosts.

Let $\mathbf{Hd}_{\mathbf{Su}[m]}$ represent the list of hosts which offer service **Su**[m].

$\mathbf{Hd}_{\mathbf{Su}[m]} \subset \mathbf{Hd}$ and $\mathbf{Hd}_{\mathbf{Su}[m]}[x]$ contains a single host.

$\mathbf{Hd}_{\mathbf{S}[m]}$ contains unique and disjoint elements.

Combining hosts and services gives, ($\mathbf{Hd}_{\mathbf{Su}[m]}[x]$,**Su**[m]) where x:1→hdmj, a list of tuples which are disjoint. This hold for all m:1→hdmj. At his point, all services and hosts who deliver the services, form tuples which are disjoint and can thus be used as a basis for creating an order independent firewall rule base. $\mathbf{Cs}_{\mathbf{Hd}_{\mathbf{Su}[m]}[x]}$ is the list of clients which have access to service **Su**[m], defined on host $\mathbf{Hd}_{\mathbf{Su}[m]}[x]$.

By using :

- **Su**[m] where m:1→suj, with suj=number of disjoint services offered on the network, for defining **R**.Port
- $\mathbf{Hd}_{\mathbf{Su}[m]}[x]$, x:→hdmj, with hdmj=number of hosts offering **Su**[m], for defining **R**.Destination
- $\mathbf{Cs}_{\mathbf{Hd}_{\mathbf{Su}[m]}[x]}$ being the list of clients requiring access to service **Su**[m] on host $\mathbf{Hd}_{\mathbf{Su}[m]}[x]$, of length = cjs, for defining **R**.Source
- “Allow”, for **R**.action

disjoint rules are being created, usable for an evolvable firewall rule base.

F. The artifact

What has been discussed in the previous section needs to be transformed into a solution usable in a real firewall. As discussed in Section II-C, firewalls work with groups. Groups can be used to represent the concepts discussed in the previous sections.

1) Starting from an empty firewall rule base **F**. Add as first rule the default deny rule **F**[1]= $\mathbf{R}_{\text{default\_deny}}$ with
    - $\mathbf{R}_{\text{default\_deny}}$.Source = ANY,
    - $\mathbf{R}_{\text{default\_deny}}$.Destination=ANY,
    - $\mathbf{R}_{\text{default\_deny}}$.Port= ANY,
    - $\mathbf{R}_{\text{default\_deny}}$.Action = “Deny”.
2) For each service offered on the network, create a group. All service groups need to be completely disjoint from each other: the intersection between groups must be empty.
    Naming convention to follow:
    - **S**_*service.name*,
    - with *service.name* as the name of the service.
3) For each host offering the service defined in the previous step, a group must be created containing only one item (being the host offering that specific service).
    Naming convention to follow:
    - **H**_*host.name*_**S**_*service.name*,
    - with *host.name* as the name of the host offering the service

4) For each host offering the service from the first step, a client group must be created. That group will contain all clients requiring access to the specific service on the specific host.
   Naming convention to follow:
   - **C_H**_*host.name*_**S**_*service.name*
5) For each **S**_*service.name*,**H**_*host.name*_**S**_*service.name* combination, create a rule **R** with:
   - **R**.Source =**C_H**_*host.name*_**S**_*service.name*
   - **R**.Destination = **H**_*host.name*_**S**_*service.name*
   - **R**.Port= **S**_*service.name*
   - **R**.Action = “Allow”

   Add those rules to the firewall rule base **F**.
   The default rule $\mathbf{R}_{\text{default}}$ should always be at the end of the rule base.

By using the artifact’s design principles, group objects are created which form the building blocks for an evolvable rule base. Each building brocks addresses one concern.
If each service of **Su** is made up of only one Port, then the **Su** will contain maximum 2^{17} elements, resulting in maximum 2^{17} service groups **S**_*service.name* being created. For each host, maximum 2^{17} services can be defined, expressed in **H**_*host.name*_**S**_*service.name* destination groups. According to the artifact, one rule per host and per service, must be created. This reduced the rule base solution space from

$$2.\left(\sum_{a=1}^{cj}\binom{cj}{a}\right)\cdot\left(\sum_{a=1}^{hj}\binom{hj}{a}\right)\cdot\left(\sum_{k=1}^{2^{17}}\binom{2^{17}}{k}\right) \tag{7}$$

where cj = fc(**N**) and hj = fh(**N**)
to:

$$fj = hdj.suj = hdj.2^{17} + 1 \tag{8}$$

with hdj = number of hosts connected to the network.
hdj = fh(**N**). The “+1” is the default deny rule $\mathbf{R}_{\text{default\_deny}}$

IV. Demonstrate artifact

To demonstrate the artifact, we will investigate the effect of different changes on the rule base (add/remove rule) and the components making up the rule base (add/remove a service, add/remove a host, add/remove a client). We also show what happens if rules would be aggregated.

A. Add and remove a rule

Creating rules according to the artifact’s design principles, leads to rules which are disjoint from each other. Disjoint rules can be added and removed from the firewall rule base without introducing CE’s.

B. Adding a new service to the network

A new service is a service which is not already defined in **Su**. The new services results in a new definition of a service being added to **Su**. The artifact prescribes that a new group **S**_*service.name* must be created for the new service. The group will contain the ports required for the service. For each new host offering the service, the artifact prescribes to create a new group destination **H**_*host.name*_**S**_*service.name*, and an associated source group **C_H**_*host.name*_**S**_*service.name*. The destination groups are populated with only one host (the host offering the service), and the source groups are populated with all clients requiring access to the service on specific host. All building blocks to create the disjoint rules are now available. For each host offering the new service, a rule must be created using the created groups. No CE’s are being introduced during these operations. Adding the new rules to the rule base does not introduce CE’s (see Section IV-A).

C. Adding a new host offering existing services, to the network

A new host is a host which is not already defined in **Hd**. The new host results in a new host definition being added to **Hd**. The artifact prescribes that a new group **H**_*host.name*_**S**_*service.name* must be created for each service delivered by the host and a corresponding source group **C_H**_*hostṅame*_**S**_*service.name* must be created as well. The destination groups are populated by their corresponding host and the sources groups are populated with all clients requiring access to the service on that host. All building blocks to create the disjoint rules are now available. For each service offered by the new host, a rule must be created using the created groups. No CE’s are being introduced during these operations. Adding the new rules to the rule base does not introduce CE’s (see sectionIV-A).

D. Adding a new host offering new services, to the network

Combining Sections IV-C and IV-B delivers what is required to complete this type of change. The artifact prescribes that new service groups must be created for new services. An equal amount of destination groups needs to be created and each populated by the new host. An equal amount of source groups needs to be created and populated by the clients requiring access to one of the new services on the new host. All building blocks to create the disjoint rules are now available. For each combination (new host, new service) a rule must be created using the created groups. No CE’s are being introduced during these operations. Adding the new rules to the rule base does not introduce CE’s (see Section IV-A).

E. Adding a new client to the network

Adding a new client to the network does not require the creation of new rule building blocks or the addition of new rules. The new client only requires to be added to those source groups which give access to the services it requires on the hosts it needs access to. No CE’s are being introduced during these operations.

F. Removing a service from the network

Let **sr** be the service which needs to be removed from the network. The name of the service is **sr**.name=sremove. The service is part of **Su**. The group corresponding with **sr** is **S**_sremove. The hosts offering the service correspond with the groups **H**_*host.name*_**S**_sremove. The clients consuming the service are defined in **C_H**_*host.name*_**S**_sremove. All building blocks to identify the rules which require removing from the rule base are now available. For each host offering **sr**, the corresponding rule

- $\mathbf{R}_{\text{default\_deny}}$.Source = **C_H**_*host.name*_**S**_sremove
- $\mathbf{R}_{\text{default\_deny}}$.Destination=**H**_*host.name*_**S**_sremove,
- $\mathbf{R}_{\text{default\_deny}}$.Port= **S**_sremove,
- $\mathbf{R}_{\text{default\_deny}}$.Action = “Allow”.

must be removed from the rule base. No CE’s are being introduced during these operations. Removing rules from the rule base does not introduce CE’s (see Section IV-A). The service **sr** needs to be removed from **Su** as well as the corresponding group **S**_remove in the firewall.

G. Removing a host from the network

Let **hr** be the host that needs to be removed from the network. The name of the host is **hr**.name=hremove. The host is part of **Hd**. There will be as much destination groups for **hr** as there are services offered by **hr**. They are defined by **H**_hremove_**S**_*service_name*. The same holds form the source groups, defined by **C_H**_hremove_**S**_*service.name*. All building blocks to identify the rules which require removal from the rule base are available. For each service offered by **hr**, the corresponding rule

- $\mathbf{R}_{\text{default\_deny}}$.Source = **C_H**_hremove_**S**_*service.name*
- $\mathbf{R}_{\text{default\_deny}}$.Destination=**H**_hremove_**S**_*service_name*
- $\mathbf{R}_{\text{default\_deny}}$.Port= **S**_*service.name*,
- $\mathbf{R}_{\text{default\_deny}}$.Action = “Allow”.

must be removed from the rule base. No CE’s are being introduced during these operations. Removing rules from the rule base does not introduce CE’s (see Section IV-A). The host **hr** needs to be removed from **Hd** and the corresponding groups **H**_remove_**S**_*service.name* in the firewall, must be removed as well.

H. Removing a service from a host

Let **sr** be the services with **sr**.name=sremove, which needs removing from host **hr** with **hr**.name = hremove. The service is part of **Su**. The group corresponding with **sr** is **S**_sremove. The destination group for service **sr** on host **hr**, is **H**_hremove_**S**_sremove. The corresponding source group is **C_H**_hremove_**S**_sremove. All building blocks to identify the rule

- $\mathbf{R}_{\text{default\_deny}}$.Source = **C_H**_hremove_**S**_sremove
- $\mathbf{R}_{\text{default\_deny}}$.Destination=**H**_hremove_**S**_sremove
- $\mathbf{R}_{\text{default\_deny}}$.Port= **S**_sremove,
- $\mathbf{R}_{\text{default\_deny}}$.Action = “Allow”.

which require removing from the rule base are available. No CE’s are being introduced during these operations. Removing rules from the rule base does not introduce CE’s (see Section IV-A). The service **sr** does not need to be removed from **Su** and neither does the corresponding group as the service is till offered on other hosts.

I. Removing a client from the network

Let **cr** be a client that needs to be removed from the network. The client is part of **Cs**. Removing a client from the network does not require removing rules from the rule base. The client needs to be removed from the different source groups which provide the client access to specific services on specific hosts. If the services and hosts to which the client has access are known, then the source group from which the client needs to be removed, are known as well. If the services and/or hosts are not known, then an investigation of all the source groups is required to see if the client is part of the group or not. If part of the group, the client needs to be removed. The client also needs to be removed from **Cs**. Determining if a client is part of a source group can be considered as a CE as all source groups require inspection. This will further be elaborated upon in Section VI.

J. The impact of aggregations

When following the prescriptions of the artifact, many groups and rules will be created (see Section V for more details). The urge to aggregate and consolidate rules into more general rules, will be a natural inclination of firewall administrators as a smaller rule base will be (wrongfully) considered as a less complex rule base. However, any form of aggregation will result in loss of information. It is because the artifact consciously enforces fine-grained information in the group naming and usages, that disjoint rules can be created and the “Zero Trust” model can be enforced. If due to aggregations it can no longer be guaranteed that rules are disjoint, then a CE-free rule base can no longer be guaranteed either. Aggregation will also lead to violations of the “Zero Trust” model.

We provide 2 examples of aggregations.

Aggregation at service level: all hosts offering the same service are aggregated into one destination group. Such an aggregation excludes the possibility of specifying that a client needs access to a specific service on a specific host. A client will have access to the service on all hosts offering the service, desired or not. In such a configuration, “Zero Trust” can no longer be guaranteed. As long as the services on the network are unique, so will be the port groups. Rules will stay disjoint and the rule base CE-free. The moment that one starts combining “Zero Trust” and non “Zero Trust” rules, non-disjoint rule will pop-up and the rule base can no longer be guaranteed to be CE-free.
Example: if for some reason, it cannot be allowed that a client has access to the service on all hosts and a special service group is being created (no longer disjoint with the existing service group) with a special associated destination group (no longer disjoint with existing destination groups), the rule created with those groups is not disjoint with existing rules in the rule base and the effect of adding this rule to the rule base is no longer guaranteed CE-free.

Aggregation at host level: all services offered on a host, are aggregated into one host-bound port/service group. Such an aggregation excludes the possibility of specifying that a client needs access to some of the services on the host. A client will have access to all services defined on the host, desired or not. In such a configuration, “Zero Trust” can no longer be guaranteed. As long as the destination groups are unique, disjoint rules can still be created. The moment that one starts combining “Zero Trust” and non “Zero Trust” rules, non-disjoint rule will pop-up and the rule base can not longer be guaranteed CE-free.
Example: if for some reason, it cannot be allowed that a client has access to all services on the host and a special service group is being created (no longer disjoint with existing service groups) with a special associated destination group (no longer disjoint with existing destination group), the rule created with those groups is not disjoint with existing rules in the rule base and the effect of adding this rule to the rule base is no longer guaranteed CE-free.

V. Evaluation

The previous section demonstrates that, when applying the artifact, the rules are guaranteed to be disjoint and adding and removing such rules has no unwanted side effects on the existing rule base. Such a rule base will be fine-grained (i.e., having many rules). One might consider this as an important drawback from using such an approach as the idea of a fine-grained structure is often considered as a complex one (but this does not necessarily has to be the case). Some operations on rules may indeed result in CE's at group level, such as adding and removing a client from the network. In Section VI, this will further be elaborated upon.

Aggregations will violate the "Zero Trust" constraint. Combining aggregation and non-aggregation based rules results in non-disjoint rules and CE's at rule base level.

"Zero Trust" has been defined at host level but one can imagine a setup where "Zero Trust" is defined at another level such as VLAN's. A VLAN is a section of the network with a continuous IP range (for example 10.10.10.1 till 10.10.10.254). In such a setup, instead of "specific service on a specific host", the "Zero Trust" could become a "specific service on a specific VLAN". If in Section III, **H** would be replaced by **V**, the list of VLAN's, and the content of **V** could be used to create destination groups **V**_vlan.name_**S**_service.name with source groups **C**_**V**_vlan.name_**S**_service.name, the building blocks would be created for disjoint rules respecting a "Zero Trust" at VLAN level model. The artifact would still be applicable and the resulting rule base would be smaller (and less fine grained). An evolvable rule base is possible as long as one sticks to a defined level of "Zero Trust" and one does not start mixing up different levels of "Zero Trust". From the moment one starts mixing different levels, evolvability can no longer guaranteed.

VI. Discussion

By means of discussion, we will cover three items. First, we will provide a short literature review to position our research with respect to earlier contributions found in literature. Second, we will reflect on the nature of the CE's which are still present when applying our proposed artifact. And third, we will highlight some opportunities for future research.

A. Literature review

The academic literature about firewalls can be divided into 3 groups. The first group (published roughly before the year 2000)focuses on the performance of the firewall and the hardware used to perform the actual package filtering. The second group (published roughly between 2000 and 2006) focuses on the complexity and issues with the rule base of the firewall. The third group (published roughly after 2006) focuses on the firewall in a Software Define Network (SDN) context, where distributed firewalls and software defined firewalls are used. As this paper focuses on the complexity and issues related to the firewall rule base, the following literature review will only focus on the second group of papers [1]–[12]. Next to academic papers, reports from Forrester and white papers from industry leaders were used as well [13]–[20]. Those reports include surveys which give information on the current state-of-affairs. One might think that, because academic publication about rule base issues have diminished after 2006, the problem is solved. However, the surveys provide a different view. Companies are still struggling with their firewall [13]–[20]. This can be due to the "knowing-doing" gap or because the issue is not fully resolved.

Most papers start by stating that there is a problem with the firewall rule base because of:

- ***Translation issues***: how to convert a high level security policy into a low-level language of firewall rules [1]–[12] [17].
- ***Size of the rule base issues***: a large rule base is considered complex [3] [7] [9] [10] [13].
- ***Error and anomalies issues***: A rule base is error-prone due to complexity and manual interventions [2] [3] [10], [13]–[20] and can contain firewall rule conflicts or anomalies [1] [2] , [3] [6] [8]–[10] [12] [13] [16].

The ***"Translation-issue"*** is tackled by proposing tools which could translate high level security concepts into low level firewall rules. FANG [6], FIRMATO [3], LUMETA [5] are artifacts proposed and described, which help translating high level security requirements into a low level firewall rule base. There are however no guarantees that these tools deliver a small and simple firewall rule base free of anamolies [3]. Companies such as TUFIN, ALGOSEC, FIREMON, VMWare also deliver commercial tools which claim to help managing the complexity of network security. The tools do not prescribe, neither enforce how a rule base should be created in order to be free of anomalies and exhibit evolvability.

The ***"Size of the rule base issue"*** is not treated as an issue related to the stability of a system under change. To the best of our knowledge, most contributions do not focus on this point, whereas it is a corner stone of NS. The different artifacts all start with ideas similar to "For each rule in the firewall, do the following ...". One might consider such an approach as a CE in itself. There is attention to reducing the rule base to a minimum list of rules, which still answer to the filtering requirements. The motivation for this "reduction of the rule base" is performance, although in [3] it was suggested that the actual size of the rule base is not related to the way in which the hardware actually applies the rules. This suggests a decorrelation between the size of the rule base and the firewall performance. If this would be the case, why bother about the reduction of the size of the rule base? Further research of the literature and real-life measurements are required to clarify this point.

Looking at the combinatorics of Section III-D, the design space is enormous. By applying the artifact, there is a conscious reduction of the design space. But the size of the rule base is still large as for each combination (host, service) a rule must be created in the rule base.

$$2^{17}.hdj \tag{9}$$

The maximum number of services is 2^{17} = 131.075. However, in reality this number will never be reached. A sample in Engie (a multinational and world leader in energy services) on 100 servers revealed that on average 39 services are exposed. The standard deviation in the sample is 14. It can be stated with a statistical probability of 98% that a host exposes less then 67 services. The sample was taken from a population of 1000 servers. Those 1000 serves are currently protected by about 890 firewall rules. If the artifact would be applied, it would mean implementing 67.000 rules. However, at Engie, a "Zero Trust" model at host level is not applied. Instead, a "Zero Trust" at VLAN level (see Section V) is present. If the realistic assumption is a made that the 1000 server are spread over 20 VLAN's, it would mean that 20 x 67 = 1340 rules are required for an evolvability rule base. This would mean 50% more rules to gain full evolvability.

Applying the artifact will probably imply a larger rule base and there may be worries about performance. It should be noted that a firewall vendor such as CheckPoint, suggests to put the rules which are most frequently hit (and applied) at the top of the firewall table. In a rule base which is order-sensitive, this may be a real issue. In a rule base which is not order-sensitive, one could monitor the firewall and see which rules are hit most and move those rules around without having to worry of the potential impact on other rules. Doing this dynamically would even be more powerful as the firewall would be able to reorganize his rules according to the traffic of the day.

The ***"Error issue"*** due to complexity and manual intervention is recognized and confirmed in recent surveys [13]–[20]. The academic papers focus more on the technical root causes of the errors, being the anomalies in the rule base. Over time, the definitions of the types of anomalies, their formal definition and proof, have evolved and resulted in a definition of how a firewall rule base should look like in order to remain stable under change: a firewall rule base should only include disjoint rules ([2], [8], [9], [10], [11], [12]). Artifacts have been put forward [2] [3], [7]–[9], [12] which allow to scan the rule base for non-disjoint rules and make them disjoint if required. The same artifacts allow to assess the impact of adding a new rule and adjusting the rules in such way that the rule base only contains disjoint rules. However, each time a rule is entered, the whole rule base needs to be scanned to detect potential anomalies between the existing rule base and the new rule. The effort of making a change to the system is thus proportional to the size of the system.

The literature review shows that the problems related to the firewall rule base are well known and the necessary condition to keep the rule base under control (i.e., having disjoint rules) is also known. However, clear architectural guidance on how to create a disjoint rule base as of the moment of conception, is lacking. It is exactly this architectural guidance, making use of NS, which is the main contribution of this paper. By structuring the rules in such a way that they are always disjoint, one can add and remove rules without having to analyze the rule base (source of CE) or worry about unforeseen side effects of the change.

B. Remaining CE's

The artifact proposed in the paper is not completely free of CE. The evaluation has shown that there is are CE's at the level of groups. However, these CE's are not related to the technical coupling within the rule base but due to the size and topology of the network. The bigger the network, the more objects and rules. Such CE's are considered acceptable given that:

- The actions leading to the CE can be automated (search for, or through, groups)
- The CE is predictable and is the logical effect of the change which needs to be applied (remove a client = look in all groups where the client is present)

CE's which cannot be automated because their impact is not predictable are not acceptable as there is no logical link between the change and the extra work one needs to do to implement the change. For example, the addition of a rule to activate a service on the network that would require the inspection of the whole rule base to find conflicting rules (not related to the newly activated service) would be considered as an unacceptable CE. Remark that the proposed artifact facilitates the removal of such unacceptable CE's.

C. Future research

Applying the artifact to create the rule base in a manual way, is highly inadvisable. The risk of introducing errors is too large. NS at the software level advocates the use of expanders, which will automate the creation of a skeleton code structure which has ex-ante proven evolvability. A similar expander should be built for the proposed artifact. The expander would enforce the application of the naming conventions and the disjoint rules, thus automatically enforcing the architectural principles. Future efforts could be invested in building a tool which would enforce the artifact on a firewall rule base. Like many of the tools discussed in literature, creating a system (with the desired configuration) which is positioned in parallel to the existing firewalls (which can later on be pushed towards the actual production firewall) is the most likely approach to apply.

The artifact currently focuses on a single firewall implementation. The same concept could now be investigated in a context where the network has multiple firewalls. How would this affect the rule base and which part of the rule base goes to which firewall? One could even consider a distributed firewall in a software defined setup. For this purpose, the concepts behind the proposed artifact should be extended towards distributed firewalls and software defined firewalls.

Because the consulted literature did not contain conclusive information on the impact of the size of the rule base on firewall performance, further literature review on this topic is required.

VII. Conclusion

Firewall rule bases are typically non-evolvable systems. Tools and literature exist on how to show and potentially reduce the complexity and conflicts in firewall rule bases, but practical guidance on how to make a rule base which has proven evolvability by design, is lacking. Using the NS paradigm and domain specific knowledge, we have proposed an artifact which has the desired evolvability for a firewall rule base. The most important drawback of the resulting rule base could be the size due to its fine-grained structure, although this should be further analyzed in future research efforts.

Acknowledgment

The authors would like to thank Prof. Dr. Herwig Mannaert for his guidance, Stefan Thys, Frederik Leemans, Stefan Biesbroeck for their extra input, feedback and editing, Sam Gozin and Bruno De Becker for the Engie operational data.

References

[1] P. Eronen and J. Zitting, "An expert system for analysing firewall rules", In Proceedings of the 6th Nordic Workshop on Secure IT Systems (NordSec 2001),pp. 100–107, November 2001.

[2] M. Abedin et al., "Detection and Resolution of Anomalies in Firewall Policy Rules", In Proceedings of the IFIP Annual Conference Data and Applications Security and Privacy, 2006, LNCS 4127, pp. 15–29

[3] Y. Bartal, A. Mayer, K. Nissim, and A. Wool, "Firmato: A novel firewall management toolkit", Proceedings of the 1999 IEEE Symposium on Security and Privacy,pp. 17-31,; Oakland, California, May 1999

[4] A. Wool, "Architecting the Lumeta firewall analyser", In Proceedings of the 10the USENIX Security Symposium, Washington DC, August 2001

[5] S. Hinrichs, "Policy-based management: Bridging the gap", In Proceedings of the 15th Annual Computer Security Applications Conference, Phoenix, Arizona, December 1999, IEEE Computer Society Press.

[6] A. Mayer, A. Wool, and E. Ziskind. "Fang: A firewall analysis engine", In Proceedings, IEEE Symposium on Security and Privacy,pp. 177-187, IEEE CS Press, May 2000

[7] S. Hazelhurst, “Algorithms for analysing firewall and router access lists”. Technical Report TR-WitsCS-1999-5, Department of Computer Science, University of the Witwatersrand, South Africa, July 1999

[8] E. Al-Shaer and H. Hamed, “Design and Implementation of firewall policy advisor tools”, Technical Report CTI-techrep0801, School of Computer Science Telecommunications and Information Systems, DePaul University, August 2002

[9] E. Al-Shaer and H. Hamed, “Discovery of policy anomalies in distributed firewalls”, In Proceedings of the 23rd Conf. IEEEE Communications Soc. (INFOCOM 2004), Vol 23, No.1,pp. 2605-2616, March 2004

[10] E. Al-Shaer and H. Hamed, “Taxonomy of conflicts in network security policies”, IEEE Communications Magazine, 44(3), March 2006

[11] E. Al-Shaer, H. Hamed, R. Boutaba, and M. Hasan., “Conflict classification and analysis of distributed firewall policies”, IEEE Journal on Selected Areas in Communications (JSAC), 23(10), October 2005

[12] A. Hari, S. Suri, and G.M. Parulkar., “Detecting and resolving packet filter conflicts”, In INFOCOM (3),pp. 1203-1212, March 2000.

[13] Firemon whitepaper, Firewall cleanup recommendations, URL https://www.firemon.com/resources/, [retrieved: April, 2019]

[14] Firemon whitepaper, 2017 State of the firewall , URL https://www.firemon.com/resources/, [retrieved: April, 2019]

[15] Firemon whitepaper, 2018 State of the firewall , URL https://www.firemon.com/resources/, [retrieved: April, 2019]

[16] Algosec whitepaper, Firewall Management: 5 challenges every company must address, URL https://www.algosec.com/resources/ [retrieved: April, 2019]

[17] D. Monahan EMA, Research Summary: “Network Security Policy Management tools – Tying Policies to Process, Visibility, Connectivity and Migration”, https://web.tufin.com/network-security-policy-management-tools-ema-research, [retrieved: April, 2019]

[18] H. Shel and A. Spiliotes, “The State of Network Security: 2017 to 2018”, Forrester Research November 2017

[19] C. Cunningham and J.Pollard, “The Eight Business and Security Benefits of Zero Trust”, Forrester Reseach November 2017

[20] M. Bennet, “Zero Trust Security: A CIO’s Guide to Defending Their Business From Cyberattacks”, Forrester Research June 2017

[21] W.R. Stevens, TCP/IP Illustrated, Volume 1, the Protocols, Addison-Wesley Publishing Company, ISBN 0-201-63346-9, 1994

[22] H. Zimmermann, and J.D. Day, “The OSI reference model - Proceedings of the IEEE”, Volume: 71, Issue: 12, Dec 1983

[23] H. Mannaert, J. Verelst, and P. De Bruyn, Normalized Systems Theory: From Foundations for Evolvable Software Toward a General Theory for Evolvable Design, ISBN 978-90-77160-09-1, 2016

[24] H. Mannaert, J. Verelst, and K. Ven, “The transformation of requirements into software primitives: Studying evolvability based on systems theoretic stability”, Science of Computer Programming : Volume 76, Issue 12 pp. 1210 to 1222, 2011

[25] H. Mannaert, J. Verelst, and K. Ven, “Towards evolvable software architectures based on systems theoretic stability”, Software Practice and Experience : Volume 42, Issue 1, 2012

[26] P. Huysmans, G. Oorts, P. De Bruyn, H. Mannaert, and J. Verelst.- “Positioning the normalized systems theory in a design theory framework”, Lecture notes in business information processing, ISSN 1865-1348 - 142, pp. 43-63, 2013

[27] G. Haerens, “Investigating the Applicability of the Normalized Systems Theory on IT Infrastructure Systems, Enterprise and Organizational Modeling and Simulation” - 14th International workshop (EOMAS) 2018, pp. 23-137, June 2018

[28] P. Johannesson, and E. Perjons, An Introduction to Design Science, ISBN 9783319106311, 2014

[29] A.R. Hevner, S.T. March, J. Park, and S. Ram, “Design Science in Information Systems Research”, MIS Quarterly: Volume 38, Issue 1 pp. 75-105, 2004

On the Modular Structure and Evolvability of Internet of Things Architectures

Tom Vermeire, Jeroen Faes, Peter De Bruyn and Jan Verelst

Department of Management Information Systems
Faculty of Business and Economics
University of Antwerp, Belgium
Email: {tom.vermeire,peter.debruyn,jan.verelst}@uantwerp.be,
jeroen.faes@student.uantwerp.be

***Abstract*—The development of the Internet of Things (IoT) is an important evolution for current businesses. The field of IoT is maturing and best practice solutions are emerging. However, as organizations are confronted with increasing and faster changes in their environment, applications using IoT should be able to adapt and evolve accordingly. This paper assesses the ability to implement changes in best practice IoT applications, using Normalized Systems Theory as a theoretical basis. Subsequently, a new architecture addressing some identified evolvability issues is proposed. In contrast to existing prescriptive work focusing on the interoperability and standardization of IoT applications, this paper evaluates design choices from the perspective of the ability to evolve.**

Keywords–Internet of Things; Evolvability; Normalized Systems Theory.

I. Introduction

Internet of Things (IoT) envisions a network of connected physical objects allowing the exchange of data. It is generally seen as a promising evolution in the current and future business landscape, with a strongly increasing impact (in terms of the number of connected devices and business spending) [1]. An increase in the number of IoT applications, their criticalness for organizations and the number of devices they are relying on, give rise to challenges regarding scalability and implementation of changes resulting from additional requirements. Consequently, it is of major importance that the design of IoT applications allows to cope with future requirements (and the modifications they imply).

This paper uses Normalized Systems Theory (NST) to assess the evolvability of design decisions concerning IoT applications. NST is a theory which provides prescriptive guidance on how to design evolvable software architectures and, more generally, modular structures. Considering IoT applications as modular structures (consisting of a set of applications, devices, etc.), we argue and demonstrate that NST can be applied in this context. It is asserted that the current best practice architecture, whereby organizations typically make use of a one-stop vendor solution for data collection, storage and processing, offers limited evolvability. Afterwards, an enhanced IoT architecture, which addresses the identified problems, is proposed. In this new architecture, the different data-related activities are separated and organizations will experience an increased flexibility to implement changes. In contrast to existing prescriptive work, mainly focusing on interoperability and standardization issues, this paper approaches the design problem of IoT applications from their potential to adapt.

The remainder of this paper is structured as follows. In Section II, an overview of related work is given. Section III explains how IoT applications can be seen as modular structures subject to change. Afterwards, Section IV presents and evaluates the current best practice IoT architecture. Section V proposes an enhanced architecture from an evolvability point of view and Section VI discusses the result. Finally, Section VII concludes and offers avenues for future research.

II. Related work

This paper focuses on IoT and its current best practice solutions, and uses NST to analyze them. Therefore, this section briefly summarizes related work regarding each of these concepts.

A. Internet of Things

The International Telecommunication Union [2] defines IoT as "*a global infrastructure for the Information Society, enabling advanced services by interconnecting (physical and virtual) things based on, existing and evolving, interoperable information and communication technologies*" (p. 1). The potential applications of IoT span many industries, including logistics, healthcare, manufacturing etc., with a varying focus on enterprises, governments and consumers [3][4] .

The first use of the term 'Internet of Things' is attributed to Kevin Ashton [5], who argued that computers were too dependent on information input from humans. Instead, he advocated a shift towards data gathering by 'things'. In order to achieve this goal, physical objects should be equipped with sensors and radio-frequency identification (RFID) technology. Although the term IoT was new then, earlier contributions demonstrated comparable ideas including a Coke machine connected to the Internet in 1982 [6] or visions on ubiquitous compution [7] in which computers would amalgamate in the environment and their actual power would originate from the connection between devices. More recently, Mattern and Floerkemeier [8] described IoT as a situation wherein the existing Internet is extended into reality through the embracement of everyday physical objects. Various IoT definitions exist having an alternating focus between the connected things, the Internet-related aspects and the semantics of the information [9]. Overall, the focus has shifted from merely identifying and monitoring physical things towards smart objects that autonomously perform computer tasks.

From a business perspective, IoT is often seen as a potential way of capturing, communicating, and processing data in more advanced ways and the ability to perform advanced analytics or provide enhanced cloud services with a vast impact on current business models [10][11]. Recently, a multitude of IoT platforms was developed allowing companies and governments to create IoT applications [12]. These platforms typically make use of the cloud to store the gathered data [4]. It is generally believed that advances in the power, size and cost of computing chips will significantly increase the number of connected objects in the following years [13].

B. IoT architectures and their challenges

Existing research on IoT architectures mainly focuses on the interoperability issues and standardization of technology.

For instance, Sethi and Sarangi [14] provide an overview of different IoT architectures. First, a layered architecture classifies the different IoT aspects on the basis of protocols (Perception, Transport, Processing, Application and Business). Second, cloud and fog based architectures take systems architectures as a starting point. Third, Social IoT attempts to mimic human social relationships in the IoT architecture. They stress the importance of middleware for data storage, analysis and processing. This middleware should abstract hardware details for programmers. In this context, they refer to a middleware platform as an appropriate solution to connect things and applications. As already stated, the use of cloud platforms within IoT architectures has become common practice.

Schmid et al. [15] also recognize the trend towards IoT platforms and argue that these platforms themselves should be interoperable in order to create IoT ecosystems comprising different industries. The BIG IoT Architecture is proposed as a solution, where IoT resources such as data or functions are offered in a standardized way on a marketplace. This marketplace offers application programming interface (API) endpoints for customers and providers and, thus, makes it possible for IoT services to operate with combined forces. Figure 1 provides a simplified overview of the BIG IoT Architecture. From left to right, a distinction is made between the customers, the market place and the providers. Although the proposed architecture consists of several other components as well, we only focus on a high-level overview in the context of this paper.

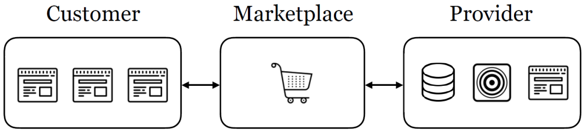


Figure 1. Simplified BIG IoT Architecture.

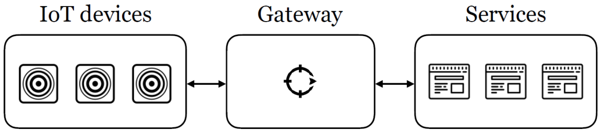


Figure 2. Simplified semantic IoT architecture.

Desai, Sheth, and Anantharam [16] see a similar challenge for IoT architectures. However, each domain can have a different standardized architecture and data model. In order to ensure the cooperation between multiple domains, standardization is required. Therefore, a semantic gateway is presented as a solution for the integration problem. The gateway aggregates annotated sensor data and connects with other physical things and (cloud) IoT services. The data annotation is realized by standard ontologies created by the semantic web community. A simplified overview of this semantic IoT architecture is shown in Figure 2. The gateway in the middle connects the IoT devices on the left side with the offered services on the right side.

Both examples provide a similar approach to encourage interoperability between different domains and their platforms: an intermediate agent ensures integration between things and IoT (cloud) services. Both approaches consider standardization as the primary challenge regarding interoperability, which the marketplace and semantic gateway each attempt to address.

Some authors within literature mention the evolvability of IoT applications as a relevant challenge. For instance, Weiser [7] pointed at the inability of existing operating systems to cope with changing hardware and software configuration. Wortmann and Flüchter [17] stated that being able to modify business models to IoT has become crucial and expressed the need for new design principles for applications to cope with updates of connected devices. Porter and Heppelmann [18] emphasized that organizations will be faced with continuously evolving IoT standards. Furthermore, the scalability of IoT applications is by many authors considered an important challenge [8][9][17]. A very specific and interesting situation was sketched by Priyantha et al. [19]. They investigated interoperable networks of sensors exposed to change and argued that current sensor-nets are not able to persist when new sensors with different protocols are added, possibly from different manufacturers. They provide two guidelines for IoT application design. First, sensors should be restricted to only generate structured data in order to be understandable for applications. Second, a programmatic description of the sensor's functionalities is prescribed. When sensors can be accessed in a structured way and programmatically by, for example, web services, the sensor-net is able to cope with newly added sensors and is, therefore, evolvable. It is stated that these findings are in line with the trend towards standardization in order to increase interoperability.

Clearly, the issue on how to provide an IoT architecture which ensures interoperability and allows for evolvability is considered relevant, challenging and open to further improvement.

C. Normalized Systems Theory

Originating from the field of software development, NST provides a number of design theorems that allow for the construction of evolvable software systems [20]. The theory is based on the domain of systems theoretic stability. Evolvability is seen by NST as the property of a software system that the impact of a change is not related to the size of the system. Assuming that the size of a software system is ever-increasing, this can be translated into Bounded Input Bounded Output (BIBO) stability.

Afterwards, NST has been formulated in a more general way, claiming that it can be applied to modular systems in general. According to the theory, every module should only contain one concern or change driver (Separation of Concerns or SoC), the use of a module by another module during its operation should be separated by a state (Separation of States or SoS), and a module used by or using other modules should be modifiable without impacting the others (Version Transparency or VT). It is shown that a violation of these theorems implies that a change of one module may impact other modules, coined as combinatorial effects. Since these effects depend on the size of a system, these are considered harmful for future evolvability.

As the IoT environment is evolving and new applications in different domains are being developed, it is plausible to apply NST to this technology in order to evaluate the ability to cope with changes. Given the expected increase in the number of connected devices and applications, the idea of maintaining stability (i.e., an impact which is not dependent on the number of devices and applications) will only become of higher relevance.

III. An IoT applications as an evolving modular structure

An IoT application typically consists of several components or modules. As the purpose of IoT is to connect physical things, these things are building blocks of an IoT application. The connected physical things are referred to as the IoT devices. The data gathered by each IoT device have to be processed and stored. One way to achieve this is by building internal applications. Another often more cost-effective way is to outsource this resposibility to (cloud) platforms. The internal business applications and external platforms are seen as separate modules of the IoT application. These three high-level building blocks constitute the basis for a typical IoT application. As these building blocks themselves consist of different modules, a deeper modular structure can be derived. The connected things, for example, typically consist of sensors, actuators and a means of communication. Figure 3 offers a visual representation of the hierarchical modular IoT application structure.

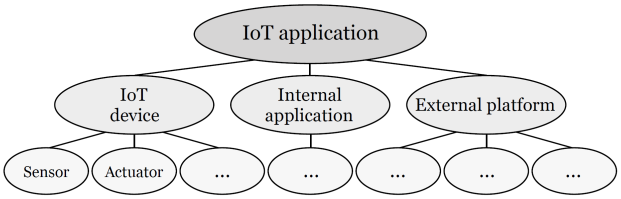


Figure 3. Modular IoT application structure.

It can be argued that the IoT environment is developing rapidly and that IoT applications will clearly be faced with changing requirements. Such changes can be due to legal requirements (e.g., General Data Protection Regulation) or technological requirements, such as the introduction of new types of sensors and actuators, new application domains and efforts towards standardization [21]. As a consequence, the modular structure might need to be changed: additional modules might need to be added or existing ones might need to be replaced by newer versions. Furtermore, different variants of the same module might be required to exist simultaneously. In the remainder of this work, it is assumed that the number of modules in a typical IoT configuration will grow over time and might become (theoretically) unlimited. This assumption is useful in order to detect NST combinatorial effects, as we will aim to do in the following sections.

IV. Best practice Internet of Things Architecture

We first discuss the current high-level architectural design of some well-known IoT platforms. Next, we evaluate their ability to adapt by using NST.

A. *Architectural design*

As mentioned earlier, several platforms have been developed to provide accessible IoT implementation capabilities to businesses. Several major technology players have developed an IoT cloud platform, each with its own vision on the architectural design of IoT applications. Typically, the cloud platform module is placed directly between the IoT devices and the internal applications of a company. Botta et al. [22] argued that the cloud is able to perform as a layer in-between things and business applications. They indicate that all complexity can be separated and that companies can focus on building the applications they need. Additionally, Gubbi et al. [3] refer to a general cloud framework as an intermediate agent between sensors on the one hand and private and public clouds on the other hand. This framework should allow developers to create applications without any complexity related to the cloud and sensor integration, as these are offered by the framework through services. The ability to create custom applications is deemed necessary, since a cloud platform usually does not offer a tailored solution for specific business problems [12].

As IoT implies, by definition, a very large number of devices and, therefore, a large amount of data, the cloud is presented by Botta et al. [22] as a solution for data storage, since it offers unlimited data storage capacity, on-demand and at low cost. Other advantages include the fact that the data stored in the cloud can be aggregated, protected by cloud security and distributed to the business applications to perform additional actions or visualizations. Furthermore, cloud platforms address a lack of sufficient computing power in IoT devices. The data is forwarded to a hub that performs data processing, in combination with aggregation. As infrastructure must be powerful enough to handle vast amounts of data, the unlimited processing power of the cloud offers a solution. In this way, the development and maintenance of IoT applications becomes more convenient and cost-effective for organizations when compared to in-house alternatives. At their turn, cloud platform providers are able to offer these services at lower prices due to economies of scale [12].

As an example of an IoT platform, AWS IoT Core can be considered. The product offers the possibility to connect all devices to the cloud platform, which in turn can integrate with other cloud and business applications via API calls [23]. Figure 4 represents this best practice architecture.

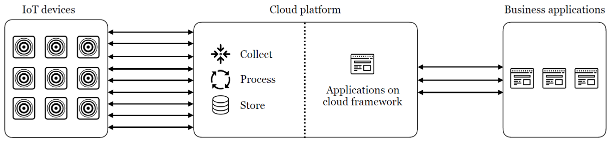


Figure 4. Best practice IoT architecture.

In some cases, such as with Google Cloud Platform, it is proposed to add a gateway between the IoT devices and the cloud data processing. The gateway is used to translate between different protocols used by connected devices. This is considered good practice because the gateway behaves in a similar way as an Enterprise Service Bus. The cloud platform only has to support the protocol of the gateway. Moreover, devices that are not directly connected to the Internet (e.g., Bluetooth devices), or cannot connect with the standards of the cloud platform, are still able to transfer data via the gateway. The IoT architecture in Figure 5 includes such a separated

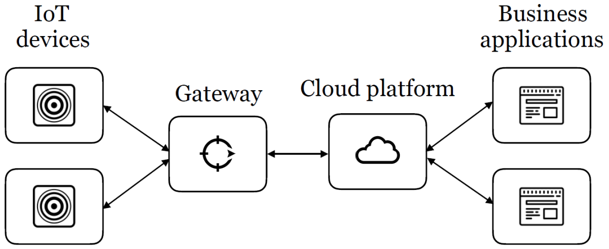


Figure 5. Simplified best practice with a gateway.

gateway [24].

In addition, a gateway should make it effortless to switch platforms, since no separate link to a specific platform is made for every IoT device. However, in this specific example, the gateway is from Google itself and works best with other cloud services of the company. At first sight, Google Cloud correlates with the use of a gateway proposed by Desai, Sheth and Anantharam [16]. However, the gateway of Google Cloud only aims at realizing integration at the technological level, whereas Desai, Sheth and Anantharam focus on semantic integration.

B. Evaluation

To evaluate the evolvability of an IoT application architecture, the following three types of modules are considered: connected devices, external platforms and internal business applications. This corresponds to the second modularity level in Figure 3. It is assumed that the number of instances of each of the three module types can become unlimited over time. In the current best practice architecture, three data activities (collection, storage and processing) are centralized in one module, the external platform. This can be considered a violation of the SoC theorem (as each of them is a concern that can change independently) and causes several issues.

Business applications are dependent on changes of the cloud platform for their internal working. To make use of services, these applications have to settle for the endpoints that are made available by the cloud platform provider. As a new version of the cloud platform can independently change internal data structures, data conversions, data aggregations and API endpoints, this possibly affects the internal working of business applications. For example, in case a certain service returns its result in a modified measurement unit, the invoking business applications should implement this change. The best practice IoT architecture proposes to connect business applications with the endpoints provided by the cloud platform. When these connections are not properly separated or encapsulated (thereby constituting a violation of VT), the impact of such a change is dependent on the size of the system (i.e., the number of service invocations) and, as a result, combinatorial effects occur. Moreover, a service used by the business for internal processes may not be available anymore in a new version of the platform. In that case, it might be necessary to look for another service provider to perform that specific task. However, since data collection, storage and processing are all centralized on the same platform, easily switching the provider for one of these services is not feasible. This is an implication of the SoC violation. Lamarre and May [12] have confirmed that businesses are usually not switching platforms. The difficulties described above may be a reason for this.

Furthermore, there are *dependencies between the cloud platform and the IoT devices of the organization*, as the platform is responsible for data collection. It is, for instance, possible that new versions of the cloud platform cause compatibility issues. In case the updated cloud platform does not support the original IoT devices, an update or a replacement of every device might be necessary. As the impact of this change clearly depends on the size of the system (i.e., the number of devices), combinatorial effects arise. Additionally, similar problems can occur as a consequence of IoT device updates. If the organization's IoT device vendor launches a new version of the device that is not supported by the used cloud platform, a normal extension of the IoT application is not possible. In that case, to be able to increase the size of the system (i.e., the number of devices), the organization might be forced to change the used cloud platform. As outlined above, a new (version of a) cloud platform will demand changes to the business applications and the IoT devices already in use. Alternatively, the organization could also look for another type of device that is compatible with its current cloud platform. However, using devices from different vendors with different technologies and protocols clearly increases the complexity of the IoT application. Again, these issues are a consequence of the fact that data collection, storage and processing are all performed on the same external platform and, therefore, not properly applying SoC.

As businesses might deploy applications on the cloud platform itself for business-specific processes, the dependency on the cloud platform increases further. These applications are typically built upon cloud frameworks for specific external platforms, making it more burdensome to switch between cloud platform providers. Therefore, we consider it safe to conclude that the centralization of data collection, storage and processing in one cloud platform potentially causes the occurrence of several combinatorial effects and, therefore, offers challenges regarding evolvability.

Two other important implications of the current best practice architecture need to be mentioned: firstly, as a consequence of the direct connection between the IoT devices and the cloud platform, the organization is not the owner of the data in its original form. The data is typically accessible for business applications by making use of API calls to the cloud platform. Companies have no control over possible conversions or aggregations that the cloud platform applies to the raw data before making it accessible. This implies (or can in the future imply) that not all raw data from the IoT devices may be accessible for the business, as the platform can implement these changes independently. Secondly, the centralization of data collection, storage and processing in one external platform possibly results in a vendor lock-in. Entrusting one external party with all these important responsibilities offers this party a considerable amount of power. As outlined above, switching platforms is a difficult undertaking, which reinforces the control of the platform provider. Additionally, outsourcing the three main data activities to a standardized platform raises the question as to what extent an organization is able to realize a competitive advantage. Opposed to what is usually desired, the strong dependency on one external provider will force businesses to adapt in function of the platform requirements.

V. Towards a Normalized IoT Architecture

Based on the evaluation above, we propose a modified architecture for IoT applications. In this architecture, *every IoT device is connected with a company gateway*, which is in turn connected with the IT landscape of the enterprise. Raw data from the IoT devices is stored and business applications can use the original data. Although the data is not directly sent to the cloud, the possibility to connect with external (cloud) platforms still exists in the proposed architecture. The gateway between the business landscape and external platforms ensures connections with the API endpoints offered by the platform providers. A graphical overview is given in Figure 6.

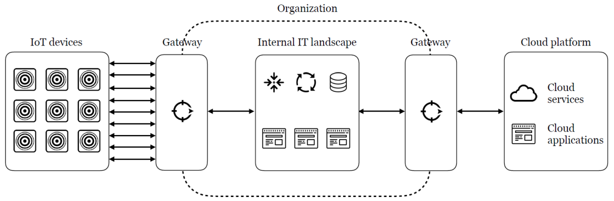


Figure 6. Proposed IoT architecture.

First, regarding *data gathering and storage*, it is the organization itself which assumes the responsibility for data gathering, since a link between the IoT devices and the information systems architecture of the organization is established (instead of working directly via an external cloud platform). This implies that the organization itself has full control over the raw data. In this way, the organization can decide freely on data types, conversions and aggregations and keeps ownership of data in its original form. The organization can also decide on how the data is stored. In case the company has data storage capacity, internal databases can be used. Furthermore, it remains possible to rely on an external cloud provider. In any case, only a lower degree of dependency on external platforms remains: there are no manipulations on original data by external parties and, since sensors are not directly connected with the external platforms, changes within these platforms have no impact on the IoT devices the company uses. In case an update of the external platform results for instance in new data requirements, the organization itself can perform the necessary data conversions if needed.

Second, regarding the *IoT devices*, it can be argued that updates can be implemented with a limited impact. The organization is free to choose which type(s) of devices it uses and by properly separating the connection between the IoT devices and in-house information systems, the impact of modifications to existing devices can be controlled and centralized into one location. Opposed to the best practice IoT architecture, the proposed architecture encapsulates the connection with external IoT devices and changes regarding these connections will not result into combinatorial effects.

Third, regarding *data processing*, the organization can still make use of an external analysis platform if preferred. It can load the relevant data from its custom or cloud databases into the platform and perform the necessary analyses. To use generated insights in custom business applications, connections with these platforms can be established via the typical API calls to the platform. Also here, if these calls are properly separated and encapsulated in the organization's business applications, the impact of future changes is contained.

In essence, our proposed architecture attempts to separate the different services offered by external platforms. Although a platform provider can still be used for multiple activities, these activities must be separated from each other and managed by a stateful controller in the organization's application. The proposed architecture improves switching opportunities by placing the internal IT landscape between the IoT devices and the external platforms. Therefore, it is expected that the likelihood of a vendor lock-in is reduced.

Furthermore, it was stated in Section IV that the positioning of an external platform between IoT devices and the internal IT landscape hampers an organization's ability to realize a competitive advantage. Indeed, making use of standardized packages for general problems offers no unique business value to the operations. The proposed architecture, however, provides decision freedom to the organization itself regarding which platforms or services to use for which functionality and which time, and to revise those decisions later on. This ensures that the business can autonomously decide where added value is created and what differentiates them from competitors.

VI. Discussion

In the proposed IoT architecture, it is possible to use various external technology providers. It can be argued that this increases complexity when compared to the current best practice architecture, where one external platform offers a one-stop solution. However, the newly proposed architecture offers an improved ability to evolve. In general, organizations that want to use IoT in their operations face a tradeoff between initial complexity and evolvability. In the short term, limiting apparent complexity with a one-stop solution might be a natural choice. Nevertheless, only an evolvable modular IoT architecture will enable an organization to create a sustainable competitive advantage.

It should be mentioned that the proposed IoT architecture correlates to some extent with previous research. Schmid et al. [15] recommend the BIG IoT architecture in which a marketplace integrates consumer applications with providers of sensors, storage, and other IoT services. The marketplace acts as an intermediate agent and translates messages between each coupled consumer and provider. The gateways in the newly proposed architecture have a similar purpose: creating a standardized means of communication between internal business applications and external parties. A difference, however, is that the proposed gateways are managed internally by the organization. The marketplace from the BIG architecture is managed by an external party. This implies that, when the BIG IoT architecture is implemented, API calls to the marketplace have to be separated properly and, in theory, another internal gateway is required between business applications and the marketplace. Similarly, Desai, Sheth, and Anantharam [16] introduced a semantic gateway between IoT devices and IoT services. However, in our approach, gateways are not placed directly between IoT devices and IoT services in order to give ownership of raw data to the business itself. Moreover, the semantic gateway immediately performs data annotations and aggregations before sending information to service providers. It can be argued that, when raw data is important, these operations should be avoided in the gateway itself.

In conclusion, some existing approaches from other perspectives have already proposed architectures similar to our solution from a modularity and evolvability perspective. However, an important difference is that our solution includes an indirect connection between IoT devices and external platforms implying first-handed control over IoT data. Moreover, our architecture stresses that all externalities should be properly separated from the internal IT applications to facilitate future changes. As every external technology should be considered as a change driver, every dependency needs to be encapsulated in a separate module (SoC) with a version transparent interface (VT).

VII. Conclusion

IoT is a promising technological trend with (potential) applications in various industries. As IoT is maturing and business agility becomes key, it is of major importance that IoT applications are able to enable such agility as well. Based on the NST, this paper assessed the extent the current state-of-the-art IoT architectures are evolvable and presented a new IoT architecture addressing the issues found.

The current best practice IoT architecture is often a one-stop vendor solution, in which the responsibilities of data collection, storage and processing are combined in one external cloud platform. To employ the gathered insights, organizations can make use of web services offered by the platform. Although the current best practice architecture has several advantages (cost reduction, use of external computing power, limited complexity), this also presents some weaknesses from an evolvability perspective: combinatorial effects (i.e., a specific type of ripple effects) may occur as a result of changes to external platforms and connected devices. This hampers the ability to extend and adapt IoT applications in order to address changing requirements. Furthermore, assigning different data-related activities to one external party results in the absence of raw data ownership and the risk of a vendor lock-in. The newly proposed IoT architecture aims to enhance evolvability and increase an organization's control by separating the different data-related activities and using indirect connections between the internal IT landscape and external modules (platforms and connected devices).

The main contribution of this paper is its analysis of current best practice IoT architectures in terms of evolvability on a theoretical grounding which is, as far as we know, new. On the one hand, this might increase our conceptual understanding of present IoT architectures. On the other hand, this might provide practitioners with additional guidance on how to design their IoT solutions in a more evolvable way. In particular, we believe that our proposed normalized IoT architecture might prove valuable in that respect. Another contribution of this paper is situated in the demonstration of the feasibility of applying NST in a new domain (i.e., IoT) for which it has not been applied earlier. However, our work is also subject to some limitations and opportunities for future work. For instance, while we have indicated the need to isolate and encapsulate all external dependencies within the internal IT landscape, this does not guarantee that the application internally is fully evolvable and free of combinatorial effects (as also here, the NST principles should be applied for that purpose). Also, our discussion of a normalized IoT architecture was purely conceptual and not tested in practice. Therefore, future research could examine the technological feasibility of our proposal in practice.

References

[1] Gartner, "Gartner says 8.4 billion connected "things" will be in use in 2017, up 31 percent from 2016," https://www.gartner.com/newsroom/id/3598917, 2017, electronically retrieved on April 8th, 2019.

[2] International Telecommunication Union, "Series y: Global information infrastructure, internet protocol aspects and next-generation networks," http://www.itu.int/ITU-T/recommendations/rec.aspx?rec=11559, 2012, electronically retrieved on April 8th, 2019.

[3] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," Future generation computer systems, vol. 29, no. 7, 2013, pp. 1645–1660.

[4] I. Lee and K. Lee, "The internet of things (iot): Applications, investments, and challenges for enterprises," Business Horizons, vol. 58, no. 4, 2015, pp. 431–440.

[5] K. Ashton, "That 'internet of things' thing," RFID Journal, vol. 22, no. 7, 2009, pp. 97–114.

[6] Carnegie Mellon University, "The "only" coke machine on the internet," https://www.cs.cmu.edu/~coke/history_long.txt, 2018, electronically retrieved on April 8th, 2019.

[7] M. Weiser, "The computer for the 21st century," Scientific American, vol. 265, no. 3, 1991, pp. 94–105.

[8] F. Mattern and C. Floerkemeier, "From the internet of computers to the internet of things," in From active data management to event-based systems and more. Springer, 2010, pp. 242–259.

[9] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," Computer Networks, vol. 54, no. 14, 2010, pp. 2787–2805.

[10] J. Bughin, M. Chui, and J. Manyika, "An executive's guide to the internet of things," McKinsey Quarterly, vol. 4, 2015, pp. 92–101.

[11] A. Bosche, D. Crawford, D. Jackson, M. Schallehn, and P. Smith, "Defining the battlegrounds of the internet of things," http://www.bain.com/publications/articles/defining-the-battlegrounds-of-the-internet-of-things.aspx, 2016, electronically retrieved on April 8th, 2019.

[12] E. Lamarre and B. May, "Making sense of internet of things platforms," https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/making-sense-of-internet-of-things-platforms, 2018, electronically retrieved on April 8th, 2019.

[13] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang, "A vision of iot: Applications, challenges, and opportunities with china perspective," IEEE Internet of Things journal, vol. 1, no. 4, 2014, pp. 349–359.

[14] P. Sethi and S. R. Sarangi, "Internet of things: architectures, protocols, and applications," Journal of Electrical and Computer Engineering, vol. 2017, 2017.

[15] S. Schmid et al., "An architecture for interoperable iot ecosystems," in Proceedings of the International Workshop on Interoperability and Open-Source Solutions, 2016, pp. 39–55.

[16] P. Desai, A. Sheth, and P. Anantharam, "Semantic gateway as a service architecture for iot interoperability," in Proceedings of the 2015 IEEE International Conference on Mobile Services (MS), 2015, pp. 313–319.

[17] F. Wortmann and K. Flüchter, "Internet of things," Business & Information Systems Engineering, vol. 57, no. 3, 2015, pp. 221–224.

[18] M. Porter and J. Heppelmann, "How smart, connected products are transforming competition," Harvard Business Review, vol. 92, no. 11, 2014, pp. 64–88.

[19] N. Priyantha, A. Kansal, M. Goraczko, and F. Zhao, "Tiny web sservice: design and implementation of interoperable and evolvable sensor networks," in Proceedings of the 6th ACM conference on embedded network sensor systems, 2008, pp. 253–266.

[20] H. Mannaert, J. Verelst, and P. De Bruyn, Normalized Systems Theory: From Fondations for Evolvable Software Toward a General Theory for Evolvable Design. Koppa, 2016.

[21] L. Columbus, "2017 roundup of internet of things forecasts," https://www.forbes.com/sites/louiscolumbus/2017/12/10/2017-roundup-of-internet-of-things-forecasts/#4d55f6451480, 2017, electronically retrieved on April 8th, 2019.

[22] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: a survey," Future generation computer systems, vol. 56, 2016, pp. 684–700.

[23] AWS, "Aws iot core," https://aws.amazon.com/iot-core/, 2019, electronically retrieved on April 8th, 2019.

[24] G. Cloud, "Overview of internet of things," https://cloud.google.com/solutions/iot-overview, 2018, electronically retrieved on April 8th, 2019.

Pattern Matching in the Era of Big Data: A Benchmark of Cluster Quality Metrics

Ole Kristian Ekseth, Per Jarle Furnes, and Svein-Olaf Hvasshovd
Department of Computer Science (IDI)
NTNU & Eltorque
Trondheim, Norway
email: oekseth@gmail.com and sophus@ntnu.no

***Abstract*—In today's quest for knowledge, there is a need for accurate and fast measures for pattern matching. While numerous new metrics and algorithms are published every year, researchers are unaware of which metric to choose. There does not exist an established strategy for pattern matching of cluster algorithms, which may explain why new hypothesis and algorithms are often forgotten. In this work, we address this issue. The paper presents a new benchmark for automated evaluation of pattern matching algorithms. From key characteristics of training data, the benchmark deduce fast and accurate cluster quality metrics, hence enabling pattern searches in big data. The benchmark address key issues in pattern analysis: while recent algorithms improve prediction accuracy by less than 2x, there is a 5x+ inaccuracy in established pattern matching algorithms. The evaluation of 100+ real-life data-sets reveals how the benchmark manages to identify patterns which are otherwise hidden, hence paving the ground for improved quality in the field of big-data pattern matching.**

***Keywords: patterns; clustering; similarity metrics; data analysis*.**

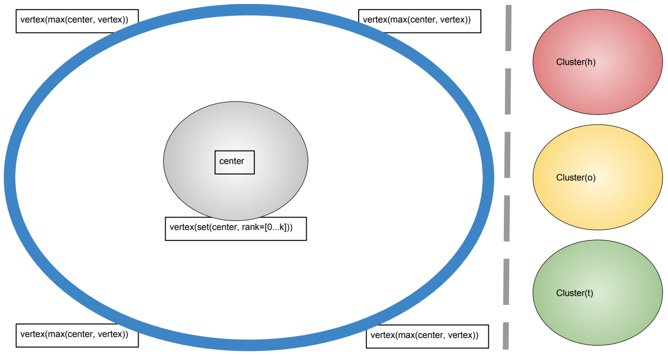

vertex(max(center, vertex))
vertex(max(center, vertex))
center
vertex(set(center, rank=[0...k]))
vertex(max(center, vertex))
vertex(max(center, vertex))
Cluster(h)
Cluster(o)
Cluster(t)

Fig. 1. Ambiguities in cluster analysis. Above figure demonstrates why it is impossible to use a a single number (*e.g.*, '0.8') to describe pattern similarities between two cluster partitions, *e.g.*, when comparing a null-hypothesis to the results of a clustering algorithm.

I. Introduction

Data mining and big data analysis have experienced a surge of interest in the recent years [1]. In data analysis, it is essential to know the trustworthiness of other's findings. An example is seen in the work of [2], where the authors report the patterns to have a difference of 1.67x: is the improvement reported by [2] sufficient to discard the earlier ground-truth?

The motivation of many research papers is to demonstrate that some algorithms are superior to other [3]–[5]. In contrast, our hypothesis is that the choice of clustering algorithms depends on both the data and the local configurations of the clustering algorithms (code-listing 1): the established strategy results in a 100x+ prediction error.

Of importance is to identify the reasons for why measurement data, presented in different research papers, diverge. The diverging recommendations, which seems to be the rule in benchmarking of algorithms, raises several questions. To exemplify, is the rarely discussed choice of *benchmark data* the determining factor?

While the choice of validity metrics (*e.g.*, Silhouette) determines the outcome of experiments (*best column* in Table II), the established strategies have a poor trustworthiness in seperating between *false* versus *true* hypothesis (Section VI). For some data-sets the VRC metric [6] is unable to spot differences in data predictions. Hence, perturbations in data will remain unknown to VRC users. The issue of VRC inaccuracy arises from how metrics weight differences in data (sub-section V-A).

To address this challenge, this paper constructs an automated method for capturing the bias in metrics and training data. The method explores the parameter space of *pattern matching algorithms* (Fig. 1). The measurements reveals how the approach addresses issues in [2]–[5].

The results demonstrate how the proposed framework increases the accuracy of data classification by 5x+ (sub-section VI-C). The benchmark captures peculiarities in turf specific data-sets. This knowledge is important in a number of domains:

1) trustworthiness: the significance of differences when no hypothesis is valid;
2) experimental design: the accurate and efficient exploration of a hypothesis in large data volumes;
3) new metrics: the automated identification of new metrics from training data (code-listing 1).

The remainder of the paper is organized as follows. Section II identifies the contributions of this paper. Section III briefly surveys related approaches, and Section IV describes a new algorithm for automated evaluation and identification of metrics. Section V describes an approach to evaluate the influence of strategies for pattern matching, a method which is applied in the result Section VI. This paper ends with a brief summary

TABLE I
APPLICABILITY OF ALGORITHMS FOR PATTERN MATCHING. THE TABLE CAPTURES THE ACCURACY AND RESOLUTION OF PATTERN MATCHING ALGORITHMS.

Year	Name	Cite	Gold	Diff. / Equal	Equal: Worst / best	Diff.: Worst / best
1971	Rand's Index		x	100x	-	-
1974	VRC		-	100x	-	
1974	Dunn		-			
1974	Dunn		x			
1979	Davis-Bouldin		x	∞	1x	1x
1979	Chi-squared		x	∞	4.5x	
1982	SSE		x	1x		
1982	SSE		-	79x		
1983	FM		x	20x		
1987	Silhouette		x		-	
1987	Silhouette			∞	-	
2001	R-squared		-	2x		
2001	ARI		x	∞		
2001	Mirkin		-	∞		
2003	Fred & Jain		x	25x	5x	
2003	Strehl & Gosh		x	25x	5x	
2007	Wallace		-	1x		
2007	VOI		-	25x	10x	
2015	MMM		x	1x		
2017	Dogen		x	∞		
2010	RMSSTD		-	78x		

of observations in Section VII.

II. CONTRIBUTIONS

This paper presents a benchmark identifying the drawbacks of clustering metrics. The work manages to both quantity the trustworthiness–threshold for algorithms, and provide a software for automated benchmarking of users own data. The results reveal how the proposed benchmark provides users with software which identifies fast and accurate cluster algorithms (Fig. 2). Hence, a new approach which deduces fast and accurate cluster quality metrics.

Today, it is impossible to check if results (*e.g.*, produced by new algorithms) makes sense. This due to configuration parameters not described in research papers, *i.e.*, hidden factors which are left unexplained to the users. Of importance is therefore to identify benchmarks, and a software tool, for capturing the algorithms which produce the best cluster predictions at lowest possible execution time.

To address this issue, this paper provides users with a tool for big-data analytics. The tool covers a big assortment of metrics and databases. The results are presented through generalization of algorithms and metrics, hence easing the accessibility (of the findings) across a wide spectrum of research domains.

The proposed method may be applied to existing algorithms and clustering (code-listing 1). Hence, the work avoids the pitfall of proposing new algorithms (instead of improving the existing). The work analysis the patterns used in clustering algorithms, and the clustering evaluation, for which a *Pareto Boundary* is identified. The *Pareto Boundary* provides the means to identify when a given algorithm provides improves beyond the noise threshold (Fig. 2).

To summarize, the paper presents a new algorithm and software. The the software enables users to select metric combinations with high prediction accuracy at low execution time, hence paving the ground for improved quality in the field of big-data pattern matching.

III. RELATED WORK

Application of data analysis requires accurate strategies to capture the similarities across measurements, hypothesis, data perturbations, etc. There are more than 30+ metrics for capturing the patterns of data, for which a subset is listed in Table I. Below section demonstrates how researchers are unaware of the ambiguity in "Cluster Comparison Metrics (CCMs)". While accurate algorithm configuration increases prediction quality by 10x+ (Fig. 2), new algorithms provides less than a 2x prediction improvement. Prediction improvements are measured through algorithms such as "Rand's Index"

Accuracy of data analysis is fundamental in all parts of research, such as bio-medicine [1], language processing [7], image recognition and reconstruction [5], [8], etc. An application of data analysis is to establish the significance of new findings: to identify the degree of correlation between hypothesis and experimental outcomes. Examples of widely used metrics are "Sum of Squared Error (SSE)", Silhouette, "Rand's Index", etc.

Pattern matching in big-data represents the performance crux in drug discovery [1], epidemiology [9], etc. Clustering is able to group mixed data into groups, called clusters, focusing on the similarity between the data points [10]. The requirements for *big data* differ from other application and domains. The work of [11] observes how "big data analytics requires technologies to efficiently process large quantities of data" [11]. Software for pattern matching suffers from high execution time, as observed for "Sci-kit learn" [12] and the "Moa" software [13].

Partitioning of data into clusters involves assumptions of the data: to use metrics for similarities between groups to partition data. For example, the default "k-means" implementation uses Euclidean distance to cluster numerical data points [14], "k-modes" groups categorical data [15], while "k-prototypes" uses cost functions to group mixed data [16].

A challenge concerns how to evaluate and interpret the identified patterns (Fig. 1). The ambiguity of CCMs is due to its purpose: from variance and agreements inside each cluster, and between multiple clusters, to infer a representative number (to capture the fit between hypothesis and data) [17].

However, the ambiguities of CCMs are not reflected in their application. In pattern analysis, there does not exist any agreement in which CCMs to use. To exemplify, [3] combines "ARI" [18] with "Silhouette index", "Jaccard index","Minkowski measure", "Silhouette index", "Dunns index" and "Davies-Bouldin index" to judge the cluster-accuracy of their proposed algorithm. [20] combines "FJ" with "ARI" in order to validate their new-proposed algorithm. The work of [4] combines "Rand's Index" with "Calinski-Harabasz (VRC)" [6], "Silhouette Index" and "logSS".

When research agrees in which CCMs to use, they disagree in how to interpret the CCM scores. To exemplify, the work of [2] asserts that a change in ARI [18] prediction score of *0.46* into *0.76* implies a significant difference in cluster accuracy. However, the authors do not discuss ambiguities in their gold standard, nor the significance of the scores. Measurements reveal how the difference may be explained by the unawareness of the metrics sensitivity (Table I).

To summarize, the established metrics for pattern matching are applied irrespective of their inaccuracy. The choice of strategy for pattern matching is applied without discussing the dependency between CCMs (Fig. 3), hence it is unknown when methods and algorithms are better than others.

Algorithm 1 An algorithm for unbiased selection of best-performing *pattern matching algorithm* in real-life data-sets. To simplify, the *selectMax(r_g, a, t, n, s)* method is omitted from the evaluation, a method which identifies the best-performing algorithm permutation.

```
procedure EVALUATE(ENSEMBLE)
    for each $a \in clustAlg$ do
        for each $t \in [0,1]$ do    ▷ are we to t(transpose)?
            for each $n \in normMetrics$ do
                for each $s \in simMetrics$ do
                    $r_M$ = ccmMatrix(ensemble, a, t, n, s)
                    selectMax($r_M$, a, t, n, s)
                    $r_g$ = ccmGold(ensemble, a, t, n, s)
                    selectMax($r_g$, a, t, n, s)
procedure CCMMATRIX(ENSEMBLE, A, T, N, S)
    $ranks = [][]$ Ranks for 'not gold' CCM (Table I)
    for each $data \in Ensemble$ do
        clusters = a(data, t, n, s)
        for each $ccm \in matricCCM$ do
            ranks[ccm,data] = ccm(clusters, data)
        ranks[ccm] = rank(ranks[ccm])
    return ranks
procedure CCMGOLD(ENSEMBLE, A, T, N, S)
    $ranks = [][]$ Ranks for each 'gold' CCM (Table I)
    $clusters_0$ = a(Ensable[0], t, n, s)
    for each $data \in Ensemble$ do
        clusters = a(data, t, n, s)
        for each $ccm \in ccmGold$ do
            ranks[ccm,data] = ccm($clusters_0$, clusters))
        ranks[ccm] = rank(ranks[ccm])
    return ranks
```

IV. METHOD: NEW BIG-DATA TOOLS FOR DATA PERTURBATION AND ALGORITHM IDENTIFICATION

This section describes a new method for capturing the trustworthiness of "Cluster Comparison Metrics (CCMs)" (code-listing 1):

1) data perturbations: a new approach and API to capture the accuracy of CCMs (Table I);
2) execution time: a strategy to reduce the time cost, hence a methodology supporting big-data analytic;
3) unbiased evaluation: a new algorithm which combines clustering with metric permutations to avoid bias in gold-data from influencing the prediction outcome.

The method enables the automated identification of best-performing metrics in an ensemble of data, hence its broad applicability. The algorithms are integrated into the "hpLysis" machine learning software [21]. Hence, users are provided with software for fast classification of large data-sets.

A. Data perturbations: a new API to detect accuracy and resolution of CCMs

Motivation is to trap the differences among CCMs. A large number of data topologies and CCMs argues for an automated approach, for which a new method and API for synthetic evaluation of CCMs is designed:

1) cluster shapes: construct different co-occurrence matrices and cluster partitions;
2) perturbations: compare *cluster shape* with exactly similar data topologies;
3) CCMs: apply permutations of the 30+ CCMs, and then select the extreme cluster predictions.

The strategy enables an automated and unbiased quantification of differences in CCM prediction.

B. Execution Time: the feasibility of big-data evaluation

The large number of CCMs requires an approach to reduce the computational complexity. The crux in CCM computation concerns the time cost of computing similarity metrics. The computation of CCMs involves the steps of 1) compute a covariance matrix (time: $O(n^3)$), and 2) similarities between features in Fig. 1 (time: $O(n^2)$).

The performance $O(n^3)$ issue is addressed through application of optimized implementation of matrix multiplication, as discussed in our earlier work [22]. The "hpLysis" software [21] provides fast access to the 320+ pairwise similarity metrics.

C. An algorithm for unbiased exploration of data

Code-listing 1 describes an algorithm for enabling the qunaitifciaotn of pattern matching algorithms. The algorithm takes as input data-sets with a well-defined order of predictions. Example input is a feature matrix combined with multiple hypotheses representing different data segmentation.

To avoid bias in clustering from hampering the prediction accuracy, the 20+ cluster algorithms supported by the hpLysis software [21] are combined with the 320+ established similarity metrics. Hence, the approach address issues in regression analysis.

The algorithm makes use of heuristics to reduce its execution time. The choice of the metrics is tuned towards different data ensembles. When partitioning the data-sets into clusters *three categories of cluster algorithms* are explored: threshold based cluster algorithms (*e.g.*, "DBSCAN" [23]); hierarchical cluster algorithms (*e.g.*, "SLINK"); randomized cluster algorithms (*e.g.*, "k-means"). For computation of the cluster algorithms the hpLysis software [21] is used, hence ensuring fast execution.

file	k means avg (Low)	k means avg (High)	k means rank (Low)	k means rank (High)	k means medoid (Low)	k means medoid (High)	hpCluster (Low)	hpCluster (High)	disjoint kdTree (Low)	disjoint kdTree (High)	disjoint kdTree CCM (Low)	disjoint kdTree CCM (High)	HCA single (Low)	HCA single (High)	HCA max (Low)	HCA max (High)	HCA average (Low)	HCA average (High)	HCA centroid (Low)	HCA centroid (High)	Kruskal HCA (Low)	Kruskal HCA (High)	k means altAlg miniBatch (Low	k means altAlg miniBatch (Higl	random best (Low)	random best (High)
msq	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0
chorSub	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9
pulpfiber	0.0	0.7	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.8	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9
randu	0.0	0.7	0.0	0.7	0.7	0.7	0.0	0.7	0.0	0.7	0.0	0.7	0.0	0.7	0.7	0.7	0.0	0.0	0.0	0.7	0.0	0.7	0.7	0.7	0.0	0.7
cf	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9
airquality	0.2	0.7	0.2	0.7	0.2	0.7	0.2	0.7	0.2	0.7	0.2	0.7	0.2	0.7	0.2	0.7	0.2	0.5	0.2	0.7	0.2	0.7	0.2	0.7	0.2	0.7
UScrime	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9
pottery	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9
Hedonic	0.0	0.8	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.8	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9
Melanoma	0.0	0.7	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.7	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.7	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9
affect	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9
Holzinger	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.8
smoking	0.2	0.7	0.2	0.7	0.7	0.7	0.2	0.7	0.2	0.8	0.2	0.8	0.2	0.7	0.7	0.7	0.2	0.2	0.2	0.8	0.2	0.8	0.7	0.7	0.2	0.7
airquality	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7
bfi	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	1.0	0.0	1.0	0.0	1.0	0.0	0.9	0.0	1.0	0.0	1.0	0.0	0.9
Hartnagel	0.0	0.7	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.8	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.8	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9
attitude	0.0	0.7	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.7	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.7	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9
gilgais	0.1	0.7	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9
cancer	0.0	0.8	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.8	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.8	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9
burt	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9
votes	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9	0.0	0.9
attitude	0.1	0.6	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.6	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.6	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9
msq	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0
Arbuthnot	0.1	0.7	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.7	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.7	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.8
LifeCycleSavings	0.1	0.5	0.1	0.8	0.8	0.8	0.1	0.8	0.1	0.5	0.1	0.8	0.1	0.8	0.8	0.8	0.1	0.8	0.1	0.8	0.1	0.8	0.8	0.8	0.1	0.8
phosphate	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.7	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.7	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.8
alcohol	0.1	0.7	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.7	0.1	0.9	0.1	0.9	0.1	0.8	0.1	0.7	0.1	0.8	0.1	0.8	0.1	0.8	0.1	0.8
aldh2	0.1	0.8	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.9	0.1	0.8	0.2	0.9	0.1	0.9	0.1	0.9	0.1	0.9

Fig. 2. How the choice of metrics influences algorithms accuracy. The above table compares 20 clustering algorithms.

V. Evaluation Framework

The section identifies an experimental benchmark setup capturing the effect of data and metric perturbations. The framework provides an unbiased strategy for measuring the accuracy of different pattern matching algorithms (Section III).

A. Axis of variability: hypothesis testing & result ranking

An accurate benchmarking of CCMs for big-data analytic requires:

1) representative: investigate the CCMs applied in big-data analytic, *e.g.*, SSE [25] and Silhouette
2) features variation: data with different spread and skewness in both column features and row features;
3) configurations: different sizes of rows, columns, clusters, and score density.

Importantly, all of the proposed metrics share a set of common artifacts.

Fig. 1 exemplifies the *axis of variability* through the use of different shapes (Table I). The figure captures complexities in cluster analysis: to correctly describe the distance between vertices versus the arbitrary clusters *h*, *o*, and *t*.

Fig. 1 identifies how the *cluster within distance* is computed through permutations of $\sum d(vertex(...), vertex(...))$: when metrics such as VRC and Dunn's Index agrees in the prediction, it is due to the agreements between different interpretations of *minimum(between–within)* distance. Hence, for controlled topologies, it is sufficient to evaluate a small subset of the proposed cluster algorithms and metrics.

B. Core characteristics captured through representative data

Motivation is to identify representative data ensembles. Therefore, data-sets are explored for different perspectives:

1) controlled: synthetic clusters and hypothesis to evaluate the the linear relationship between hypothesis, data topology, and cluster segmentation;
2) real-life: an evaluation of 100+ real-life data-sets taken from [26].

The 100+ real-life data-sets. are modified through increased use of Gaussian noise. The application of linearly distorted data enables users to capture the effects of randomness, *e.g.*, when evaluating the CCMs ability to correctly rank different hypothesis.

The CCMs are evaluated through comparison of outputs from different algorithms. As input, the evaluation takes both randomized data and randomized cluster partitions. An issue concerns the different scores (and ranges) provided by metrics (Table I). To address the issue of different scales, the prediction results are ranked separately for each $[data\ permutations]$ x $metric$.

Zero clusters for the SSE CCM:

vertices	1	2	3	4	5	6	7	8	9
0	0.58	0.22	0.17	0.18	0.17	0.16	0.17	0.17	0.17
100	0.56	0.17	0.11	0.09	0.08	0.08	0.08	0.08	0.08
300	0.56	0.16	0.09	0.07	0.06	0.05	0.05	0.05	0.05
600	0.56	0.15	0.08	0.06	0.05	0.04	0.04	0.04	0.04
1000	0.56	0.15	0.08	0.05	0.04	0.04	0.03	0.03	0.03
1500	0.56	0.15	0.07	0.05	0.04	0.03	0.03	0.03	0.03

Zero clusters for the Silhouette CCM:

vertices	1	2	3	4	5	6	7	8	9
10	0.57	0.72	0.67	0.65	0.6	0.6	0.6	0.6	0.6
100	0.5	0.89	0.87	0.87	0.87	0.87	0.87	0.87	0.86
300	0.5	0.93	0.92	0.92	0.92	0.92	0.92	0.92	0.92
600	0.5	0.95	0.95	0.94	0.94	0.94	0.94	0.94	0.94
1000	0.5	0.96	0.96	0.96	0.95	0.95	0.95	0.95	0.95
1500	0.5	0.97	0.97	0.97	0.97	0.97	0.97	0.96	0.96

Zero clusters for the Dunn CCM:

vertices	1	2	3	4	5	6	7	8	9
10	0.43	0.04	0.13	0.04	0.04	0.04	0.04	0.04	0.04
100	0.5	0	0	0.01	0.01	0.01	0.01	0.01	0.01
300	0.5	0	0	0	0	0	0	0	0
600	0.5	0	0	0	0	0	0	0	0
1000	0.5	0	0	0	0	0	0	0	0
1500	0.5	0	0	0	0	0	0	0	0

Zero clusters for the VRC CCM:

vertices	1	2	3	4	5	6	7	8	9
10	19.55	1.22	0.62	0.39	0.27	0.13	0	0	0
100	141.6	0.79	0.62	0.49	0.41	0.33	0.27	0.23	0.19
300	379.45	0.52	0.46	0.41	0.35	0.3	0.26	0.23	0.2
600	739.44	0.38	0.36	0.33	0.3	0.27	0.24	0.22	0.19
1000	1,221.6	0.3	0.3	0.29	0.27	0.25	0.22	0.2	0.18
1500	1,819.44	0.25	0.25	0.25	0.24	0.22	0.21	0.19	0.18

Fig. 3. Prediction difference when no clusters are present. The above figure captures the result (of pattern matching algorithms) when no clusters are present in the input data.

Table I introduce the notation of *Gold*. The *Gold* column refers to cases where two hypothesis (*e.g.*, for the use-case where cluster partitions are compared), which is an alternative to comparing a hypothesis with a feature matrix. On the other hand, the table's *Equal Clusters (Equal)* column identifies the metrics sensitity to comparing two equal cluster partitions, an effect is compared to the case where they do not *(Diff.)*. In the measurements, each algorithm is evaluated across multiple feature matrices and multiple hypothesis (*e.g.*, the result of a cluster algorithm).

VI. Result: Empirical evaluation

This paper presents an automated approach for unbiased evaluation of 30+ CCMs. For brevity, the details of the 30+ CCMs are included in the benchmark scripts (appended into the hpLysis software). The results reveal how the proposed method outperforms established metrics and algorithms (Fig. 2), answering questions such as:

1) 30x+: How CCMs differ in their prediction scores? (Fig. 3);
2) 4x+: How differences in data size influence the CCM score? (Table II);
3) 0x–79x: Is SSE able to separate between *false* versus *true* hypothesis? (Table I).

While the above results are specific for the evaluated topologies, they capture the pitfall of making strong conclusions from inaccurate pattern metrics.

A. How disagreements in CCMs capture topological features

The motivation of CCMs is to grasp the differences between data using a few numeric indicators (Table II): to apply independent metrics to capture similarities and distortions in data.

The correct applicability of CCMs depends on both the datasets and the gold standards (Table I). A linear increase in random perturbations is not recognized in the Davids-Bouldin metric. In contrast, SSE and Silhouette detect a variation in data perturbed with Gaussian noise. When compared to Silhouette and SSE, the VRC metric is distinctively different. The results demonstrate how the combination of different CCMs provides users with a unique ability to reject a false hypothesis: there is no uniform agreement in which CCMs to select.

TABLE II
HOW CATEGORIZATION OF CCMS REVEALS UNDERLYING DATA TOPOLOGY. THE TABLE SUMMARIZES THE OBSERVATIONS FROM FIG. 3: WHILE $n = 10$ REFERS TO A MATRIX WITH ROWS=COLUMNS=10, $n = 1500$ CAPTURES THE RESULT OF EVALUATING A MATRIX WITH ROWS=COLUMNS=1500; "THE *n=10 – n=1500*" DESCRIBES THE RELATIVE DIFFERENCE BETWEEN *worst–best*; THE "*worst–best*" COLUMNS IDENTIFIES THE SPREAD IN CCM SCORE; THE "*best:column*" IDENTIFIES THE HYPOTHESIS WHICH IS FARTHEST AWAY FROM THE INPUT DATA.

CCM	n=10 – n=1500	n=10: worst–best	n=1500: worst–best	best: column
SSE	2.6x – 3.5x	0.22 – 0.58	0.56 – 0.15	6
Silhouette	1.3 – 2x	0.7 – 0.57	0.97 – 0.50	2
Dunn	10.5 – ∞	0.42 – 0.04	0.58 – 0.00	*all*
VRC	16.1x – ∞	19.6 – 1.22	1819.4 – 0.18	9

B. The correct choice of CCM provides accurate predictions

The performance of CCMs is determined by:

1) metric choice: 10x+ difference when using "Davids-Bouldin" instead of "Dunn's" or "Euclidean" (Table I);
2) topology sensitity: while metrics such as VRC are highly sensitive to score perturbations, metrics such as Silhouette and SSE provides higher granularity (as derived from underlying measurements);
3) score difference: a 100x+ score-difference between matrix with rows=columns=[10, 1500] (Fig. 3).

The above differences is due to the metrics definition, hence the importance of relating CCMs to topolgoical traits. An example is an assumption that the *within cluster distance* has a uniform distribution, for which CCMs, such as VRC and Dunn's Index, becomes overlapping. When evaluating the algorithm for CCM identification (code-listing 1), the results demonstrates how the new-identified CCMs outperforms metrics for regression analysis. Hence, the benchmark of cluster

ISBN: 978-1-61208-708-5

quality metrics improves the broad turf of *regression analysis*.

Fig. 3 demonstrates how CCMs have different score sensitivity: while VRC indites an $= 1819.4/0.18 = \infty$ separation between *correct hypothesis* versus *wrong hypothesis*, SSE has a sensitivity of $0.56/0.04 = 14x$, as summarized in Table II.

C. Summary: pattern recognition versus data topologies

The measurements identifies the importance of applying *independent CCMs* to capture differences in data-sets and cluster predictions. Hence, the choice of CCMs should reflect the given use-case. For the same data, there is a 5x prediction difference between "Fred & Jain" [27] versus Davies-Bouldin (Table I). Similarly, SSE and Silhouette disagrees in which of the cluster prediction is the best (Fig. 3).

While the established strategy is to apply CCMs irrespective of the topologies, this paper has demonstrated how the implicit assumptions of data topology directly influences the accuracy of pattern matching (Fig. 2). While this knowledge is known among authors of algorithms (*e.g.*, [23]), users of pattern matching are unaware of these findings (Section III).

VII. Conclusion and Future Work

In this paper, we have identified how established pattern matching strategies in big data suffers from bias. The 30x+ inaccuracy of metrics for pattern recognition goes undetected in large research projects (Fig. 3). The proposed benchmark software enables quantification of the trustworthiness of established recommendations (Section IV).

Importantly, the approach may be applied for big data-sets, which is due to the combination of optimized software implementation and heuristics deduced from metrics (code-listing 1). The benchmark deduces fast and accurate cluster quality metrics: code-listing 1 identifies the metric combination to be used (for a given data ensemble), hence enabling an increase in the accuracy of algorithms.

The new methodology and software provide users with a tool enabling the insight into when and how data is captured by hypothesis: to identify patterns which are otherwise hidden. The results highlight the importance of not always following the established guidelines for cluster validity.

In the future, we plan to apply the proposed method and benchmark to the 1000+ recently proposed cluster algorithms, hence easing the applicability of our findings into all turfs relying on pattern matching algorithms.

Acknowledgements

The authors would like to thank Dr. O.V. Solberg at SINTEF, Dr. B.H. Helleberg at NTNU–medical, K.T. Dragland at NTNU, and MD K.I. Ekseth at UIO for their support.

References

[1] Ekseth, O.K., Meyer, J.C., Hvasshovd, S.O.: A new database for drug discovery through application of data-integration and semantics. In: Semantic Computing (ICSC), 2018 IEEE 12th International Conference On, pp. 403–410 (2018). IEEE

[2] Hahsler, M., Bolaños, M.: Clustering data streams based on shared density between micro-clusters. IEEE Transactions on Knowledge and Data Engineering **28**(6), 1449–1461 (2016)

[3] Chiu, T.-Y., Hsu, T.-C., Yen, C.-C., Wang, J.-S.: Interpolation based consensus clustering for gene expression time series. BMC bioinformatics **16**(1), 1 (2015)

[4] Lord, E., Diallo, A.B., Makarenkov, V.: Classification of bioinformatics workflows using weighted versions of partitioning and hierarchical clustering algorithms. BMC bioinformatics **16**(1), 1 (2015)

[5] Yazdani, M., Chow, J., Manovich, L.: Quantifying the development of user-generated art during 20012010. PLOS ONE **12**(8), 1–24 (2017). doi:10.1371/journal.pone.0175350

[6] Halkidi, M., Vazirgiannis, M., Batistakis, Y.: Quality scheme assessment in the clustering process. In: European Conference on Principles of Data Mining and Knowledge Discovery, pp. 265–276 (2000). Springer

[7] Garla, V.N., Brandt, C.: Semantic similarity in the biomedical domain: an evaluation across knowledge sources. BMC bioinformatics **13**(1), 261 (2012)

[8] Solberg, O.V., Lindseth, F., Torp, H., Blake, R.E., Hernes, T.A.N.: Freehand 3d ultrasound reconstruction algorithmsa review. Ultrasound in medicine & biology **33**(7), 991–1009 (2007)

[9] Bhaskaran, K., Smeeth, L.: What is the difference between missing completely at random and missing at random? International journal of epidemiology **43**(4), 1336–1339 (2014)

[10] Ferrari, D.G., De Castro, L.N.: Clustering algorithm selection by meta-learning systems: A new distance-based problem characterization and ranking combination methods. Information Sciences **301**, 181–194 (2015)

[11] Lau, L., Yang-Turner, F., Karacapilidis, N.: Requirements for big data analytics supporting decision making: A sensemaking perspective. In: Mastering Data-Intensive Collaboration and Decision Making, pp. 49–70. Springer, ??? (2014)

[12] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., *et al.*: Scikit-learn: Machine learning in python. Journal of Machine Learning Research **12**(Oct), 2825–2830 (2011)

[13] Fan, W., Bifet, A.: Mining big data: current status, and forecast to the future. ACM sIGKDD Explorations Newsletter **14**(2), 1–5 (2013)

[14] Szalkai, B.: Generalizing k-means for an arbitrary distance matrix. arXiv preprint arXiv:1303.6001 (2013)

[15] He, Z., Deng, S., Xu, X.: Approximation algorithms for k-modes clustering. In: International Conference on Intelligent Computing, pp. 296–302 (2006). Springer

[16] Ji, J., Bai, T., Zhou, C., Ma, C., Wang, Z.: An improved k-prototypes clustering algorithm for mixed numeric and categorical data. Neurocomputing **120**, 590–596 (2013)

[17] Ekseth, O.K., Gribbestad, M., Hvasshovd, S.-O.: Inventing wheels: why improvements to established cluster algorithms fails to catch the wheel. In: The International Conference on Digital Image and Signal Processing (DISP19), Springer (2019)

[18] Yeung, K.Y., Ruzzo, W.L.: Details of the adjusted rand index and clustering algorithms, supplement to the paper an empirical study on principal component analysis for clustering gene expression data. Bioinformatics **17**(9), 763–774 (2001)

[19] Dunn, J.C.: Well-separated clusters and optimal fuzzy partitions. Journal of cybernetics **4**(1), 95–104 (1974)

[20] Zhao, W., Chen, J.J., Perkins, R., Wang, Y., Liu, Z., Hong, H., Tong, W., Zou, W.: A novel procedure on next generation sequencing data analysis using text mining algorithm. BMC bioinformatics **17**(1), 1 (2016)

[21] Ekseth, Ole Kristian: hpLysis: a high-performance software-library for big-data machine-learning. https://bitbucket.org/oekseth/hplysis-cluster-analysis-software/. Online; accessed 06. June 2017

[22] Ekseth, O.K., Hvasshovd, S.-O.: How an optimized DBSCAN implementation reduce execution-time and memory-requirements for large data-sets. (2017)

[23] Ester, M., Kriegel, H.-P., Sander, J., Xu, X., *et al.*: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Kdd, vol. 96, pp. 226–231 (1996)

[24] Sibson, R.: Slink: an optimally efficient algorithm for the single-link cluster method. The computer journal **16**(1), 30–34 (1973)

[25] Lloyd, S.: Least squares quantization in pcm. IEEE transactions on information theory **28**(2), 129–137 (1982)

[26] Arel-Bundock, V.: Rdatasets r datasets: An archive of datasets distributed with r, 2014. URL http://vincentarelbundock. github. io/Rdatasets

[27] Ana, L., Jain, A.K.: Robust data clustering. In: Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference On, vol. 2, p. 128 (2003). IEEE

Web Security and Privacy for Novices – Part 1

A Pattern Collection and Two Meta-Patterns

Alexander G. Mirnig*, Artur Lupp*, Alexander Meschtscherjakov* and Manfred Tscheligi†
*Center for Human-Computer Interaction
University of Salzburg, Salzburg, Austria
Email: firstname.lastname@sbg.ac.at
†Center for Human-Computer Interaction &
Austrian Institute Of Technology, Salzburg & Vienna, Austria
Email: firstname.lastname@sbg.ac.at

***Abstract*—Fostering security and privacy in online interactions is important for developers, providers, and users. Since a substantial amount of content on the web today is developed by nonprofessionals (e.g., teenagers, small company owners, hobbyists, etc.), it is important to provide development guidance that is suitable for these user groups. In this paper, we present two meta-patterns intended for novice web developers. The patterns intend to address some of the most common issues pertaining to privacy and security in websites and are intended to guide the reader through the development process from a privacy- and security-centered perspective.**

Keywords–Patterns; On-line; Security; Privacy; Novice Users.

I. INTRODUCTION

From its rather humble beginnings in the nineties of the last century [1], the world wide web boasts a high number of websites today, with more and more being created and put online every year. According to Internet Live Stats [2], there were 1,766,926,408 websites online in 2017, which is an increase of 69% to 2016, where 1,045,00,534,808 had been counted. Otherwise, the average number of users per website has been consistently decreasing from 24 in 2000, to 9.1 in 2008, to 3.7 in 2015. This simultaneous increase and decrease is, in large part, due to a growing number of *inactive* websites, such as parked domains (i.e., the domain is reserved but not used) or expired sites. The latter often occurs when a website is abandoned by its owner, be it due to a lack of resources, interest, or both. And such abandoned websites are only one of many potential risks for unsuspecting visitors, as they are likely to be out of date and, therefore, more susceptible to injections or other attacks.

Whatever an individual's motivation might be in the end, it can be assumed that many individuals, who are not an experts in general web design, security, or privacy, have basic capabilities and desire to set up their own web presence. At the same time, these individuals may or may not be interested in improving their craft and eventually attain such expert knowledge. While the aforementioned teenager might decide to take the path of a professional web designer later in their life, he/she might just as well lose interest and abandon the website. The small-size company web developer might be a capable individual with a strong interest in web design, or they might just as well be the only available person to do what needs to be done (i.e., the only one possessing basic computer skills and a 10-year old web development handbook) without any further aspirations in this regard.

Available literature should reflect this difference in demographics and the reality of widespread web access - both from a user and developer standpoint. The aim should not be to replace standard literature but instead provide a resource containing the essentials for beginners, who are not necessarily students aiming at attaining an expert level eventually. What these individuals should have access to is an *essential minimum knowledge* about web design for privacy and security, in order to endanger neither themselves nor others who visit their sites or use their services.

This paper constitutes the first part of a series of three thematically connected papers, which describe patterns from the same pattern collection. The contribution of this paper is a set of two meta-patterns, which describe common security and privacy issues and how they can be addressed on a high level. Papers two and three will contain sets of patterns that address concrete issues (e.g., frequency of backups, data protection compliance, etc.). In this paper, we first outline relevant related work from the domains of web privacy, web security, and design patterns in Section II. We then describe the problem mining and pattern writing process in Section III. Section IV contains the two meta-patterns. In Section V, we discuss general aspects regarding ease of access of security-critical information for novices and conclude the paper in Section VI.

II. RELATED WORK

In the following, we provide a brief background on privacy and security in relation to novice users, together with an introduction to design patterns. Security and privacy are often considered to be inter-related, where an appropriately secure environment protects an individual's privacy acting within it. In the context of the Internet, both are connected to the information in relation to the user. Microsoft [3] defines both concepts as follows: "*Information privacy* refers to the user's ability to control when, how, and to what extent information about themselves will be collected, used, and shared with others. *Information security* refers to the ability of businesses and individuals to secure their computers from vulnerabilities and maintain the integrity of the stored information." When brought into relation with usability, *usable privacy* and *usable security* [4] refer, broadly speaking, to the ease (or difficulty) to interact with or implement a solution that fosters privacy, security, or both. This ease or difficulty is relative to an individual's level of expertise, which is why usability must be considered with the prospective user and their capabilities in mind. For the purposes of this paper, we focus on novice or layman web developers.

A. Layman knowledge about Privacy and Security

A crucial aspect of security decisions regarding home computers and websites, is the existing knowledge about computers, the internet and their security issues. LaRose et al. [5] found that more knowledge about general computer security issues correlates frequently with the intention to behave securely. Their results also show that people who agree on the statement that online safety is their personal responsibility, are more likely to protect themselves compared to those who do not agree. Generally, people familiar with common security measures are more likely to engage in security behaviors [6]. In comparison, Shillair et al. [7] did not find any correlation between expert and layman knowledge regarding security measures. Even though knowledge is a very important factor when making decisions regarding security and privacy, additional motivations are needed for people to tackle security decisions efficiently [8], [9].

One motivation could be the protection of important information, such as online banking passwords. Internet applications often provide guidance when creating a password. Users tend to reuse passwords across most of their accounts once a user needs to manage a larger number of password [10]. A countermeasure could be a password creation process provided by an application. On the one hand, this could make the password creation process easier. On the other hand, the result may be weaker passwords. Shay et al. [11] suggested that service providers should present password requirements with additional (visual) feedback to increase usability, carefully considering the representation of feedback and guidance.

B. Design Patterns in Software Engineering and Related Domains

Originally conceived by Christopher Alexander to capture solutions in Architecture [12], [13], the pattern approach was later adopted by the computer science community and adapted to capture problem solutions in software engineering [14], [15]. The pattern collection by Gamma et al. [16] (also known as the "Gang of Four", or "GoF" for short) is probably still the best known contemporary pattern collection for software engineering, and can, at the same time, be considered one of the fundamental pieces of modern pattern literature, as it lays out a basis for pattern elements and structure along with the actual patterns themselves.

Design patterns are structured documentations of solutions to reoccurring problems. Since individual problems are usually parts of larger problems, patterns are often collected in pattern collections, which are also referred to as *pattern languages*, which dates back to Alexander's original use of the term [13]. Patterns can occur on different levels of abstraction and are referred to as *high-* or *low-level* patterns [17], depending on whether they describe a high- or low-level problem. Patterns on the highest level of abstraction are also referred to as *meta-patterns*.

Patterns have been adopted by various disciplines [18], among them Human-Computer Interaction (HCI) (e.g., [19], [20]) and Interaction Design (e.g., [17], [21]). Munoz-Arteaga et al. [22] proposed a pattern-based methodology for information security feedback design, which is, like most domain-specific literature, intended for advanced users. According to Vlissides [23], one key attribute of patterns is that they can make expertise accessible to non-experts. Bach et al. [24] make use of this feature in their design patterns for data comics, where data-related information is communicated in an easy to read comic-format.

III. Pattern Generation and Structure

The Pattern generation process began with an interview-based problem mining process in order to address issues with a high degree of relevancy for online privacy and/or security. The pattern writing was conducted by an HCI expert with experienced in writing patterns and who had also been involved in the problem mining. The pattern contents are based on an internal state-of-the art containing guidelines [25]–[27], topic relevant scientific publications (primarily ACM, Springer, and IEEE), and information gained from the interview protocols.

The pattern format was adapted from Mirnig et al. 2016 [28]. This structure was chosen due to its relative simplicity, which should make it easier for novice readers to comprehend the pattern contents. It looks as follows:

- *Name*: A short and descriptive name describing the solution
- *Intent*: A short paragraph intended to allow the reader decide whether or not the solution applies to the context in question
- *Problem*: The problem statement
- *Scenario* One example of a suitable application context
- *Solution*: The solution description
- *Example*: At least one descriptive example of the solution
- *References*: To source the solution and provide access to more in-depth resources, where available
- *Keywords*: Intended to help structure the pattern collection

The finished initial versions then underwent one iteration workshop with two HCI researchers and two web developers, in which each pattern was rated, adapting the approach proposed by Wurhofer et al. [29], Krischkowsky et al. [30], and Mirnig et al. [28], [31]. Each pattern was rated individually for each of its subcategories (Name, Intent, etc.) and then discussed in plenum. The result was a collection of 16 patterns. The process, and resulting pattern structure are described in more detail in Mirnig et al. 2019 [32].

IV. Patterns

In the following, we present the two meta-patterns on aspects that contributes to the general security of a website as well as how to evaluate it.

A. What contributes to the security of a website?

Intent: This Pattern lists various points contributing to the security of a website. Apart from that, it also provides methods that may be used to secure your own website.

Problem Statement: The amount of cyber-attacks on websites has increased over the last few years. The main targets are especially webshops and websites dealing with user data, for example forums or service websites.

Scenario: Setting up a website without thinking about security issues is negligent. An unprotected website is a security risk not only to yourself, but for all visitors of that site.

Solution: To quote Sun Tzu, ”If you know the enemy and know yourself, you need not feat the result of a hundred battles” [33]. Thus, if you know of potential dangers and common security issues in the web world, you can protect yourself more efficiently. The following list introduces some measurements that can be used to increase the security of your website.

- Access the web server only with a safe and secure computer.
  - ○ A computer system can be considered as safe and secure, if the operating system as well as all the installed applications are up-to-date. Especially security updates play a very important role in this case.
- Dont plug in hardware from third parties to any system without checking them first.
  - ○ External storage devices can contain malware or viruses that may infect a system. Thus, they have to be checked by anti-virus software before using them on important devices.
- Limit the amount of admin accounts.
  - ○ Admin accounts usually have rights to access almost everything on a system. Therefore, only experts should have access to them.
  - ○ Limit the amount of admin accounts to a bare minimum to minimize abuse.
- Use safe and secure admin passwords.
  - ○ Studies show that longer passwords (10 characters or more) are more secure in comparison to shorter ones, even when the shorter passwords contain special characters or symbols.
- Usage of two-factor authentication.
  - ○ If a cyber-criminal is able to get hands on an admin-password a two-factor authentication can provide an additional safety barrier.
- Set up user groups and manage the access rights.
  - ○ Reading and writing privileges for users on a web server should be managed by administrators. This way important or system-relevant files can be protected against access and manipulation from unauthorized persons.
- Keep yourself up-to-date!
  - ○ Inform yourself regularly about the latest web-security issues and countermeasures.
  - ○ https://www.heise.de/security/alerts/ [ger] this website provides useful information about the latest security alerts.
- Encrypt the communication with your website (SSL (Secure Sockets Layer / TLS (Transport Layer Security)
  - ○ An encrypted communication with a website offers data security and data integrity.
- Only allow access to your website via an encrypted connection
  - ○ All communication with your website should only be possible using an encrypted connection.
  - ○ If your site is accessed via http://example.com it should be automatically redirected to https://example.com.
  - ○ Use HSTS Header to force a secured connection.
- Regular backups
  - ○ Backups may help you to be on the safe side. Damaged or corrupted files can be resorted using a backup.
  - ○ Some offer automatic backups for your site. It is advised to check the services of your own web host if they offer something similar.
  - ○ For more on this topic please refer to ”When and how often should I install updates?” [34] [ger].
- Only use software or plugins from trustworthy and credible sources.
  - ○ Software and plugins, for example for an CMS like WordPress, should only be acquired from trustworthy sources. If the source of the piece of software or plugin is unknown it should not be used.
  - ○ A trustworthy and credible source for WordPress Plugins can be found here: https://wordpress.org/plugins/. Even when the source is trustworthy, do not forget to keep an eye on comments and ratings from other users.
- Use security plugins for your CMS.
  - ○ Security plugins for a CMS are a good addition and offer various features that may improve the security of your CMS even more.
- Use scan tools to test your website.
  - ○ Scan tools can be used to scan your site for known security issues, for example the implementation of certain HTTP security headers.
- Check your anti-virus software.
  - ○ Only use anti-virus software which is known to be credible, trustworthy and updated frequently.
- XSS-Prevention.
  - ○ Check user input before sending it to the web server to prevent injection of any code (e.g., HTML, URL or JavaScript).
  - ○ XSS-prevention plugins are available for well-known CMS.
- Prevention of SQL- or code-injection (Figure 1.
  - ○ For more on this topic please refer to this blog post:https://blog.varonis.de/sql-injection-verstehen-erkennen-und-verhindern/ [ger]

```
<html>
<head>
<title>basic html sample page<
</head>
<body>
<h1>a simple sample web pa e</h1>
<img src=".../images/contact.jpeg">
<h4>by sheldon brown</h4>
<h2>demonstrating a few html features</h2>
<p>html is really a very simple language. it consists
of ordinary text, with commands that are enclosed by
"<" and ">" characters, or bewteen an "&" and a ";". <p>
</html
```

Figure 1. Metaphoric visualization of Code-Injection

Examples: Regular Updates - To ensure the security of a website and other systems it is necessary to pay attention to operating system and software updates. Especially security updates should not be neglected. Pattern ”When and how often should I install updates?” [34] [ger] provides more insight into this subject.

Encrypt the communication with your Website - Pattern How do I encrypt the communication with my website [35] [ger] explains how to implement a SSL/TLS Encryption into your own website.

ScanTools - After securing the website and the associated server, it is possible to test them with ScanTools for known vulnerabilities. ScanTools examine a variety of security aspects of websites and may help you to find out whether your site is lacking in terms of security. More on this topic can be found in Pattern ”How do I check the security of my website?” [36] [ger].

Keeping yourself up-to-date! - Following websites can be used to keep yourself informed about the latest web-security issues.

- https://heise.de/security/
- http://seclists.org

Prevent XSS (Cross Site Scripting) - The following pages provide the necessary knowledge on how to prevent XSS:

- Cross-Site Scripting (XSS) unterbinden [37]
- Cross Site Scripting Prevention Cheat Sheet [38]

HSTS Header on an Apache Server - The following code can be inserted into your .htaccess file to enforce an encrypted connection with your website if it is implemented:

```
# Use HTTP Strict Transport Security to force
    client to use secure connections only

Header set Strict-Transport-Security
    "max-age=3600" env=HTTP
```

For more information, please visit: How do I activate HSTS for my website? [39] [ger]

References: Studies on the subject of password security -

Guess Again (and Again and Again): Measuring Password Strength by Simulating Password-Cracking Algorithms [40]

The true cost of unusable password policies: password use in the wild [41]

Multifactor Authentication -

Multifactor Authentication [42] [ger]

Keywords: Security, Encryption, Authentication

B. How do I check the security of my website?

Intent: This Pattern shows common methods used to test the security of websites.

Problem Statement: To ensure the safety of computers users usually trust in anti-virus and malware software. Unfortunately, it is not that simple for websites. Keeping a website safe and secure is more demanding and requires more attention rather than only installing a certain piece of software or plugin. Websites can be different and diverse, so are their weaknesses and vulnerabilities.

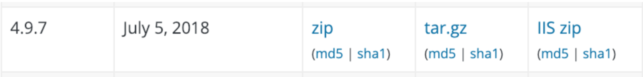
Scenario: Before you open your website to the world, it is recommended to perform a vulnerability scan to test the security of your site.

Solution:

- Check whether your site offers encrypted communication.
  - ○ The website should only accept and allow encrypted communication.
  - ○ For more information on this topic, please refer to Pattern ”How do I encrypt the communication with my website?” [35] [ger].
- Run a virus and malware scan over all files on the web server.
  - ○ Check whether your web hosting service provider offers virus protection services.
  - ○ In case users are allowed to upload files to your web space, scan them before they are actually uploaded onto the web server.
  - ○ To be on the safe side, avoid file upload features for users in general.
- Verify check sums of software and downloaded files you want to install onto the server.
  - ○ Software and Plugins from trusted sources usually offer check sum values for the files you can download.
  - ○ In case no check sum values are provided double check the credibility of the site offering the download.
- Use online ScanTool websites to have your site tested by a third party.
  - ○ In general, ScanTool websites scan your website for known security issues and vulnerabilities (e.g., whether SQL- or Code-injection is possible or whether a n encrypted connection is mandatory to access your website).
  - ○ Some ScanTool websites even offer explanations and solutions for certain vulnerability if they find one on your site.
- Provide minimal errors to your users.
  - ○ Do not provide full exception details in your error messages. Keep them simple. The more information you reveal in your error messages, the easier it is to possibly exploit them.

Examples: Verifying Check Sums -

There are several types of file check sums that are used to verify downloaded files. One of the widely used and most popular is the MD5 check sum. In this example, we will check the MD5 check sum of a WordPress 4.9.7 installation package in a .zip file (see Figure 2. The MD5 check sum in this case is provided on the WordPress download page and can be accessed by clicking the md5 link 2 under the file type downloading. Clicking this ink will open a new page, showing the associated MD5 check sum for the file. In this example, the check sum for the zip file is: 075a6e7585c61e3aa2874d91d32bc336.



4.9.7	July 5, 2018	zip (md5 | sha1)	tar.gz (md5 | sha1)	IIS zip (md5 | sha1)

Figure 2. WordPress Version 4.9.7 Download Page

After downloading the zip file, use the terminal to acquire the check sum of that file.

Terminal Command for macOS:

```
# Command
md5 wordpress-4.9.7-de_DE.zip

# Output
MD5 (wordpress-4.9.7-de_DE.zip) =
    075a6e7585c61e3aa2874d91d32bc336
```

Terminal command for Windows:

```
# Command
certUtil -hashfile wordpress-4.9.7-de_DE.zip
    MD5

# Output
MD5-Hash of wordpress-4.9.7-de_DE.zip:
    075a6e7585c61e3aa2874d91d32bc336
```

Now, compare the check sum from the terminal with the check sum provided on the website where you downloaded the file. If they match, data integrity is given and the file is safe to use.

For more information on the subject of check sums please refer to https://itsfoss.com/checksum-tools-guide-linux/.

ScanTools -

Mozilla Foundation Security Check: This security check provided by the Mozilla Foundation aims to help you to configure your site safely and securely. If you want to test your site, just type in the URL of the site you want to be scanned in here: https://observatory.mozilla.org.

SSL Labs - SSL Labs performs a deep analysis of the performs a deep analysis of the configuration of any SSL web server on the public Internet. It is pointed out by the provider, that this service is purely for information purposes and is not recommended for commercial purposes.

- https://www.ssllabs.com/ssltest/

Additional ScanTool Service Websites

- https://www.virustotal.com/de/ [ger]
- https://www.htbridge.com/websec/

Important Note: Please carefully observe the terms and conditions of the respective providers before you use their services.

References: Mozilla's security test for websites [43] [ger].

Does a web server need an anti virus software installed? [44].

Keywords: Security, Integrity, ScanTools, Malware

V. Discussion

Keeping things simple is important to keep them understandable and it is at this point where we must ask the question on how we can lower the educational access barrier to fit in with the ease of on-line content creation of today. There are many individuals out there who have access to a great number of content management systems and similar tools right at their fingertips. By using these with a "website first, security last"-mentality, they will endanger both themselves and those using their creations. Many of these creators have no aspirations to ever become experts, be it out of necessity or simple disinterest.

Restricting the access of these creators is not an attractive option and would run counter to the freedom of today's online world. What these individuals need are resources that reflect not only their level of expertise but also their goals and needs. If all these individuals need is an adequately secured website, then they should have the information on how to do so. This seems contrary to most sound educational intuitions, as it would essentially mean to aim for mediocrity. However, this kind of mediocrity would not compete with a superior solution but with no solution at all instead.

Using such patterns to inform one's web development will not lead to not fully secured websites. Instead, they are a resource provide an adequate minimum of security and privacy on-line. This should not be seen as to be in competition with available professional and educational literature. Rather, it supplements it. The internet of today is a much more diverse place than it was even a decade ago. In order to increase everyone's security and privacy, available information must reflect this diversity. And when some individuals aim either low or not at all as far as privacy and security are concerned, then giving them the means to at least aim low seems to be an overall improvement.

VI. Conclusion

Designing a website with privacy and security in mind from the beginning is a nontrivial task for both novices and experienced professionals. There are limits to what can be communicated in a way so that it can be understood and implemented by novices. Nonetheless, a certain minimum is as helpful as it is necessary in order to improve the online experience for everyone. In this paper, we provided two meta-patterns as guidance regarding what constitute privacy- and security-relevant aspects of a website, and how these attributes can be verified in the end. Concrete instructions on the implementation of such solutions is still necessary. Such instructions must, just like the high-level meta-patterns, be written in a way that is suitable for novices. Thus, additional lower-level patterns are necessary. In Parts 2 and 3, we provide two sets of such pattern solutions. Future work will focus on continually extending the pattern solutions to cover more issues, as well as updating existing patterns in order to keep their solutions valid and usable.

Acknowledgment

The financial support by the Internet Privatstiftung Austria (IPA) under the program "netidee" with the title "SecPatt" under grant number 2390 is gratefully acknowledged.

References

[1] "World Wide Web (W3)," http://info.cern.ch/hypertext/WWW/TheProject.html, (accessed April 10, 2019).

[2] "Total number of Websites," https://www.internetlivestats.com/total-number-of-websites, 2019 (accessed April 10, 2019).

[3] Privacy and security on the microsoft developer network. [Online]. Available: https://msdn.microsoft.com/en-us/library/ms976532.aspx (2018)

[4] A. Amran, Z. F. Zaaba, M. M. Singh, and A. W. Marashdih, "Usable security: Revealing end-users comprehensions on security warnings," Procedia Computer Science, vol. 124, 2017, pp. 624 – 631, 4th Information Systems International Conference 2017, ISICO 2017, 6-8 November 2017, Bali, Indonesia. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050917329666

[5] R. LaRose, N. J. Rifon, and R. Enbody, “Promoting personal responsibility for internet safety,” Commun. ACM, vol. 51, no. 3, Mar. 2008, pp. 71–76. [Online]. Available: http://doi.acm.org/10.1145/1325555.1325569

[6] N. Kumar, K. Mohan, and R. Holowczak, “Locking the door but leaving the computer vulnerable: Factors inhibiting home users’ adoption of software firewalls,” Decis. Support Syst., vol. 46, no. 1, Dec. 2008, pp. 254–264. [Online]. Available: http://dx.doi.org/10.1016/j.dss.2008.06.010

[7] R. Shillair, S. R. Cotten, H.-Y. S. Tsai, S. Alhabash, R. LaRose, and N. J. Rifon, “Online safety begins with you and me,” Comput. Hum. Behav., vol. 48, no. C, Jul. 2015, pp. 199–207. [Online]. Available: http://dx.doi.org/10.1016/j.chb.2015.01.046

[8] D. Lee, R. Larose, and N. Rifon, “Keeping our network safe: A model of online protection behaviour,” Behav. Inf. Technol., vol. 27, no. 5, Sep. 2008, pp. 445–454. [Online]. Available: http://dx.doi.org/10.1080/01449290600879344

[9] C. L. Anderson and R. Agarwal, “Practicing safe computing: A multimedia empirical examination of home computer user security behavioral intentions,” MIS Q., vol. 34, no. 3, Sep. 2010, pp. 613–643. [Online]. Available: http://dl.acm.org/citation.cfm?id=2017470.2017481

[10] S. Pearman, J. Thomas, P. E. Naeini, H. Habib, L. Bauer, N. Christin, L. F. Cranor, S. Egelman, and A. Forget, “Let’s go in for a closer look: Observing passwords in their natural habitat,” in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, ser. CCS ’17. New York, NY, USA: ACM, 2017, pp. 295–310. [Online]. Available: http://doi.acm.org/10.1145/3133956.3133973

[11] R. Shay, L. Bauer, N. Christin, L. F. Cranor, A. Forget, S. Komanduri, M. L. Mazurek, W. Melicher, S. M. Segreti, and B. Ur, “A spoonful of sugar?: The impact of guidance and feedback on password-creation behavior,” in Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, ser. CHI ’15. New York, NY, USA: ACM, 2015, pp. 2903–2912. [Online]. Available: http://doi.acm.org/10.1145/2702123.2702586

[12] C. Alexander, The Timeless Way of Building. New York, USA: Oxford University Press, 1979.

[13] C. Alexander, S. Ishikawa, and M. Silverstein, A Pattern Language: Towns, Buildings, Construction. New York, USA: Oxford University Press, 1997.

[14] J. O. Coplien, Software Patterns. New York, USA: SIGS Books, 1996.

[15] K. Quibeldey-Cirkel, Design Patterns in Object Oriented Software Engineering. *O*rig. title: Entwurfsmuster: Design Patterns in der objektorientierten Softwaretechnik. Berlin, Germany: Springer, 1999.

[16] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Pearson, 1994.

[17] J. O. Borchers, “A pattern approach to interaction design,” AI & SOCIETY, vol. 15, no. 4, 2001, pp. 359–376.

[18] S. Köhne, A Didactical Aproach towards Blended Learning: Conception and Application of Educational Patterns. *O*rig. title: Didaktischer Ansatz für das Blended Learning: Konzeption und Anwendung von Educational Patterns. Hohenheim, Germany: University of Hohenheim, 1995.

[19] A. Dearden and J. Finlay, “Pattern languages in hci: A critical review,” Human-Computer Interaction, 2006, pp. 49–102, Sheffield Hallam University. [retrieved: 02, 2016] URL: http://research.cs.vt.edu/ns/cs5724papers/dearden-patterns-hci09.pdf.

[20] A. F. Blackwell and S. Fincher, “PUX: Patterns of User Experience,” Interactions, vol. 17, no. 2, 2010, pp. 27–31.

[21] M. V. Welie and G. C. V. D. Veer, “Pattern languages in interaction design: Structure and organization,” in Proc. Interact ’03, M. Rauterberg, Wesson, Ed(s). IOS. IOS Press, 2003, pp. 527–534.

[22] J. Muoz-Arteaga, R. M. Gonzlez, M. V. Martin, J. Vanderdonckt, and F. lvarez Rodrguez, “A methodology for designing information security feedback based on user interface patterns,” Advances in Engineering Software, vol. 40, no. 12, 2009, pp. 1231 – 1241, designing, modelling and implementing interactive systems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0965997809000519

[23] J. Vlissides, Pattern Hatching: Design Patterns Applied. Addison-Wesley, 1998.

[24] B. Bach, Z. Wang, M. Farinella, D. Murray-Rust, and N. Henry Riche, “Design patterns for data comics,” in Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, ser. CHI ’18. New York, NY, USA: ACM, 2018, pp. 38:1–38:12. [Online]. Available: http://doi.acm.org/10.1145/3173574.3173612

[25] Oecd privacy guidelines. [Online]. Available: http://www.oecd.org/sti/ieconomy/privacy-guidelines.htm (2013)

[26] Privacy and security on google for education. [Online]. Available: https://edu.google.com/k-12-solutions/privacy-security/?modal_active=none (2018)

[27] Microsoft security guide. [Online]. Available: https://technet.microsoft.com/en-us/library/bb794718.aspx (2018)

[28] A. Mirnig, T. Kaiser, A. Lupp, N. Perterer, A. Meschtscherjakov, T. Grah, and M. Tscheligi, “Automotive user experience design patterns: An approach and pattern examples,” International Journal On Advances in Intelligent Systems, vol. 9, 2016, pp. 275–286.

[29] D. Wurhofer, M. Obrist, E. Beck, and M. Tscheligi, “A quality criteria framework for pattern validation,” International Journal On Advances in Software, vol. 3, no. 1 and 2, 2010, pp. 252–264.

[30] A. Krischkowsky, D. Wurhofer, N. Perterer, and M. Tscheligi, “Developing patterns step-by-step: A pattern generation guidance for hci researchers,” in PATTERNS 2013, The Fifth International Conferences on Pervasive Patterns and Applications. IARIA, 2013, pp. 66–72. [Online]. Available: http://www.thinkmind.org/index.php?view=article&articleid=patterns_2013_3_30_70053

[31] A. G. Mirnig, A. Meschtscherjakov, N. Perterer, A. Krischkowsky, D. Wurhofer, E. Beck, A. Laminger, and M. Tscheligi, “User experience patterns from scientific and industry knowledge: An inclusive pattern approach,” International Journal On Advances in Life Sciences, vol. 7, no. 3 and 4, 2015, pp. 200–215. [Online]. Available: https://www.thinkmind.org/index.php?view=article&articleid=patterns_2015_2_30_70011.

[32] A. G. Mirnig, A. Lupp, A. Meschtscherjakov, E. Economidou, and M. Tscheligi, “Security patterns for webdesign: a hierarchical structure approach,” in CHI Conference on Human Factors in Computing Systems Extended Abstracts, ser. CHI’19 Extended Abstracts. New York, NY, USA: ACM, 2019. [Online]. Available: http://doi.acm.org/10.1145/3290607.3312789

[33] S. B. Griffith, Sun Tzu: The art of war. Oxford University Press London, 1963, vol. 39.

[34] SecPatt, “When and how often should I install updates? [ger],” https://www.secpatt.at/patterns/pt_1/, 2018 (accessed April 10, 2019).

[35] SecPatt, “How do I encrypt the communication with my website? [ger],” https://www.secpatt.at/patterns/pt_4/, 2018 (accessed April 10, 2019).

[36] SecPatt, “How do I check the security of my website? [ger],” https://www.secpatt.at/patterns/pt_6/, 2018 (accessed April 10, 2019).

[37] “Prevent Cross-Site Scripting (XSS) [ger],” https://www.php-kurs.com/cross-site-scripting-xss-unterbinden.htm, (accessed April 10, 2019).

[38] “Cross Site Scripting Prevention Cheat Sheet,” https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.md, (accessed April 10, 2019).

[39] “How do I activate HSTS for my website?” https://www.cyon.ch/support/a/wie-aktiviere-ich-http-strict-transport-security-hsts-fur-meine-website, (accessed April 10, 2019).

[40] “Guess Again (and Again and Again): Measuring Password Strength by Simulating Password-Cracking Algorithms.” https://ieeexplore.ieee.org/abstract/document/6234434/?part=1, 2012 (accessed April 10, 2019).

[41] “The true cost of unusable password policies.” https://dl.acm.org/citation.cfm?id=1753384, 2010 (accessed April 10, 2019).

[42] “Multifactor Authentication,” https://www.onlinesicherheit.gv.at/praevention/konten_und_passwoerter/mehrfaktor-authentifizierung/249584.html, 2017 (accessed April 10, 2019).

[43] “Mozilla introduces free security test for websites,” https://www.heise.de/ix/meldung/Mozilla-bringt-kostenlosen-Sicherheitstest-fuer-Websites-3306197.html, 2016 (accessed April 10, 2019).

[44] “Does a web server need an anti virus software installed?” https://security.stackexchange.com/questions/245/does-a-webserver-need-an-antivirus-software-installed, 2010 (accessed April 10, 2019).

Web Security and Privacy for Novices – Part 2

Updates, Mail Servers, and E-Commerce

Artur Lupp*, Alexander G. Mirnig* and Manfred Tscheligi†
*Center for Human-Computer Interaction
University of Salzburg, Salzburg, Austria
Email: firstname.lastname@sbg.ac.at
†Center for Human-Computer Interaction &
Austrian Institute Of Technology, Salzburg & Vienna, Austria
Email: firstname.lastname@sbg.ac.at

***Abstract*—This paper is the second part of a series of three thematically connected papers and presents four patterns for nonprofessional web developers. The patterns in this part of the series address updates, setting up and maintaining mail servers and securing a web shop. Starting with a pattern that provides basic information necessary for any type of web presence, while the subsequent patterns provide more in-depth information specializing on specific topics.**

***Keywords*–Patterns; On-line; Security; Privacy; Novice Users.**

I. Introduction

Never has it been that easy to construct and set up a website. The internet offers a vast amount of easy to use tools and websites that allow nonprofessionals to develop, design and shape a personal web presence. This web presence can then be put online with only a few mouse clicks. Without the proper knowledge about web security and web privacy however, these websites may be turned into a potential threat to the internet community. Poorly set-up and maintained websites can easily be hijacked by cyber criminals. The criminals usually compromise the site in a way, that they start to distribute malware or ransomware to infect the computers of inexperienced users visiting the website.

We created a set of patterns describing common security and privacy issues and how they can be addressed on a high level. This paper constitutes the second part of a series of three thematically connected papers, which describe patterns from the same pattern collection. Paper one contains two meta-patterns and a general description how the patterns were generated and structured. Paper three contains a set of four additional patterns addressing concrete issues (e.g., frequency of backups, data protection compliance, etc.).

In this paper, we will first outline relevant related work from the domains of web privacy and web security focusing on updates, mail servers and the security of web shops. Section II provides a short outline of relevant related work from the aforementioned focus areas. Section III introduces the four patterns with the titles: ”When and how often should I install updates?”, ”Should I set up or operate a mail server myself?”, ”How should I set up / operate a mail server myself?” and ”How do I secure a web shop and what should be taken into consideration?”. Section IV concludes this paper.

II. Related Work

Updates are intended to improve previously installed programs. The necessary changes may range from simple and invisible changes, such as bug fixes or improved security patches up to major changes in the user interface, as well as the addition of new functions or changes of already existing features that may affect user workflow. Non-expert users tend to skip updates due to previous negative experiences, e.g., unwanted and unexpected changes in the user interface or annoying ”update now” pop-up messages [1]. Skipping updates leads to more compromised computers, even though they could have been protected as security updates were available but had not yet been applied [2]. A more recent Security Intelligence Report from Microsoft [3] shows that the amount of compromised computers is still rising mainly because cyber criminals prefer easy targets, such as novice users and non-experts. It is easier to compromise computers and servers that are not efficiently protected due to missing protection software, security updates or poorly configured environments. Even when a computer, server or website uses protection software and is up-to-date, the exploitation of access control mechanisms with weak passwords is one of the bigger problems. Pearman at al. [4] show that a lot of users reuse passwords when they have to manage a large number of them. Additionally, there is trend to use weak passwords, if the system is allowing the creation and use of them [5].

III. Patterns

A. When and how often should I install updates?

Intent: This pattern addresses the topic of updates for software applications and operating systems on the personal computers, as well as on servers. The main aim is to answer when, how often and why updates should be installed in general.

Problem Statement: Keeping software or systems up to date is very important. Systems which are not updated frequently are more vulnerable to attacks from the outside compared to systems that are continuously held up to date.

Scenario: In order to secure your own system and reduce the vulnerabilities attackers may use to get into your system, it is necessary to maintain your system with the newest updates.

Solution: Updates should be installed in **regular intervals**. However, this definition does **not precisely define the actual time period**.

Therefore, we recommend the usage of automatic updates if an application is offering this feature. Important security-related programs (e.g., anti-virus software or operating systems) should be checked for updates on a daily basis. While, for all other installed programs its sufficient to check for updates on a weekly or monthly basis.

Before the installation of any kind of update, we recommend to create a system backup for safetys sake.

In unfavorable cases, updates may lead to problems or incompatibilities. Especially major updates or version jumps in programs may be irreversible. Therefore, it is advised to plan ahead and calculate some time to test the system thoroughly after after bigger version jumps, upgrades and updates. More about backups can be found in pattern: When and how should I create backups? [6].

- It is recommended to keep **systems and software up-to-date from the beginning**.
  - ○ Up-to-date means, that all (primarily security-relevant) updates for a system or software currently available are installed.
- Systems, software or websites working with **sensitive data** have **higher priority**.
  - ○ It is recommended to check for updates on a daily basis. This applies to systems and computers accessing the web server and the applications running on the server. This point is especially important, if the website is working with personal or sensitive data (account data, addresses, etc.).
  - ○ Usually, the hosting provider takes care core application updates (e.g., operating system, mySQL, PHP, etc.).
- Establishing of an **update day** is recommended. For example, **once a month** for the whole system or **once a week or day** for systems or applications that **handle important or personal data**.
- Automated updates are recommended, but you should also look **manually** regularly **(once a week)**.
  - ○ In certain cases, it is possible that an automatic update messes up the automatic update function. In this case, manual updates are the way to go.
- Set up an information network which informs you about update releases. A couple of software and hardware providers offer mailing lists and info websites for that purpose.

In case certain software is not being updated anymore, it is advisable to look for alternatives that are kept up to date.

Examples: Automated Updates - At certain time intervals (e.g., hourly, daily or weekly), the system automatically searches for updates.

Patch Day - On a patch day, all updates currently available will be applied at a previous determined fixed day (e.g., once a week or once a month). The updates are usually tested on a separate system before being applied onto the main system, in order to eliminate any problems and / or compatibility issues.

Software Update Strategies - Create your own update strategy. Set up a schedule for update checking and deployment. This allows you to have a complete overview to when and if an update was deployed, which program the update was for and whether a certain machine already received an update.

WordPress update - WordPress offers two ways of updating the core application. The first option is the automatic update function and the second is the manual update.

- To utilize the automatic update function of WordPress, go to the WordPress Admin Panel and select ”Updates” on the left sidebar. In case that the WordPress version was installed with the default settings, you can also access this update function directly using the following url:www.example.com/wp-admin/update-core.php (replace example.com with your own url). This page shows you the currently available updates for the WordPress core installation, plugins, translations and themes. If updates are available, they can be applied by clicking the update button for the corresponding application or component.
- In order to update WordPress manually, please refer to the instructions provided on this page: Manually Update WordPress [ger].

Important Mailing Lists - The article linked below, offers interesting facts regarding updates. Furthermore, it provides important mailing lists at the end of the article: Important Mailinglists [7].

Additional mailing lists and more information can be found here: Additional Links, Mailinglists and Newsgroups [8].

References

D. L. Parnas, Software aging, Proceedings of 16th International Conference on Software Engineering, Sorrento, 1994, pp. 279-287. [9]

Bellissimo, Anthony, John Burgess, and Kevin Fu. Secure Software Updates: Disappointments and New Challenges. HotSec. 2006. [10]

Understanding Patch and Update Management: Microsofts Software Update Strategy [11]

Apple security updates [12]

SCCM Software updates Strategy [13]

Why updating your Software is a Must Do [14]

WordPress Security Category Archive [15]

Keywords

Updates, Security, Software, Backups

B. Should I set up or operate a mail server myself?

Intent: This pattern aims to help users to make a decision whether they should run their own mail server or not.

Problem Statement: The administration and maintenance of a mail server complicated and unfortunately often underrated. There are a vast amount of guides on the internet, that aim to explain how to set up a mail server. However, setting up, running and maintaining a mail server requires a lot of specialist knowledge. Unfortunately, laymen do not or only insufficiently possess this kind of knowledge.

Scenario: Sufficient web space and a domain were acquired. The page is now online. For obvious reasons, it seems like a good idea to get mail addresses matching your domain by setting up a mail server by yourself. But is that a good idea?

Solution

Operating and maintaining Mail servers is a complex task. If you are unsure whether you have the skills or not, don not risk anything and leave it to professionals.

The following points should be considered before the decision, whether it makes sense to operate your own mail server:

- **The optimal setup of a mail server is complicated.**
  - It is deceptively easy to set up a mail server. Myriads of guides and a hand full of easy to use applications can be found online. Unfortunately the security aspect is neglected in many of these guides, as it is time consuming and complicated.
  - Setting up a mail server requires expertise and certain knowledge of UNIX systems.
  - The default settings are definitely neither the best, nor are they recommended! Running a mail server using the default setting should be avoided at all costs!
- **Set-up a spam filter.**
  - A spam filter reduces the number of spam emails that would otherwise end up in users' mailboxes.
  - Spam filters should always be adjusted very carefully. Additionally, the filter needs readjustments every now and then, as the content of the spam emails will change over time. Setting up a good and reliable spam filter is very time consuming.
- **Antivirus Software**
  - Emails, especially with attachments should always be checked by an antivirus software before redirecting them to the receiver.
  - Security mechanisms like this have to be installed and adjusted. Additionally, they need to be kept up-to-date in order to do their work.
- **Offering webmail access.**
  - Usually, users want to retrieve their emails not only through a software. A web platform might be mandatory.
  - This web platform (usually, a piece of software installed on a webserver, e.g., roundcube [16]) has to hosted and adjusted.
  - This software has to be kept up-to-date. Backups and the maintenance are mandatory.
- **Blacklisting.**
  - The main task of a mail server is to send emails to the addressed receivers. This might not be possible if your own mail server is on a blacklist. Blacklisted mail servers usually fail to deliver the messages, as they will be blocked by the mail server at the receivers end.
  - Once a mail server is on a blacklist, removing it from the list might take a while.
  - A mail server is blacklisted in case it is used to send out spam mails. This usually happens when the mail server is compromised due to incorrect or faulty configuration.
- **Time consuming maintenance.**
  - Backups are mandatory and data recovery must be guaranteed.
  - All updates must be tested in advance.
    - Even the smallest changes can render the mail server dysfunctional.
  - Finding a problem in case the server is malfunctioning, is difficult.
    - Assuming that emails can no longer be received by other users; where should you start to look for the problem?

These are just but a few of many issues, illustrating how **complex** the operation and maintenance of a mail server is. Please consider whether it is worth the risk and the time hosting a mail server by yourself. There are a lot of companies offering low-priced email services with included maintenance, antivirus checks and spam filters.

The easiest and safe way to a mail server is though a email service provider!

Examples

Comparison of Mail Servers: The following link provides a comparison of mail servers (e.g., mail transfer agents, mail delivery agents, and other related software proving e-mail services):

https://en.wikipedia.org/wiki/Comparison_of_mail_servers

References

Roundcube - Free and Open Source Web mail Software [16]

Why You May Not Want To Run Your Own Mail Server [17]

Keywords

Mail Server, Security, Blacklist

C. How should I set up / operate a mail server myself?

Intent: This pattern elucidates the available options to secure your own mail server.

Problem Statement: Secure communication and data integrity are very important and highly valued in digital communications. Imagine that your email end up in the recipient's spam folder, or even worse, it will be blocked by the recipients mail server before it even reaches the in box. Whether your mails will be blocked before they reach the destination or not, is usually decided by the security and configuration of your own mail server.

Scenario: An mail address using the domain name should has to be set up by you. How can you set it up a secure mail service on your own server?

Solution: **Before you start to setting up your own mail server, please read the pattern Should I set up or operate a mail server myself?" [18]. Setting up and running a mail server is time consuming and complex.**

Definition of Terms - **SMTP** is the acronym for Simple Mail Transfer Protocol. This protocol is used to send or receive emails from different mail servers. **IMAP** and **POP3** are transmission protocols used to retrieve emails from a mail server.

In case a mail software is already installed on the server, step 1 can be skipped.

1. Install an email software application on the server - Independent of the mail application which will be used, following points need to be clarified in advance:

- Check whether your server meets the requirements for the software you intend to use.
  - Only one SMTP application per server is recommended.
- Pay attention to possible incompatibilities.
  - Remove previous SMTP applications completely, before installing a new one.
- Use only SMTP applications that are general available.
  - Applications handling any kind of communication should always be kept up-to-date.

It is recommended to use the IMAP protocol when retrieving the emails from the mail server. IMAP synchronizes the client (mail program on the smartphone or computer) with the server, when the client connects to the server. This synchronization allows other devices and applications to catch up to the most recent status. Therefore, you will not see already read emails as unread.

2. Securing the mail server -

Following points should always be kept in mind:

- **NEVER** use the default settings when setting up or securing a mail server.
- Immediately change the default passwords.
- Change passwords in regular intervals (especially admin accounts).
  - More information regarding passwords can be found in the pattern ”How do I secure a web shop and what should be taken into consideration?” [19] please refer to the sections ”Solution” and ”Examples”.
- All settings, configurations and functions should be tested **before** applying them.
- The functionality and security of the mail server has to be tested regularly.

After installing the mail server - please check your configuration and adjust the options as listed below to increase the security:

- Activate Secure Sockets Layer (SSL) / Transport Layer Security (TLS) encryption.
- Deactivate SSLv2 and SSLv3.
- Activate SMTP-AUTH.
- Activate TSL encryption for incoming and outgoing emails.
- Reduce the number of possible connections to mitigate DDoS attacks.
- Decrease the number of failed login attempts.
- Use DNS-blacklists to intercept spam mails.
- Use reverse DNS LookUp to verify the sender.
- Activate SPF (Sender Policy Framework).
- Create local blacklists. In case it is necessary to block IP-addresses manually.

Examples: A good guide for setting up a mail server

- https://workaround.org/ispmail

rspam Anti Spam Tool

- https://www.openhub.net/p/10349

Install Postfix on RedHat

- https://tecadmin.net/install-and-configure-postfix-on-centos-redhat/

Install Postfix on Ubuntu

- https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-postfix-on-ubuntu-16-04

Secure Postfix with Lets Encrypt (SSL)

- https://www.upcloud.com/support/secure-postfix-using-lets-encrypt/

Additional guide for setting up a mail server with Postfix

- https://www.codeproject.com/Articles/847650/How-to-Install-Configure-Email-Server-with-Postfix

SMTP authentication for Postfix

- http://postfix.state-of-mind.de/patrick.koetter/smtpauth/smtp_auth_mailclients.html

DNS-Blacklist in Postfix

- https://docs.iredmail.org/enable.dnsbl.html

Further configuration examples and improvements for Postfix

- http://www.postfix.org/TUNING_README.html

SSL/TLS Encryption

- Pattern ”How do I encrypt the communication with my website?” [20] explains how to acquire an SSL/TLS certificate. This certificate can also be used for the encryption of your email transfer. However, in order to use the same certificate make sure, that the mail server is using the same domain (e.g., example.com) as your website and not a sub domain (e.g., mail.example.com)
- Change DNS MX entry (record) of the mail application from @mail.example.com to @example.com in order to get it working.

SMTP diagnostic tool to test your own SMTP configuration - https://mxtoolbox.com/diagnostic.aspx

References

https://www.upcloud.com/support/secure-postfix-using-lets-encrypt/

https://webmasters.stackexchange.com/questions/83442/single-ssl-certificate-for-web-and-email

https://wiki.archlinux.org/index.php/Virtual_user_mail_system

https://workaround.org/ispmail/jessie

https://www.howtoforge.com/effective_mail_server_defense

https://www.syn-flut.de/mit-postfix-spam-blockieren

https://blog.returnpath.com/blacklist-basics-the-top-email-blacklists-you-need-to-know-v2/

https://www.alienvault.com/blogs/security-essentials/basic-best-practices-for-configuring-email-servers

https://www.linode.com/docs/email/postfix/postfix-smtp-debian7/

http://www.postfix.org/BASIC_CONFIGURATION_README.html

Keywords

eMail, Mail Server, Security, SMTP, POP3, IMAP

D. How do I secure a web shop and what should be taken into consideration?

Intent: This pattern explains which aspects you need to keep in mind if you want to designing a good web shop focusing on security.

Problem Statement: Online shopping is getting more and more important. With its growth, the number of users accounts on e-commerce platforms exploded. Nowadays web shops, e-commerce places and other web presences with a lot of users are popular target for cyber criminals.

Scenario: Creating and setting up an online shop is quite simple with the right guides. Keeping the website and customer data safe however, can be challenging. Storing customer data safe and secure should be the highest priority!

Solution

Maintaining and operating an online shop is a lot of work. The recently introduced EU GDPR has strengthened the rights of customers. Thus, if you want to avoid any problems especially legal ones - consider to consult a professional web developer for the creation and maintenance of a web shop.

Attention: This pattern is not a legal advice! We addressed the GDPR (https://www.dsb.gv.at/gesetze-in-osterreich [ger]) and applicable data protection regulations during our research for the patterns, however, we are no legal advisors, nor are we lawyers or privacy experts. We shall not have any liability whatsoever for the accuracy, completeness, timeliness, or correct sequencing of the provided information.

If you still want to run a web shop by yourself after this warning, please pay attention to the following points:

GDPR - As the owner and operator of a web shop, you have to design and set up the shop in a GDPR compliant way. Some of the changes are simple and require only the addition of business information and contact details on your site, other changes may be technically more challenging. Please consider visiting this side https://www.wko.at/service/wirtschaftsrecht-gewerberecht/AGB_im_Internet_-_im_Detail.html [ger] in order to get informed on what information you have to provide to your visitors to be GDPR compliant.

Encrypted Communication - It is absolutely necessary to encrypt the communication with your web shop. Especially if you offer direct payment options on your site. The encrypted communication is used to ensure data integrity.

Data and System Security - The GDPR requires you to keep your system up-to-date! The data security on the systems has to be guaranteed.

Shop certificates and quality seals - Shop certificates and seals of approval create trust in the web shop. These certificates suggest the customer, that the web shop meets certain quality standards (e.g., data security, etc.). Many trusted certificate or seal of approval providers check the website and the associated system, before they evaluate them. Depending on different factors, they decide whether the web shop is worthy of receiving a seal of approval or a trusted certificate.

Secure Passwords - Most of the time users are required to create an account, before they can order from a web shop. During the account creation process, they usually have to pick a password. It is recommended to aid the users during the password creation process, persuading them to use stronger passwords. Recent studies show a longer password (8 characters or more) is much more secure than a password with less characters (even when they are using special characters).

The following rules can be set as an requirement to persuade users to create a stronger password:

- The password should contain at least one uppercase and lowercase character.
- One special character is required.
  - , (,), =, !, etc.
- Avoid simple words, birthday dates, names or repeated letter sequences, such as:
  - asdfasdf, 123456, Heinz1, 12.10.2019, 121212, Superman
- Minimum length should be 8 characters.

Examples

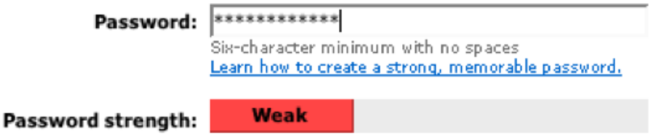
Safety Aspects - How to set up an SSL/TLS encrypted communication on a website is explained in the pattern ”How do I encrypt the communication with my website?” [20]. As a web shop works with personal data (e.g., names, addresses, banking information, etc.) the safety of this data has to be assured. The following patterns explain how to assure this: What contributes to the security of a website? [21], When and how should I create backups? [6], How do I store data securely? [22], Which data am I allowed to save? [23], and What information do I have to provide to visitors of my website? [24].

A List of Providers for Trust Certificates and Seals of Approval

- https://www.trustedshops.at
- https://www.tuev-sued.de/fokus-themen/it-security/safer-shopping/onlinehaendler
- https://www.datenschutz-cert.de/ips-internet-privacy-standards.html
- https://ehi-siegel.de

Data Informative - The GDPR gives individuals a right to be informed about the collection and use of their personal data. If a user requests his personal data, the operator of the site has to provide the request information within one month. More Information concerning this topic can be found here: https://www.dsb.gv.at/fragen-und-antworten#Wie_beantworte_ich_ein_Auskunftsersuchen_

Password meter for secure passwords - The following Figures 123 show a password meter indicating the strength / security of a password depending on length, characters used and complexity. Figure 1 uses a password with more than the required six characters, however it is considered weak due to lack of complexity or the use of repeated strings, such as ”abcdabcd”. The approach in Figure 2 is better compared to the password shown in Figure 1, thus, it can be considered as a medium strength password. Even though the password is shorter, it may contain numbers apart from standard characters, making dictionary attacks more difficult. Figure 3 displays how a password meter would indicate the use of a strong password. A strong password should be at least 8 characters long, containing uppercase and lowercase characters, as well as numbers and special characters if possible.



Password: ************

Six-character minimum with no spaces

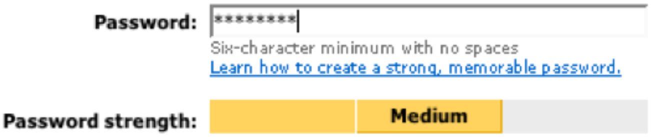
Learn how to create a strong, memorable password.

Password strength: Weak

Figure 1. Password strength comparison - weak

weak

Figure 1 indicates that the password is longer than the required 6 characters, but a repeated string may have been selected (e.g., ”asdfasdf”), making the password less secure.



Password: ********

Six-character minimum with no spaces

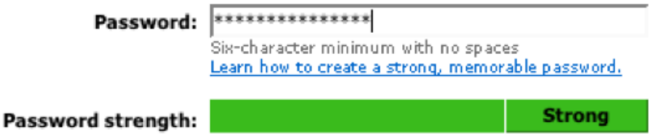
Learn how to create a strong, memorable password.

Password strength: Medium

Figure 2. Password strength comparison - medium

medium

In Figure 2, the password is shorter compared to Figure 1. However, it meets the minimum of the required 6 characters. Since it meets the requirements and probably uses numbers apart from characters it is considered more secure but not overwhelmingly safe.



Password: ***************

Six-character minimum with no spaces

Learn how to create a strong, memorable password.

Password strength: Strong

Figure 3. Password strength comparison - strong

strong

Figure 3 shows a much longer password using more characters as required and therefore, the most secure password out of the three shown examples.

References

86% of Passwords are Terrible (and Other Statistics) [25]

Wirtschaftsrecht/Gewerberecht Muster für den Bestellablauf - Info der WKO Datenschutzbehörde Österreich [26]

Keywords

GDPR, Data Privacy, Personal Data

IV. Conclusion

The patterns presented in this paper are interdependent and based on one another. The first pattern explains the necessity of updates for secure systems and offers recommendations for handling the update routine. This knowledge is the basis for the subsequent patterns in this work, addressing the set up / operation of a mail server, as well as security improvement of web shops, as they rely on safe, secure and up-to-date systems in order to keep the personal data of potential users safe and secure. While these patterns can address only a fraction of possible problems and questions that may occur during web development, they selected the most common ones, with the aim to provide applicable solutions and examples for them. Future work will focus on the extension of the problem list, while keeping the existing patterns updated to ensure the validity and usability.

Acknowledgment

The financial support by the Internet Privatstiftung Austria (IPA) under the program “netidee” with the title “SecPatt” under grant number 2390 is gratefully acknowledged.

References

[1] K. E. Vaniea, E. Rader, and R. Wash, “Betrayed by updates,” Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI ’14, 2014, doi: 10.1145/2556288.2557275.

[2] Microsoft, “Microsoft Security Intelligence Report, Volume 13, January through June 2012.” https://www.microsoft.com/en-us/download/details.aspx?id=34955, 2018 (retrieved April 10, 2019).

[3] Microsoft, “Microsoft Security Intelligence Report, Volume 23, March 2018.” https://info.microsoft.com/rs/157-GQE-382/images/EN-US_CNTNT-eBook-SIR-volume-23_March2018.pdf, 2018 (retrieved April 10, 2019).

[4] S. Pearman, J. Thomas, P. E. Naeini, H. Habib, L. Bauer, N. Christin, L. F. Cranor, S. Egelman, and A. Forget, “Let’s go in for a closer look: Observing passwords in their natural habitat,” in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, ser. CCS ’17. New York, NY, USA: ACM, 2017, pp. 295–310, doi: 10.1145/3133956.3133973.

[5] J. A. Cazier and B. D. Medlin, “Password security: An empirical investigation into e-commerce passwords and their crack times,” Information Systems Security, vol. 15, no. 6, 2006, pp. 45–55, doi: 10.1080/10658980601051318.

[6] SecPatt, “When and how should I create backups? [ger],” https://www.secpatt.at/patterns/pt_6/, 2018 (retrieved April 10, 2019).

[7] “Important Mailinglists [ger],” http://www.linux-magazin.de/ausgaben/2004/08/keine-wissensluecke/, 2004 (retrieved April 10, 2019).

[8] “Additional Links, Mailinglists and Newsgroups [ger],” http://www.netzmafia.de/skripten/sicherheit/sicher9.html, 2002 (retrieved April 10, 2019).

[9] D. L. Parnas, “Software aging,” in Proceedings of the 16th International Conference on Software Engineering, ser. ICSE ’94. Los Alamitos, CA, USA: IEEE Computer Society Press, 1994, pp. 279–287, http://dl.acm.org/citation.cfm?id=257734.257788 (retrieved April 10, 2019).

[10] A. Bellissimo, J. Burgess, and K. Fu, “Secure software updates: Disappointments and new challenges,” in Proceedings of the 1st USENIX Workshop on Hot Topics in Security, ser. HOTSEC’06. Berkeley, CA, USA: USENIX Association, 2006, pp. 7–7, http://dl.acm.org/citation.cfm?id=1268476.1268483 (retrieved April 10, 2019).

[11] Microsoft, “Understanding Patch and Update Management: Microsofts Software Update Strategy,” https://technet.microsoft.com/en-us/library/cc768045.aspx, 2003 (retrieved April 10, 2019).

[12] “Apple security updates,” https://support.apple.com/en-us/HT201222, 2019 (retrieved April 10, 2019).

[13] “SCCM Software updates Strategy,” http://www.sccm.ie/how-to/8-sccm-software-updates-strategy, (retrieved April 10, 2019).

[14] “Why updating your Software is a Must Do,” https://www.techlicious.com/tip/why-you-should-update-software-when-prompted/, 2012 (retrieved April 10, 2019).

[15] “WordPress Security Category Archive,” https://wordpress.org/news/category/security/, 2019 (retrieved April 10, 2019).

[16] “Roundcube - Free and Open Source Web mail Software,” https://roundcube.net, 2018 (retrieved April 10, 2019).

[17] "Why You May Not Want To Run Your Own Mail Server," https://www.digitalocean.com/community/tutorials/why-you-may-not-want-to-run-your-own-mail-server, 2014 (retrieved April 10, 2019).

[18] SecPatt, "Should I set up or operate a mail server myself? [ger]," https://www.secpatt.at/patterns/pt_14/, 2018 (retrieved April 10, 2019).

[19] SecPatt, "How do I secure a web shop and what should be taken into consideration? [ger]," https://www.secpatt.at/patterns/pt_10/, 2018 (retrieved April 10, 2019).

[20] SecPatt, "How do I encrypt the communication with my website? [ger]," https://www.secpatt.at/patterns/pt_4/, 2018 (retrieved April 10, 2019).

[21] SecPatt, "What contributes to the security of a website? [ger]," https://www.secpatt.at/patterns/pt_3/, 2018 (retrieved April 10, 2019).

[22] SecPatt, "How do I store data securely? [ger]," https://www.secpatt.at/patterns/pt_8/, 2018 (retrieved April 10, 2019).

[23] SecPatt, "Which data am I allowed to save? [ger]," https://www.secpatt.at/patterns/pt_9/, 2018 (retrieved April 10, 2019).

[24] SecPatt, "What information do I have to provide to visitors of my website? [ger]," https://www.secpatt.at/patterns/pt_12/, 2018 (retrieved April 10, 2019).

[25] "86% of Passwords are Terrible (and Other Statistics)," https://www.troyhunt.com/86-of-passwords-are-terrible-and-other-statistics/, 2018 (retrieved April 10, 2019).

[26] "Wirtschaftsrecht / Gewerberecht Muster für den Bestellablauf - Info der WKO Datenschutzbehörde Österreich," https://www.wko.at/service/wirtschaftsrecht-gewerberecht/Webshop__In_7_Schritten_zur_Bestellung.html, 2019 (retrieved April 10, 2019).

Web Security and Privacy for Novices – Part 3

Backups, Data Security, and GDPR Compliance

Artur Lupp*, Alexander G. Mirnig* and Manfred Tscheligi†
*Center for Human-Computer Interaction
University of Salzburg, Salzburg, Austria
Email: firstname.lastname@sbg.ac.at
†Center for Human-Computer Interaction &
Austrian Institute Of Technology, Salzburg & Vienna, Austria
Email: firstname.lastname@sbg.ac.at

***Abstract*—In this paper, we present four patterns for nonprofessional web developers. This is the third part of a series of three thematically connected papers. The patterns in this part of the series address backups, secure data storage and the European General Data Protection Regulation. The reader will be introduced to the basic knowledge of backups, followed by an introduction to the new data protection regulations with an explanation how to handle the associated additional responsibilities regarding the handling and storage of personal data in the internet.**

Keywords–Patterns; On-line; Security; Privacy; Novice Users.

I. Introduction

With the aid of guides and simple to use programs, setting up websites, web shops, blogs and other web presences is as easy as it ever was. The problem is however, the lack of knowledge transfer about web security and privacy. Nonprofessional web developers setting up web shops without proper knowledge about the EU General Data Protection Regulation (GDPR) [1] may face charges for not obeying the European law when it comes to safe data storage or data handling. Even the lack of certain pieces of information on a commercial website can lead to irreversible financial data. Thus, blindly trusting in guides is a bad idea. This is one of many reasons why we decided to create web security and web privacy themed design patterns to aid nonprofessional web developers and to help to make the internet a safer place.

This paper constitutes the third part of a series of three thematically connected papers, describing patterns from the same pattern collection. The contribution of this paper is an additional set of four patterns addressing backups, safe and GDPR compliant data storage and GDPR compliance in general. The first paper of the three thematically connected papers contains two meta-patterns. The second paper provides a set of four patterns addressing updates, mail servers and the security of web shops. Section II provides a short overview over related word, Section III introduces four patterns with the titles: ”When and how should I create backups?”, ”How do I store data securely?”, ”Which data am I allowed to save?” and ”What information do I have to provide to visitors of my website?”. We will conclude this paper in Section IV.

II. Related Work

On 25 May 2018, the new GDPR was applied across the European Union, enforcing new legal requirements as a necessity to get data controllers to protect personal data even more than before. However, data controllers are now confronted with new challenges, to ensure the safety and to comply with the GDPR. Two legal requirements which are very important prove themselves as very controversial [2]–[4]. The first one is the right to be forgotten and second the right to withdraw consent. In the era of big data, cloud computing and the Internet of Things, these laws might yield unforeseen problems and consequences depending on interpretation and implementation, especially for nonprofessionals in law and web development as shown by Alnemr et al. [5] and Bob Duncan [6]. Especially the right to be forgotten has a heavy impact on backups and their archivation [7]. Due to this circumstances operating websites, especially when handling and working with personal data might prove as challenging for novices in web development.

III. Patterns

A. When and how should I create backups?

Intent: Creating regular backups can make your life easier and less frustrating. Accidentally deleted, compromised or lost files can easily be restored trough backups instead of being lost forever.

Problem Statement: If files are deleted from a server, they usually are irrevocably lost. There is no recycling bin to store the deleted files in case you want to recover them. Corrupted or compromised files and database entries can render a website inoperable and in some rare cases they might even destroy the whole website. In cases like these, backups can be a life saver. A web shop for example that is offline for too long, due to missing backups or faulty data, can lead to significant financial losses. Apart from that, on-commercial websites that are offline for a longer period of time, will notice a drop in the rankings of searching engines.

Scenario: System relevant files were deleted due to a system error. The functionality of the website is greatly limited. After fixing this problem with great effort because no backups were available, you decide to inform yourself about backups.

Solution: Backups are, in simple words, basically duplicated version of (all or certain) files. A modern website consists of multiple files ranging from files that contain the design and style information, images, plugins and database files where the posts and comments of users are commonly saved. The best way to make a copy of all of those files is to create a (full) backup. There are multiple ways to create backups - either you create them manually by copying the files one by one, or you use an automated script, a plugin or a special backup software. However the backups are made, you should always keep in mind that things can go wrong. Therefore, it is mandatory to understand the backup process in order to avoid mistakes and it is even more important to validate the backups after they are created. A corrupt, faulty or not working backup is as good as having no backup at all.

Backup Frequency - **Backups** should be created in **regular intervals**. But especially before:

- (major)updated of the operating system or important(server) software
- before moving to a new server (e.g., migration)

There is no golden rule for the frequency of backups. Generally, it is up to you, to decide when it is the time to create a backup. You have to think about the **importance and amount of the data** and whether it is **worth the time creating a backup**. It is possible to save some time, and skip backups with the possible risk of minimal data loss in case something happens. It is important to find the **balance**.

A web shop or a well visited bulletin board lively and active community, should do **daily or real-time backups**, as a data loss could have a very high impact and could lead to financial consequences. For more information about web shops please refer to pattern: How do I secure a web shop and what should be taken into consideration? [8]. If the site in question is **only for information purposes** or if the website **does not receive daily input**, it is sufficient to backup the site **once a week**.

Speaking of backups, there are a lot of possible forms. It is possible to secure only parts (e.g., images, files or only the database) of a page, or to do a full backup (i.e., saving the whole website with all of its components). If, for example, you have a well-visited website with only few postings and comments, a **full backup** (all files and the database) should be done **once a week**. A backup of the database, containing the user postings and comments should be done in more frequently. **Two or three times a week** seem reasonably if the page is well visited and the user base is posting or commenting frequently.

Note, however, not every type of backup makes it possible to restore individual files. Therefore, each and every backup method has to be tested and verified before you start relying on a certain method.

Verification of Backups - Backups are important! In case of a unfortunate event or a disaster, they might help you to recover individual files or even a whole system. Since backups might be the last resort in critical moments, you have to be able to count on them. This is why all backups should be tested and verified if it is feasible.

- Test whether they are corrupt or functional.
- Verify whether the Backups can be used to recover a system or files.

Automated Backups - Backing up a system manually (i.e., copy and paste important files by hand) can take a long time, especially when it is a large and complex system. Therefore, it is recommended to use backup software which offers the possibility to automate the backup process. Examples of such programs can be found in the ”Examples” section of this pattern. Some hosting provider also offer automated backups in their offers. It is recommended to keep that in mind, in case you forgot to do backups by yourself. This might save your day.

File and Database Backups - Before going deeper into specific backup solutions, it is important to be able to distinguish between data/file backups and database backups. Because depending of the type of data you want to back up, the processes vary. A WordPress page will be used as an example to explain the differences in the backup process depending on what is going to be saved.

Files: Apart from the mandatory main installation files of a WordPress installation,the term files also includes themes, designs, plugins, images, scripts (e.g., JavaScript, PHP, etc.), as well as other files and static pages. The files can be saved by following methods:

- Files may be backed up by the hosting providers (if they offer this service).
- Special backup software (e.g.,https://winscp.net/eng/docs/introduction) can create backups of your site and store them either locally or on a different server.
- There are WordPress plugins that create backups automatically at a certain time you can define (https://wordpress.org/plugins/updraftplus/).
- Manually transfer data to your own computer or a hard drive using a FTP (File Transfer Protocol) program (e.g.,https://filezilla-project.org) or command line tools.

Basically, its about saving files from one location and then copying them to a different location or storage device.

Databases: Database backups work differently compared to file backups. However, a database backup is always done the same way. Even automatically generated database backups by WordPress plugins, are identical to those that have been generated manually. First of all, you have to access the database by logging into it by using the database login credentials. Then you have to select the database entries you want to save and export them. This will result in a file, that can be used to recover the database entries. In the case of standard WordPress and phpMyAdmin installations, the database is called ”wp”. Login into the database, select the database ”wp” and export its contents. Viola, there you have the database backup.

It is recommended to make about **3 backup copies** and store them in **different locations (e.g., hard disk, cloud or server)**. This will help to be safe in case one or two of the storage locations fail.

Examples: Backup Software - The following programs can be used to back up individual files or complete systems. Please follow the links provided below if you want to know how the software works and whether it is suitable for your needs.

- tar
  - https://wiki.ubuntuusers.de/tar/
- Bacula
  - https://blog.bacula.org/what-is-bacula/
- dump
  - http://www.willemer.de/informatik/unix/unixdasi.htm [ger]

References: Backing Up Your WordPress Site [9]

WordPress Database Backup Instructions [10]

Backing up Your Website: The Ultimate Guide [11]

What kinds of Google Penalties are there and what are the differences? [12]

MariaDB - Backup and Restore Overview [13]

MySQL - Database Backup Methods [14]

MariaDB - mysqldump [15]

How Often Should You Backup Your WordPress Sites? [16]

Keywords: Backup, Security Backup, Database, Data Files

B. How do I store data securely?

Intent: This pattern addresses the problems of the secure storage of user generated content and user data.

Problem Statement: The secure storage of data has to be handled according to the EU General Data Protection Regulation [1]. This is especially important if the data in question is personal data.

Scenario: A website is allowing its users to write comments and to upload data (images or similar). The comments and content have to be transferred and stored on the web server securely.

Solution:

- Keep your system and server safe and up-to-date.
  - An up-to-date system equipped with an up-to-date anti virus program and the latest (security) updates offers less attack vectors for cyber criminals.
- SSL/TLS Encryption Communication
  - An encrypted connection allows a secure data transfer and guarantees data integrity.
- Save Received Data in Anonymous and/or Encrypted Format.
  - Important information, like passwords, should never be stored directly in plain text. Encrypt the data before storing it in the database.
  - Some data may only be stored in **encrypted and/or anonymous form (e.g., personal data)**.
    - For more information, please refer to pattern: Which data I am allowed to save? [17].
- Define and Set Access Rights.
  - The fewer people have access to the data, the safer the data generally is.
  - Adjust access rights for users individually.
    - Everyone should only have access to files and folders meant for them.
- Database Backup.
  - It is not just about storing the data safely. Securing the stored data is important as well.
- Encrypted Backups.
  - Encrypting backups is an additional step to take to secure the data even more. While this is totally optional, an encrypted backup can make it much harder for unauthorized people to access the files inside of them.

Examples: Ensure Security Your System - An up-to-date system offers less attacking vectors for potential cyber criminals. It is mandatory to ensure the installation of the latest (security) updates. More about this topic can be found in pattern ”When and how often should I install updates?” [18]. Securing your own computer or server is only the first step to ensure a secured system. It is also necessary to secure the website itself. Pattern How do I check the security of my website?” [19] explains in depth how this can be done.

Ensuring Data Integrity - Pattern ”How do I encrypt the communication with my website? [20] explains how to enable an SSL/TLS encrypted connection (HTTPS) on your own website. With a secure HTTPS connection, the communication between the users web browser and the encrypted website is encrypted. This ensures, that all the data that comes from the user is really from him.

Encrypt Packed Data Backups (Example uses 7Zip)

- How to protect your ZIP-Archives with a Password? [21][ger].

How to Encrypt Passwords using PHP

- Password Hashing [22].

References: Backup & Recovery [ger] [23]

Function Reference/wp hash password [24]

WordPress Password Hash Generator PHP [25]

Keywords: Data Security, Encoding, Data Storage, Personal Data

C. Which data am I allowed to save?

Intent: This pattern addresses the question, what data you are allowed to save on your website and what there is to consider when doing so.

Problem Statement: With all the advances in web development, nowadays it is possible to gather a lot of data from website visitors. But what kind of data are you allowed to gather and save according to GDPR (EU General Data Protection Regulation) ?

Scenario: Website operators have to follow the new GDPR and are required to adjust their website to match these regulations.

Solution: **Attention: This pattern is not a legal advice! We addressed the GDPR (https://www.dsb.gv.at/gesetze-in-osterreich [ger]) and applicable data protection regulations during our research for the patterns, however, we are no legal advisors, nor are we lawyers or privacy experts. We shall not have any liability whatsoever for the accuracy, completeness, timeliness, or correct sequencing of the provided information.**

Current regulations for **files, links and general user content**:

According to the Austrian E-Commerce Act 17 (exclusion of responsibility for links) [26], the website operator is not responsible for the content of links posted on his site (by other persons), as long as he has no knowledge of content provides via this link. However, if the operator determines that the link contains or links to illegal activity or information (for example, a link to a movie or a song file), the link has to be removed immediately.

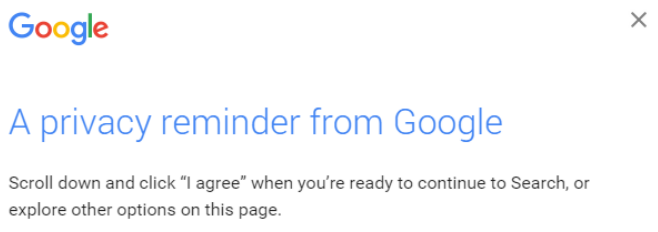
It is recommended to apply regulations and define whats allowed on the website or not in the form terms of services. The terms clarify which and whether content may be uploaded by users, or that if content is uploaded, the user has to have the appropriate usage rights for that content. **Attention :** Never write anything in the terms and service, that you simply can not do! For example, **never specify that any content will be checked before it is posted on your website**. As this makes you liable for each and any content on the website, regardless who posted it. As this states, that you have checked and verified that link in general and have knowledge about its content. The operator of a website is also liable for the content of a link posted by an employee or a person who is supervised by the operator.

Special regulations for **personal data** currently described in the GDPR:

- Website operators have to ensure the security of personal data.
  - It is defined in the GDPR as Privacy by Design / Privacy by Default, meaning, that everybody must use appropriate technical measures and procedures (e.g., pseudonymization) to ensure data security.
- Users need to be informed. Especially about what happens with their data and how their data will be used. In addition, users must actively agree that data may be stored by teh website.
  - More information can be found in pattern:What information do I have to provide to visitors of my website? [27].
- Users have the right to have their data deleted (i.e., ”the right to be forgotten”).
  - For more information please refer to :https://www.wko.at/service/wirtschaftsrecht-gewerberecht/EU-Datenschutz-Grundverordnung:-Pflicht-zur-Berichtigung.html#heading__Recht_auf_Loeschung_ [ger].
- Storage, transfer and distribution of personal data is not allowed without the users consent.
- If working with personal data, it is required to document the processing activities.
  - More information on that topic can be found here: https://www.wko.at/service/wirtschaftsrecht-gewerberecht/EU-Datenschutz-Grundverordnung:-Dokumentationspflicht.html
- The Court of Justice of the European Union has ruled that both static and dynamic IP addresses are to be considered as personal data.

A comprehensive blog post about GDPR and blogs can be found here: https://datenschmutz.net/dsgvo-checkliste-fuer-blogs/

Examples: Data Privacy - Inform Users and get their Consent - Example of Google search page:



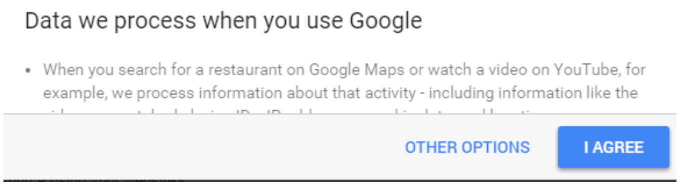
A privacy reminder from Google

Scroll down and click “I agree” when you’re ready to continue to Search, or explore other options on this page.

Figure 1. Google Privacy Reminder Info Page

A lot of companies changed their privacy and data regulations to address the GDPR. In reaction to that, Google and other companies informed their users of these changes. Figure 1 shows the reminder google used when users accessed the search page, listing and explaining what happens to the data collected by Google

You must actively agree to these conditions in order to be able to continue using Google without detours, as shown in Figure 2.



Data we process when you use Google

- When you search for a restaurant on Google Maps or watch a video on YouTube, for example, we process information about that activity - including information like the

Figure 2. Google Privacy Reminder Info Page - Consent

What is personal data?

- General personal Data
  - Name, birthday date, age, address, e-mail-address, phone numbers, pictures from the person, education, job, marital status, nationality religious, as well as political attitude, sexuality, health data, holiday planning and police record.
- Identification (ID) numbers.
  - Social security number, tax number, health insurance number, ID card number, matriculation number.
- Banking Information
- Online Data
  - Internet Protocol (IP) address, a cookie ID, location data (for example the location data function on a mobile phone).
- Physical Characteristics.
  - Gender, skin color, hair color and eye color.
- Possession Features
  - Car, property ownership, land registry entries, license plate number.
- Customer data
  - Orders, bank account informations, account data, etc.
- Personal data can be defined as any information that relates to an identified or identifiable living individual. For example, a birthday alone can not always be linked to a specific person without additional information. However, if a name is added to this birthday, it is way easier to pin it down to a certain person. Thus, you have to be careful when displaying or providing personal data in general.

More information provided by the European Commission on Personal Data can be found here: What is Personal Data? [28]

References: Information about the GDPR provided by the Austrian Economic Chambers:

EU GDPR [29][ger]

EU GDPR - Data Security Measures [30][ger]

A blog post about user generated content [31][ger]

What is personal data? [32][ger]

Keywords: DSGVO, GDPR, Data Protection, Personal Data

D. What information do I have to provide to visitors of my website?

Intent: This pattern aims to inform website operators what kind information they legally have to provide their users on websites.

Problem Statement: The Internet is not a lawless place. There are laws intended to protect internet users which have to be respected by websites and theirs operators. Website owners should know their rights, as well as theirs responsibilities and should be at least be acquainted with the law.

Scenario: A website is almost ready to go online. What information does it have to provide to its users in order to be GDPR compliant?

Solution: **Attention: This pattern is not a legal advice! We addressed the GDPR (https://www.dsb.gv.at/gesetze-in-osterreich [ger]) and applicable data protection regulations during our research for the patterns, however, we are no legal advisors, nor are we lawyers or privacy experts. We shall not have any liability whatsoever for the accuracy, completeness, timeliness, or correct sequencing of the provided information.**

The website imprint, the terms of service and the privacy policy must be easy to find and access has always to be guaranteed. It is advisable to place these things in a good position where its visible all the time (e.g., in the header, or in the footer of a website).

Attention! - For web shops additional conditions apply. Please visit the following site for more information: General Terms and Conditions - Details [ger] [33].

- Imprint
  - The imprint is not an obligation for small private websites (e.g., a travel blog or a page only for friends). This is stated in Austrian Federal Law Consolidated Version, Media Act 24. Nevertheless, it is advisable to provide an imprint for the reasons of transparency.
  - The imprint includes:
    - Name or Company Name of the page owner and operator.
    - Registration number and place of registration.
    - Place of residence or registered office of the page owner.
  - If the site is serving (directly or indirectly) commercial purposes, the Austrian Federal Law Consolidated Version, Media Act 25 applies.
    - Disclosure obligation according to Austrian Federal Law Consolidated Version, Media Act 25 [34].
- Terms and Conditions
  - The terms and conditions should include the following:
    - A clear indication that users are responsible for the content of their posts.
    - Users have to agree to the terms if they want to use the website.
    - The posts from the users are not allowed to violate the terms or the applicable law.
    - The user holds the rights to his contents and contributions as long as it does not violate the law.
    - Contributions must not violate the rights of third parties (e.g., copyright, trademark law or personality rights).
    - Exemplary enumeration of content that should not be uploaded:
      * Copyrighted content, if no authorization exists (e.g., photos, images, videos).
      * Pornographic or adult content.
      * Racial, xenophobic, discriminatory or offensive content.
      * Content that violates applicable law.
    - Rules of conduct.
    - Restrictive measures and sanctions for violation of the terms of service.
    - Release from claims from third parties.
- Privacy Policy and use of Cookies:
  - Are cookies used at the website to save personal data or does the personal data general stored on the side (e.g., IP addresses)? Is there an information and an active consent requirement for site visitors?
    If cookies are used to save personal data (e.g., geo location) on a website, it is obliged to **inform the users and it is required to get an active consent from the users in order to be allowed to save the date**.
  - The use of cookies is only permitted if:
    - The user is informed in detail in advance.
    - Cookie use needs active consent from the user if saving personal data.
    - The consent must be given voluntarily, without doubt and through an active act.

Examples: Information for Storing Data (Including Personal Pata) - The patterns How do I store data securely? [35] and Which data am I allowed to save? [17] address the storage of data and person data.

Examples of Imprints and Disclosures

- https://datenschmutz.net/impressum/ [ger]
- https://www.wko.at/service/Offenlegung_Salzburg.html [ger]
- https://de.wikipedia.org/wiki/Wikipedia:Impressum [ger]
- https://www.guteguete.at/impressum [ger]

Example: Liability Disclaimer

- https://www.conrad.at/de/ueber-conrad/impressum.html [ger]

Example for ABG (General Terms and Conditions / Conditions of Use)

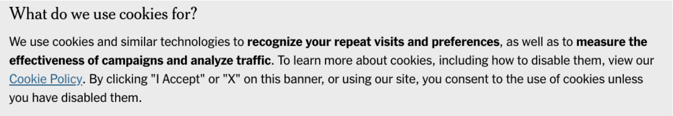
- https://www.amazon.com/gp/help/customer/display.html?nodeId=201909000
- https://www.amazon.de/gp/help/customer/display.html?nodeId=201909000 [ger]

Example: Privacy Statements / Policy

- https://policies.google.com/privacy?hl=en
- https://www.guteguete.at/datenschutzerklaerung [ger]

Example for a Cookie Notice for GDPR - See Figure 3 for an example of a Cookie Notice for European countries shown on https://www.nytimes.com.

ISBN: 978-1-61208-708-5



What do we use cookies for?

We use cookies and similar technologies to **recognize your repeat visits and preferences**, as well as to **measure the effectiveness of campaigns and analyze traffic**. To learn more about cookies, including how to disable them, view our Cookie Policy. By clicking "I Accept" or "X" on this banner, or using our site, you consent to the use of cookies unless you have disabled them.

Figure 3. Cookie Notice shown on nytimes.com

Privacy Statement

- Sample of the WKO Privacy Statement [36][ger]

References: Establishment of an Online Store - Website [37] [ger]

Your own Website [38] [ger]

Austrian Federal Law Consolidated Version: Media Act 24, Version of 20.06.2018 [39] [ger]

Austrian Federal Law Consolidated Version: Media Act 25, Version of 20.06.2018 [34] [ger]

User Generated Content - Minimize your Liability [40] [ger]

Keywords: GDPR, EU, Personal Data, Privacy, Law

IV. Conclusion

This paper is the third and final part of a series of three thematically connected papers. It presents four additional patterns with the aim to aid nonprofessional web developers understanding common privacy and security problems frequently surfacing during the creation of websites. These patterns explain the importance of updates and the difficulties of saving and handling user data in a GDPR compliant way. While the explanation of the underlying concept and benefit of backups is quite straightforward and can easily be stated in one pattern, it is a different matter for the GDPR. Depending on the type of website and which data is being handled, the regulations and requirements defined by the GDPR can vary extremely. Thus, it is not possible to explain the complete GDPR in this format. We tried to cover the most important question by providing solutions, examples for possible problem cases in order to provide a decent knowledge base for novice web developers. Future work will mainly focus on the extension of the pattern solutions while keeping the existing patterns up-to-date to ensure future validity and usefulness.

Acknowledgment

The financial support by the Internet Privatstiftung Austria (IPA) under the program "netidee" with the title "SecPatt" under grant number 2390 is gratefully acknowledged.

References

[1] "EU General Data Protection Regulation (GDPR," https://en.wikipedia.org/wiki/General_Data_Protection_Regulation, 2019 (retrieved April 10, 2019).

[2] E. Alepis, E. Politou, and C. Patsakis, "Forgetting personal data and revoking consent under the GDPR: Challenges and proposed solutions," Journal of Cybersecurity, vol. 4, no. 1, 03 2018, pp. 1–20, doi: 10.1093/cybsec/tyy001.

[3] C. Tankard, "What the gdpr means for businesses," Network Security, vol. 2016, no. 6, 2016, pp. 5 – 8, doi: 10.1016/S1353-4858(16)30056-3.

[4] J. Krystlik, "With gdpr, preparation is everything," Computer Fraud & Security, vol. 2017, no. 6, 2017, pp. 5 – 8, doi: 10.1016/S1361-3723(17)30050-7.

[5] R. Alnemr, E. Cayirci, L. D. Corte, A. Garaga, R. Leenes, R. Mhungu, S. Pearson, C. Reed, A. S. de Oliveira, D. Stefanatou, K. Tetrimida, and A. Vranaki, "A data protection impact assessment methodology for cloud," in Privacy Technologies and Policy, B. Berendt, T. Engel, D. Ikonomou, D. Le Métayer, and S. Schiffner, Eds. Cham: Springer International Publishing, 2016, pp. 60–92.

[6] B. Duncan, "Can eu general data protection regulation compliance be achieved when using cloud computing," in CLOUD COMPUTING 2018 : The Ninth International Conference on Cloud Computing, GRIDs, and Virtualization, ser. Cloud Computing 2018, B. Duncan, Y. Lee, and A. Olmsted, Eds. IARIA, 2 2018, pp. 1–6.

[7] E. Politou, A. Michota, E. Alepis, M. Pocs, and C. Patsakis, "Backups and the right to be forgotten in the gdpr: An uneasy relationship," Computer Law & Security Review, vol. 34, no. 6, 2018, pp. 1247 – 1257, doi: 10.1016/j.clsr.2018.08.006.

[8] SecPatt, "How do I secure a web shop and what should be taken into consideration? [ger]," https://www.secpatt.at/patterns/pt_10/, 2018 (retrieved April 10, 2019).

[9] "Backing Up Your WordPress Site," https://codex.wordpress.org/WordPress_Backups#Backing_Up_Your_WordPress_Site, 2019 (retrieved April 10, 2019).

[10] "WordPress Database Backup Instructions," https://codex.wordpress.org/WordPress_Backups#Database_Backup_Instructions, 2019 (retrieved April 10, 2019).

[11] "Backing up Your Website: The Ultimate Guide," https://webdesign.tutsplus.com/articles/backing-up-your-website-the-ultimate-guidewebdesign-4748, 2019 (retrieved April 10, 2019).

[12] "What kinds of Google Penalties are there and what are the differences?" https://www.sistrix.com/ask-sistrix/google-penalties/what-kinds-of-google-penalties-are-there-and-what-are-the-differences, 2019 (retrieved April 10, 2019).

[13] "MariaDB - Backup and Restore Overview," https://mariadb.com/kb/en/library/backup-and-restore-overview/, 2019 (retrieved April 10, 2019).

[14] "MySQL - Database Backup Methods," https://dev.mysql.com/doc/mysql-backup-excerpt/8.0/en/backup-methods.html, 2019 (retrieved April 10, 2019).

[15] "MariaDB - mysqldump," https://mariadb.com/kb/en/library/mysqldump/, 2019 (retrieved April 10, 2019).

[16] "How Often Should You Backup Your WordPress Sites?" https://blogvault.net/how-often-should-you-backup-your-wordpress-sites/, 2016 (retrieved April 10, 2019).

[17] SecPatt, "Which data am I allowed to save? [ger]," https://www.secpatt.at/patterns/pt_9/, 2018 (retrieved April 10, 2019).

[18] SecPatt, "When and how often should I install updates? [ger]," https://www.secpatt.at/patterns/pt_1/, 2018 (retrieved April 10, 2019).

[19] SecPatt, "How do I check the security of my website? [ger]," https://www.secpatt.at/patterns/pt_7/, 2018 (retrieved April 10, 2019).

[20] SecPatt, "How do I encrypt the communication with my website? [ger]," https://www.secpatt.at/patterns/pt_4/, 2018 (retrieved April 10, 2019).

[21] "How to protect your ZIP-Archives with a Password?" https://www.heise.de/tipps-tricks/ZIP-Archiv-mit-einem-Passwort-schuetzen-So-geht-s-3907870.html, 2017 (retrieved April 10, 2019).

[22] "Password Hashing," https://paragonie.com/blog/2017/12/2018-guide-building-secure-php-software#secure-php-passwords, 2017 (retrieved April 10, 2019).

[23] "Backup & Recovery [ger]," https://www.onlinesicherheit.gv.at/praevention/datensicherung_und_loeschung/datensicherung_und_wiederherstellung/249920.html, 2018 (retrieved April 10, 2019).

[24] "Function Reference/wp hash password," https://codex.wordpress.org/Function_Reference/wp_hash_password, 2019 (retrieved April 10, 2019).

[25] "WordPress Password Hash Generator PHP," http://www.kvcodes.com/2016/09/wordpress-password-hash-generator/, 2016 (retrieved April 10, 2019).

[26] "Austrian Federal Law Consolidated Version: E-Commerce Act 17, Version of 10.04.2019," https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=Bundesnormen&Gesetzesnummer=20001703, 2018 (retrieved April 10, 2019).

[27] SecPatt, “What information do I have to provide to visitors of my website? [ger],” https://www.secpatt.at/patterns/pt_12/, 2018 (retrieved April 10, 2019).

[28] “What is Personal Data?” https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-personal-data_en, 2019 (retrieved April 10, 2019).

[29] “EU GDPR,” https://www.wko.at/service/wirtschaftsrecht-gewerberecht/EU-Datenschutz-Grundverordnung.html, 2018 (retrieved April 10, 2019).

[30] “EU GDPR - Data Security Measures,” https://www.wko.at/service/wirtschaftsrecht-gewerberecht/EU-Datenschutz-Grundverordnung:-Datensicherheit-und-Daten.html, 2018 (retrieved April 10, 2019).

[31] “A blog post about user generated content,” http://www.rechtzweinull.de/archives/108-Haftung-fuer-User-Generated-Content-Grundsaetze-und-Hinweise-fuer-die-Praxis.html, 2009 (retrieved April 10, 2019).

[32] “What is personal data?” https://www.datenschutz.org/personenbezogene-daten/, 2018 (retrieved April 10, 2019).

[33] “General Terms and Conditions - Details,” https://www.wko.at/service/wirtschaftsrecht-gewerberecht/AGB_im_Internet_-_im_Detail.html, 2019 (retrieved April 10, 2019).

[34] “Austrian Federal Law Consolidated Version: Madia Act 25, Version of 20.06.2018,” https://www.ris.bka.gv.at/NormDokument.wxe?Abfrage=Bundesnormen&Gesetzesnummer=10000719&FassungVom=2018-06-20&Artikel=&Paragraf=25&Anlage=&Uebergangsrecht=, 2018 (retrieved April 10, 2019).

[35] SecPatt, “How do I store data securely? [ger],” https://www.secpatt.at/patterns/pt_8/, 2018 (retrieved April 10, 2019).

[36] “Sample of the WKO Privacy Statement,” https://www.wko.at/service/wirtschaftsrecht-gewerberecht/muster-informationspflichten-website-datenschutzerklaerung.html, 2018 (retrieved April 10, 2019).

[37] “Establishment of an Online Store - Website,” https://www.usp.gv.at/Portal.Node/usp/public/content/gruendung/gruendung_online-shop/website/Seite.70064.html, 2019 (retrieved April 10, 2019).

[38] “Your own Website,” https://www.help.gv.at/Portal.Node/hlpd/public/content/172/Seite.1720902.html, 2018 (retrieved April 10, 2019).

[39] “Austrian Federal Law Consolidated Version: Madia Act 24, Version of 20.06.2018,” https://www.ris.bka.gv.at/NormDokument.wxe?Abfrage=Bundesnormen&Gesetzesnummer=10000719&FassungVom=2018-06-20&Artikel=&Paragraf=24&Anlage=&Uebergangsrecht=, 2018 (retrieved April 10, 2019).

[40] “User Generated Content - Minimize your Liability,” https://www.it-recht-kanzlei.de/agb-user-generated-content-blog-forum-wiki.html, 2009 (retrieved April 10, 2019).