Attribute Based Access Control Metamodel for Spreadsheet Programs

Miro Zdilar
Faculty of Organization and Informatics,
University of Zagreb
Pavlinska 2, 42000 Varaždin, Croatia
mizdilar@student.foi.hr

Abstract*. Spreadsheets are one of the most popular tools used for various purposes in organizations. With recent technological advancement and new features added, spreadsheets have become powerful tool for complex analysis and modelling. However, spreadsheets are associated with high incidence of errors and unauthorized changes in multiuser environments causing companies significant losses. The goal of this work-in-progress paper is to present Attribute-Based Access Control metamodel for spreadsheet programs that empowers organizations to control unauthorized activities with spreadsheets in multiuser environments.*

Keywords. Authorizations, Attribute Based Access Control, Metamodel, Spreadsheets, Spreadsheet Error Detection, Spreadsheet Programs

1 Introduction

Spreadsheets are one of the most successful end-user programming systems used in business and academia (Panko, 2008). End-user programming systems empowers end-users to build and execute powerful computer programs without the use of traditional programming languages. In U.S. alone, it has been estimated that the number of end-user programmers outnumber traditional software programmers (Scaffidi et al., 2005). Spreadsheets are traditionally used in almost all companies in the U.S and Europe for accounting and financial reporting purposes (Panko & Ordway, 2008). Several researchers have studied spreadsheet usage patterns (Reschenhofer & Matthes, 2015) and they identified following business processes supported by spreadsheets; strategic and capacity planning, financial reporting, stakeholder analysis, risk management, performance calculation, data transformation, cash-flows analysis, time-series transformations, and simulations. Since the first introduction of electronic spreadsheets for personal computers in 1979, they have been developed as single user application targeted for personal computers. With recent technological advancement and new features added, spreadsheets have evolved to cloud-powered computational platform.

However, spreadsheets still lack effective access control functionality capable to adapt to technical and organizational dynamics of enterprises and growing complexity of spreadsheet programs. Even though spreadsheet allows Discretionary Access Control (DAC) at the level of cell and worksheet, this type of access control does not allow granularity for different roles, centralised administration with access policy and monitoring data flow becomes almost impossible as the spreadsheet program grows in complexity.

Uncontrolled access to spreadsheet programs and data have been linked with many spreadsheet errors causing companies and organizations significant financial losses and negative publicity. Non-profit and voluntary organization European Spreadsheet Risk Interest Group (EuSpRIG) maintains list of horror stories with details of spreadsheet errors and uncontrolled access to spreadsheet programs and data (European Spreadsheet Risk Interest Group, 2023).

This work-in-progress article presents Attribute Based Access Control (ABAC) metamodel for spreadsheet programs which can describe authorizations and its enforcement (access control) to prevalent spreadsheets, including dynamics of multiuser organizations. Therefore, the work presented in this paper answers following research questions:

- RQ1: What are common access control models for spreadsheets, both on technical level of modern cloud-powered spreadsheets and on organizational level within multiuser enterprises?
- RQ2: What is a suitable meta model to describe those access control models for spreadsheets?

The remainder of this paper is organized as follows. Section 2 provides summary of related work in the field of access controls models and their suitability for modern cloud-powered spreadsheets. In section 3, research method is presented for deriving ABAC metamodel for spreadsheet programs based on existing literature review, and architecture of modern cloud-powered spreadsheets. Afterwards, in section 4, the derived ABAC metamodel for spreadsheet programs is presented with descriptions of design challenges and model components. In section 5, initial results of the evaluation of the proposed metamodel are presented. Finally, in section 6, conclusions and critical reflections on research and model evaluation are provided with proposals for future research opportunities.

2 Related Work

This paper followed critical review of the literature presented in other studies on spreadsheet errors (Powell et al., 2008). Rather than give a chronological account of the literature, discussion in this paper is focused around formulated research questions.

Authorization and its enforcement (access control) are key components of enterprise information technology systems (Korman, 2016). Focus of researchers have been around modelling different access control systems, evaluation and comparison of access control models deployed to various technical and operational environments, and formal verification of access control models in context of specific algorithms and protocols. Following is the summary of common access control models (Kashmar et al., 2021) that have been analysed for their suitability in development of access control metamodel for spreadsheets;

- Discretionary Access Control (DAC): is access control model built with three major components – objects, subjects, and permissions. DAC allows owners (subjects) to control permissions to their objects and is commonly implemented with Access Control List (ACL). DAC is implemented as integral part of many information technology systems, such as operating systems and databases. DAC is part of commercial spreadsheet implementation for decades and allows spreadsheet owner to control user's access to specific cell or worksheet. Issues with DAC have been studied by researchers (Downs et al., 1985) and this type of access control does not allow granularity for different roles, centralised administration with access policy and monitoring data flow becomes almost impossible as the spreadsheet program grows in complexity.
- Mandatory Access Control (MAC): is access control model managed in centralized manner and is built with four key components – a set of objects, a set of subjects, permissions, and security level. Even though MAC allows centralized policy management, it is very complex to implement this type of access control on all spreadsheet resources due to mandatory security level assigned to both subjects and objects. In addition, security levels specified in traditional MAC has been considered as antiqued (Dholakia, 2017) and do not seem to be in accordance with technical and organizational level of spreadsheet use in modern enterprises.
- Role-Based Access Control (RBAC): is access control model based on following key components – subjects, roles, permissions, actions, operations, and objects. In context of RBAC, role means a group of permissions to use object(s) and perform certain action(s). Only designated administrator has the right to control system security and manage roles assigned to users. RBAC implementation has been studied in hospital management where roles allow modeling complex relationships between doctors, nurses, etc. (Boadu, 2014). However, modeling and maintaining roles for different spreadsheet user roles and groups is very complex, especially in dynamic organizations where business processes and corresponding user roles are rapidly changing.
- Attribute-Based Access Control (ABAC): is access control model based on following three types of dynamic attributes – subjects, objects, and environments. User requests are resolved and determined based on subject attributes, objects attributes, environmental attributes as well as set of conditions specified by access policy. ABAC model is dynamic as it uses state of attributes at the time of access mode resolution. Even though we find limited literature on application of ABAC model to spreadsheets, we decided to further research applicability of ABAC to development of our meta model due to following promising characteristics of ABAC model:
  a. ABAC model is based on dynamic attributes, where object attributes fit to our proposed model of spreadsheet resources and corresponding attributes.
  b. Hierarchy of spreadsheet resources and objects can be modelled with set of ABAC conditions and access rules determinations. This property prevents conflicts in access resolutions and simplifies prototype implementation.
  c. Deployment opportunities for ABAC with spreadsheets are flexible and allows early prototype implementation as detective access control system. This minimizes impact on users and familiarized spreadsheet user interface.
  d. Complexity of ABAC model for spreadsheets depends on number of spreadsheet resource attributes.
  e. Dynamic nature of modern cloud-powered spreadsheets and extensions to spreadsheet formula language fits nicely to ABAC dynamic attribute concept. Potential new functionalities and modules added in cloud-powered spreadsheet can be integrated within existing ABAC concepts.

Unified metamodel of enterprise authorizations (Korman, 2016) is summarizing existing models of access controls. In addition, authors provided mapping between presented unified metamodel and ArchiMate tool that is frequently used in modern enterprises as architecture modeling language. List of generic metamodels for expressing different configurations of access models is valuable starting point for our research and design of ABAC metamodel for spreadsheets.

ABAC modelling and implementation has been recognized by U.S. government as important access control modelling concept for large enterprises and federal information technology systems. National Institute of Standards and Technology (NIST) published in 2014 Special Publication 800-162 " Guide to Attribute Based Access Control (ABAC) Definition and Considerations" (Hu et al., 2014). This publication provides definitions and considerations for using ABAC to improve information sharing, design of systems, while maintain control of that information. Concepts and terminology for ABAC presented in this document has been instrumental for design of our ABAC metamodel for spreadsheets.

To further explore our second research question, in continuation of this literature review we focused our discussion on spreadsheet modelling. Popularity and success of spreadsheets in various domains triggered significant interest of research community (Jannach et al., 2014,), including visualization techniques, static code analysis and reporting, testing approaches, automated fault localization and repair, model-driven development, and design support.

In one of the first publications structured around spreadsheet modelling (Isakowitz, 1995), authors presented model of spreadsheet with separation between logical and physical view. Presented spreadsheet model consists of four key components- "schema" which captures spreadsheet program's logic, "data property" which holds input values of the input cells, "binding property" which maps logical view of schema and data to two-dimensional spreadsheet grid, and finally "editorial property" which provides visual description for headings, labels, tables and documentation on two-dimensional spreadsheet grid. With the help of tools, logic can be extracted from simpler spreadsheet programs and presented in "schema" component with relational logic model. Ideas presented in this paper inspired our work and research related to spreadsheet modelling.

Formal set-oriented spreadsheet modelling approach has been presented by authors as part of their effort to research type inference for spreadsheet programs (Abraham, 2006). A spreadsheet is defined as collection of formulas and values embedded into a spatial structure. Authors provided generalization of spatial structure where two-dimensional grid is just one possible representation. In this paper, spreadsheet formula language is formalized with grammar based on set-oriented and logical representation. Concept of spreadsheet types introduced is still relevant and applicable to modern cloud-based spreadsheets. Ideas presented by authors has been influential in design of presented ABAC metamodel for spreadsheets, specifically in design of access rule resolution and prototype implementation.

The most influential conceptual model of spreadsheet for research presented in this paper have been published by authors researching complexity metrics for spreadsheets (Reschenhofer at al., 2017). Presented spreadsheet metamodel captured and integrated all relevant aspects of spreadsheet, including spreadsheet formula language. Main idea presented by authors has been further explored in this work with inclusion of modern cloud-based spreadsheets.

3 Research Method

Work in this paper follows Design Science Research (DSR) approach (Hevener et al., 2004). DSR is structured around three interconnected components or research phases that should jointly deliver design science research artefact. Specifically, application of DSR to this research resulted with following method and research phases:

- Environment – defines set of requirements to the design artifact. In this research proposed ABAC metamodel for spreadsheet represents theoretical foundation for addressing business needs for spreadsheets in multiuser environments and reduction of unauthorized change and spreadsheet errors. Proposed metamodel is agnostic to actual spreadsheet commercial products. In addition, proposed metamodel should address the needs of different business domains and organizations.
- Knowledge Base – we followed focused approach to related work and literature review and actual knowledge base for this work is presented in Section 2.
- Design Science Research – main deliverable of this research is ABAC metamodel for spreadsheets which was designed with iterative approach and continuous refinements and verifications. In addition to design of research artifact, our work contributes with new additions to spreadsheet and access control knowledge base. Our work has been influenced by model of spreadsheet with separation between logical and physical view (Isakowitz, 1995), formal set-oriented spreadsheet modeling approach (Abraham, 2006), conceptual model for measuring the complexity of spreadsheets (Reschenhofer at al., 2017) and unified metamodel for enterprise authorizations (Korman, 2016). The evaluation as part of design science research was done by applying the

designed ABAC metamodel for spreadsheets on use case in multiuser environment.

4 ABAC metamodel

According to research method described in Section 3, we designed ABAC metamodel for spreadsheets which is capable to control unauthorized activities with spreadsheets in multiuser environments. Due to limited space in this publication, basic model is depicted in Fig. 1, spreadsheet formula model is further explained in Fig. 2 and lastly unified view to key components of ABAC metamodel for spreadsheet is depicted in Fig 3.

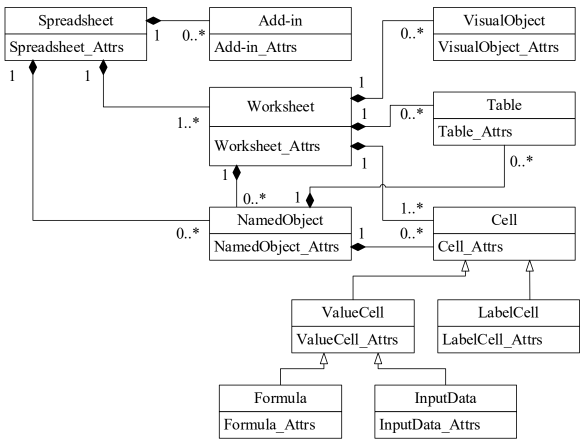


Figure 1. Metamodel of spreadsheet resources and associated attributes

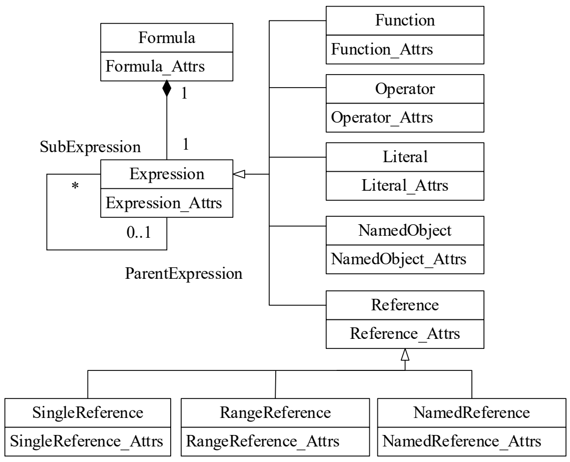


Figure 2. Spreadsheet formula metamodel (Continuation from Figure 1)

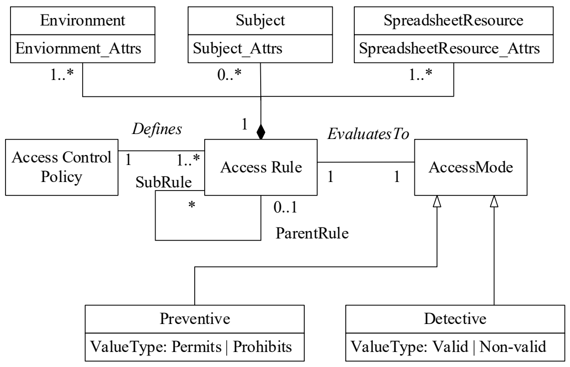


Figure 3. Proposed ABAC metamodel for spreadsheets

Design considerations in proposed metamodel are structured around resources that constitute modern cloud-powered spreadsheet. Spreadsheet resources are building blocks for spreadsheet programs and are manipulated by spreadsheet users or change their state during lifecycle of spreadsheet as result of spreadsheet program execution. Spreadsheet resources and their attributes are bounded with ABAC rules and permissible actions performed by spreadsheet users. Key spreadsheet resources unique to proposed ABAC metamodel for spreadsheets are following:

- The class *NamedObject* represents resource in modern spreadsheets that is controlled by user through Named Manager functionality. This naming convention is used in Micorsoft Excel 365 (Excel) product, but other cloud-based spreadsheets offer similar functionality. With recent introduction of *LAMBDA* functions (Gross, 2024) and collection of supporting functions (*MAP, REDUCE, SCAN, MAKEARRAY, BYROW, BYCOL, ISOMITTED)*, Excel is additionally empowered for computational tasks previously reserved for plugins or scripting with embedded macro language. Great demonstration of new Excel capabilities and powerful development strategies using only Excel formulas has been provided by Bartholomew at EuSpRIG conference (Bartholomew, 2023). Lambda functions are essential for creation of reusable software components, empowering spreadsheet developers to develop models quicker with fewer errors (Hatmaker, 2023). Other exciting new feature added to Excel is native support for Python programming language (Microsoft Excel, 2023). Powerful Python computational engine is embedded in Excel and users can integrate Python language code with existing formula language at the level of cell. Python in Excel is compatible with existing tools and libraries for charting and numerical analysis. Among many exciting new features that Python in Excel offer to users is ability to create and dynamically control complex tabular and visual objects directly from python code. This functionality was previously available only through Excel predefined toolbar. The class

*NamedObjec*t represents all these powerful resources added to Excel and through ABAC rules allows controlled access by users.

- The class *Table* represents Excel named table resources that could be added to worksheet through application interface. However, after manual creation of named table, this spreadsheet resource can be controlled directly from spreadsheet formula language. Structured references are powerful extension to formula language and allows spreadsheet users to reference in their code entire table, columns, rows, and table ranges. Through designated attributes, named table manipulations by users can be controlled with ABAC rules.
- The class *Expression* represents powerful spreadsheet language ability for nesting formulas. Original design introduced by Retschenhofer et al. (Reschenhofer at al., 2017) have been extended to include *NamedObject* as parameters and *NamedReference* as one possible realization of cell references.
- Spreadsheet resources represented in proposed metamodel contains associated set of attributes. We used simple naming convention and set of attributes are represented with suffix *_Attrs* concatenated to Class name that represents spreadsheet resource. For example, set of attributes associated with class Worksheet is represented in model as *Worksheet_Attrs*. Depending on nature and characteristics of modeled spreadsheet resource, corresponding attributes are represented with enumerated lists or key-value HashMap. For example, attribute *type* for *InputData* is represented with enumerated list *[Boolean, Integer, Number, String, Date, Array]* (Reschenhofer at al., 2017). Attribute name for Worksheet is represented with key-value HashMap *{"name":"Worksheet_Name"}*.
- The class *AccessMode* is generalization of *Preventive* and *Detective* classes. In optimal deployment case, proposed model should be implemented as preventive ABAC control system that prevents users for performing unauthorized actions. However, proposed ABAC metamodel can be deployed as detective control mechanism to determine validity of performed user actions with spreadsheet under control.

4.1 Access Rules

In proposed ABAC metamodel, access rules are modelled as quadruple with following structure:

$$<S, A, SR, E> \qquad (1)$$

S represents set of spreadsheet users or roles of the spreadsheet user. Theoretically, *S* represents any *Agent* that might interact with spreadsheet and controlled access is required. For example, other IT system might access spreadsheet via predefined interface or background job might update spreadsheet input data during predetermined period of times.

A is set of actions that subject might perform on spreadsheet resource represented with following enumerated list:

$$A \in [CREATE, READ, UPDATE, DELETE] \qquad (2)$$

Above enumerated list is well known in computer science as CRUD acronym. Important to note is that proposed ABAC metamodel does not have restrictions to number of actions and if needed in specific deployment scenarios, number of actions could be reduced or extended.

SR represents set of spreadsheet resources and corresponding resource attributes on which subject *S* can perform action *A*.

E are dynamic conditions, independent of subject and spreadsheet resources that may be used as attributes at decision time to influence an access decision. Examples of environment conditions include time, location, threat level, and temperature (Hu et al., 2014).

4.2 Access Mode Determination and Conflict Resolution

In order to formalize access mode determination in proposed metamodel and prevent conflict in access rules resolution, we introduce following definitions:

Atomic Access Rule (AAR):
AAR is quadruple of the from (1);

$$<S_i, A_j, SR_k, E_l> \qquad (3)$$

Where $S_i \in S$, $A_j \in A$, $SR_k \in SR$, $E_l \in E$.

□

Composed Access Rule (CAR):
CAR is conjunction statement composed with AARs:

$$AAR_1 \wedge AAR_2 \wedge AAR_3 \wedge \ldots \wedge AAR_n \qquad (4)$$

□

Access Policy (AP):
AP is set of all CARs applicable to single organization, and all its users and spreadsheet programs.

□

Deny-by-default Policy (DP):
If not explicitly defined in AP, all actions for all users and all spreadsheet resources are prohibited.

□

Priority of Actions (PA):
Actions should be evaluated in following order:

$$DELETE > CREATE > UPDATE > READ \qquad (5)$$

□

Delete action has highest priority and operator > in context of above relation should be interpreted as “Allow to Perform”. For example, if access rule permits user to delete specific spreadsheet resource, according to above definition, user is also allowed to create, update, and read corresponding resource.

Rule Evaluation Hierarchy (REH):
All child spreadsheet resources inherit rules applicable to their parents.

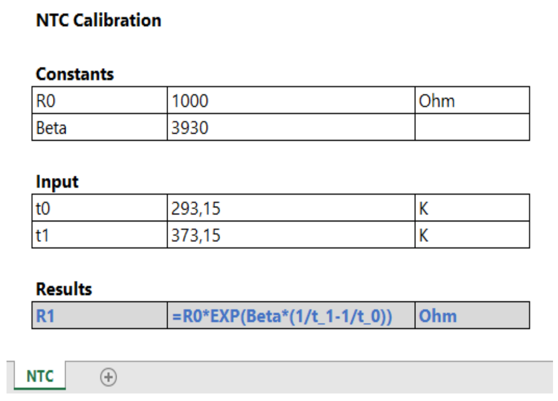
□

Proposed metamodel is structured around natural hierarchy of spreadsheet resources. For example, *Spreadsheet* class contains (“HAS A” relationship), *Add-in*, *Worksheet* and *NamedObject* classes, and all applicable access rules for *Spreadsheet* are inherited by its children, *Add-in*, *Worksheet* and *NamedObject*. Determining hierarchy for dynamic spreadsheet resources, such as composed formula expressions and large data tables is challenging, however strict conformance with REH rule during all stages of spreadsheet lifecycle is important for consistent rule’s evaluation and user’s activity control.

5 Model Evaluation

Proposed ABAC metamodel for spreadsheets has been evaluated on use case within analytical laboratory. In order to minimize impact on established organizational processes and spreadsheet use patterns, we deployed ABAC detective controls to determine validity of performed user actions with spreadsheet under control.

In agreement with laboratory management, we selected spreadsheet program for Negative Temperature Coefficient (NTC) probe calibrations. NTC spreadsheet supports important laboratory processes for calibration and management of 65 temperature probes utilized for temperature measurement of air, surface or liquids. Template spreadsheet “NTC” used for calibration of NTC probed in analytical laboratory is depicted in Figure 4.


NTC Calibration

Constants

R0	1000	Ohm
Beta	3930	

Input

t0	293,15	K
t1	373,15	K

Results

R1	**=R0*EXP(Beta*(1/t_1-1/t_0))**	**Ohm**

Figure 4. Worksheet “NTC” used for ABAC model evaluation.

NTC probe calibration spreadsheet is used on daily basis by different team members. Every Monday, calibration expert performs checks and potentially calibrations on all 65 probes utilized in laboratory. One spreadsheet instance is managed for each calibration probe. Laboratory Manager reviews all calibration spreadsheets and if results of NTC probe calibration complies with laboratory guidelines, Manager changes colour of result cell and corresponding worksheet to green as evidence of review and compliant status of performed calibration. Lastly, on Tuesdays, laboratory administrator edits header label and adds information about probe serial number, date of calibration, prints calibrations results on preformatted stickers and attaches them on probe housing. All calibrated probes should be ready for
Based on above process description following sentences in natural language describes user’s role:

- Calibration Expert is responsible for calibration of NTC probes. Calibration should be performed every Monday at the beginning of working week. Results of checks and calibrations should be recorded in dedicated spreadsheet for each calibrated NTC probe, in designated cells for calibration results. Other elements of spreadsheet should remain unchanged.
- Laboratory Manager is responsible for reviewing and approving performed NTC probe calibrations. As evidence of approval and satisfactory calibration results, Manager should change color of result cell and of NTC worksheet to green. Yellow color should be used if probe requires recalibration and red color if probe should be inspected by technician or discarded for future use.
- Laboratory administrator is responsible to edit header of all green (approved) spreadsheets every Tuesday, add probe serial number and date to worksheet header label, print final worksheet on sticker and apply sticker to probe housing. This activity should be completed by end of Tuesday.

Based on above role descriptions in natural language we translated them to ABAC access policy (AP) for three different user roles within analytical laboratory.

<expert, update, R0, Monday>∧
<expert, update, Beta, Monday>∧
<expert, update, t0, Monday>∧
<expert, update, t1, Monday>

<manager, update, R1.color=”Green”, Monday>∧
<manager, update, R1.color=”Yellow”, Monday>∧
<manager, update, R1.color=”Red”, Monday>∧
<manager, update, NTC.color=”Green”, Monday>∧
<manager, update, NTC.color=”Yellow”, Monday>∧
<manager, update, NTC.color=”Red”, Monday>

<admin, update, header1.type="Date", Tuesday>∧
<admin, update, header2.type="String", Tuesday>

To align with provided definitions for access mode determination and conflict resolution, above policy is structured with conjunction statements for each user's role that participate in evaluation.

In continuation of evaluation process, we requested analytical laboratory team to continue with their standard weekly procedures and provide us access to all spreadsheet instances generated during the weekly calibration procedures. From the perspective of laboratory users, our evaluation had no impact to their well-established procedures and routines.

To determine users' activity and generate dynamically hierarchy of spreadsheet resources under evaluation we utilized Abstract State Machine (ASM) model for spreadsheets (Zdilar, 2023). ASM model was instrumental to identify all manipulations to spreadsheet resources including formula expressions. ASM has been implemented in Python 3 programming language (Van Rossum & Drake, 2009), with OpenPyXL library utilized for parsing spreadsheet formulae (Zumstein, 2021). ASM model generates directed graph as result of changes performed by users in spreadsheet under evaluation. Edges in graph represents state transitions and corresponds to changes and modifications to spreadsheets resources performed by users. We further utilized NetworkX python library (Hagberg et al. 2008) to generate graph queries and compare identified user changes with allowed actions in defined access policy.

We successfully parsed and analysed all 65 spreadsheets generated by laboratory users during one week of calibration activities. Due to experience of laboratory stuff, only two minor unauthorized changes have been detected by proposed model – cell colour in one spreadsheet was not properly changed and laboratory administrator completed printout of spreadsheet stickers following working day due to other priorities. To confirm correctness of determined changes by our proposed ABAC model, we manually review results for all 65 laboratory spreadsheets.

6 Conclusion and Future Research

In this paper we have presented initial results of our research on conceptual modelling of access control for spreadsheets. We structured our research around Design Science paradigm and two formulated research questions. We believe that proposed Attribute-Based Access Control metamodel for spreadsheets and results of its evaluation provides initial answers to formulated research questions. Our initial evaluation in analytical laboratory provided valuable feedback for proposed ABAC metamodel. However, our evaluation is limited to one use case and in our future research we will conduct model evaluation in more comprehensive case studies and large multi-user environments. We will also explore opportunities to apply formal verification techniques for model requirements and correctness of access mode determination and rule conflict resolution. Based on valuable feedback from evaluation participants, we noticed that generation of machine-readable access rules and translation from users' roles documented in natural language is challenging process. We will explore opportunities to automate this process with minimal manual intervention and represent access rules with standardized formats. We focused our initial research on key components of ABAC metamodel, and deployment aspects documented in NIST Special Publication 100-162 (Hu et al., 2014) were out of scope in this research.

Acknowledgments

The author would like to thank Prof. dr. sc. Markus Schatten from Artificial Intelligence Laboratory, for valuable comments and suggestions.

References

Abraham, R., & Erwig, M. (2006, July). Type inference for spreadsheets. *In Proceedings of the 8th ACM SIGPLAN international conference on Principles and practice of declarative programming* (pp. 73-84).

Bartholomew, P. (2023). Excel as a Turing-complete Functional Programming Environment. *arXiv preprint arXiv:2309.00115.*

Boadu, E. O., & Armah, G. K. (2014). Role-based access control (RBAC) based in hospital management. *Int. J. Softw. Eng. Knowl. Eng, 3*, 53-67.

Dholakia, Y. (2017). Mandatory Access Control – Problems in it and propose a model which overcomes them. *International Research Journal of Engineering and Technology (IRJET), (4)4*, pp.2031-2035

Downs, D. D., Rub, J. R., Kung, K. C., & Jordan, C. S. (1985, April). Issues in discretionary access control. In *1985 IEEE symposium on security and privacy* (pp. 208-208). IEEE.

European Spreadsheet Risk Interest Group. (2023). EuSpRIG Horror Stories. https://eusprig.org/research-info/horror-stories/ (June 10, 2024)

Gross, C. (2021). Announcing LAMBDA Helper Functions: Lambdas as arguments and more. https://techcommunity.microsoft.com/t5/excel-blog/announcing-lambda-helper-functionslambdas-as-arguments-and-more/ba-p/2576648 (June 10, 2024)

Hatmaker, C. (2023). Reducing Errors in Excel Models with Component-Based Software Engineering. *arXiv preprint arXiv:2309.00650.*

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science-Hevner. Design Science in Information Systems Research. *MIS Quarterly, 28 (1)*, 75–105.

Hu, V. C., Ferraiolo, D., Kuhn, R., Schnitzer, A., Sandlin, K., & Scarfone, K. (2014). Sp 800-162. guide to attribute based access control (abac) definitions and considerations. *Nat'l Inst. Standards and Technology*, 800-162.

Isakowitz, T., Schocken, S., & Lucas Jr, H. C. (1995). Toward a logical/physical theory of spreadsheet modeling. *ACM Transactions on Information Systems (TOIS), 13*(1), 1-37.

Jannach, D., Schmitz, T., Hofer, B., & Wotawa, F. (2014). Avoiding, finding and fixing spreadsheet errors–a survey of automated approaches for spreadsheet QA. *Journal of Systems and Software, 94*, 129-150.

Kashmar, N., Adda, M., Atieh, M., & Ibrahim, H. (2021). A review of access control metamodels. *Procedia Computer Science, 184*, 445-452.

Korman, M., Lagerström, R., & Ekstedt, M. (2016). Modeling enterprise authorization: a unified metamodel and initial validation. *Complex Systems Informatics and Modeling Quarterly*, (7), 1-24.

Microsoft Excel. (2023). Announcing Python in Excel: Combining the power of Python and the flexibility of Excel. https://techcommunity.microsoft.com/t5/excel-blog/announcing-python-in-excel-combining-the-power-of-python-and-the/ba-p/3893439 (June 10, 2024)

Panko, R. R. (2008). Spreadsheet errors: What we know. what we think we can do. *arXiv preprint arXiv:0802.3457.*

Panko, R. R., & Ordway, N. (2008). Sarbanes-oxley: What about all the spreadsheets*?. arXiv preprint arXiv:0804.0797.*

Powell, S. G., Baker, K. R., & Lawson, B. (2008). A critical review of the literature on spreadsheet errors. *Decision Support Systems, 46*(1), 128-138.

Reschenhofer, T., Waltl, B., Shumaiev, K., & Matthes, F. (2017). A conceptual model for measuring the complexity of spreadsheets. *arXiv preprint arXiv:1704.01147.*

S Scaffidi, C., Shaw, M., & Myers, B. (2005, September). Estimating the numbers of end users and end user programmers. In *2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)* (pp. 207-214). IEEE.

Van Rossum, G., & Drake, F. L. (2009). Python 3 Reference Manual. *CreateSpace.*

Zdilar, M. (2023). Towards Automated Detection of Qualitative Spreadsheet Errors in Multi-user Environments. In *Central European Conference on Information and Intelligent Systems* (pp. 419-424). Faculty of Organization and Informatics Varaždin.

Zumstein, F. (2021). *Python for Excel: A Modern Environment for Automation and Data Analysis* (pp. 155-179). O'Reilly Media.