IMSA - Intra Model Security Assurance

Qiang Zhi*, Shuichiro Yamamoto, and Shuji Morisaki
Nagoya University, Nagoya, Aichi, 464-8601, Japan
zhiqiang0728@gmail.com, syamamoto@acm.org, morisaki@is.nagoya-u.ac.jp

Abstract

Security assurance cases for architecture diagrams are developed independently by using traditional approaches. This paper proposes a new method, Intra Model Security Assurance, IMSA, approach to develop both security assurance cases and architectures in the same diagrams by using ArchiMate. IMSA enables to efficiently assure security by reducing the cognition and operation gaps caused by manipulating different diagrams such as security assurance cases and architecture diagrams. The effectiveness of the proposed method is also showed by experimental evaluation. According to the experimental results, proposed approach is superior to traditional approach for assuring security.

Keywords: Security case, Assurance case, Intra Model Security Assurance, Enterprise Architecture, ArchiMate.

1 Introduction

Assurance cases had been developed separately to assure safety, security, and dependability for architectural artifacts. We have proposed security and dependability case development methods so far. These methods assumed assurance cases are different artifact from those of architectural diagrams. Using different diagrams for describing assurance cases leads to the following problems.

- Linking architectural elements and the corresponding claims in assurance cases is necessary.
- Understanding and memorizing different diagram structures is necessary
- Looking different diagrams simultaneously is necessary.

Although we developed a method [28] to generate assurance cases exhaustively based on architecture diagrams, the method has not been widely accepted by field engineers in Japan. Our investigations on Japanese engineers showed that they did not want exhaustive assurance cases, but wanted more focused local assurance cases. For the local assurance cases, it is troublesome to manage different two diagrams such as architectural and assurance case diagrams. If we describe assurance claims and evidence in a united architectural diagram, the local assurance of architecture will be realized by only using a united architectural diagram consolidating architecture and security case descriptions.

If assurance case elements could be represented in the architectural diagrams, it will be easy to assure architectures because there is no need to change different diagrams to check elements among diagrams. This paper proposes an Intra Model Security Assurance (IMSA) approach for developing security cases by relating elements of security case with those of architecture in architecture diagrams. Using the approach will reduce the gap of exchanging architecture and security case diagrams. The approach will also have the possibility of efficiently assuring security of architectures. To clarify the superiority of the proposed method against to the traditional approach, a case study and an experiment are conducted.

Journal of Internet Services and Information Security (JISIS), volume: 8, number: 2 (May 2018), pp. 18-32

*Corresponding author: Room 566, South Building of Integrated Build, Graduate School of Informatics, Nagoya University, Furo-cho, Chikusa-ku, Nagoya, Aichi, 464-8601, Japan, Tel: +81-052-789-5989

On the case study, diagrams for architecture and security case for both methods were developed and compared for a secure retrieval on cloud storage service. On the experiment, diagrams for architecture and security case for both methods were also developed and compared for Healthcare device and Smart house systems. Two groups of subjects are assigned to answer questions on investigating diagrams of both approaches. The time and correctness of questions are evaluated to compare both methods.

The rest of the paper is organized as follows. Section 2 describes related work. Section 3 proposes IMSA. An example of cloud security architecture design is provided by using IMSA in Section 4. The comparative experiment is described in Section 5. The effectiveness of IMSA is discussed in Section 6. Finally, section 7 summarizes and provides the future work of the paper.

2 Related work

The Goal Structuring Notation (GSN) was proposed [10, 11] and widely used to develop assurance and safety cases. The Open Group (TOG) Real Time & Embedded Systems Forum focuses on standardizing for high assurance, secure, and dependable open systems [2]. The purpose of the O-DA (Open Dependability through Assuredness) standard is providing the concept of modeling dependability, building assurance cases, and achieving agreement on accountability on the event of actual or potential failures. Dependability cases are used to assure dependable system architectures.

The argument patterns [33] have been proposed to help engineers develop assurance cases. A security argument pattern and security case based on common criteria [30, 26, 7] have been proposed for assuring security. Mobile security assurance approaches based on attributed GSN [29, 31] had been proposed. Attributed GSN can manage values assigned to claim nodes to represent goal achievement ratio. These attributes are useful to compare solutions.

The more generic approach to generate assurance cases based on the elements and relationships of the architecture described in ArchiMate [28, 34, 13, 15, 25] had been proposed. ArchiMate [9, 1] is a visual modeling language for describing Enterprise Architecture. TOG standardizes ArchiMate. UML (Unified Modeling language) focuses only for modeling software. SysML (Systems Modeling language) focuses on systems modeling. ArchiMate can be used to model business, application, and technology architectures as well as motivation aspects. The motivation aspects include business drivers, principles, assessments, goals, requirements, and constraints. The motivation elements can be used to represent claim goals and evidences.

Grandy and others [8] proposed an integrated approach on EA and security risk management. Their approach was limited by the capability of ArchiMate 2.0 that is a former version of the current ArchiMate 3.0. For example, their approach did not use the influence relationship between requirements and countermeasure concepts. Feltus and others [6] described a meta-model for SCADA (Supervisory Control and Data Acquisition) systems and then by using ArchiMate they described components behavior for mitigating cyber-crime actions. The approach did not clarify the influence impacts on assurance between particular crime mitigation components and risk management policies. Korman and others [14] compared ArchiMate 2.0 concepts with various risk assessment methods. They clarified the concept coverage of ArchiMate. Although they concluded that ArchiMate models might be a source of guidance for risk assessments, they did not clarify the detailed security assurance method using ArchiMate. Band and others [5] published a White Paper to provide guidelines to ArchiMate users on modeling the enterprise risk and security. The contribution of this White Paper has been called a “Risk and Security Overlay” (RSO) of the ArchiMate language. Abbass and others [3] described an Information System Security Risk Management (ISSRM) model by the constructs of ArchiMate. Mayer and Feltus evaluated the risk and security model of Archimate using ISSRM[16]. The completeness of the RSO visual expressions has been evaluated with regard to ISSRM and cognitive effectiveness by using the nine principles of Moody

[17].

3 Intra Model Security Assurance approach

This section explains IMSA approach to relate architecture and security case diagrams in the same diagrams. The IMSA approach provides the high efficiency to assure security, because it can directly assure security of assets in the same diagrams without exchanging diagrams.

3.1 Meta model of Architecture and Security case

It is necessary to represent architecture and security case in the same diagrams. Figure 1 integrates meta-models of architecture and security case. The meta-model of architecture consists of the target of assurance, elements and relationships. The target of assurance represents the system as a whole. The meta-model of security case consists of target of assurance, property, risk, counter measure, and evidence. The evidence will be realized by elements of the target system.

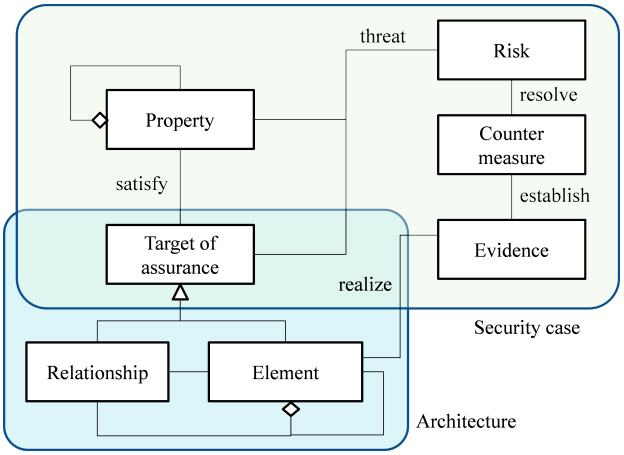

Figure 1: Meta model of architecture and security case

3.2 Using ArchiMate for Security case

As ArchiMate provides elements to describe motivations of architectures. The motivation elements are driver, assessment, goal, and requirement. The Table 1 shows an inter relationship among the meta-model, security case and motivation elements.

Table 1: Meta model, Security case and Motivation elements of ArchiMate.

Meta model	Security case	Motivation elements
Property	Top Claim	Driver
Risk	Context	Assessment
Counter measure	Sub claim	Goal
Evidence	Evidence	Requirement

In security case, property, risk, counter measure, and evidence of meta-model are described by top claim, context, sub claim, and evidence, respectively. In ArchiMate, property, risk, counter measure, and

evidence of meta-model are described by driver, assessment, goal, and requirement, respectively. Figure 2 shows a generic example of security case configuration in ArchiMate. Although the figure only shows one risk for property, many numbers of risks can be allocated to the property. In the same way, other numbers of counter measure and evidence can also be added for risk and counter measure, respectively.

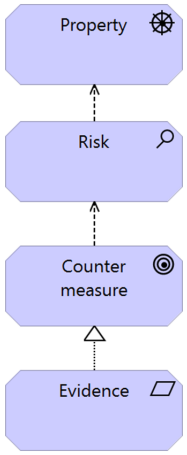

Figure 2: Security case configuration in ArchiMate motivation elements

4 Case Study

This section gives an example about secure retrieval on cloud storage [35], and explains in ArchiMate.

4.1 Target System

The target system is composed of search server, cloud storage server and user who wants to search files. All data stored on cloud storage server is encrypted, including the file that stores search information. Cloud storage server can't decrypt any data because it does not have decryption keys. Search server can only get the search information file from cloud storage server and only has the decryption keys of search index. The search server decrypts the corresponding index file based on user's attributes so that the user only can search in the corresponding scope by his own attribute. The encryption algorithm in this target system is Ciphertext-Policy Attribute-Based Encryption [4], CP-ABE, which is a new Public-Key cryptography and it is very suitable for data sharing. This target system is shown in Figure 3.

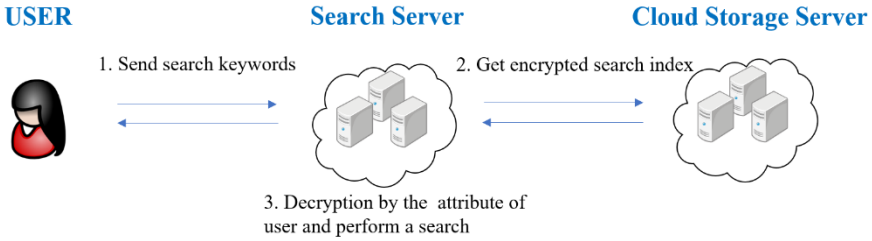

Figure 3: Secure Retrieval on Cloud Storage

4.2 Target System in IMSA

The target architecture described in ArchiMate is shown in Figure 4. The Figure 4 shows application and technology architecture. In the technology layer, Cloud Storage Server stores Encrypted Index and Encrypted file data. The CP-ABE tool on Search Server has Encrypting and Decrypting functions. Network realizes Communication Protection.

In the application layer, Search Processing is composed of Attribute Checking, Decoding and Searching. Attribute Checking function checks the Attribute from user, Decoding function decrypts Encrypted Index, then Searching function performs a retrieval. Attribute Checking, Decoding and Searching realize Checking Service, Decoding Service and Searching Service respectively. Checking Service, Decoding Service and Searching Service serve Secure Search. User Device, Search Server and Cloud Storage Server are connected by Network. Communication Protection which realized by Network encrypts communication. System software on server realize Login Check and Identity Check.

The security of the target system architecture is analyzed from the point of confidentiality includes data confidentiality, communication confidentiality and retrieval confidentiality. The security case for the confidentiality on the target system architecture is shown in Figure 5. Figure 5 integrates the target system architecture and the corresponding security case in the same ArchiMate diagram.

Security issues include Storage Server Data Leakage, Index Data Tampering, exceeding authority, Search Server Data Leakage, Attribute Forgery, Keyword Disclosure and Keyword Tampering. Detect Illegal Users will be effective countermeasure for Server Data Leakage. The countermeasures for Index Data Tampering, Keyword Disclosure, Keyword Tampering and exceeding authority are to increase the difficulty of implementing these actions.

The evidence for the countermeasures of Server Data Leakage is Login Authentication and User Authentication, The evidence for the countermeasure of Index Data Tampering is Pairing, The evidence for the countermeasure of exceeding authority is Search Scope Judgement, the evidence for the countermeasure of Attribute Forgery is Digital Signature, the evidence for the countermeasures of Keyword Disclosure and Keyword Tampering are Communication Encryption and Hash Check, respectively.

These requirement are realized by Identity Check function, Login Check function, Pairing function, Secure Search process, Attribute Checking function, Encoding function and Hush function, respectively.

4.3 Target System in D-case

The security case for target system is also described by using D-case as follows. D-Case is a tool to describe GSN which had been developed in the course of DEOS project. Figure 6 shows the security case of target system.

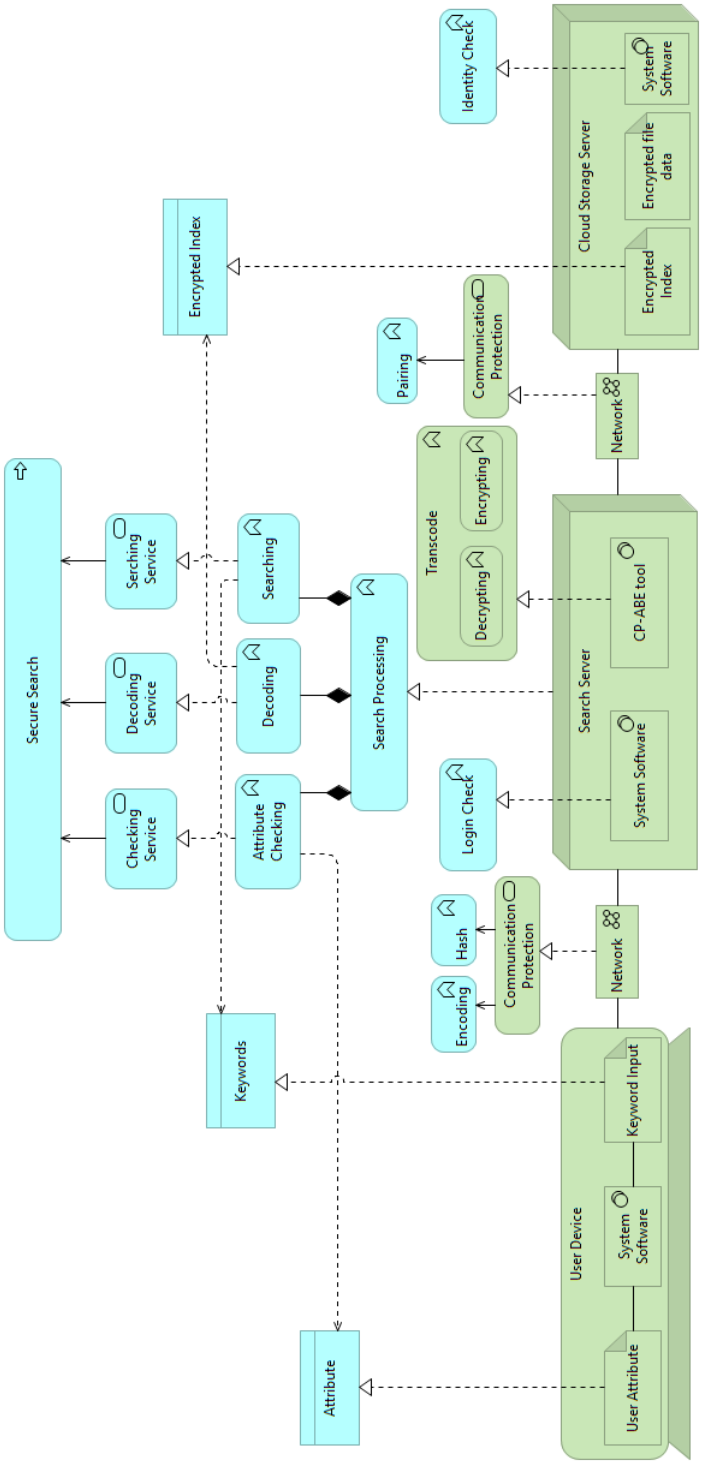

Figure 4: Secure Retrieval on Cloud Storage Architecture in ArchiMate

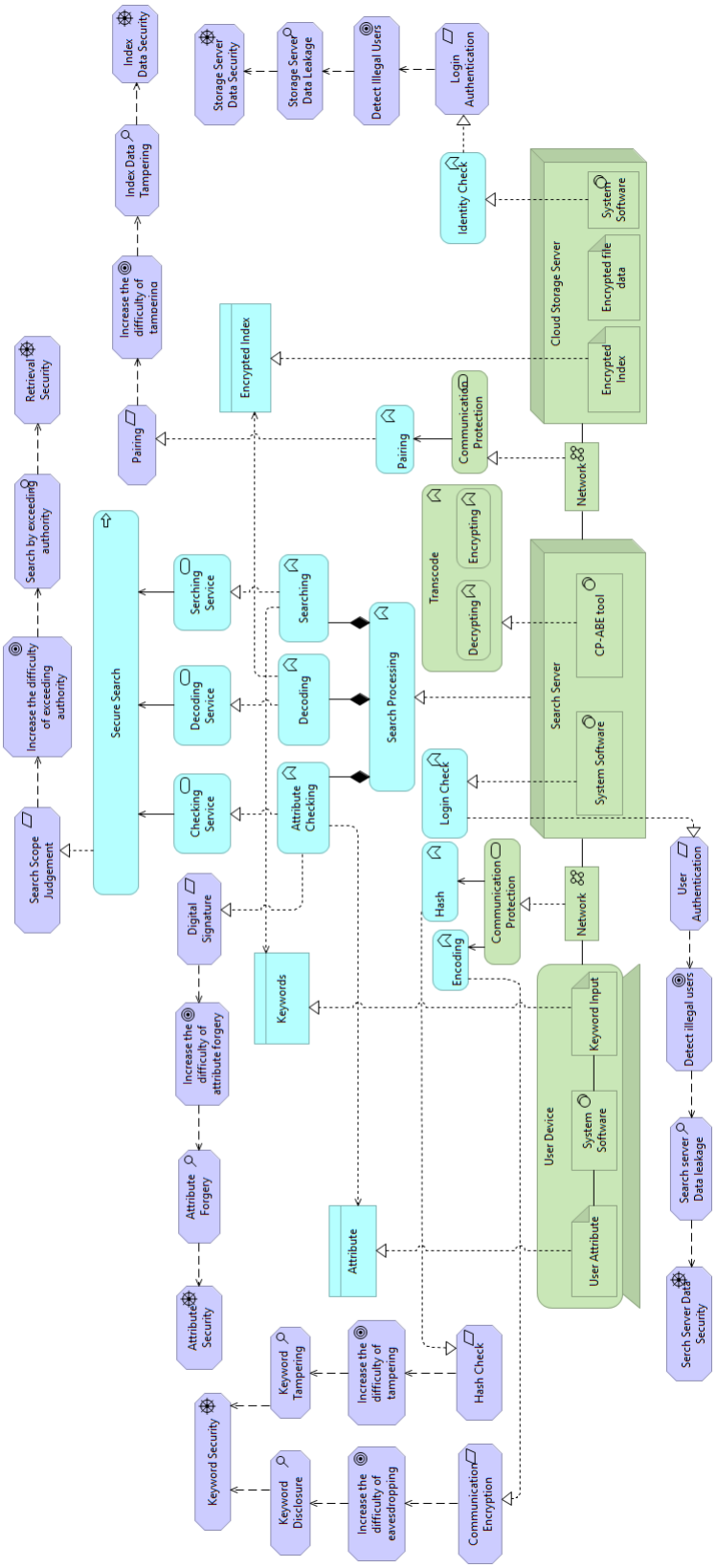

Figure 5: Intra Security Case Example for Secure Retrieval on Cloud Storage.

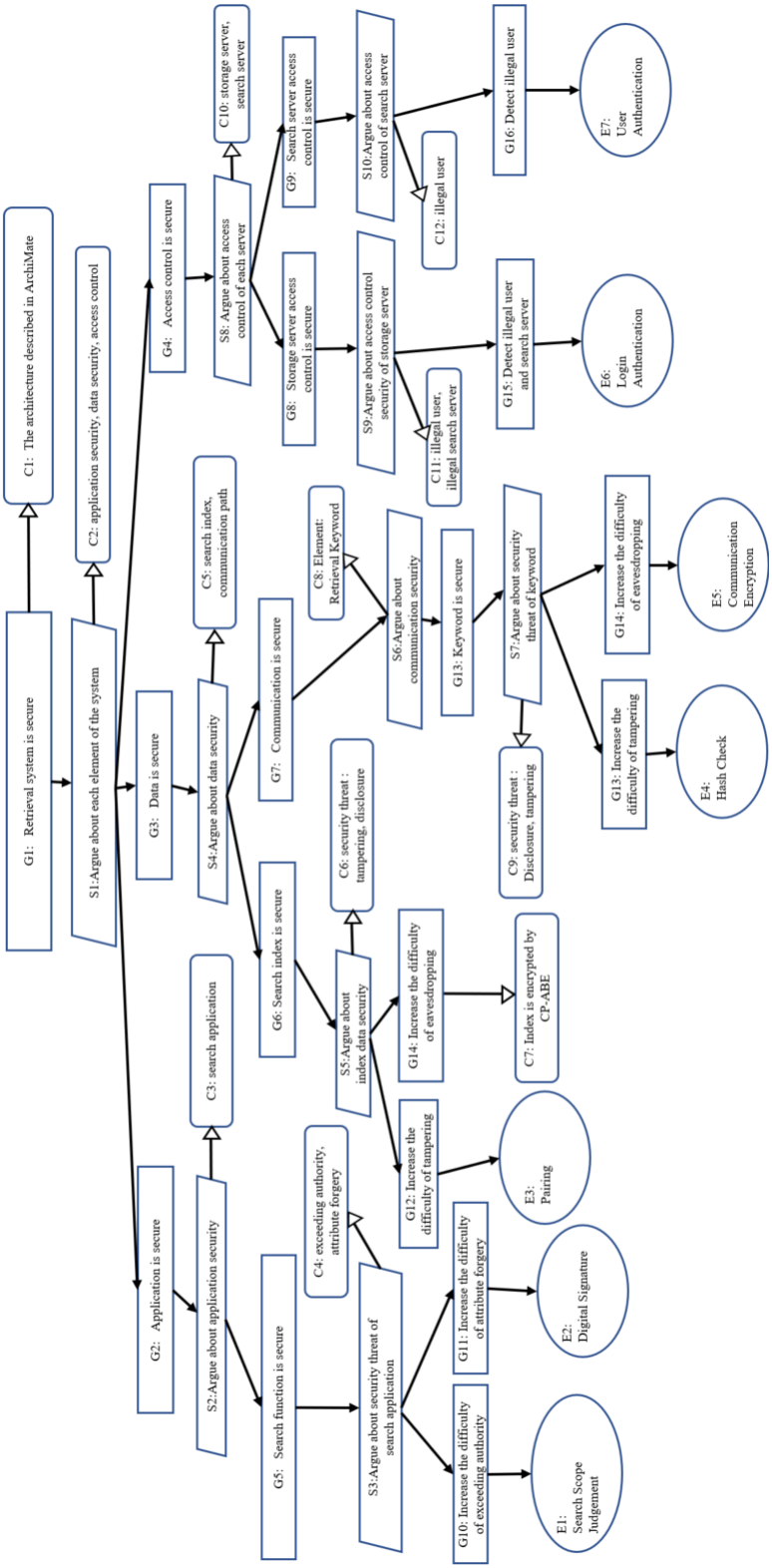

Figure 6: Security Case Using D-case

4.4 Comparison

Table 2 shows the number of nodes of the system. The number of architecture nodes for the target system is 30. The number of security case nodes for D-Case in the target system is 48. The number of security case nodes for ArchiMate in the target system is 27. The number of nodes for ArchiMate are smaller than those of D-Case and the reduction is about 44%. The reason is there is no need to describe context and decomposition nodes in ArchiMate. In D-Case, 12 context nodes are used to correspond with architecture elements and claims. Moreover, 10 decomposition nodes are necessary to decompose claim nodes into sub claim nodes in D-Case.

Table 2: Number of Nodes of models and security cases.

Meta model	Number of Nodes
Model in ArchiMate	30
Security case in D-Case	48
Security case in ArchiMate	27

Table 3 shows the number of relationship of the system. The number of relationship for the target system is 30. The number of relationship for D-case in the target system is 44. The number of security case relationship in ArchiMate for the target system is 28. The number of relationship for ArchiMate are smaller than those of D-Case and the reduction is about 36%. The reduction of relationship is smaller than the reduction of nodes and the reason is that there are 7 relationships between security case and system model.

Table 3: Number of Relationship of models and security cases.

Meta model	Number of Relationship
Model in ArchiMate	30
Security case in D-Case	44
Security case in ArchiMate	28
Between security case and model	7

5 Experiment

In order to evaluate the proposed method, we compare the method with the traditional approach. We conducted experiments on two systems to improve the accuracy of the experimental results, and selected Healthcare device system and smart house system as the experiment object. Table 4 shows the number of nodes of the systems. The number of architecture nodes for the Healthcare device and Smart house systems are 28 and 36, respectively. The numbers of security case nodes for D-Case in the Healthcare device and Smart house systems are 103 and 150, respectively. The number of security case nodes for ArchiMate in the Healthcare device and Smart house systems are 55 and 72, respectively.

Six subjects are decomposed into two groups A and B. Each group contains three subjects. Subjects are all students who knows D-case and ArchiMate. In the comparative experiment, six subjects are ordered to answer the four questions defined for each system in Table 5.

Table 4: Number of Nodes of models and security cases.

Method	Healthcare device system	Smart house system
Model in ArchiMate	28	36
Security case in D-Case	103	150
Security case in ArchiMate	55	72

Table 5: Questions of experiments.

Questions	Healthcare device system	Smart house system
What is the Threat for the asset?	Healthcare data	Operation information
What is the evidence for the counter measure of the threat?	Threat for the pairing key	Threat for the smart device states
What element realizes the evidence?	Elements to realize above evidence	Elements to realize above evidence
What is the function does not realize the evidence?	Missing functions for the above evidence	Missing functions for the above evidence

Table 6 shows the combination of subject groups and target systems. Each group consists of three members.

Table 6: Combination of the experiment.

Group	Healthcare device system	Smart house system
A	Traditional	Proposed(IMSA)
B	Proposed(IMSA)	Traditional

In the experiment, the average time to answer questions and the average ration of correct answers are collected. The results of the experiment are shown in Figure 7 and Figure 8.

Figure 7 shows the comparison of average time to answer questions. The Figure 7 shows the average time to answer for proposed approach is less than those of traditional approach for two groups.

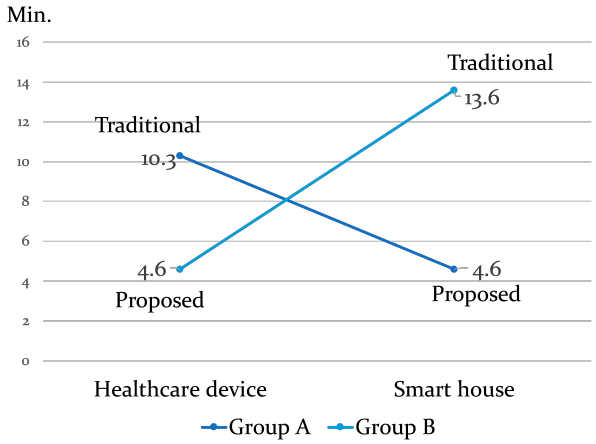

Figure 7: Comparison of average time to answer questions.

Figure 8 shows the comparison of average correct ratio of questions. The Figure 8 shows the average correct answer ratio for proposed approach is greater than those of traditional approach for two groups.

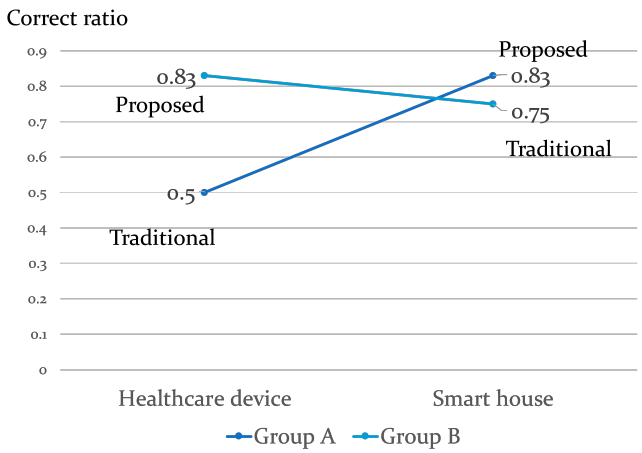

Figure 8: Comparison of average correct ratio of questions.

6 Discussion

6.1 Comparison of the traditional and proposed approaches.

Table 7 summarizes the comparison between proposed and traditional approaches to assure security of architectures. The proposed approach can easily correspond with security cases and architecture elements. Traditional approach can not directly relate elements of assurance cases and architectures. In case of traditional approach, it is necessary to use specific security case editor other than architecture editors for developing and exploring security cases. The mapping rules between security case and architecture diagrams are necessary to develop security cases for traditional approach. The mapping rules of traditional approach should be collected as security case patterns for different architecture domains. Different from traditional approach, the proposed approach only necessary to define the usage rule of motivation elements for security case as shown in Table 7.

Table 7: Comparison of approaches.

Items	Traditional approach	Proposed approach
Artifacts	Architecture diagram and GSN	ArchiMate
Relations among elements	By using names indirectly	By relating nodes directly
Editing operations	By using two diagram editors	By using ArchiMate editor
Rules of diagramming	Mapping rules between two diagrams	Description rules of security cases in ArchiMate

6.2 Effectiveness.

The case study in section 4 showed that proposed approach can reduce the number of nodes and relationships approximately 40%. The correctness ratio of the proposed method was higher than those of the

traditional method. The time to answer questions of the proposed method was also less than those of the traditional method. These result showed the effectiveness of the proposed method. This result positively verified our hypothesis that using one diagram is superior to using two different diagrams for assuring security.

6.3 Limitation.

The number of subjects on the experiment was small. The experiments using more number of subjects are necessary to generalize the result of the paper. The productivity and quality of the proposed method were not evaluated. It is also necessary to evaluate these fundamental properties for the proposed approach.

7 Conclusion

This paper proposed an IMSA approach to develop security cases and architectures in the same diagram. So far security cases are described by using a specific goal oriented diagram notation. The goal oriented diagram is different from the architectural diagrams. Therefore engineers have the problem to manage two different diagrams for developing architecture and assuring security.

The contribution of the paper is to clarify and the specific security assurance method with one EA modeling language. To consolidate different diagrams, we clarify the Meta model of architecture and security assurance as shown in Figure 1 . Then we develop a mapping from the Meta model to EA modeling language, ArchiMate.

To clarify the effectiveness of the proposed approach, we compared it with D-Case which is a derivative of GSN. The comparison consist of a case study and a quantitative experiment.

The case study for a secure retrieval on cloud storage service showed that the proposed approach reduced the number of diagram node and relationship for those of traditional approach. The reduction ratios for nodes and relationships were about 44% and 36%, respectively.

The experiment on Healthcare device and Smart house systems showed that the proposed approach improved the working time and correctness of investigation works on assuring security for those of traditional approach.

Future work include to extend the proposed method for the d* framework [32], [19], [22], [18], [20], [21], [23], and [24]. The claim goals of assurance cases are able to have attributes [27, 12]. The security case in ArchiMate is also extensible to have quantitative attributes.

References

[1] `https://www.cetis.org.uk/projects/` [Online; accessed on May 15, 2018]. Archi.

[2] The open group standard: Real-time and embedded systems: Dependability through assuredness™ (o-da) framework. `https://publications.opengroup.org/c13f`, [Online; accessed on May 15, 2018], 2013.

[3] W. Abbass, A. Baina, and M. Bellafkih. Improvement of information system security risk management. In *Proc. of the 4th IEEE International Colloquium on Information Science and Technology (CiSt'16), Tangier, Morocco*, pages 182–187. IEEE, October 2016.

[4] J. Bethencourt, A. Sahai, and B.Waters. Ciphertext-policy attribute-based encryption. In *Proc. of the 28th IEEE Symposium on Security and Privacy (S&P'07), Oakland, California, USA*, pages 321–334. IEEE, May 2007.

[5] I. B. et al. Modeling enterprise risk management and security with the archimate language. `https://publications.opengroup.org/w172` [Online; accessed on May 15, 2018], January 2015. The Open Group, White Paper.

[6] C. Feltus, M. Ouedraogo, and D. Khadraoui. Towards cyber-security protection of critical infrastructures by generating security policy for scada systems. In *Proc. of the 1st International Conference on Information and Communication Technologies for Disaster Management (ICT-DM'14), Algiers, Algeria*, pages 1–8. IEEE, March 2014.

[7] A. Finnegan and F. McCaffery. A security argument pattern for medical device assurance cases. In *Proc. of the 2014 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW'14), Naples, Italy*, pages 220–225. IEEE, November 2014.

[8] E. Grandry, C. Feltus, and E. Dubois. Conceptual integration of enterprise architecture management and security risk management. In *Proc. of the 17th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW'13), Vancouver, British Columbia, Canada*, pages 114–123. IEEE, September 2013.

[9] T. O. Groupe. Archimate 3.0 specification. `http://pubs.opengroup.org/architecture/archimate3-doc/` [Online; accessed on May 15, 2018]. C162,Van Haren (2016).

[10] T. Kelly. A six-step method for the development of goal structures. York Software Engineering, September 1997. `https://www.researchgate.net/publication/245072237_A_Six-Step_Method_for_the_Development_of_Goa\unhbox\voidb@x\bgroup\@xxiil\egroup_Structures` [Online; Accessed on May 10, 2018].

[11] T. Kelly and J. McDermid. Safety case construction and reuse using patterns. In *Proc. of the 16th International Conference on Computer Safety, Reliability and Security (Safe Comp'97), York, UK*, pages 55–69. Springer, London, September 1997.

[12] N. Kobayashi, S. Morisaki, N. Atsumi, and S. Yamamoto. Quantitative non functional requirements evaluation using softgoal weight. *Journal of Internet Services and Information Security (JISIS)*, 6(1):37–46, February 2016.

[13] N. Kobayashi and S. Yamamoto. An evaluation of o-da template. In *Proc. of the 2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI'17), Hamamatsu, Japan*, pages 263–268. IEEE, July 2017.

[14] M. Korman, T. Sommestad, J. Hallberg, J. Bengtsson, and M. Ekstedt. Overview of enterprise information needs in information security risk assessment. pages 42–51. IEEE, September 2014.

[15] M. Lankhorst. *Enterprise Architecture at Work – Modeling Communication and Analysis*. Springer-Verlag Berlin Heidelberg, 3 edition, 2013.

[16] N. Mayer and C. Feltus. Evaluation of the risk and security overlay of archimate to model information system security risks. In *Proc. of the IEEE 21st International Enterprise Distributed Object Computing Workshop (EDOCW'17), Quebec City, Quebec, Canada*, pages 106–116. IEEE, October 2017.

[17] D. Moody. The 'physics' of notations: Toward a scientific basis for constructing visual notations in software engineering. *IEEE Transactions on Software Engineering*, 35(6):756–779, December 2009.

[18] T. Saruwatari, T. Hoshino, and S. Yamamoto. Method to share responsibility knowledge of dependability cases. *Procedia Computer Science*, 22:1073–1082, 2013.

[19] T. Saruwatari and Yamamoto. Definition and application of an assurance case development method (d*). *Springerplus*, 2(1):224–231, May 2013.

[20] T. Saruwatari and S. Yamamoto. Creation of assurance case using collaboration diagram. In *Proc. of the 2nd IFIP TC5/8 International Conference on Information and Communication Technology (ICT-EurAsia'14), Bali, Indonesia*, volume 8407 of *Lecture Notes in Computer Science*, pages 413–418. Springer, Berlin, Heidelberg, April 2014.

[21] T. Saruwatari and S. Yamamoto. D* framework creation procedure from collaboration diagram. *IT CoNvergence PRActice (INPRA)*, 2(2):43–54, 2014.

[22] T. Saruwatari, S. Yamamoto, and Matsuno. A comparative study of d*framework and gsn. In *Proc. of the 2013 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW'13), Pasadena, California, USA*, pages 315–320. IEEE, November 2013.

[23] M. Tokoro. *Open Systems Dependability, Dependability Engineering for Ever-Changing Systems*. CRC Press, 2 edition, 2012.

[24] M. Tokoro. *Open Systems Dependability*. CRC Press, 2015.

[25] G. Wierda. *Mastering ArchiMate – A Serious Introduction to the ArchiMate Enterprise Architecture Modeling Language*. R&a, 2 edition, 2014.

[26] S. Yamamoto. An evaluation of argument patterns based on data flow. In *Proc. of the 2nd IFIP TC5/8 International Conference on Information and Communication Technology (ICT-EurAsia'14), Bali, Indonesia*, volume 8407 of *Lecture Notes in Computer Science*, pages 432–437. Springer, Berlin, Heidelberg, 2014.

[27] S. Yamamoto. An approach for evaluating softgoals using weight. In *Proc. of the International Conference on Information and Communication Technology (EurAsia'15), Daejeon, Korea*, volume 9357 of *Lecture Notes in Computer Science*, pages 203–212. Springer, Cham, October 2015.

[28] S. Yamamoto. An approach to assure dependability through archimate. In *Proc. of the 2015 International Conference on Computer Safety, Reliability, and Security (Safe Comp'15), Delft, The Netherlands*, volume 9338 of *Lecture Notes in Computer Science*, pages 50–61. Springer International Publishing, September 2015.

[29] S. Yamamoto. Assuring security through attribute gsn. In *Proc. of the 5th International Conference on IT Convergence and Security (ICITCS'15), Kuala Lumpur, Malaysia*, pages 1–5. IEEE, August 2015.

[30] S. Yamamoto, T. Kaneko, and H. Tanaka. A proposal on security case based on common criteria. In *Proc. of the International Conference on Information and Communication Technology (ICT-EurAsia'13), Yogyakarta, Indonesia*, volume 7804 of *Lecture Notes in Computer Science*, pages 331–336. Springer, Berlin, Heidelberg, March 2013.

[31] S. Yamamoto and N. Kobayashi. Mobile security assurance through archimate. *IT Convergence Practice (INPRA)*, 4(3):1–8, September 2017.

[32] S. Yamamoto and Matsuno. d* framework: Inter-dependency model for dependability. In *Proc. of the 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'12), Boston, USA*, pages 1–1. IEEE, June 2012.

[33] S. Yamamoto and Y. Matsuno. An evaluation of argument patterns to reduce pitfalls of applying assurance case. In *Proc. of the 2013 1st International Workshop on Assurance Cases for Software-Intensive Systems (ASSURE'13), San Francisco, California, USA*, pages 12–17. IEEE, May 2013.

[34] S. Yamamoto and S. Morisaki. A case study on architecture quality assurance service using o-da. In *Proc. of the 2016 International Conference on ENTERprise Information Systems (CENTERIS'16), Porto, Portugal*, pages 189–193, October 2016.

[35] Q. Zhi, T. Ohigashi, R. Aibara, and K. Nishimura. Implementation and evaluation of secure search functions in cloud storage. `http://www.ieee-jp.org/section/hiroshima/En/doku.php?id=start` [Online; accessed on May 15, 2018], 2015. 17th IEEE Hiroshima Section.

Author Biography

Qiang Zhi is a Ph.D student in the Graduate School of Informatics at Nagoya University, Japan. Previously, he was a programmer in the Japanese software industry. He received a master's degree in information science from Hiroshima University in Japan in 2016. His research interests include software engineering and information security.

Shuichiro Yamamoto received the Dr. Eng. degree from Nagoya University in 2000. He is a professor of Graduate School of Informatics at Nagoya University. He joined Nippon Telegraph and Telephone Public Corporation (now NTT) in 1979 and engaged in the development of CASE tools, network-based smart card environments, and distributed application development platforms. He moved to NTT DATA in 2002. He became the first Fellow of NTT DATA Research and Development Headquarters in 2007. He moved to Nagoya University as a professor in 2009.

Shuji Morisaki is an associate professor in the Graduate School of Informatics at Nagoya University, Japan. Previously, he has been a software engineer in the Japanese software industry. He received a DE in information science from Nara Institute of Science and Technology, Japan in 2001. His research interests include empirical software engineering and software quality.