A New Approach to Develop a Dependable Security Case by Combining Real Life Security Experiences (Lessons Learned) with D-Case Development Process

Vaise Patu and Shuichiro Yamamoto

Nagoya University, Furo-cho Chikusa-ku, Nagoya City 〒464-8601, Japan
{dr.vpatu,yamamotosui}@icts.nagoya-u.ac.jp

Abstract. Modern information and distributed systems runs for extensive periods of time and are being constantly improved in service objectives under evolving technologies and changing regulations and standards. These systems have become extremely complex and therefore, it is very important that they are to be dependable in order for them to execute their functionalities and purposes correctly or to an acceptable level of services. However, due to the ever-growing complexity of information and distributed systems, it is very difficult to achieve dependability by relying only on conventional technologies such as development processes and formal methods. And therefore the idea of Assurance Case or D-Case (dependability case) has become more and more a popular notion. Recently, D-Case which is an extension form of Assurance Case, is more commonly associated with the safely aspect of dependability. And because of this regard, safety cases are more well known in comparison to other aspects of dependability such as availability, integrity and confidentiality witch are all related to the security domain. In this paper, we introduce our new approach to the development of a dependable security case.

1 Introduction

It is very difficult to define what exactly a secure system means, because the concerns in the security domain are too diverse. For example, some secure systems are more concerned with unauthorized access to information, while some are more concerned with denial-of-service attacks and so on. In this paper, we discussed how we developed a dependable security case for our e-learning distributed system by utilizing the knowledge we gained from previous occurred attacks as a mechanism to build solutions that counters any more future attacks of the same kind. As we all know, it is common for network systems to contain security vulnerabilities that allow unauthorized personnel to compromise the systems, steal intellectual property, or disclosure sensitive data. To combat these vulnerabilities, a proactive approach to building secure network systems and strong evident based security case is necessary.

A. Cuzzocrea et al. (Eds.): CD-ARES 2013 Workshops, LNCS 8128, pp. 457–464, 2013.

There are many well-written documents on how to write safety cases and reliability cases but very few on security cases. The reason for this is not the concern of this paper but to propose our method on how we developed our security case by complying with previous systems risks and attacks and the lessons we learned from it.

2 Our New Approach to the Development of a Dependable Security Case

How we decide on the aspect of the system to be assured using an assurance case, is directly derives from the system under review risk analysis results and its set of requirements. However in our approach, we also added in our experience (lessons learned) gained from the attacks our system faced in the past as shown in figure 1. Since we are building a security case, it was close to impossible for us to predict the vulnerabilities and the security risks that reside in our system architecture when our distributed e-learning system starts its operation. But as we continued our operation for almost 6 years now, we gradually improved our system performance and security by building counter measures to any severe exploitation we faced in the past. As an example in Table 1, we have listed down some of the most common security vulnerabilities that are most popularly delt with in the domain of security especially when dealing with information and network systems. From the this known list of security vulnerabilities / claims, we build from it a correponding list of solutions or evidence that comes in handy when we go through the process of creating of our security case.

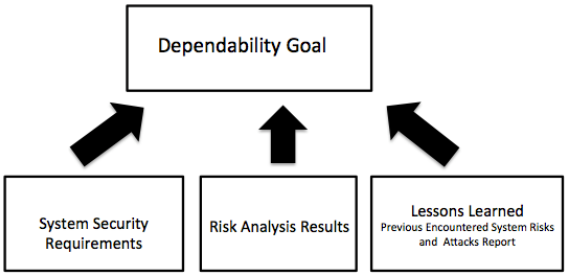

Fig. 1. To achieve the goal of system dependability, we proposed the addition of a 3rd requirement, which are the reports of previous encountered system risks and attacks

The solution column is a result of our discussion, on how we should change our system configurations in order to minimize the effects of these attacks if we encountered if we encounter them in the future. We developed our 4-step process to register new attacks once we discover them.

A) Identifying the type of attack
B) Describe the nature of the attack
C) Build a counter strategy for the attack via discussions
D) Store the completed report to our system repository for future reference

Table 1. Our completed table of the known list of the most common security vulnerabilities traced back to the risk initiated network component and the solutions we have to limit the risk causal factors

Network Component at Risk	Security Vulnerabilities (- Claims)	Proposed Solution / Evidence
Boarder Router	Inadequate router access control: Misconfigured router ACLs can allow information leakage through ICMP, IP, NetBIOS and lead to unauthorized access to services on your DMZ servers.	All Router ACLs configurations should be checked and monitored constantly for accuracy
Remote Access Server	Unmonitored remote access points provide one of the easiest means of access to your corporate network. Telecommuters often connect to the Internet with little protection, exposing sensitive files to attack.	(i) Limit the number of Remote Access Servers within the system. (ii) All Remote Access Servers should be under monitoring continuously
Firewall	Misconfigured firewall ACLs can allow access to internal systems directly or once a DMZ server is compromised.	(i) Firewall ACLs configurations should be checked and monitored constantly for accuracy. (ii) Backup the completed configurations into a secure separate location.
Internet / DMZ Server	Information leakage can provide the attacker with operating system and application versions, users, groups, shares, DNS information via zone transfers and running services like SNMP, finger, SMTP, telnet, rusers, rpcinfo, NetBIOS.	<Undeveloped>
	Hosts running unnecessary services such as RPC, FTP, DNS, SMTP are easily compromised.	Limit the using of services such as RPD, FTP, DNS and SMTP
	Misconfigured Internet servers, especially CGI and ASP scripts on web servers, anonymous FTP with world-writable directories and XSS vulnerabilities	(i) All CGI and ASP scripts should be double checked by a second party for accuracy before the system is deployed (ii) Forbid the use of FTP with anonymous accounts and passwords on the system
	Inadequate logging, monitoring and detection capabilities at the network and host level	Boost continuous monitoring of the full system log-files in a daily or weekly bases to up the security threats early detection capability of the system
Branch Office	Excessive trust relationships such as Windows Domain Trusts, UNIX rhosts, and SSH files can provide attackers with unauthorized access to sensitive systems	<Undeveloped>
Workstation	Weak, easy to guess passwords at the workstation level can compromise your company's server	<Undeveloped>
	Unauthenticated services like X Windows allow users to capture remote keystrokes or workstation keystrokes after software is installed	<Undeveloped>
Internal LAN Server	Excessive file and directory access controls (Windows shares and UNIX NFS exports)	<Undeveloped>
	Software that is unpatched, outdated or left in default configurations, especially web servers are vulnerable to attacks	<Undeveloped>

3 Building of the Security D-Case

Our new approach of creating a dependable security case was first introduced and tested as an experiment to provide assurance to the security aspect of our distributed e-learning system. However, more experiment and tests (trail and error) is still needed in order to take much of the work into the next stage. The good news is, security is no stranger to network and information systems. These systems have been associated with security for as long as networking and information systems existed. What is really new here is the urge to associate network systems with assurance case to give some kind of confidence to the system stakeholders that the system they are getting will behave as predicted and function well at a certain level when it faced by security risks and attacks in the future.

3.1 Phase 1 of the Development Process - Set a Top Goal

Depending on the aspect of dependability we are writing the assurance case for, we could easily identify the most upper goal or the top goal for your D-Case diagram. In our example, since we are building a security case, therefore our top goal also known as the main claim should be: "System is reasonably secure".

In support of the top goal or top claim, we need some input documents into the context nodes. These documents provide the environment information about the system. This includes any sort of attachment that helps to make the main argument truthful or convincing. For example, documents like the system requirements, which tells of what kind of security requirements that the system has in order to secure the system from unwanted attacks, or any kinds of security design architecture diagram if any, and so forth. Figure 2 is to provide a visualized view of what this paragraph is trying explain. Figure 1 shows only the top most part of our security case diagram.

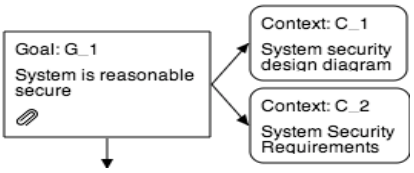

Fig. 2. The top part of the security case tree diagram

3.2 Phase 2 of the Development Process - Set an Argument Strategy to Decompose the Main Goal to Sub-goals

After the top goal is set with all the necessary contexts. What follows is the decomposition step or stage of the main argument or the 'main goal' into two or more sub-goals. However, prior to the decomposition stage, where the 'sub-goal nodes' comes into the picture, the node called 'strategy' is inserted between the main goal and

the sub-goals. The strategy node should explain or give a sense of justification or reasoning to why the security case builders decided to decompose the main goal into such and such number of sub-goals. In figure 3, the strategy links us to some of the identified security vulnerabilities displayed in table 1. In this example, we took 6 identified security vulnerabilities as sub-goals. And the strategy is to argue over each of the 6 identified security vulnerabilities.

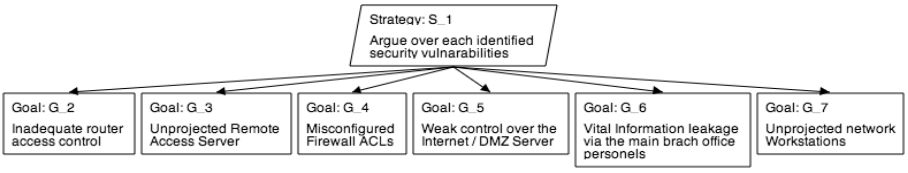

Fig. 3. This figure shows the decomposition process from the top goal into sub-goals with the strategy node in between

No matter what the assurance case is, whether it is a safety case, reliability case or a security case, they all follow a kind of pattern. For example, security case patterns are claims-argument-evidence structures that can be reused in many different security cases. The security case method offers the opportunity for security and domain experts to organize security knowledge and mitigation strategies in the form of security case patterns. Such patterns can then be shared among the security community and other stakeholder communities and continually built upon, refined, and improved. A growing repository of security case patterns is a huge possibility for a variety of domains and operational contexts that not only would provide greater opportunities for reuse and standardization of assurance arguments, but also could allow the security community to associate an historical record of security performance and return on investment with particular security case patterns.

In the last measure to our security case, we would see that some of the sub-goals got themselves a straight forward evident or solution that satisfy the final objective in supporting of the main goal, while in some sub-goals, they have to be expanded more widely in order to get to the heart of where the evident truly exist. Figure 4 shows how sub-goals one, two and three are decomposed until the evident that satisfy all the objectives of the security case main goal.

And finally as shown in Figure 5, we decomposed sub-goal 4 into four more sub-goals of sub-goal 4.1, sub-goal 4.2, sub-goal 4.3 and sub-goal 4.4. Note that for sub-goal 4.1, we added one property of GSN called undeveloped which is truly meant that this sub-goal 4.1 is not yet completed within the security case. Many factors could contribute to be the reason why a sub-goal is labeled undeveloped. For example, one of the factors could be that the evident or evidence provided to satisfy the objective of sub-goal 4.1 is not well defined by both the networking engineers and the security case builders.

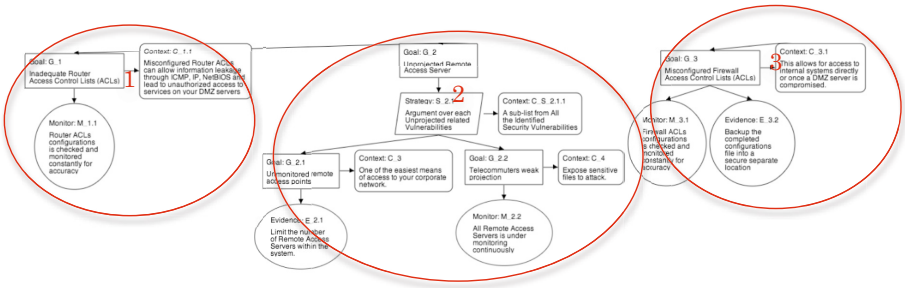

Fig. 4. This part of the Security Case shows how sub-goals 1,2 and 3 are being decomposed right down to the evident nodes

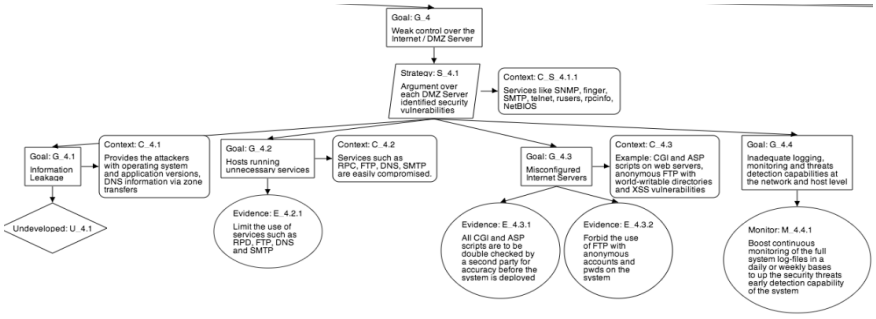

Fig. 5. This figure shows how sub-goal 4 is decomposed into 4 more sub-goals of 4.1, 4.2, 4.3 and 4.4

4 Evaluations and Conclusion

The nature of network system attacks are consequently unpredictable. Therefore relying solely on risk assessment and analysis results may not be sufficient enough to argue the security case of your system to its stakeholders. We definitely need to collect previous attacks reports (lessons learned) from systems with similar functions and design with your system. Because of this, the "**Time factor"** that took for developing our security case reduces significantly. Another benefit is **Cost**. Developing Assurance Cases at the moment can cost a lot according to a report from Carnegie Mellon University. Then **Simplicity**. Building a dependability case can be a major challenge to those who just entered the field of Assurance Cases. However, our approach has proven to be very straightforward and therefore easy to absorb by the newbies.

We proposed the combination of security engineering knowledge and experiences with D-Case to develop strong and valid assessment security cases. This is done by integrating past experience of security breach and system development, in fact turning the security assurance cases into a system development tool. In this methodology, requirements and system goals become high-level goals in the security case tree, which

is subsequently extended in a way that reflects each stage of development, later stages corresponding to lower level claims in the security case tree. Strategies for deriving sub-goals from parent goals can be based on the strategies for deriving more concrete views and models of the system under development, and should include an extensive vulnerability and risk analysis of the system view at hand. The sub-goals produced in this way at one stage of development should be regarded as requirements for the subsequent stages. For reuse, patterns of decomposition of goals/claims into sub-goals for different types of systems and security requirements might turn out to be a useful by-product.

We conducted an experiment to prove our evaluation. 4 graduate students and 3 undergraduate students of Nagoya University carried out the experiment, plus 3 network engineers whom are looking after the KISSEL system. At the end of our experiment, we find out that more than 95% of the participants agreed that our new method was very straight forward and very easy to follow in creating their own security case diagram. 90% agreed that depending on the length of the teams' discussions, the new method takes very few hours to complete once the solutions are final from the discussions. Then about the cost of the work, 99% of the participant felt that the cost of the new method if applied to all the factors shown in the experiment, should be much less than predicted before the experiment was carried out.

It's very difficult to collect previous attack reports from other similar systems. We find this quite a challenging task. On one occasion, we got a reply back from one of the system administrator we collaborated with saying that his University refuse to assist us with our request to use their network reports due to confidentiality reasons. However, some big companies like IBM for one are quite open about these kinds of reports. They even share them for free online via white papers. Also, we found out that our security cases diagram contains very sensitive information about our system. Therefore another new issue that arose after completing our work was to decide whether to save the assurance case diagrams via our system database or to store them as paper file documents.

In this paper we proposed a new approach for developing security cases. The ideas mentioned were (1) how we derived our security case diagram from complying with previous attacks reports and risk (lessons learned). (2) We shared our experiment of using our approach by applying it to our server distributed e-learning system called KISSEL. (3) Our new approach gives us the benefits in Time Efficiency, Low Cost and Simplicity. We have shown an experiment of using our method by non-experts on a web-server demo system. Currently we have preparing the feasibility experiments of our method on a few cases targeting university students and engineers. In the experiments we plan to do comparison evaluation with/without our method. Furthermore, applying our method to cyber-security issue is one of our important next goals. We would like to present the results in near future.

References

1. Ankrum, T.S., Kromholz, A.H.: Structured assurance cases: three common standards. In: Proceedings of the Ninth IEEE International Symposium on High-Assurance Systems Engineering (HASE 2005), pp. 99–108 (2005)
2. Avizienis, A., Laprie, J.-C., Randell, B.: Fundamental Concepts of Dependability. In: Proceedings of the Third Information Survivability Workshop, ISW 2000 (2000)
3. Bloomfield, R., Littlewood, B.: Multi-legged Arguments: The Impact of Diversity Upon Confidence in Dependability Arguments. In: Proceedings of 2003 International Conference on Dependable Systems and Networks, San Francisco, California. IEEE Computer Society Press (2003)
4. Jackson, D., Thomas, M., Millett, L.I. (eds.): Software for Dependable Systems: Sufficient Evidence? Committee on Certifiably Dependable Software Systems, Computer Science and Telecommunications Board, National Research Council. National Academies Press, ISBN:0-309-66738-0, http://www.nap.edu/catalog/11923.html
5. Kelly, T.P.: Arguing Safety—A Systematic Approach to Safety Case Management. DPhil Thesis, York University, Department of Computer Science Report YCST (May 1999)
6. DoD. Ministry of Defence, Defence Standard 00-56, Issue 4 (Publication Date June 01, 2007)
7. Howell, C.: Workshop on Assurance Cases: Best Practices, Possible Obstacles, and Future Opportunities. In: DSN 2004 (2004)
8. http://www.adelard.com/web/hnav/ASCE/choosingasce/cae.html
9. Bishop, P., Bloomfield, R.: A Methodology for Safety Case Development. In: Proc. of the 6th Safety-critical Systems Symposium, Birmingham, UK (February 1998)
10. Toulmin, S.: The Use of Argument. Cambridge University Press (1958)
11. Besnard, P., Hunter, A.: Elements of Argumentation. The MIT Press (2008)
12. Leveson, N.: The Use of Safety Cases in Certification and Requlation. ESD Working Paper Series. MIT, Boston (2011)
13. Kelly, T., Weaver, R.: The Goal Structuring Notation – a safety argument notation. In: Proc. of DSN 2004, Workshop on Assurance Cases (2004)
14. Jackson, D., Thomas, M., Milett, L.: Software for Dependable Systems: Sufficient evidence? National Academic Press (2007)
15. D-Case Editor (2011), http://www.dependable-os.net/tech/D-CaseEditor/
16. Matsuno, Y., Takamura, H., Ishikawa, Y.: A Dependability Case Editor with Pattern Library. In: Proc. IEEE HASE, pp. 170–171 (2010)
17. Despotou, G.: Managing the Evolution of Dependability Cases for Systems of Systems. PhD Thesis, YCST-2007-16, High Integrity Research Group, Department of Computer Science, University of York, United Kindgom (2007)
18. Weinstock, C.B., Goodenough, J.B., Hudak, J.J.: Dependability Cases. Technical Note CMU/SEI-2004-TN-016, SEI, Carnegie Mellon University (2004)