On the Impact of Generative Policies on Security Metrics*

Dinesh Verma[1] and Elisa Bertino[2] and Geeth de Mel [3] and John Melrose [4]

Abstract— **Policy based Security Management in an accepted practice in the industry, and required to simplify the administrative overhead associated with security management in complex systems. However, the growing dynamicity, complexity and scale of modern systems makes it difficult to write the security policies manually. Using AI, we can generate policies automatically. Security policies generated automatically can reduce the manual burden introduced in defining policies, but their impact on the overall security of a system is unclear. In this paper, we discuss the security metrics that can be associated with a system using generative policies, and provide a simple model to determine the conditions under which generating security policies will be beneficial to improve the security of the system. We also show that for some types of security metrics, a system using generative policies can be considered as equivalent to a system using manually defined policies, and the security metrics of the generative policy based system can be mapped to the security metrics of the manual system and vice-versa.**

I. INTRODUCTION

Policy based management [1] is an approach for managing computer systems and networks using higher level constructs called policies. A policy specifies the requirements of system management in terms of high level directives. Each directive would express some constraints that the system ought to comply with. A policy is usually expressed as a set of conditions and associated actions to be taken when the conditions are met, although there are other variations such as using events which trigger the conditions and actions, or defining goals that the system should meet. Policy based management has been used successfully to simplify the network administration in many domains, such as computer communications networks [2], semantic web-services [3], storage management [4], computer security [5], etc. Within the specific area of security management, policy based management has been used to simplify the specification of network firewall configuration [6], access control [7], content distribution [8] among others.

*This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

[1]D. Verma is with IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA dverma at us.ibm.com

[2]E. Bertino is with the Department of Computer Science, Purdue University, West Lafayette, IN 47907, USA bertino at purdue.edu

[3]G. de Mel is with the IBM Research UK, Hartree Center, Warrington, UK geeth.demel at uk.ibm.com

[4]J. Melrose is with the UK Ministry of Defense, UK jmelrose at dstl.gov.uk

While policy based management has proven its value for simplifying the task of systems management, the growth in the complexity, scale and dynamicity of systems has made the task of writing the policies more difficult over time. These conditions are found in many environments, e.g. mobile ad-hoc networks in military coalitions [9], virtualized cloud and data center environments [10], solutions using moving target defense [11], etc. In many situations, it is hard to find people who have the requisite expertise and skills to write the security policies correctly. As a result, the concept of generative policies, in which the system can generate its own policies instead of relying on human-defined policies has been proposed [12]. The generation of policies is applicable in any domain where policies need to be specified, but manual definition of policies is difficult or cumbersome. wit policies, reduce the chances of human error, and reduce the time it takes to define the applicable security policies. However, it does have a drawback. The automated generation of policies adds to the complexity of the system, which makes it more vulnerable to attacks.

The concept of automatically generating security policies in a system can lead to challenges in the Validation and Verification (V & V) of a system that needs to be deployed. In this paper, we show that a system using generative policies can be mapped to an equivalent system using manually defined policies, with a one-to-one correspondence between the security metrics of the two systems. By mapping such a system to an equivalent manual system, we can apply the V & V techniques for systems using manual policies to systems using dynamically machine generated policies.

We begin this paper with a brief review of policy based management and the approach for generative policies for security systems. We then review the state of the art in identifying and defining security metrics for information systems security. We propose an approach for associating security metrics with a system using security policies, and discuss the impact on those security metrics when policy based management is coupled with generative policies, leading to a general rule for identifying conditions under which security policy generation would be beneficial. Finally, we show how a system with generative policies can be transformed (for the purposes of security metrics calculation) into a system which uses manually defined policies, and compute the equivalent human mistake rate and equivalent breach probability for such policies.

II. REVIEW OF POLICY BASED MANAGEMENT

The task of a management system is to deal with the various situations that may arise during system operation. In

order to deal with those situations, the management system provides a configuration to the system, and handles any alerts that the system may generate during its operation. While the traditional system operation would expect some human administrator to deal with the situations which lead to an alert being generation, a policy based management allows the human administrator to specify policies as part of the system configuration. These policies specify the actions that the system should take when the situations leading to an alert generation is required, thereby decreasing the number of alerts and the cognitive burden on the human administrator. In the specific context of security management, policies would define access control for any computer or information resource, authorization requirements for accessing the network, instructions on dropping or allowing specific types of network traffic into a network, restrictions on the type of protocols that must be followed when communicating on the network, or allowing/disallowing some types of devices to exist on the network.

Using policy based management approach, a human administrator defines the policies, usually in an intuitive format, e.g. an English sentence or a selection from a graphical user interface. A policy refinement process would convert these to a set of policies that a machine can understand, which are provided to a policy decision point(PDP). The policy decision point can send those policies over to a policy enforcement point(PEP), which uses the policies to define the configuration, or handle the alerts as they happen. The policy decision point may choose to translate the policies into a different format for different types of enforcement points.

Usually, the policy refinement step will be done within a system management module, e.g. at a network operations center. The PEP is a part of the device or service being managed, while the PDP could be a part of the system management module, be a part of the device/service being managed, or be an independent service. Different implementations of policy based management would implement different bindings of the PDP and PEP in their environments.

When policies are being generated automatically, the architecture for management is much simpler on the management system side. In these cases, the human administrator sends some human guidance over to the device or service being managed. The PEP, the PDP and the PRF are all included in the managed device/service, along with a new module, the policy generator. The policy generator is responsible for creating policies that the device/service uses. It can generate those policies in a human interpretable format so that the policies being generated can be analyzed and updated manually, if needed.

Different types of human guidance result in different approaches for generating policies. The specific type of guidance would depend on the domain to which policies are being applied, and an overview of different approaches for generating access control policies is described in [13].

In the generative approach, the human guidance consists of two pieces of information, a definition of context and a generation grammar. The generation grammar may be either an Answer Set Grammar [14], or a template to generate policies, or any other mechanism to create the policies. The context would usually consist of an interaction graph, which defines which types of other services or devices a specific device/service could expect to see in the environment. The grammar specifies a policy generation language that is defined over the attributes of the devices/services found in the environment.

When the device or service receives human guidance, it looks for situations where the context changes, and new policies needs to be generated. A typical instance for security management will be when the set of devices and services in the system a change, e.g. a device/service enters or leaves the system, or changes its role within the system. In those cases, the device/service uses the grammar to generate the relevant policies.

III. Review of Security Metrics

The definition of metrics that can quantify different aspects of the security of a computer system is an open problem in the field of computer security. A security metric should be measurable, quantifiable, and provides an estimate of an aspect of the system security, e.g. number of security tests it passes, or the risk of a potential breach. The benefits of defining such a metric is well recognized [15], [16], but there are also concerns that the goal of a good security metric may be unattainable in practice [17].

The problem is further exacerbated by the fact that security is highly dependent on the specifics of the system. The security metrics associated with a cloud computer service [18], those associated with a smartphone operating system like Android [19], those associated with firewalls in communication networks [20] and those associated with an electronic Health application [21] are going to be fundamentally different from each other.

Notwithstanding the challenges associated with defining security metrics, different approaches for defining security metrics have been proposed in the literature. Several good surveys of security metrics in different contexts have been done, which try to put a systematic framework on defining security metrics for systems using different sets of criteria. In this section, we look at the types of security metrics that have been proposed, as well as the approaches for determining those security metrics.

A. Types of Security Metrics

There have been several types of security metrics used in different systems, and different survey papers have tried to classify them into different types.

A survey of systems security metrics which classifies them into different categories based on interactions between an attacker and a defense mechanism has been proposed by Pendleton et. al [22]. They classify all security metrics into four categories: vulnerability metrics, defense metrics, attack metrics, and situation metrics. Vulnerability metrics measure the implicit vulnerabilities in the system, such as programming errors or design flaw; defense metrics measure

the strength of security mechanisms put around the system, such as the type of firewalls, honeypots and access control mechanisms, attack metrics measure the strength of the type of attacks that are launched on the system, and situation metrics measure the environmental issues around a system such as the amount of investment in the security and the evolution of security vulnerabilities.

An alternative classification of security metrics is provided by Ramos et. al [23] who classify security metrics based on the target type (what is the metric intended for – e.g. a process, system, or enterprise), objective (what the security metrics is intended for – compliance with a policy, economic trade-off analysis, or assessing the security strength of a system), how it is constructed (using an abstract model or by running experiments to measure it) and how the security metrics will be measured (is it automated, is it subjective or objective, is it based on quantitative metrics or qualitative metrics).

Savola [24] provides a broader and simpler classification of security metrics dividing them into three categories, organizational, technical and operational. Organizational metrics describe attributes of organizational programs and processes, technical metrics describe security requirements on code, designs and specifications, and operational metrics describe attributes of operational systems.

Another survey [25] proposes a classification based on security goals (confidentiality,integrity, availability, authentication), area of system being measured (i.e., management, operations, technical), intention of measurement (preventing flawas, detecting security flaws, recovery etc), or targeted audience for the metrics (technical people, business leaders, external authority etc.)

Other surveys include a taxonomy of security flaws in computer programs [26], security metrics to evaluate Intrusion Detection Systems [27], and security metrics of software process life cycle [28].

As evident by the large number of surveys, there are a very large number of security metrics, which can be classified in many different ways. This makes the task of using a commonly accepted security metric in a general way very difficult.

B. Measuring Security Metrics

Any metric is not useful if it can not be measured. On the basis of the surveys described in section III-A, the approaches can be divided into three broad categories: checklist based approaches, experimental evaluation, and model based assessment.

The checklist based approaches create security checklists which are used to go through different items relevant to the security of the system. These checklists can then be converted into a security score. The most well-known of these checklists is the Common Vulnerability Scoring System [29] defined by NIST, which ranks the vulnerability of the system into a score between 0 and 1. However, many other types of security checklists have been developed within different organizations and companies, and are used widely within the industry.

The checklist based approach has the benefit that it allows human input, incorporates qualitative metrics, as well as social and environmental factors that may be hard to incorporate otherwise. Due to its ease of use, it remains the most prevalent method by which enterprise security metrics are evaluated. An alternative to checklists is to run validation tests which can estimate a quantitative security metric. The experimentation approach is applicable only to those security metrics for which a suitable experiment can be devised. As an example, penetration testing [30] can be used to determine the number of vulnerable open ports on a server, and measuring the number of vulnerable servers in the enterprise provides an assessment of its vulnerability. Experimentation techniques have been used to assess security in enterprise networks [31], financial services [32], automotive [33], [34], etc. In many of these, experimentation is used to supplement the checklist based approach, and provide the input for specific items in the checklists.

Both checklists and experimental validation are used in practice, but they do not provide any assurances about the security metrics of a system. They are empirical approaches and can not make general statements on the security risk associated within a system. Within the academic community, approaches have been proposed to create theoretical models of risks and security vulnerabilities, which would allow for a more provable approach for assessing security. Some examples of these are an approach to define theoretical models for information security [35], risk assessment for information [36], adversary modeling [37], and analyzing time-dependent risks associated with security [38]. The theoretical models can provide generalized statements about the security of the system, but the assumptions made within the models may often not match the complexity of the real-world systems.

One practical use of model based security metrics is to understand the cost benefit analysis of different security alternatives [39]. The risk associated with any particular approach for security is evaluated, and compared to the cost of implementing that security solution. This allows for the choice of a security approach which performs the most benefit (least risk) for a given cost.

IV. Security Metrics with Generative Policies

As apparent from the discussions in III, there are many different types of security metrics, which are highly dependent on the system under consideration, and are evaluated using a mixture of pragmatic approaches like checklists and experimentation, along with theoretical models of security risks. The definition of the security risk associated with a system depends on the nature of the system. Policy based security management is a general technique that is applicable to many different systems, and different aspects of security of these systems. Therefore, it is difficult to come up with a specific definition of security metrics that are applicable to all systems using policy based security management.

Instead of trying to define specific metrics, we chose to model the impact of generating system policies for a system using the cost-benefit analysis approach. This will allow us to determine the conditions under which generating policies automatically would be better (have more benefit) than having a human define policies manually.

We assume that an original system is designed to have a human write the security policies. There may be several security metrics associated with this system, which would be different depending on the type of system, as well as the type of policies that are written to the system. With the cost-benefit approach, the system using manual policies can be characterized with the following attributes:

TABLE I
ATTRIBUTES OF SYSTEM WITH MANUALLY DEFINED POLICIES

Attribute	Explanation
S	The value that the system delivers when it is operational
R	The cost associated with any security breach
p	The probability of any security breach
t	The time taken to define the manual policies
q	The probability of human writing wrong policies

Two of the attributes shown in Table I can be considered as security metrics, namely the probability of system breach and costs of the breach. In practice, there may be many different types of system breaches, with different probabilities and risk values associated with each type of system breach. Nevertheless, we can get a good estimate of the cost-benefit analysis by computing the averages over such distribution. In the above formulation, S and R need to be measured in comparable units, and are measured over some period of time. The probability of breach needs to be estimated over the same period of time, and we assume that this time-period is much larger than t, the time it takes to define policies.

The value of p is associated with a specific value of the security metric, and can be viewed as a proxy measurement of the metric itself. We will make the conservative assumption that there will be a breach if a wrong policy is defined. In that case, the probability of a breach happening either due to human mistake or to the system vulnerabilities will be $1-(1-p)(1-q)$, or $p+q-pq$. Since both p and q will be fairly small, we can approximate this as $p+q$. The net benefit delivered by the system is $S-(p+q)R$, which is the long term value offered by the system reduced by the expected risk due to occurrence of a breach. Note that the time taken for the manual definition of policies does not play a role in it, since it is assumed to be negligible compared to the period when the costs are computed.

We would now explore how the system benefit changes when dynamic policies are used. For the dynamic generation of policies, we would need the following attributes to characterize the system. Note that S,R and p are the same attributes that would remain unchanged from the system that used a manual policy definition.

Policies are generated every time the context of the system changes. if we used the manual approach for policy definition every time the context changes, then the system would be operational for $x-t$ time units out of the total period of x. This makes the system operational for part of the period, namely $(x-t)/x$ or equivalently as $(1-t/x)$. Therefore the value delivered by the system will be $(1-t/x)S-(p+q)R$. This reflects the fact that when the system is being configured, it does not deliver the value that it could.

TABLE II
ATTRIBUTES OF SYSTEM WITH GENERATIVE POLICIES

Attribute	Explanation
S	The value that the system delivers when it is operational
R	The cost associated with any security breach
p	The probability of a security breach for the system
x	The average time between context changes
e	The probability of machine generating wrong policy

When policies are being generated automatically by the system, they can be generated faster than a human. Thus, the value delivered by the system with generative policies would be $(S-(p+e)R)$.

We can now compare the conditions under which generation of policies will be better than the manual definition of them. It is obvious that if $t > x$, manual definition of policies will be infeasible. Therefore, we are considering the case where $t \leq x$. Generation of policies is better for security of a system if:

$$(S-(p+e)R) > (1-t/x)S-(p+q)R$$

which after some arithmetic manipulations can be shown to be equivalent to:

$$(t/x) > (e-q)R/S$$

Since t/x is between 0 and 1, it follows that the above condition is always satisfied with $e < q$. Furthermore, as the amount of time taken for manual policies becomes longer, the generation of policies becomes a better approach for system design.

These findings can be summarized in the following three observations:

- When machine generated policies are less likely to contain errors than human defined policies, it is better to use a generative approach.
- When manual definition of policies take longer, the generative approach becomes more attractive; and
- If system has a high degree of dynamicity, then generative policies are the only viable approach for policy based security management.

A. Equivalence Expressions with Generative Policies

We can define an equivalence between a system that is using generative policies and a system that uses manually defined policies. In this section, we consider two types of equivalences – the equivalent mistake rate of a generative policy system, and the equivalent breach probability of a generative policy system.

To define equivalence, we consider two systems, both performing the same task, but one of the systems uses manual policies while the other system uses generative policies. In the manual system, there is a given probability of breach, and a probability of a human making a mistake. If both systems provide the same overall cost-benefit trade-off, we would like to find out the attributes that characterize the manual system system given the attributes of the generative policy system. Note that in both systems, the attributes of S, and R, t and x would remain unchanged as they are dependent on the function the system is designed to deliver and the environment in which the system operates. Therefore, the goal is to find out the expression for the human system attributes of p and q which would correspond to the same benefit as that of the generative policy system.

The equivalent human mistake rate of a generative policy system will be the mistake rate a human administrator writing policies can make while delivering the same benefit to the system. In order to compute the equivalent human mistake rate, we would need to assume that the probability of breach remains the same in both the systems.

With a manual definition of policies, the benefit delivered by the system is $(1 - t/x)S - (p + q)R$. With a generative policy approach, the benefit delivered by the system is $(S - (p + e)R)$.

We can rewrite the benefit using the generative policies as the following: $(1 - t/x)S - (p + q_a)R$ where

$$q_a = (e - (t/x)(S/R)$$

Since negative error rates are meaningless in the context of any real system, we can define the equivalent human mistake rate q_equiv as

$$q_equiv = max\{0, (e - (t/x)(S/R)\}$$

An alternative equivalent metric would be to translate the effect of policy generation into an impact on the breach probability of the system. Since the net benefit delivered by a manual depends on both the breach probability and the human mistake rate, we will assume that the human being is a perfect administrator and makes no mistake. In other words, the q associated with the system using manually defined policies is zero.

In this case, we can rewrite the net benefit of the system based on generative policies to be $(1 - t/x)S - p_equivR$ where

$$p_equiv = p(1 - (tS/xRp - e)$$

p_equiv is the equivalent breach probability in the human system. The effect of using generative policies is to reduce the preach probability by a factor of $(tS/xRp - e)$

V. Conclusion and Future Work

In this paper we have focused on policy-based management systems for which security is critical. We have formulated a model by which one can compare the impact of manually authored policies vs automatically generated policies, such as policies generated according to the novel generative policy paradigm [12]. By establishing an equivalence among systems that generate policies automatically and systems that use manual policy definitions, we can apply existing processes and approaches defined for manually defined policy based systems to be applicable to systems created using the new paradigm.

Our future work includes comparing the performance of humans and machines in generating policies. One problem with humans generating policies is that there could be a lot of variability among different humans, whereas a machine would be more consistent. A study focusing on such a comparison has been carried out for the task of image annotation [40]. An interesting result from the study is that the three best human annotators are better that the best machine algorithm, but the difference is low. However the best machine algorithm is much better than the worst human annotators, and the difference is very large.

One conclusion from this study is that using machine algorithms would be better in general, unless one has very critical image annotation tasks and the human annotators are very good at their task. However another study, focusing on image recognition, showed that algorithms and humans have about the same accuracy for very clean images, but when the images are distorted, the humans are better [41]. It is clear that more experiments are needed especially to take into account different contextual conditions. Also whereas a large body of work has focused on tasks related to annotating/classifying image data, no equivalent work exists in the area of policy specifications, especially in cases in which policies have to generated under time constraints.

An interesting direction for our future work is to carry out a study to compare humans and machines when generating policies. One possibility would be to give policy decision examples (plus context information) to a human and to some machine learning algorithms (such as statistical machine learning algorithms or symbolic learning algorithms [42]), ask both to write the policies and then compare the results. The starting point for the experiment would be a policy specification expressed in natural language, perhaps in some controlled form as today we have powerful machine learning techniques for analyzing natural language. It is important to point out, though, that depending on the type of natural language documents different machine learning techniques may have to be used. See for example the approach applied by Jero et al. [43] in order to extract formal protocol specifications from RCF documents for different communication protocols, e.g. GRE, IPv6, IP, TCP, DCCP and SCTP. Therefore as part of our future work it will be also interesting to determine which machine learning algorithms are more suitable to extract formal specifications of policies from natural language policy documents.

References

[1] J. Strassner, *Policy-based network management: solutions for the next generation*. Elsevier, 2003.

[2] D. C. Verma, "Simplifying network administration using policy-based management," *IEEE network*, vol. 16, no. 2, pp. 20–26, 2002.

[3] A. Uszok, J. M. Bradshaw, M. Johnson, R. Jeffers, A. Tate, J. Dalton, and S. Aitken, "Kaos policy management for semantic web services," *IEEE Intelligent Systems*, vol. 19, no. 4, pp. 32–41, 2004.

[4] M. Devarakonda, D. Chess, I. Whalley, A. Segal, P. Goyal, A. Sachedina, K. Romanufa, E. Lassettre, W. Tetzlaff, and B. Arnold, "Policy-based autonomic storage allocation," in *International Workshop on Distributed Systems: Operations and Management*. Springer, 2003, pp. 143–154.

[5] M. Sloman and E. Lupu, "Security and management policy specification," *IEEE network*, vol. 16, no. 2, pp. 10–19, 2002.

[6] S. Hinrichs, "Policy-based management: Bridging the gap," in *Computer Security Applications Conference, 1999.(ACSAC'99) Proceedings. 15th Annual*. IEEE, 1999, pp. 209–218.

[7] N. Li, Q. Wang, W. Qardaji, E. Bertino, P. Rao, J. Lobo, and D. Lin, "Access control policy combining: theory meets practice," in *Proceedings of the 14th ACM symposium on Access control models and technologies*. ACM, 2009, pp. 135–144.

[8] N. Shang, M. Nabeel, F. Paci, and E. Bertino, "A privacy-preserving approach to policy-based content dissemination," in *Data Engineering (ICDE), 2010 IEEE 26th International Conference on*. IEEE, 2010, pp. 944–955.

[9] P. Michiardi and R. Molva, "Analysis of coalition formation and cooperation strategies in mobile ad hoc networks," *Ad Hoc Networks*, vol. 3, no. 2, pp. 193–219, 2005.

[10] R. Moreno-Vozmediano, R. S. Montero, and I. M. Llorente, "Iaas cloud architecture: From virtualized datacenters to federated cloud infrastructures," *Computer*, vol. 45, no. 12, pp. 65–72, 2012.

[11] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang, *Moving target defense: creating asymmetric uncertainty for cyber threats*. Springer Science & Business Media, 2011, vol. 54.

[12] D. Verma, S. Calo, S. Chakraborty, E. Bertino, C. Williams, J. Tucker, and B. Rivera, "Generative policy model for autonomic management," in *2017 IEEE SmartWorld, Distributed Analytics InfraStructure and Algorithms for Multi-Organization Federations Workshop*. IEEE, 2017.

[13] S. Calo, D. Verma, S. Chakraborty, E. Bertino, E. Lupu, and G. Cirincione, "Self-generation of access control policies," in *Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies*. ACM, 2018, pp. 39–47.

[14] S. Calo, I. Manotas, D. Verma, E. Bertino, M. Law, and A. Russo, "Agenp: An asgrammar-based generative policy framework," *Proceedings of Policies for Autonomic Data Governance*, 2018.

[15] S. Stolfo, S. M. Bellovin, and D. Evans, "Measuring security," *IEEE Security & Privacy*, vol. 9, no. 3, pp. 60–65, 2011.

[16] W. H. Sanders, "Quantitative security metrics: Unattainable holy grail or a vital breakthrough within our reach?" *IEEE Security & Privacy*, vol. 12, no. 2, pp. 67–69, 2014.

[17] S. M. Bellovin, "On the brittleness of software and the infeasibility of security metrics," *IEEE Security & Privacy*, vol. 4, no. 4, pp. 96–96, 2006.

[18] J. Luna, H. Ghani, D. Germanus, and N. Suri, "A security metrics framework for the cloud," in *Security and Cryptography (SECRYPT), 2011 Proceedings of the International Conference on*. IEEE, 2011, pp. 245–250.

[19] D. R. Thomas, A. R. Beresford, and A. Rice, "Security metrics for the android ecosystem," in *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*. ACM, 2015, pp. 87–98.

[20] S. Al-Haj and E. Al-Shaer, "Measuring firewall security," in *Configuration Analytics and Automation (SAFECONFIG), 2011 4th Symposium on*. IEEE, 2011, pp. 1–4.

[21] R. M. Savola and H. Abie, "Metrics-driven security objective decomposition for an e-health application with adaptive security management," in *Proceedings of the International Workshop on Adaptive Security*. ACM, 2013, p. 6.

[22] M. Pendleton, R. Garcia-Lebron, J.-H. Cho, and S. Xu, "A survey on systems security metrics," *ACM Computing Surveys (CSUR)*, vol. 49, no. 4, p. 62, 2017.

[23] A. Ramos, M. Lazar, R. Holanda Filho, and J. J. Rodrigues, "Model-based quantitative network security metrics: A survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2704–2734, 2017.

[24] R. M. Savola, "Quality of security metrics and measurements," *Computers & Security*, vol. 37, pp. 78–90, 2013.

[25] C. Villarrubia, E. Fernández-Medina, and M. Piattini, "Towards a classification of security metrics." in *WOSIS*, 2004, pp. 342–350.

[26] C. E. Landwehr, A. R. Bull, J. P. McDermott, and W. S. Choi, "A taxonomy of computer program security flaws," *ACM Computing Surveys (CSUR)*, vol. 26, no. 3, pp. 211–254, 1994.

[27] A. Milenkoski, M. Vieira, S. Kounev, A. Avritzer, and B. D. Payne, "Evaluating computer intrusion detection systems: A survey of common practices," *ACM Computing Surveys (CSUR)*, vol. 48, no. 1, p. 12, 2015.

[28] P. Morrison, D. Moye, R. Pandita, and L. Williams, "Mapping the field of software life cycle security metrics," *Information and Software Technology*, 2018.

[29] K. Scarfone and P. Mell, "An analysis of cvss version 2 vulnerability scoring," in *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*. IEEE Computer Society, 2009, pp. 516–525.

[30] B. Arkin, S. Stender, and G. McGraw, "Software penetration testing," *IEEE Security & Privacy*, vol. 3, no. 1, pp. 84–87, 2005.

[31] F. El-Hassan, A. Matrawy, N. Seddigh, and B. Nandy, "Experimental evaluation of network security through a hierarchical quantitative metrics model." in *Communication, Network, and Information Security*, 2006, pp. 156–164.

[32] M. Gupta, A. R. Chaturvedi, S. Mehta, and L. Valeri, "The experimental analysis of information security management issues for online financial services," in *Proceedings of the twenty first international conference on Information systems*. Association for Information Systems, 2000, pp. 667–675.

[33] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno *et al.*, "Comprehensive experimental analyses of automotive attack surfaces." in *USENIX Security Symposium*. San Francisco, 2011, pp. 77–92.

[34] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham *et al.*, "Experimental security analysis of a modern automobile," in *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010, pp. 447–462.

[35] A. J. A. Wang, "Information security models and metrics," in *Proceedings of the 43rd annual Southeast regional conference-Volume 2*. ACM, 2005, pp. 178–184.

[36] L. Sun, R. P. Srivastava, and T. J. Mock, "An information systems security risk assessment model under the dempster-shafer theory of belief functions," *Journal of Management Information Systems*, vol. 22, no. 4, pp. 109–142, 2006.

[37] E. LeMay, M. D. Ford, K. Keefe, W. H. Sanders, and C. Muehrcke, "Model-based security metrics using adversary view security evaluation (advise)," in *Quantitative evaluation of systems (QEST), 2011 eighth international conference on*. IEEE, 2011, pp. 191–200.

[38] F. Arnold, H. Hermanns, R. Pulungan, and M. Stoelinga, "Time-dependent analysis of attacks," in *International Conference on Principles of Security and Trust*. Springer, 2014, pp. 285–305.

[39] S. A. Butler, "Security attribute evaluation method: a cost-benefit approach," in *Proceedings of the 24th international conference on Software engineering*. ACM, 2002, pp. 232–240.

[40] R. Ewerth, M. Springstein, L. An Phan-Vogtmann, and J. Schtze, "Are machines better than humans in image tagging? - a user study adds to the puzzle," in *Advances in Information Retrieval, Proceedings of the 39th European Conference on IR Research, ECIR 2017, Aberdeen, UK, April 8-13, 2017*. Springer, 2017, pp. 186–198.

[41] S. F. Dodge and L. Karam, "A study and comparison of human and deep learning recognition performance under visual distortions," in *26th International Conference on Computer Communication and Networks, ICCCN 2017, Vancouver, BC, Canada, July 31 - Aug. 3, 2017*. IEEE, 2017, pp. 1–7.

[42] E. Bertino, A. Russo, M. Law, S. Calo, I. Manotas, D. Verma, A. A. Jabal, D. Cunnington, G. de Mel, G. White, J. Lobo, J. Ingham, and G. Cirincione, "Generative policies for coalition systems a symbolic learning framework," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 1–7.

[43] S. Jero, M. L. Pacheco, D. Goldwasser, and C. Nita-Rotaru, "Leveraging textual specifications for grammar-based fuzzing of network protocols," *arXiv preprint arXiv:1810.04755*, 2018.