Dynamic and adaptive policy models for coalition operations

Dinesh Verma[a], Saraphin Calo[a], Supriyo Chakraborty[a], Elisa Bertino[b], Chris Williams[c], Jeremy Tucker[c], Brian Rivera[d], and Geeth de Mel[e]

[a]IBM Research, Yorktown Heights, NY, USA
[b]Purdue University, West Lafayette, IN, USA
[c]UK Dstl, Porton Down, Wiltshire SP4 0JQ, UK
[d]Army Research Labs, Adelphi, MD, USA
[e]IBM Research (UK), Warrington WA4 4AD, UK

ABSTRACT

It is envisioned that the success of future military operations depends on the better integration—organizationally and operationally—among allies, coalition members, inter-agency partners, and so forth. However, this leads to a challenging and complex environment where the heterogeneity and dynamism in the operating environment intertwines with the evolving situational factors that affect the decision-making life cycle of the war fighter. Therefore, the users in such environments need secure, accessible, and resilient information infrastructures where policy-based mechanisms adopt the behaviours of the systems to meet end user goals. By specifying and enforcing a policy based model and framework for operations and security which accommodates heterogeneous coalitions, high levels of agility can be enabled to allow rapid assembly and restructuring of system and information resources. However, current prevalent policy models (e.g., rule based event-condition-action model and its variants) are not sufficient to deal with the highly dynamic and plausibly non-deterministic nature of these environments. Therefore, to address the above challenges, in this paper, we present a new approach for policies which enables managed systems to take more autonomic decisions regarding their operations.

Keywords: Coalition Operations, Policy based Management, Generative Policies, Autonomic Systems, Software Defined Architectures

1. INTRODUCTION

Policy based management has proven useful to simplify the management and operations of complex distributed environments in many different domains, enabling management advances in many domains such as industrial systems,[1] computer network management,[2] enterprise access control,[3] distributed systems management,[4] security and identity management,[5] storage area networks,[6] military sensor networks,[7] self-managing cells,[8] and many other areas. Due to their wide applicability, several policy management frameworks have been proposed, among which the IETF/DMTF framework[2] has been widely adopted. Several policy specification languages and information models, both general purposes and for specific domains, have been proposed such as the IETF/DMTF Policy Core Information ModelPCIM,[9] Ponder,[10,11] XACML,[12] KaoS,[13] Rei,[14] OWL-POLAR,[15] IBM Autonomic Computing Policy Language- ACPL[16] and many others as discussed in survey papers[17] and used to create autonomic management systems.[18]

Despite the tremendous benefits and usefulness of policy based management in many different domains, currently deployed policy management systems have effectively followed an approach in which a management system provides relatively constrained instructions to a managed device, without leaving much room for autonomic behavior for the device. While some aspects of autonomic behavior are shown when managed devices and management systems are treated as a single unit, current policy based systems do not permit significant autonomic behavior at the level of the managed device. In this paper, we present an architecture which can let the managed device obtain more autonomic behavior, and illustrate its usefulness by means of an example.

Further author information: (Send correspondence to D. Verma)
D. Verma: E-mail: dverma@us.ibm.com
G. de Mel: E-mail: geeth.demel@uk.ibm.com

The rest of the paper is organized as follows: in Section 2, we define the high-level goals and objectives for autonomic behavior. We illustrate the autonomic behavior goal with a common security management situation in Section 3. We then propose an approach which allows devices to generate their own policies and demonstrates its applicability w.r.t. to illustrative scenario in Section 4. In Section 5, we conclude the document by summarizing our contributions and sketching some directions for future research.

2. GOALS FOR AUTONOMIC MANAGEMNT

A traditional management system can be modeled by the abstracted physical layout shown in Figure 1, which shows a set of managed devices connected to a management system by a communication network. The managed devices are given configuration information by the management system. Operational information (e.g. alerts or logs) is provided by the managed device to the management system. The management system would have a set of processing algorithms, e.g. policies and rules to deal with the set of alerts and logs that are processed. In order to handle the alert or log, the system may decide to send a reconfiguration command to the managed device. The syntax and semantics of alerts, logs and configurations are determined by the domain of management, such as fault management, security management, performance management etc. When the system is not able to deal with the alerts or logs by itself, the human operator intervenes to deal with the situation, diagnoses the underlying cause, and then reconfigures the system to react to the unexpected situation.

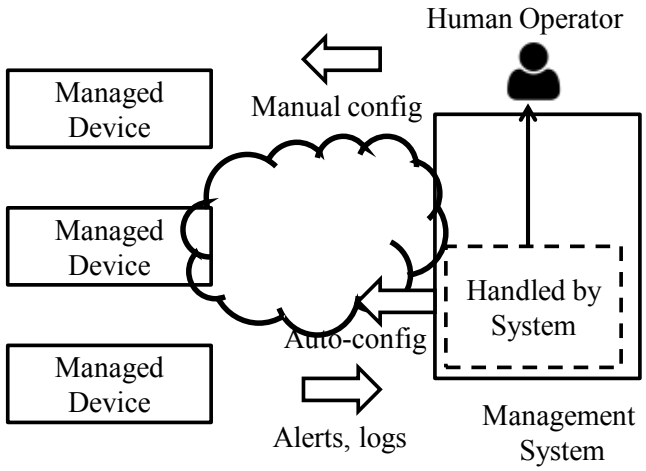

Figure 1. Simplified model of systems management

Several technologies have been developed with the field of network and systems management to reduce the amount of alerts that needs to be handled by the human operator. The existing state of the art for policy management provides one such approach. The typical architecture for policy management is shown in Figure 2. The human operator is provided an easy to specify objective for the managed device to use (human view of policy). The management system translates them into a machine view of policy through the process of refinement or transformations.[2] The machine view of policy is provided to entities known as policy decision points (PDP). Each managed device embeds a policy enforcement point which can consult a PDP and use the information provided by the PDP to reconfigure itself. When a situation requiring an alert arises, or any entry in a system log needs to be processed, the PEP converts it into a standard request that can go to the PDP. The PDP makes a decision on the appropriate situation and informs the PEP on how to handle the situation. The PEP can then change the system configuration to react to the environment.

The policy management system, PDP and PEP make up the logical constructs which can then be mapped into the physical layout of managed devices and management system is a couple of different ways, as shown in Figure 3. The PEP is usually embedded as a component of the managed device, while the policy refinement process is embedded as a part of the management system. The PDP could be embedded either as a part of the

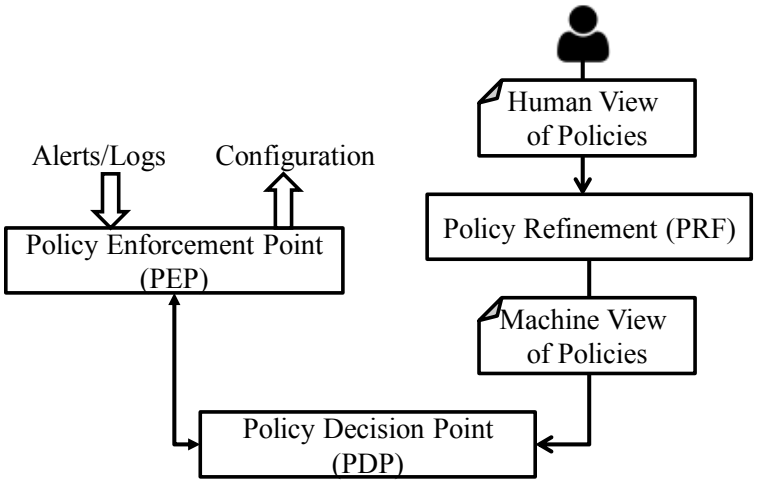

Figure 2. Policy based Management Architecture

management system, or it could be embedded within the managed device. When the PDP is embedded within the managed device, the device exhibits limited autonomic behavior, by operating according to the policies defined by the management system. .

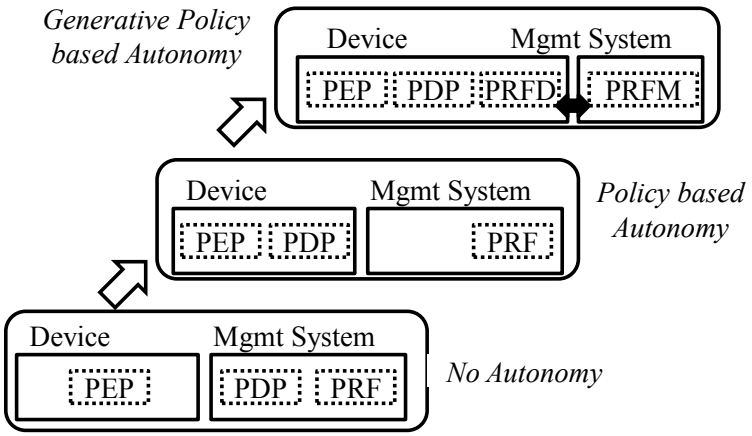

Figure 3. Generative Policy based Management

As mentioned earlier, the current model for policy based management has the semantics where the machine view of policies are determined by the refinement process, and the managed device with the embedded PDP has no ability to define its own policies. Our goal is to enable a situation where the managed device can derive its own policies, which means embedding a part of the policy refinement process within the device itself. Embedding policy refinement within the device has the most significance when modular design principles are used, enabling the same refinement process within the device to be used across multiple management domains. Within the context of coalition operations, where a device belonging to one coalition partner may need to interact with the management systems of another partner, policy refinement within the device increases the autonomy and responsiveness of the managed device.

In Figure 3, the refinement component embedded in the device is shown as PRFD, while the refinement component within the management system is shown as PRFM. PRFD uses refinement algorithms that can work without requiring a human input, since the PRFM in the management system can be improved significantly

with interactive human input. The semantics of the interaction PRFM and PRFD, shown by the black arrow in Figure 3, is a key component that enables this stage of autonomy. Our goal for autonomic management is to enable devices to reach the Generative Policy based Autonomy stage shown at the top of Figure 3.

We call our approach a generative model for policies, since policies are generated by the device itself. Generative policies allow a flexibility and freedom of action for the managed device to determine their own behavior characteristics. The current model of policies used in the state of the art is very prescriptive in that it defines the actions to be taken under various conditions, and does not permit much freedom to the managed device.

The broad approach for generative policy based management can be illustrated in an intuitive manner using an example from a common security management situation in data centers and cloud sites. This example is discussed in the next section.

3. ILLUSTRATIVE PROBLEM SCENARIO

Maintaining secure access to documents is a common problem in any data center/cloud site. We can consider a situation where we have a set of documents, some of which are considered sensitive, and others that are not. A set of users have access to sensitive documents, which can be accessed either using a web-based application or via a secure shell based system. In order to prevent attacks on the systems, a packet filtering firewall is provided to safeguard access to both the web-based system and the secure shell based system. The configuration is shown in Figure 4. The scenario is a common one encountered in almost any site requiring access control to a set of documents.

The current practice in securing such systems is for a human administrator to manually configure filtering and access control policies for the firewall, web-server, secure shell server and the document server. In a typical scenario, the ports on a firewall need to be configured to allow access to the web server. However, if we provide a higher degree of autonomy to the web-server, e.g. assume it is using a moving target defense,[19] and changes its port for the web-server at some regular periods, the configuration of the packet filter firewall needs to be repeated manually every-time such a change happens. The management tool will have to create the appropriate policies for each of the devices.

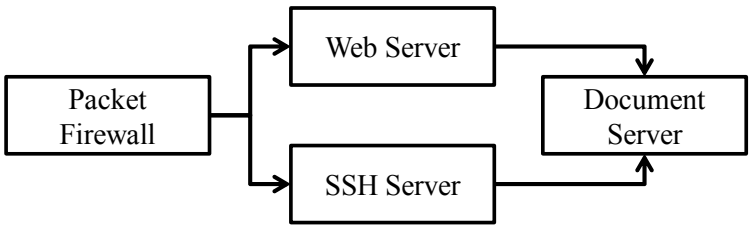

Figure 4. Illustrative Problem Scenario

Instead of a manual reconfiguration of access controls after each change, it would be highly desirable if the human operator (the human view of policies) simply specified the access requirements on the documents. Based on those access control requirements, the packet firewall, the web server, the SSH server, and the document server would each derive their policies to comply with the desirable policies on their own. If the web-server switched its port as part of the moving target defense, the packet firewall would automatically adjust its filtering policies accordingly. In these cases, the policy refinement process happens within the devices themselves, and does not require any human intervention until access control on specific documents are changed.

Note that a similar policy refinement will be desirable if the devices were to be managed not for access control security, but also for fault management, configuration management or performance management. When a fault happens in one of the components, we would desire all the components impacted by the fault to generate new policies to deal with that fault. Similarly, for performance management, the different components should be able to determine their policies for managing performance, e.g. by increasing the number of parallel instances, etc.

In all aspects of autonomic behavior, we would like the devices and components to define their policies on their own.

In the next section, we present the general solution for generative policies.

4. GENERATIVE POLICY ARCHITECTURE

In the generative policy architecture, the PDP is embedded within the managed device, and it gets its policies from the PRFD module that is also embedded within the managed device. The PRFM is responsible for sending the overall coordination guidelines to the PRFD. The architecture overall is as shown in Figure 5.

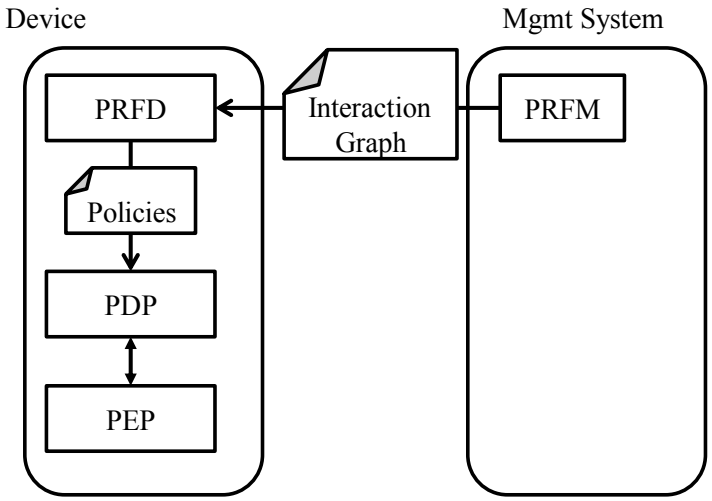

Figure 5. Generative Policy Architecture

The PRFM provides two types of information to each PRFD. One is the representation of an interaction graph. An interaction graph is an abstract description of the various entities within the environment that the PRFD needs to interact with. The interaction graph is defined as a relationship between entities in different roles in the system, not as an exhaustive listing of all the different devices in the system. The role of each PRFD in the interaction graph is defined by the PRFM. The PRFM also associates a resource model, i.e. a set of attributes with each link in the interaction graph.

Each PRFD receives the interaction graph from the management system, and uses it to generate the policies for its local PDP. Each PDP discovers other PDPs in the system it should connect with according to the interaction graph, and finds out the attributes of the connecting links. In order to generate these policies, the PRFD uses a policy grammar that defines the syntax of the policies that can be generated locally. The grammar is used to generate expressions that are defined over an alphabet that includes the local attributes of the managed devices and the attributes of the links that the managed device has with other devices as defined in the interaction graph. As the managed device discovers other devices in the roles identified in the interaction graph, new policies conforming to the grammar and including the attributes of the newly discovered links are generated.

The grammar can be provided to the PRFD by an independent management system. As an example, a U.S. operator may provide the grammar to its drones, while in the context of a coalition operation, the management system PRFM may belong to another coalition partner. For each distinct domain to which the policy architecture is applied, the set of valid roles, the attributes which define the mapping of the nodes in the interaction graph to the terminals in the policy grammar, and the specific policy grammar are specified.

Let us consider the application of this architecture to the illustrative problem scenario. In this scenario, three roles for different entities can be identified, a network protection role(N), a protocol protection role(P), and a document protection role(D). In the scenario instance shown in Figure 3, two devices are in the role of protocol

protection, while only one device is in the other two roles. The global interaction graph is shown in Figure 6 with the letters N, P and D indicating the different roles, and the attributes required for each link. The specific interaction graph that will be provided to the PRFD in each of the devices is also shown, with the role of the device marked in black circles in each device specific interaction graph.

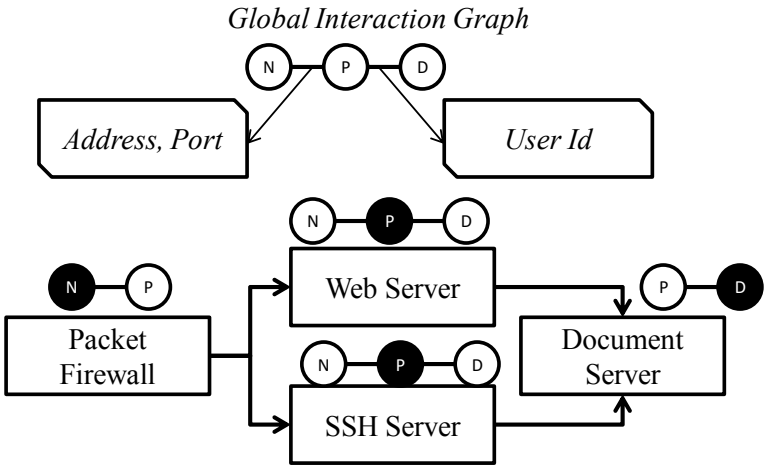

Figure 6. Interaction Graphs for the Illustrative Problem

When the PRFD for each of the devices receives the interaction graph, it searches for the other nodes that are associated with adjacent roles in the interaction graph. The discovery module finds out the other devices in those roles, and finds out the attributes identified by the devices in those roles in the interaction graph. Then, the PRFD uses the grammar available with it to generate its own set of policies to be used for its PDP.

In the illustrative scenario, once the address and port for the protocol protection role are identified, the packet firewall can generate the appropriate packet filtering policies for the firewall. Similarly, the web-server can receive the set of user-ids that are authorized to access the document server, and install its protection policies to only allow those user-ids to access the document server. Note that the document server will provide several user-ids to the protocol protection servers, while each of the web-servers and SSH server will only provide a single network address and port for itself to the packet firewall. Eventually, the document server would need to have its document protection policies. In this case, these policies come directly from the PRFM.

The grammar for policy generation for the packet filter firewall will generate the 5-tuple (source and destination addresses, source and destination ports, and the protocol) that determine whether or not a packet is allowed within the network. The address and port numbers of each protocol protection device (the web-server and SSH server in this scenario) will provide this information to the firewall which can use it to create its network packet filtering policies. When the attributes for the web-server and SSH-server change, e.g. due to a scheme such as moving target defense[19] changing those attributes, the discovery process will again be triggered and the policies in the packet filtering firewalls will be regenerated.

While the previous example was for access control, the same scheme can be generalized for other aspects for the same system setup, e.g. increasing the number of instances of each node in Figure 4 dynamically for performance management, or for a node to dynamically restart the other node for resiliency management. When nodes are dynamically instantiated using technologies like Network function virtualization (NFV) or Software-Defined Networking (SDN), roles can also be dynamically assigned to different instances running in the system.

Moreover, we believe this architecture can enable software-defined coalition (SDC) creation and enablement. SDC is an approach to provide support for a community of interest (CoI) where assets and information products from multiple parties are brought together to provide a holistic and actionable information space for the decision maker. SDC combines and extends concepts from software defined networking[20] to operate in the context of

coalition operations. We are currently working towards instantiating the generative policy architecture on such SDC setting, and will present the finding in a future related conference.

5. CONCLUSIONS

In this paper, we have examined the problem of enabling autonomic behavior in managed devices, and enabling them to generate policies for their operations on their own. The policies are generated for each domain according to an interaction graph which is provided by a management system, and which defines the scope of activities for the devices. We have then provided a running commentary on how this architecture can be instantiated w.r.t. to the illustrative scenario of this document. The early applications show that this architecture can provide a significant advancement over the current state of the art in policy based management.

While the approach and the architecture provide a promising approach for autonomic behavior, several new directions need to be explored to make the architecture truly useful. Algorithms for checking the validity and effectiveness of generated policies need to be developed to the framework. We need to apply the framework to several other system management scenarios, and validate that this will address autonomy challenges in those scenarios.

ACKNOWLEDGMENTS

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 copy-right notation hereon.

REFERENCES

[1] Illner, S., Pohl, A., Krumm, H., Luck, I., Manka, D., and Stewing, F.-J., "Policy-based self-management of industrial service systems," in [*Industrial Informatics, 2006 IEEE International Conference on*], 492–497, IEEE (2006).

[2] Verma, D. C., "Simplifying network administration using policy-based management," *IEEE network* **16**(2), 20–26 (2002).

[3] Bhatti, R., Ghafoor, A., Bertino, E., and Joshi, J. B., "X-gtrbac: an xml-based policy specification framework and architecture for enterprise-wide access control," *ACM Transactions on Information and System Security (TISSEC)* **8**(2), 187–227 (2005).

[4] Maullo, M. J. and Calo, S. B., "Policy management: An architecture and approach," in [*Systems Management, 1993., Proceedings of the IEEE First International Workshop on*], 13–26, IEEE (1993).

[5] Ganek, A., Nadalin, A., Nagaratnam, N., and Verma, D., "An autonomic approach for managing security and identity management policies in enterprises," *Journal of High Speed Networks* **15**(3), 291–300 (2006).

[6] Agrawal, D., Giles, J., Lee, K.-W., Voruganti, K., and Filali-Adib, K., "Policy-based validation of san configuration," in [*Policies for Distributed Systems and Networks, 2004. POLICY 2004. Proceedings. Fifth IEEE International Workshop on*], 77–86, IEEE (2004).

[7] Verma, D. C., Cirincione, G., and Pham, T., "Policy enabled interconnection of sensor networks using a message queue infrastructure," in [*SPIE Defense and Security Symposium*], 698108–698108, International Society for Optics and Photonics (2008).

[8] Keoh, S. L., Dulay, N., Lupu, E., Twidle, K., Schaeffer-Filho, A. E., Sloman, M., Heeps, S., Strowes, S., and Sventek, J., "Self-managed cell: A middleware for managing body-sensor networks," in [*Mobile and Ubiquitous Systems: Networking & Services, 2007. MobiQuitous 2007. Fourth Annual International Conference on*], 1–5, IEEE (2007).

[9] Moore, B., Ellesson, E., Strassner, J., and Westerinen, A., "Policy core information model–version 1 specification," tech. rep. (2001).

[10] Damianou, N., Dulay, N., Lupu, E., and Sloman, M., "The ponder policy specification language," in [*Policies for Distributed Systems and Networks*], 18–38, Springer (2001).

[11] Twidle, K., Dulay, N., Lupu, E., and Sloman, M., "Ponder2: A policy system for autonomous pervasive environments," in [*Autonomic and Autonomous Systems, 2009. ICAS'09. Fifth International Conference on*], 330–335, IEEE (2009).

[12] Godik, S. and Moses, T., "Oasis extensible access control markup language (xacml)," *OASIS Committee Secification cs-xacml-specification-1.0* (2002).

[13] Uszok, A., Bradshaw, J., Jeffers, R., Suri, N., Hayes, P., Breedy, M., Bunch, L., Johnson, M., Kulkarni, S., and Lott, J., "Kaos policy and domain services: Toward a description-logic approach to policy representation, deconfliction, and enforcement," in [*Policies for Distributed Systems and Networks, 2003. Proceedings. POLICY 2003. IEEE 4th International Workshop on*], 93–96, IEEE (2003).

[14] Kagal, L., Finin, T., and Joshi, A., "A policy language for a pervasive computing environment," in [*Policies for Distributed Systems and Networks, 2003. Proceedings. POLICY 2003. IEEE 4th International Workshop on*], 63–74, IEEE (2003).

[15] Şensoy, M., Norman, T., Vasconcelos, W., and Sycara, K., "Owl-polar: Semantic policies for agent reasoning," *The Semantic Web–ISWC 2010* , 679–695 (2010).

[16] Agrawal, D., Lee, K.-W., and Lobo, J., "Policy-based management of networked computing systems," *IEEE Communications Magazine* **43**(10), 69–75 (2005).

[17] Han, W. and Lei, C., "A survey on policy languages in network and security management," *Computer Networks* **56**(1), 477–489 (2012).

[18] Huebscher, M. C. and McCann, J. A., "A survey of autonomic computingdegrees, models, and applications," *ACM Computing Surveys (CSUR)* **40**(3), 7 (2008).

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

[20] Nunes, B. A. A., Mendonca, M., Nguyen, X.-N., Obraczka, K., and Turletti, T., "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys & Tutorials* **16**(3), 1617–1634 (2014).