Toward Cybersecurity Personalization in Smart Homes

Daniele Bringhenti, Fulvio Valenza, and Cataldo Basile | Politecnico di Torino

Security personalization has become a critical need for smart homes in recent years. The current approaches cannot fully satisfy this requirement of user-centered security. We propose a user-friendly approach for the automatic configuration of home security solutions through policy-based management, minimizing human interventions, and improving security usability.

Today, individuals are exposed to more threats from the outside world as more risks can undermine cybersecurity in domestic environments. In particular, this larger exposition surface has impacted both the privacy and the wellness of individuals. These considerations are supported by recent investigations about the effects of cyberattacks. According to the most recent Data Breach Investigations Report by Verizon (https://enterprise.verizon.com/resources/reports/2021-data-breach-investigations-report.pdf), 2020 has seen social engineering become the most common way to make breaches in networks, and 8% of all of the breaches involved humans as unintentional factors.

One of the causes of these worrisome statistics may be that inexpert people have started to use complex devices connected to the Internet. In the last decade, smart homes have become ecosystems where a massive variety of devices coexist, from more traditional personal computers or laptops to Internet of Things (IoT) or domotics nodes, such as locks, sensors, or wearables. The privacy of sensitive data stored on these devices is threatened by malicious attacks today more so than in the past.[1] Another possible cause is the heterogeneity of users. Their need for remote connectivity from home has been progressively increasing since last year when the outbreak of the COVID-19 pandemic instigated new ways to live for people of any age—children, middle-aged persons, and the elderly.

For example, children may be the victims of episodes of cyberbullying when using social networks, or they may watch videos with inappropriate content. Older people might have their credit card numbers stolen, inadvertently activate paid subscriptions, or fall victim to false alarms and fictitious threats. Coping with multiple types of cyberattacks simultaneously is a complex task, but it is essential so that today's houses will not only be smart places where citizens can find more opportunities than in the past but also safe places where they can enjoy those opportunities.

To date, several heterogeneous off-the-shelf solutions already exist to challenge these issues: home gateways, routers, and application programming interfaces (APIs) of devices, such as smart TVs and wearables. Examples are the TP-Link Router AX1800 and TCL Smart Roku TV 5058435. Despite their low price (for instance, the

Digital Object Identifier 10.1109/MSEC.2021.3117471
Date of current version: 26 October 2021

router costs less than US$100 and can be easily installed in home networks without the intervention of a technician), they offer a large number of security services. These services may be locally integrated into the product or remotely accessible through the connection with access points or servers of the Internet Service Provider (ISP), such as Vodafone Secure Net. Besides, they are typically complimentary and could offer all-around protection in smart homes. For example, a parental control allows one to filter content that is unsuitable for children, a mail spammer searches for malicious emails to prevent social engineering attacks, and a packet filtering firewall offers a thinner granularity for access control. Some of these products may also integrate more complex services, such as identifying intrusions based on attack patterns. In any case, it seems that the best solution for protecting our own smart homes is already available in the nearest supermarket, and there is no problem that is still worth researching in this environment.

However, people commonly look for and buy off-the-shelf products, such as home gateways or ISP services, mainly to improve the connection quality and solve networking problems related to bandwidth and latency. Even though these solutions offer all of the previously mentioned security functionalities, people typically do not use them.[2] On the one hand, they struggle to understand and endorse them. Security already involves complex concepts by definition, and, unfortunately, having to use technical jargon does not help people approach this context. It also happens that the same service is named with different terms in as many solutions, and this does nothing but increase the general confusion.

On the other hand, the personalization of these security services is minimal for end users. We have made a nonexhaustive analysis of off-the-shelf products that marketed user friendliness or simplicity and noted that they also expose security functionalities. However, the behavior of these functionalities is hardcoded by the manufacturers and can only be turned on or off. For example, parental controls integrated into home gateways can filter content depending on a specific time of the day, but the end users are not always enabled to change this setting. Facing this limitation, people often decide not to use the services as the general settings may not match their specific needs. Consequently, solutions that are currently available for enforcing security in home networks are not entirely user friendly.

One may argue that this decision in the security design is taken to avoid a situation in which end users who are inexperienced in security matters involuntarily open the door to external cyberattacks. However, the problem should be studied from a different point of view. The challenge should not be to understand how to protect home networks by limiting end users but learning how to protect these users even when allowing them to customize the security requirements for their smart homes. The direction to pursue should be a user-centered design (UCD), where usability goals and user characteristics are given extensive attention during the design of a solution. Specifically, in this context, user-centered security[3] should be the ultimate objective for home networks: security should be designed so that individuals can understand the main principles behind its services and personalize them in a user-friendly manner.

However, the achievement of this goal is not trivial in the context of smart homes. Manual approaches are not feasible because of the high problem complexity caused by the heterogeneity of available solutions and user needs. Therefore, alternatives must be investigated to improve the quality of life and personalization for the users of cybersecurity.

In light of these considerations, we have searched for a solution following the "keep it simple, stupid" principle, which suggests counterbalancing the native intricacy of security problems with the simplicity of their solutions. This direction strictly follows the main ideas behind UCD. To this end, we propose leveraging and adapting a well-known paradigm, already used for business networks, for security enforcement and personalization in smart home networks: policy-based management (PBM).[4] In particular, the main contributions of this article are the following: 1) the definition of a user-friendly security language that human beings can easily use to define their security requirements in smart homes and 2) the creation of a complete workflow for automatic cybersecurity personalization that can establish the configuration of security products with minimum external interventions.

Related Work

In the literature, user security languages have been proposed to fill the gap between the complexity of security configurations and the simplicity requirement of human users. Originally, standard languages, like the Policy Core Information Model proposed in Request for Comments 3060 (https://datatracker.ietf.org/doc/html/rfc3060) and the eXtensible Access Control Markup Language proposed by OASIS3 (http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html), have been presented in the literature. On the basis of these languages, in more recent years, other access control policy languages have been studied, such as Ponder,[5] SecPAL,[6] and other usage control languages.[7] However, all of these languages have three main limitations compared to the language that we are proposing.

1. They cannot simply be used by people with no expertise in computer information; they mainly target security-savvy people.

2. They do not cover the whole area of network security but exclusively deal with access control.
3. The related research is generally focused on corporate computer networks without focusing on the specificity of home networks.

Similar considerations apply to the other main contribution of our article: the application of PBM for automating security configuration. This technique lays its foundations on intent-based networking,[8] an orchestration principle that aims at minimizing human interventions. In fact, the configuration of network security is automatically generated from high-level intents, also called *policies*, that describe how the network should behave without specifying all of the details. On the one hand, access control and firewalls have been deeply studied in the literature, for example, in the work of Bartal[9] and Bringhenti.[10] On the other hand, little research has been spent on other security controls, such as virtual private network gateways and intrusion detection systems.[11] In all cases, policy refinement is not applied to security solutions that are tailored to smart homes.

In light of this literature review, our work is the first combined proposal of a user-friendly security language and automatic security configuration workflow that is specifically designed for smart home networks. At the same time, it also aims to overcome the limitations of studies carried out in other areas, for example, the complexity of usage for security languages and the limitation of the configuration computation to specific kinds of security controls.

The Approach

We proposed an approach for automatic cybersecurity personalization in smart homes based on PBM, which is a management paradigm that aims to abstract the enforcement of the rules governing the behavior of systems from their complexity. In this approach, individuals are not required to configure systems manually; they only need to define a set of requirements in the form of security policies. Then, these policies are employed by an automated process to establish the system configuration, which becomes transparent to the end users. PBM is a solution that has proven to be successful in solving the same problem (coping with the high complexity of security configuration and customization) in networks of bigger dimensions, such as those of business companies or universities.[10,11] PBM has become necessary in that context because even expert security administrators cannot manually manage huge networks. In smart homes, even though the networks are small, the people accessing them are less experienced in security; hence, PBM might come in handy.

However, this approach cannot be applied to home networks without a proper adaptation. As previously anticipated, it is crucial to consider the heterogeneity of both security solutions and user categories. Children use home networks to study and play with their friends, parents to work and keep themselves informed, grandparents to remotely interact with relatives who live afar. Each person has multiple and different needs, and their harmonization should be an integral part of the solution. A single security level for a whole smart home is not acceptable, but multiple levels should coexist and not interfere with each other.

The PBM-based approach that we propose to pursue network security personalization in smart homes is illustrated in Figure 1. Three main pillars compose the workflow through which this approach is feasible:

- *User security language*: A user-friendly language allows human beings ease of access to the personalization workflow so that expertise in the security fields is not required.
- *Automated generation of configuration*: The user-defined policies are refined into an enriched representation that contains all of the information for their enforcement on the security products and services.
- *Policy harmonization*: Inconsistencies among the policies derived from user requirements must be identified and solved to avoid incorrect security behavior.

In the remainder of this article, we will illustrate each pillar, focusing on its objectives and working mechanisms.

User Security Language

Individuals accessing home networks are commonly nontechnically savvy users. They should not learn the needed skills to understand how security services must be configured fully. Instead, they should be allowed to request protection for themselves and their loved ones by expressing security intents, for example, in natural language, in a way that abstracts the complexity of the security configuration. Then, these security intents should be transposed to a more structured language, called a *user security language*, which represents a link between humans and the automated processes that must actually configure the security. The specification of the security policies is the main operation human beings are required to perform in a PBM-based approach. All of the other operations are mostly automated and typically do not involve external interventions.

A language that could match the user needs should have four main characteristics:

- *Simplicity*: The language should allow each user to define sophisticated policies intuitively, with statements

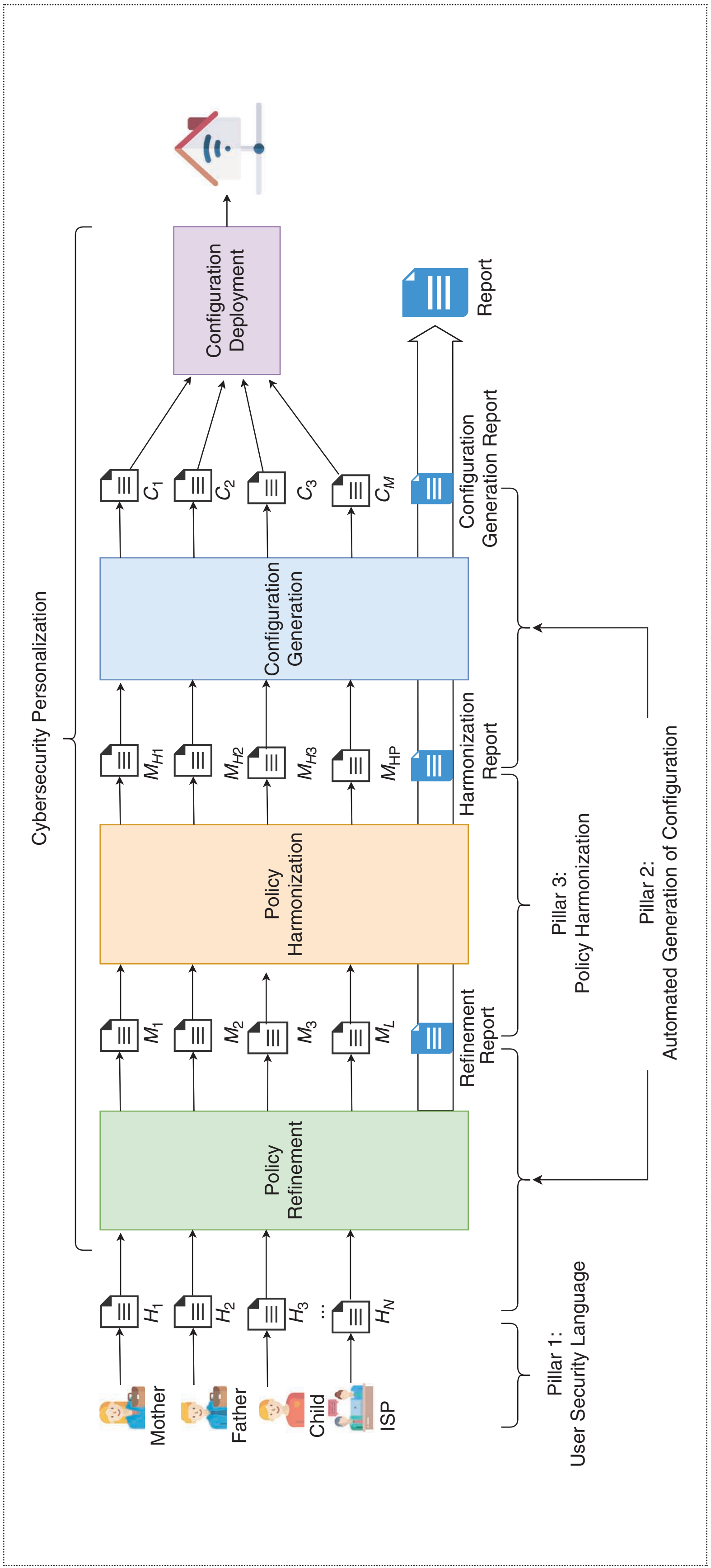

Figure 1. The workflow of automatic cybersecurity personalization. *H*: high-level security policy; *M*: medium-level security policy; *C*: configuration.

that can be expressed in a form as similar as possible to natural language.

- *Precision*: The language should accurately represent all the security intents that humans may specify so that it is never ambiguous and can always be used by a refinement process to precisely produce the configuration of a corresponding security device or service.
- *Flexibility*: The language should support both the heterogeneity of users and security solutions. On one hand, it should allow the definition of policies expressing needs for each age group (for example, from studying to working). On the other hand, it should not require one to cope with the technicalities of the specific security solutions.
- *Extensibility*: Every day, new products are manufactured, and new services are available to end users. If a PBM-based approach does not foresee their possible creation, it fails the objective to achieve user-centered security, as it is not future proofed. Therefore, the language should support extensions without impacting the structure of the syntax. Users can thus still use the same language by just learning a few new words.

In light of these considerations, we propose a user-oriented language, called a *high-level security policy language* (*HSPL*),[11] that has the necessary characteristics. The HSPL represents a tradeoff between human languages and the traditional approach to configuring products for home security, where users may only enable or disable predefined settings. Therefore, it allows one to transpose the security intents (which could also be sentences recognized by voice assistants) to structured statements required by the subsequent operation of the approach, that is, policy refinement. The structure of these statements is as follows:

$$[<subject>]\ <action><object>$$
$$[<(field_{type} = value)> \ldots <(field_{type} = value)>]$$

In this structure:

- *<subject>* represents the person for whom the enforcement of the policy is requested (such as the child or the grandparent). The subject may be implicit in case there is a single person who accesses the home network. For the same policy, multiple subjects may be specified.

- < *action* > represents the security operation that must be enforced to fulfill the policy (for instance, block, allow, protect, enable, and permit access).
- < *object* > represents the policy target on which the requested operation must be performed (such as email and Internet traffic).
- < ($\textit{field}_{\text{type}}$ = *value*) > represents an optional condition to characterize the action (for example, "time = from 9 a.m. to 5:59 p.m., GMT," "content type = social networks," and "domain = youtube.com"). Multiple conditions may be specified simultaneously to enrich the expressiveness of the policy.

A pair of examples will show how this language can be used for policy specification.

Timed Content Restriction

Alice is a 10-year-old girl who has been remotely attending school for the last several months. Her parents do not want her to access social networks during school time. Manual enforcement of this requirement would require configuring multiple services, such as time filtering and web-application firewalls. Instead, with the proposed user security language, her parents can express all of this information in a single security intent: "Alice must not access social networks during school time." Then, this intent can be easily transposed in the following HSPL statement:

< Alice >< must not access >< Internet >
[< time = from 10.00 AM to 1.59 PM CET >;
< days = from Monday to Friday >;
< content type = social networks >]

Phishing Attenuation

Bob is a 75-year-old man who has been a victim of phishing multiple times. His son, Charles, wants to limit the communications (for example, through emails and Skype) that his father may establish. Specifically, Bob should be allowed to communicate only with his son and grandchildren. Even though multiple services [such as the email system and voice over Internet Protocol (IP)] are involved, again, a single statement is enough to express this security requirement: "Bob can only interact with Charles, David, and Sophie." The corresponding HSPL statement would be the following:

< Bob >< can exclusively start >< communications >
[< source/destination = Charles, David, Sophie >;
< communication means = voice, text messages, emails >]

The HSPL has already proved to be a valid policy specification language; it has been validated in studies[12,13] where it has been respectively cast into contexts such as smart home and loT environments. Interested readers can find more details about this high-level description in past work,[11] where the formal specification and examples of the application of the security language are presented.

Automated Generation of Configuration

After the policy specification, our approach envisions an automated process that transforms the policies into an equivalent, yet enriched, formulation that captures all of the required information for the configuration of security products and services to protect smart homes. In the literature, this operation is commonly known as *policy refinement*,[14] and in the approach we propose, it works as follows.

First, the policy refinement process identifies the security functionalities required for the enforcement of the HSPL policies. A one-to-one mapping is not always possible as complex policies might require multiple functionalities to be fully enforced. For instance, referring to the section "Timed Content Restriction," two security functionalities would be identified: time filtering and web-application filtering. They may be present on the same product (such as the home gateway), or at least one of them may be only remotely available. In any case, if in the home network of Alice's parents, these functionalities cannot be accessed, this means that the policy cannot be enforced. Second, the configuration of the security products and services that have the identified functionalities is automatically generated. This operation deals with two main issues: 1) conciseness of the HSPL statements and 2) heterogeneity of the security solutions.

Indeed, the HSPL simplifies security management for users, but, at the same time, for ease of specification, it must omit details needed for the security configuration. For example, the effective IP addresses that may be used in communications or the network topology are overlooked by this high-level language. Also, because of the high heterogeneity of the solutions, each product requires setting commands characterized by languages with a different syntax, even though they are semantically similar.

To overcome the first issue, this policy refinement process must access external knowledge bases to retrieve information needed for determining the security configuration. Suppose an HSPL statement specifies that a child cannot visit social networks. In that case, there must exist a list of uniform resource locators (for example, a list that is stored in a database and can easily be modified by third-party services or national/international organizations) to be retrieved for computing the configuration of a web-application firewall. A similar consideration can be made for the IP addresses of the devices. Traditionally, a mapping between devices and currently assigned IP addresses used to be present in the home gateway, and it used to be employed for computing the rules of a packet-filtering firewall. As our vision is to abstract from the specific solutions that individuals may use in their smart homes, our approach aims to embrace a comprehensive view of all of the possible databases of IP addresses (for example, blacklists) and use them to compute the security configurations.

To overcome the second issue, the computed configurations must be expressed with a solution-independent language, that is, a language that allows the representation of security configurations in a form independent from a specific vendor's implementation. To this end, we propose a language called a *medium-level security policy language* (*MSPL*). An MSPL has a lower level of abstraction than an HSPL because an MSPL statement must provide all of the required pieces of information, including those retrieved by the mentioned external knowledge bases. At the same time, it is characterized by a generic syntax that abstracts the vendor-specific syntaxes of the different security solutions.

MSPL statements are machine readable and can be represented as JavaScript Object Notation (JSON) or XML snippets. Referring again to the section "Timed Content Restriction," Figure 2 reports an excerpt of the JSON file representing the configurations of the two security functionalities (time filtering and web-application filtering) that are identified to enforce the requested HSPL policy. As seen from this excerpt, the representation is richer than the equivalent HSPL formulation; for example, it specifies that, for each functionality, the first matching rule (FMR) is adopted as a resolution strategy, and the complete list of blocked social networks is present.

After the generation of these MSPL statements, they cannot be immediately enforced on the security solutions that users might access locally in their homes or remotely through their ISP network access points. Therefore, a final translation from the MSPL to the syntax of the specific solution is required. However, this operation only consists of a syntax change for the automatic computation of the MSPL configuration. Figure 2 also reports the low-level configuration that would be produced from the corresponding MSPL statement for Squid, a widely used reverse proxy and web-application firewall for Unix-based operating systems. The same information is represented in the two languages (the MSPL and Squid language), simply with a different format.

Then, after this translation, the output needs to be deployed on the security products and services. The ideal case would be to perform deployment through interaction with their APIs. However, currently, not all security solutions expose APIs for their configuration. Alternative methods (e.g., Secure Socket Shell channels and Message Queue Telemetry Transport) can be used to cope with this limitation, hoping for standardization of these APIs.

At that point, the process automatically produces a report to inform the people who requested the security enforcement about the outcome of their demand. The

MSPL statement:

```
{
  “subject”: “Alice”,
  “configurations”: [
    {
      “functionality”: “timeFiltering”,
      “defaultAction”: “allow”,
      “filteringRules”: [
                  {
                   “action”: “deny”,
                   “condition”: {
                          “startTime”: “10.00 AM”,
                          “endTime”: “1.59 PM”,
                          “timeZone”: “CET”,
                          “day”: “*”
                          }
                  }
                 ],
      “resolutionStrategy”: “FMR”
    },

    {
      “functionality”: “L7Filtering”,
      “defaultAction”: “allow”,
      “filteringRules”: [
           {
              “action”: “deny”,
              “condition”: {
                    “url”: “facebook.com”
                  }
           },
           {
             “action”: “deny”,
             “condition”: {
                    “url”: “twitter.com”
                  }
             },
           ...,
         ],
      “resolutionStrategy”: “FMR”
    }
  ]
}
```

Squid configuration:

```
acl studying-time time MTWHF 10:00-13:59
acl blacklisted-domains dstdomain www.facebook.com
  www.twitter.com ...
acl all src 0/0
http_access deny studying-time blacklisted-domains
http_access allow all
```

Figure 2. An example of an MSPL statement and low-level configuration. FMR: first matching rule.

best result that may be conveyed through this report is that all of the policies have been successfully enforced without changing their original specification. Alternatively, it may happen that the policies cannot be fully enforced. For example, if a policy requiring parental control cannot be enforced on a smart TV, it should be enforced on each application installed. However, it may be that some applications do not offer this security functionality. In that case, the system reports the effects of the proposed partial enforcement and lists the modified policies. Consequently, the user knows that some applications are still unsafe and may decide to uninstall them or specify a more restrictive policy (for instance, any Internet access from the smart TV is prohibited).

The proposed process for the automatic generation of security configuration has been originally validated in the context of the European Commission-funded Project SECURED (https://github.com/SECURED-FP7?language=html). The same concepts have been reused in the Anastacia Project (https://gitlab.com/anastacia-project?sort=created_asc), in which an entirely different consortium decided to adopt and customize the SECURED approach. Full details about the use cases and approach validation can be found in the Git repositories of the two projects.

Policy Harmonization

Traditionally, for each product or service, a single security policy used to be defined for all family members. Even when some devices could allow a per-user configuration, this operation was performed by a single individual (for example, a parent), thus limiting the capabilities offered by the devices. Instead, in our vision, the best approach would be to enable each user the possibility of defining his/her own policies. On the one hand, this solution simplifies the policy specification itself because the task is distributed among multiple people. On the other hand, additional actors, such as ISPs, may want to define policies, and they cannot let their customers personally enforce them in their smart home networks. Nonetheless, inconsistencies may emerge when policies written by different users need to be enforced.

Our approach contemplates the definition of personalized network security policies by envisioning an additional operation called *policy harmonization*. As the naming suggests, the objective of this task is to identify all of the policies specified for a specific user and harmonize them. This case is envisioned to consider scenarios where complete enforcement of all of the policies is not feasible because contradictions affect them. Contradictions may emerge when users have different security requirements, for example, when parents can overrule the requests of their children. They can also appear when users wrongly define contradicting policies without noticing the errors, for instance, if the parents have to define large policy sets to set up the security for them and their children and do not properly check for inconsistencies.

Policy harmonization is performed according to a reconciliation process, which defines how inconsistencies need to be customized by the experts (such as the ISPs or third-party vendors) to allow the users to solve inconsistencies transparently. Policy harmonization is performed in our network security personalization approach after the policy refinement and before generating the low-level configuration. Policies are both specific enough to permit precise identification of contradictions and abstract enough to avoid getting lost in useless details (like devices' syntax). The reconciliation process we have adopted, which is presented next, is based on the feedback of a set of smart home use cases. It is also based on the same formal models that have been successfully employed for the reconciliation of firewall policies in other studies.[15]

The proposed reconciliation starts from the MSPL statements, obtained by refinement of the HSPL policies expressed by all family members. The HSPL policies are grouped according to their subject (for example, all of the policies for Alice specified by herself, her parents, and the ISP). Then, each group of statements is analyzed to identify the inconsistencies resolved using multiple strategies, which are applied in a fixed order. The complexity of the reconciliation derives from the purpose of hiding to users the complexity of explicitly managing default actions, exceptions, and blacklisting versus white listing, and propagating them in different policy sets written by different authors. Among these strategies, we describe the first ones, which are known as the *security-first* and *family-role-first* strategies.

The *security-first* strategy implements the security-by-default principle. When two policies for the same subject are contradictory, the most restrictive statement is always enforced. The other one is modified to be enforced as far as possible, i.e., by "subtracting" the part that the other policy has overruled. This strategy prioritizes the actions that are requested by the policies. For example, "deny," "block," and "prohibit" are considered more restrictive than (and thus prevail on) "allow" and "permit." To clarify this strategy, let us consider the example shown in Figure 3, where the father has requested that his daughter Alice can access social networks every day, whereas her mother has explicitly prohibited this access type for all of the children during weekdays. Policy harmonization first groups these policies and identifies a partial overlapping. The latter policy is the most restrictive and is thus preserved in the harmonized set without changes. Instead, the former is modified to grant access to social networks to Alice only on Saturday and Sunday.

According to the *family-role-first* strategy, in case of inconsistencies among policies specified by different users, the policy requested by the person with a higher family role in the home is enforced (for example, parents can overrule their children's requests), while the other policy is refused or partially modified. This strategy has a lower priority than the *security-first* strategy. Therefore, the process of policy harmonization applies only when an anomaly still persists for similarly conservative policies, that is, for policies whose actions have equal priority in terms of security. For instance, if Alice requests to access the Internet with all her devices, whereas the father decides that Alice can only access the Internet outside of school time and without accessing illegal websites from the laptop, the father's policy is enforced.

Finally, the report produced at the end of the automatic configuration also describes the outcome of policy harmonization. Users must be informed if their policies have only been partially enforced and if other policies have overruled them. The overruling policy is also specified in the report. This information allows users to understand how different needs were conflicting so that individuals living in the same home can discuss the result of the automatic configuration and decide accordingly. However, this may not always be possible. For example, if the ISP requests the overruling policy, the hidden visibility of this policy would make the report omit it.

Today, many limitations for personalizing cybersecurity in smart homes still exist in current solutions, and individuals commonly struggle to use the security features offered by off-the-shelf products and services. Therefore, to overcome these limitations, we have suggested the possibility of employing an automated approach to simplify and customize security configurations in domestic environments, with minimum human intervention. This approach bases its foundation on PBM and provides users with a language for policy specification that represents a tradeoff between human languages and machine-like representations. An operation of policy harmonization is also envisioned to identify and solve possible anomalies in the policy specification, for example, when two family members define conflicting policies.

As future work, we are further optimizing the implementation of this approach to minimize the number of APIs required to deploy the automatically computed

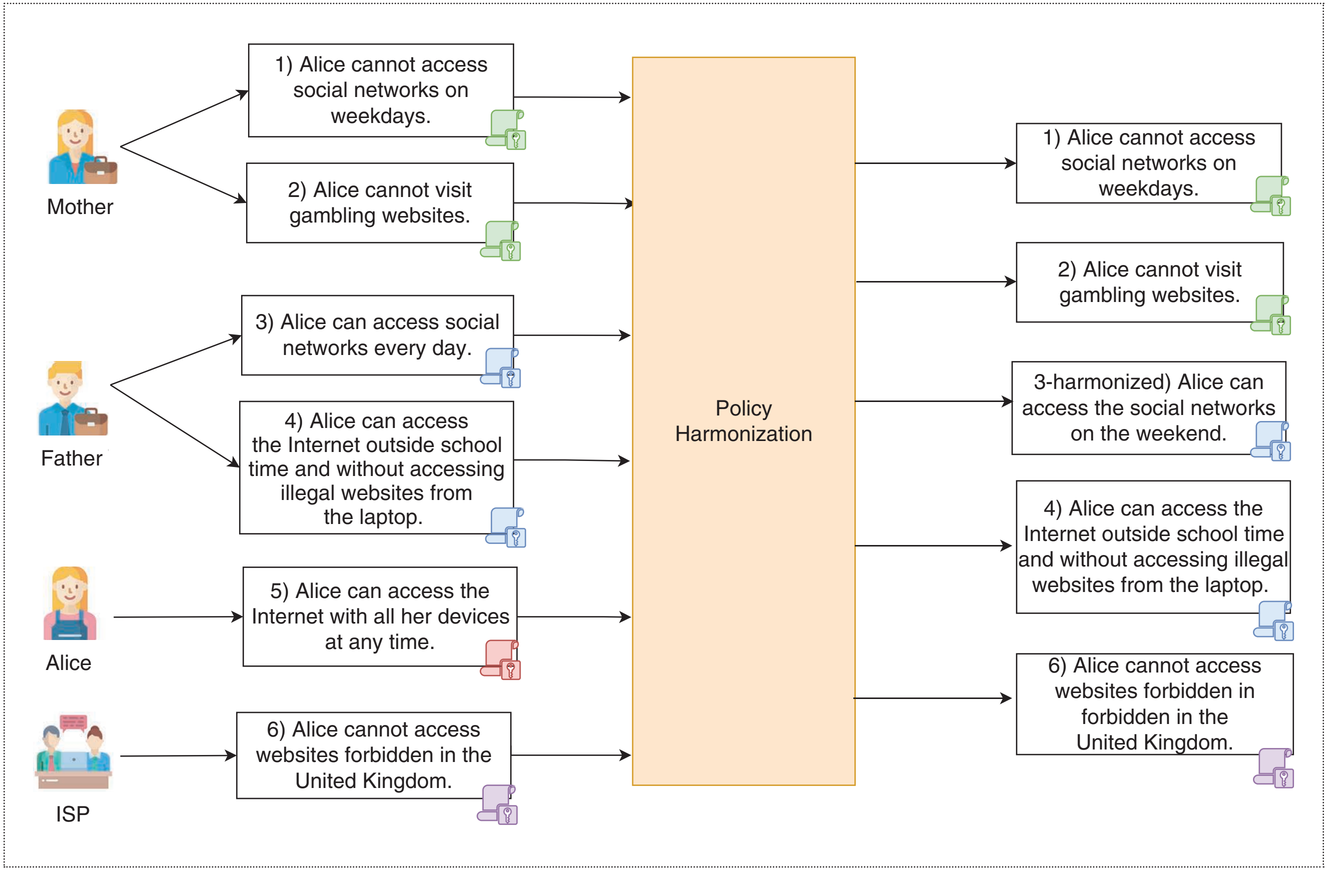
Mother
Father
Alice
ISP
1) Alice cannot access social networks on weekdays.
2) Alice cannot visit gambling websites.
3) Alice can access social networks every day.
4) Alice can access the Internet outside school time and without accessing illegal websites from the laptop.
5) Alice can access the Internet with all her devices at any time.
6) Alice cannot access websites forbidden in the United Kingdom.
Policy Harmonization
1) Alice cannot access social networks on weekdays.
2) Alice cannot visit gambling websites.
3-harmonized) Alice can access the social networks on the weekend.
4) Alice can access the Internet outside school time and without accessing illegal websites from the laptop.
6) Alice cannot access websites forbidden in the United Kingdom.

Figure 3. An example of policy harmonization.

configurations. We also plan to extend the proposed user security language and the whole approach to work in smart homes and other contexts, such as more general IoT networks or fog/edge computing environments. ■

Acknowledgment

This work has been partially supported by the European Union's Horizon 2020 Project FISHY (grant agreement 952644).

References

1. P. P. Gaikwad, J. P. Gabhane, and S. S. Golait, "A survey based on smart homes system using Internet-of-Things," in *Proc. Int. Conf. Comput. Power, Energy, Inf. Commun. (ICCPEIC)*, 2015, pp. 330–335. doi: 10.1109/ICCPEIC.2015.7259486.
2. N. Taha and L. Dahabiyeh, "College students information security awareness: A comparison between smartphones and computers," *Educ. Inf. Technol.*, vol. 26, no. 2, pp. 1721–1736, 2021. doi: 10.1007/s10639-020-10330-0.
3. M. Lacoste *et al.*, "User-centric security and dependability in the clouds-of-clouds," *IEEE Cloud Comput.*, vol. 3, no. 5, pp. 64–75, 2016. doi: 10.1109/MCC.2016.110.
4. A. A. Jabal *et al.*, "Methods and tools for policy analysis," *ACM Comput. Surveys (CSUR)*, vol. 51, no. 6, pp. 1–35, 2019. doi: 10.1145/3295749.
5. N. Damianou, N. Dulay, E. Lupu, and M. Sloman, "The ponder policy specification language," in *Proc. Int. Workshop Policies Distrib. Syst. Netw. (POLICY 2001)*, Bristol, U.K., Jan. 29–31, 2001, vol. 1995, pp. 18–38. doi: 10.1007/3-540-44569-2_2.
6. M. Y. Becker, C. Fournet, and A. D. Gordon, "SecPAL: Design and semantics of a decentralized authorization language," *J. Comput. Security*, vol. 18, no. 4, pp. 619–665, 2010. doi: 10.3233/JCS-2009-0364.
7. A. Lazouski, F. Martinelli, and P. Mori, "Usage control in computer security: A survey," *Comput. Sci. Rev.*, vol. 4, no. 2, pp. 81–99, 2010. doi: 10.1016/j.cosrev.2010.02.002.
8. E. Zeydan and Y. Turk, "Recent advances in intent-based networking: A survey," in *Proc. 91st IEEE Veh. Technol. Conf., VTC Spring 2020*, Antwerp, Belgium, May 25–28, 2020, pp. 1–5. doi: 10.1109/VTC2020-Spring48590.2020.9128422.
9. Y. Bartal, A. J. Mayer, K. Nissim, and A. Wool, "Firmato: A novel firewall management toolkit," *ACM Trans. Comput. Syst.*, vol. 22, no. 4, pp. 381–420, 2004. doi: 10.1145/1035582.1035583.
10. D. Bringhenti, G. Marchetto, R. Sisto, F. Valenza, and J. Yusupov, "Automated optimal firewall orchestration and configuration in virtualized networks," in *Proc. NOMS 2020—IEEE/IFIP Netw. Operations Manag. Symp.*, Budapest, Hungary, Apr. 20–24, 2020, pp. 1–7. doi: 10.1109/NOMS47738.2020.9110402.
11. C. Basile, F. Valenza, A. Lioy, D. R. López, and A. P. Perales, "Adding support for automatic enforcement of security policies in NFV networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 2, pp. 707–720, 2019. doi: 10.1109/TNET.2019.2895278.
12. D. Montero *et al.*, "Virtualized security at the network edge: A user-centric approach," *IEEE Commun. Mag.*, vol. 53, no. 4, pp. 176–186, 2015. doi: 10.1109/MCOM.2015.7081092.
13. S. Ziegler, A. F. Skarmeta, J. B. Bernabé, E. E. Kim, and S. Bianchi, "ANASTACIA: Advanced networked agents for security and trust assessment in CPS IoT architectures," in *Proc. Global Internet of Things Summit (GIoTS 2017)*, Geneva, Switzerland, June 6–9, 2017, 2017, pp. 1–6. doi: 10.1109/GIOTS.2017.8016285.
14. A. C. Riekstin *et al.*, "A survey of policy refinement methods as a support for sustainable networks," *IEEE Commun. Surv. Tuts.*, vol. 18, no. 1, pp. 222–235, 2016. doi: 10.1109/COMST.2015.2463811.
15. C. Basile, A. Lioy, C. Pitscheider, and S. Zhao, "A formal model of policy reconciliation," in *Proc. 23rd Euromicro Int. Conf. Parallel, Distrib. Netw.-Based Process. (PDP 2015)*, Turku, Finland, Mar. 4–6, 2015, pp. 587–594. doi: 10.1109/PDP.2015.42.

Daniele Bringhenti is currently pursuing a Ph.D. in control and computer engineering at Politecnico di Torino, Turin, 10129, Italy. His research interests include novel networking technologies, automatic orchestration, the configuration of security functions in virtualized networks, and formal network security policies. Bringhenti received an M.Sc. in computer engineering from Politecnico di Torino, Italy. He is a Student Member of IEEE. Contact him at daniele.bringhenti@polito.it.

Fulvio Valenza is currently an assistant professor with a time contract at Politecnico di Torino, Turin, 10129, Italy, where he works on orchestration and management of network security functions in next-generation networks. His research interests include network security policies. Valenza received a Ph.D. in computer engineering from Politecnico di Torino, Turin, Italy. He is a Member of IEEE. Contact him at fulvio.valenza@polito.it.

Cataldo Basile is currently an assistant professor at Politecnico di Torino, Turin, 10129, Italy. His research interests are software security, software attestation, policy-based security management, and general models for the detection, resolution, and reconciliation of security policy conflicts. Basile received a Ph.D. in computer engineering from Politecnico di Torino, Turin, Italy. He is Member of IEEE. Contact him at cataldo.basile@polito.it.