Viz^{4}NetSec: Visualizing Dynamic Network Security Configurations of Everyday Interconnected Objects in Home Networks

Noëlle Rakotondravony[1], Henrich C. Pöhls[2(✉)], Jan Pfeifer[2], and Lane Harrison[1]

[1] Worcester Polytechnic Institute, Worcester, MA, USA
[2] Passau Institute of Digital Security, University of Passau, Passau, Germany
hp@sec.uni-passau.de

Abstract. Controlling the communication of devices within a network by compartmentalization or segmentation is one of many techniques to protect and improve the overall security of networked systems. Meaningful instrumentation of network segmentation requires having an overview of the devices that participate in the network; only then users can seize control of what devices are allowed to do in terms of communication. In this work, we consider a minimized set of network security controls (i.e. allow connections, allow in-bound-only, allow out-bound-only) that can be implemented using modern routers with built-in firewall or even Software Defined Networking (SDN) capabilities. We present Viz^{4}NetSec, a node-link diagram for visualizing typical home user network scenarios. The visualization is integrated in an existing smart home control software and provides an interactive interface through which a smart home user can find, dynamically interact with, and isolate devices by setting SDN flow rules. The aspect of *dynamicity* in this paper is important as we envision that everyday users would reasonably need interactions to trigger configuration changes that directly change the network's or the device's behavior because this enabled users to configure network rules in a trial and error or 'gamified' fashion. Thus, dynamicity empowers adapting security decisions (mostly in the sense of privacy) to their ever-changing everyday digital lives. We conclude this work with an evaluation of the proposed network visualization, discussing how home network users can use the available functionalities in Viz^{4}NetSec to perform the isolation of devices within the network as the most simple security related task.

Keywords: Network Security · Smart Home · Dynamic Interaction · SDN

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-61382-1_11.

A. Moallem (Ed.): HCII 2024, LNCS 14729, pp. 164–185, 2024.
https://doi.org/10.1007/978-3-031-61382-1_11

1 Introduction

Predictions estimate the number of Smart Home Devices (SHDs) is rising and the IoT market is set to reach 483 billion US$ between 2022 and 2027 [15]. The connectivity of those SHD is between the home's Internet router to reach cloud services and to other devices within the same home. However, research shows that security deficiencies [21] and privacy problems of numerous such SHDs cause "digital harms" [6]. According to Sagar Joshi [21], an average smart home could face over 12,000 hacker attacks every week, underscoring the need for security measures in smart homes. While "privacy was not a primary consideration in users' adoption decisions [for smart speakers] but did serve as a deterrent for some non-users" [26], we assume that the inhabitants of smart homes —when educated on privacy-related topics and empowered by usable tools— would decide that at least some network communication capabilities of their devices are neither deemed really necessary nor always favored.

Empowering users is crucial. In the case of smart speakers, Lau *et al.* argued that "current smart speaker privacy controls are rarely used, as they are not well-aligned with users' needs" [26]. Examples of how this can be facilitated by physical interaction, like a physically closable lid on a webcam working as a "physical kill switch" [49], have been proposed. In this paper, we work from the following hypothesis:

> Smart home users, once enabled with usable controls,
> will use such controls to increase their privacy
> and the security-posture of their networks.

We emphasize two goals for such a usable control interface:

- First, the visualization shall be suitable for novice users that will have to cope with the ever-growing number of interconnected devices in modern households; globally the average was already at 17.1 devices per home in 2022 [48].
- Second, the interaction shall allow a dynamic control of the communication of the devices. In the prototype we simply control the devices' network connectivity, but filtering traffic can be done more granularly [57,67].

1.1 First Goal: Visualization for Simple Smart Home Tasks

This work's focus is put on visualizations of typical home user scenarios. The network's hierarchy in home deployments is usually flat, e.g. devices have a direct wireless connection to the router, and there are usually only few layers in the technical network infrastructure. To limit the possibilities of what to visualize, Viz[4]NetSec starts simply: it shows the connections between the device and the router. We also simplify the control of these connections and only make rules based on the endpoint and the direction of the communication. As a result, we have incoming and/or outgoing traffic or no communication with the smart home device. The control and visualization can be further fine-grained by adding additional network segmentation, like microsegmentation [44], and then disabling

communication links between the segment and the router. This will result in some SHDs being able to talk to others within their segment.

For Viz[4]NetSec, we leverage tree-based visualizations and node-link layouts for displaying relationships, and group entities with simple hierarchies [20,54]. In more detail, we use a force-directed graph implemented within the open-source application for smart home control, Home Assistant [16]. The representation of Viz[4]NetSec depicted in Fig. 5 shows the visual representation of all communication links of the home network. While this study primarily aims to simplify network visualization for smart home users, it recognizes the vastness of the IoT landscape. The focus for the prototype was on widely-used Wi-Fi devices, as Wi-Fi is a comfortable and widely adopted protocol [9].

1.2 Second Goal: Dynamicity of the Exercised Control

We consider the aspect of *dynamicity* to be of paramount importance as we hypothesize

- firstly, that everyday users would reasonably need interactions that trigger configuration changes immediately;
- secondly, that everyday users would be enabled to dynamically adjust the allowed actions of a device according to what communications they feel are tolerable due to changes in their situations;
- finally, that everyday users might want to explore what communication they minimally need to allow in order to achieve the required functionality.

On the one hand, *dynamicity* allows users to configure network rules in a trial and error fashion, *e.g.*, "What happens to the functionality if I disable the Internet for this smart light bulb?". On the other hand, it can empower users to adapt their security decisions (mostly in the sense of privacy) to the dynamically changing environment of their digital life, *e.g.*, "Enable the security camera to stream images to the Internet when I leave my home, not when I am at home". They also might want to adapt to situations, *e.g.* "Allow communication of a SHD with the Internet when performing a software update, while in normal daily operation the device stays unconnected to the internet." We integrated the capabilities of SDN [33] to monitor the network's connections for the Viz[4]NetSec visualization and to directly control the network's flow. Our prototype offers the functionality to isolate a device on the network using SDN flow rules when clicking on the device in the visual representation.

Isolating and controlling the communication of devices within a network is not a new technology. Among the strategies to achieve this are compartmentalization [60] or segmentation [34]. Both strategies have been successfully used in small and large networks [39]. However, configuration possibilities grow with an increased number of devices in the network, and in most situations, the task to configure them in an optimal way is in the hand of network administrators [64]. Professional system administrators could therefore benefit from tools to support such simple tasks, and home users that are not professionally trained even more

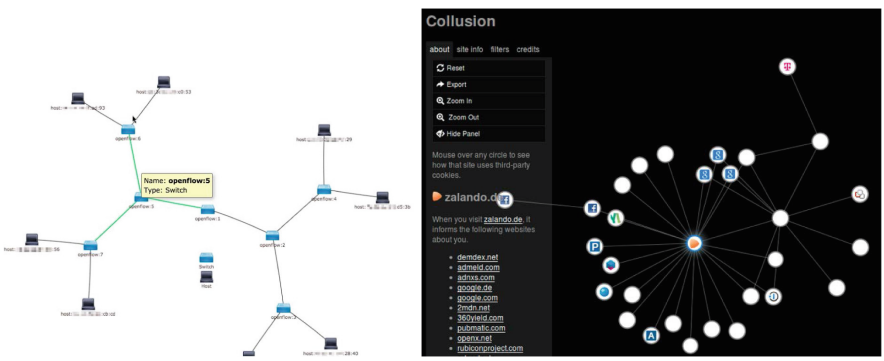
so. From this basic interaction, the tool can be enhanced with more commands, offering an extended version for more sophisticated smart home users. In this work, we start from a minimized set of network security controls (*i.e.* allow connections, allow in-bound-only, allow out-bound-only) that can be implemented using modern routers with built-in firewall or SDN capabilities.

2 Related Work

2.1 Network Security Configuration

Software-Defined Networking (SDN) is the technical basis of our (and others' e.g. [18,44]) solution to heighten the network security in smart homes, and to enforce segmentation and isolation in detail [44]. At the center of the SDN architecture lies a network controller, which comes in various forms, from open-source implementations to proprietary ones [40]. The concept of programmable networks isn't new (see Nunes *et al.* [40] for an overview) and predates today's well-known protocols like OpenFlow [41]. We opted for SDN because its flexibility also supports more complex network configurations [40] still offering reasonable speed [37] for enough dynamicity.

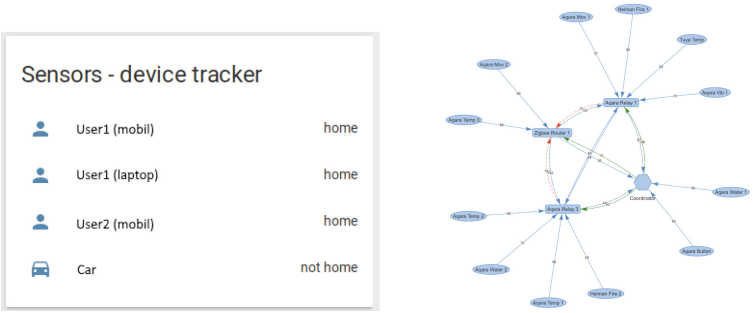
SDN controllers, such as OpenDaylight (ODL), ONOS, Cisco APIC, and Juniper's Contrail, often feature Graphical User Interfaces (GUI) that present a topological view of the network (see Fig. 1a). Node-link layouts are commonly used for security and privacy related network visualizations, e.g. the tool EtherApe [61] (see Fig. 4a) showing the communicating devices, or the browser extension Lightbeam[1] (see Fig. 1b) visualizing externally loaded web content.



(a) OpenDaylight Topology View [42]. (b) Firefox extension Lightbeam [27, 23].

Fig. 1. Visualizations of networks using graphs and node-link layouts.

[1] formerly known as Collusion.



(a) Nmap Tracker example [25]. (b) ZHA Map example [46].

Fig. 2. Visualizations within Home Assistant.

Especially for Home Assistant [16], natively integrated visualizations like the NMAP Tracker [17] (Fig. 2a) or ZHA Map [68] (Fig. 2b) have been developed. The NMAP Tracker's visualization serves more to detect if mobile IoT devices are "reachable" within the network to trigger actions, e.g. when the car is at home. ZHA Map maps devices utilizing the Zigbee Protocol [68]. However, it does not provide any dynamic interaction with the visualized network topology and is mainly used for troubleshooting.

In this work, we implement a graphical tool to simplify interacting with the many capabilities offered by an SDN. Our visualization Viz^{4}NetSec aims at representing the underlying network logically segmented and structured based on SDN configurations, in order to facilitate the dynamic configuration of the network itself by the user.

2.2 Network Security Visualization

Traditional home network systems come with interactive router dashboards [53] (*e.g.* Fig. 3-top). User studies showed that dashboards can help users implement

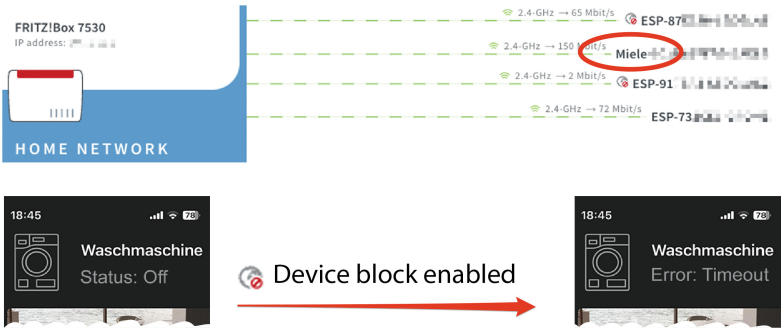
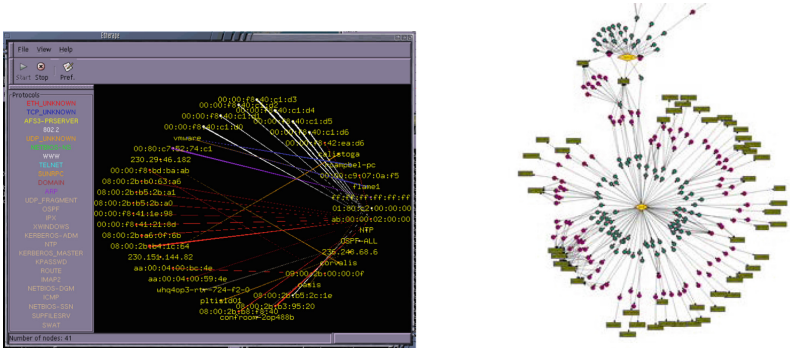


Fig. 3. Blocking Internet connectivity per device on a commercial router's UI for a connected washing machine (top) and the result in the user's app (bottom).

fundamental network configurations [59], scheduling [4], and security measures for their devices and smart appliances [11,66].

Network visualization is widely researched [38,55]. Graph-based views like the one from the open-source project EtherApe in Fig. 4a address shortcomings of tabular representations in built-in commercial interfaces [19]. Shiravi *et al.* [56] and Guimarães *et al.* [13] surveyed the state of the art visualization techniques for network security and management. They identified node-link and topological views as being prevalent representation techniques [29,30].

Additionally, modern approaches to network visualizations underline the importance of interactivity and controls. The ability to fine-tune devices' configurations through an interactive interface can help users protect their resources from misuse in a timely and responsive manner [22,45]. In both small and large networks, the ability to control or customize the visual attributes and representations of the network visualization itself can help a user make sense of the underlying network in a faster and more personalized way [35,38].



(a) EtherApe [61] (b) Worm impact by Ed Blanchfield [32]

Fig. 4. Visualizations for computer networks using node-link diagrams.

While market and off-the-shelf tools are often geared at advanced users [19], extending the table-like representations they offer can also help novice home-users benefit from a visual monitoring of their network infrastructure and configurations. Ball *et al.* show that the graphical approach to security can increase users' awareness and confidence when the visualizations maintain simplicity, usability, and enable controls [3]. Poole *et al.* argue that the graphical representation of a home network should mimic the way households conceptualize their home networks, for example, in a simple entity-link fashion [50], also suggesting the adequacy of node-link diagram proposed in other research. Other researchers propose that the design and usability of home network visualization sould fit into both the users' technical skills but also the daily routines of the households [7,62]. Thus, masking complexities to users while providing interactions to low-level details for more advanced administrators is essential to home users' experience. For instance, network segmentation is a challenging security

concept to visualize. It requires a clear overview of the network structure [14]. To address that, tree-like visualizations can be leveraged to simplify the modeling of network layouts and its hierarchy [36,54]. Using a tree diagram, the layout of an SDN-based IoT structure is simplified as a connection between all things at home and the Internet [1].

In this paper, we build on node-link diagram based approaches to implement Viz^{4}NetSec, a visualization of dynamic network security configurations for home networks. We visualize the software-defined layout of the network in simple and customizable tree-like representations. We implement interactivity that translates complex security concepts (such as segmentation) to actionable tasks that allow (non-expert) home users to dynamically update their devices' network connectivity (e.g., to the Internet) and receive instant real network feedback upon every interaction with the visualization.

2.3 Home User Scenarios

Numerous studies have delved into the security challenges and privacy implications associated with Internet of Things (IoT) devices, particularly in home user scenarios. In their study of the privacy perception of smart speaker users, Lau *et al.* posit that users shall take "more agency" but that it is "difficult to effectively manage one's privacy", and that users are lacking the tools to empower them in light of "the current choice [of] architectures of technology in general" [26]. In line with this, Kumar *et al.* show that from the technological and social perspectives on the growing ubiquity of connected IoT devices, security and privacy are major directions of studies for smart home systems [24].

Yan *et al.* [65] suggest that security and privacy measures must be implemented at every layer of the IoT infrastructure. In particular, home users of different technical skill levels could benefit from visually understanding the communication not only between end-point devices, but also between the home network and the Internet, which Ul Rehman *et al.* [63] highlight as an important element of security threats to home network.

The works by Waseem *et al.* [18] and Osman *et al.* [44] demonstrate the added value of SDN-enabling a smart home network to reinforce security. Osman *et al.* show that *microsegmentation*, *i.e.* putting a device into one of potentially many small and isolated network segments based on the device's functional description, can help to prevent that an attacked device can spread the attack to other devices using the case of the famous Mirai botnet [2]. Mirai, found first in 2016, targeted "a wide range of networked embedded devices such as IP cameras, home routers (many vendors involved), and other IoT devices" [47]. While both approaches are based on automatically grouping devices and configuring the SDN, the authors also acknowledge the necessity of a human intervention for when a microsegmentation interferes and blocks any desired connectivity[2].

In this work, we aim at enabling everyday users to dynamically adjust the allowed communication of devices in their smart homes' SND-enabled network to

[2] "[...] when it "breaks the Internet" and stalls the vital functions of the smart home." [2].

what they judge are tolerable in their use cases. To achieve that, the implemented visualization is interactive and has dynamicity: it is based on information from the SDN and it translates interactions into changed SDN flow rules to directly influence devices' connectivity within the network.

3 Viz[4]NetSec: A Network Visualization for Improving Home Network Security Through Dynamic Interaction

Figure 5 shows our approach Viz[4]NetSec where each connected device is represented as a node in a force-directed graph which represents its logical position within the SDN. When a node is selected in the visualization, the corresponding entry in a tabular view (displayed next to it) is highlighted and the selected node increases in size slightly. Additionally other devices for which the SDN has logged recent network communication get highlighted as well (see yellow nodes in Fig. 5). A demo video[3] as well as the source code[4] are available online.

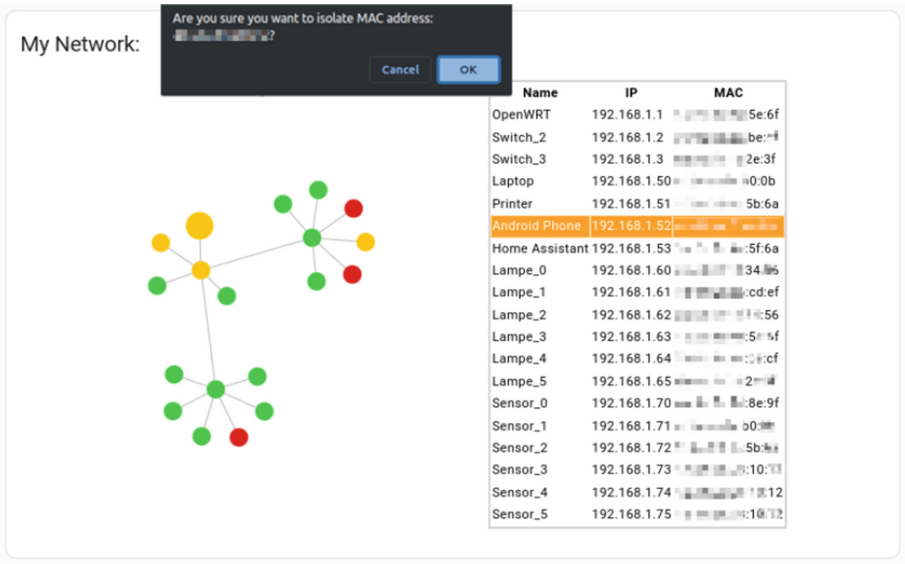


Fig. 5. Viz[4]NetSec integrated as a card in the Home Assistant interface. Selected node appears slightly larger, and the corresponding table row is highlighted. Yellow are nodes that previously communicated with the one selected; green = online; red = offline. Pressing *DELETE* opens a pop-up and asks for user's permission to isolate selected node. Shown network has six smart lamps, six smart sensor devices and laptop, printer and mobile phone; devices are grouped logically into network segments using two SDN switches (See also Fig. 7b). (Color figure online)

[3] Video of Viz[4]NetSec: https://youtu.be/Q57DYiuhmBY or https://henrich.poehls.com/papers/video_viz4netsec_interactions.mp4.

[4] Home Assistant integrated Viz[4]NetSec: https://github.com/pfeifer-j/visualization.

Additionally, the user is asked whether there should be a change in the device's connectivity, *i.e.* if the device selected shall become isolated.

We tried the visualization within a real world setup with smart devices like smart lights (e.g., Shelly LED DUO E27, tapo LED L530E), smart plugs (e.g., tapo Smart Plug Mini Smart Wi-Fi Socket) and other smart devices (e.g., GW1100 weather station). Figure 6 shows the general setup for our prototypical implementation. All used smart devices were chosen just as examples of devices that are commercially available[5] and supported by the commonly used Home Assistant [16].

3.1 Technical Prototype Using Open-Source Components

The Viz^{4}NetSec visualization (Fig. 5) is implemented as a card in the user interface of the open-source home automation platform Home Assistant [16]. Home Assistant markets itself by emphasizing its support for local control and increased privacy. Due to its vast array of community-driven extensions and robust customizability, Home Assistant has emerged as a go-to solution for smart home beginners and enthusiasts [16]. We choose to implement Viz^{4}NetSec in Home Assistant because like Home Assistant consolidates various smart home devices under a single interface regardless of their manufacturers, Viz^{4}NetSec is also vendor-agnostic when it comes to the visualization and control of network communication of various devices. Our prototypical setup is depicted in Fig. 6.

The visual interface of Viz^{4}NetSec is rendered using the d3.js library [8] based on data acquired from the SDN. It is displayed using a web browser on a standard tablet with a touchscreen for user interaction with Home Assistant. Viz^{4}NetSec is realized as a module within Home Assistant. All software necessary to generate the visualization and send the users' isolation commands runs alongside with Home Assistant on the first Raspberry Pi 4 Model B[6] [31].

To receive an overview of the local network and to take control of the devices' communications, the prototype uses an SDN-capable router running open-source OpenWRT [43], and Open vSwitch (OVS) [28]. OpenWRT is an alternative firmware adding SDN-related functions among other customization capabilities for many commercially available standard routers. In tandem, (OVS) is a multilayer, open-source virtual switch, optimized for network automation through programmatic extensions, all while accommodating standard management interfaces and protocols. Together, they offer a powerful suite for precise network traffic management and control [28,43]. The LuCi API of the OpenWRT router facilitates the extraction of information about connected devices [5]. We ran a second Raspberry Pi [31] as router using OpenWRT and Open vSwitch (OVS) and USB 3.0 Ethernet adapters for wired connectivity.

[5] The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

[6] Raspberry Pi 4 Model B running Raspbian OS v.11 with quad-core Cortex-A72 at 1.5GHz with 8GB LPDDR4-3200 SDRAM memory.

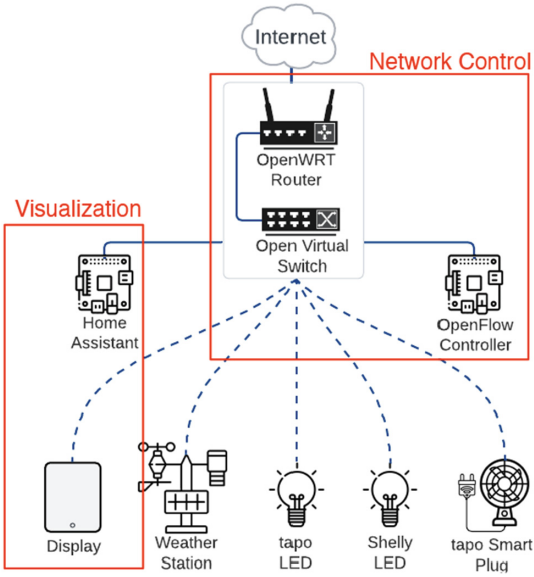


Fig. 6. An overview of the physical components in the prototype; wired connections are blue lines, dotted blue lines represent connections via Wi-Fi. (Color figure online)

In an SDN, a controller is responsible for managing network functions. On the third Raspberry Pi [31] the open-source, python-based software Ryu [52]. It receives instructions from the Viz[4]NetSec interface and controls SDN network devices, like switches and routers using the OpenFlow protocol [52] accordingly.

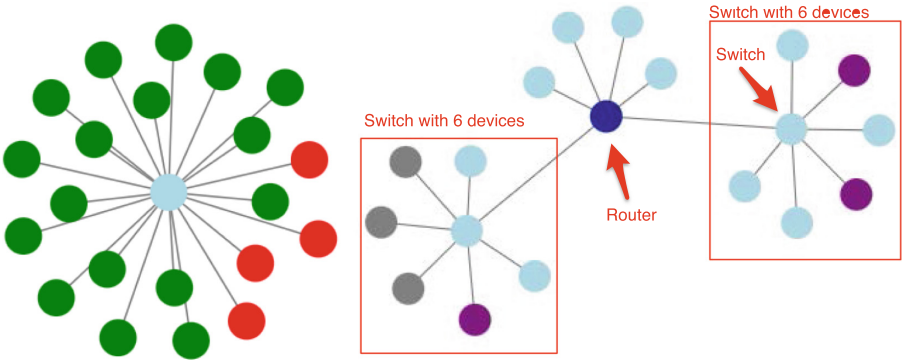
3.2 Features of Viz[4]NetSec

In the following, we provide the background on Viz[4]NetSec's main features.

3.2.1 Feature 1: Network and Communication Flow Visualization

The main visualization in Viz[4]NetSec consists of circle objects built using d3.js library. The relationships between the circles or nodes are sourced from the JSON data obtained from API calls to the SDN controller. Each circle object represents one node or device in the network. Each entry in the JSON data contains details such as name, IP address, MAC address, host, and information on whether the device is currently reachable[7]. These details are added as attributes to the nodes. Based on the host information, a link connects the nodes to a central circle, which symbolizes the OpenWRT router by default. This first of two visualization modes, called the *Physical Visualization*, is shown in Fig. 7a

[7] Note: The "reachable" flag is not accurately set in latest LuCi.rpc release due to a bug [51].



(a) Physical Mode: Online nodes = green, offline nodes = red.

(b) Software-Defined Mode: Online nodes = blue, offline = purple, isolated nodes = gray (due to user's customization). Red annotations added.

Fig. 7. Viz[4]NetSec: Different modes auto generated from SDN information.

The second mode, *Software-Defined Visualization*, is shown in Fig. 7b. It portrays device interconnections based on OVS switch statistics. The API call to OVS results in a specific JSON format for each flow rule. The information gained from the flow rule can be utilized to visualize the communications between devices. When a switch receives a packet, a flow rule is generated, signaling active communication between the source and destination devices[8] specified in the rule. Flow rules can be retained for a variable duration and would enable our visualization to depict either only very recent communication between devices or communication over extended periods.

Upon clicking a node, flow rules with the device's IP address as the source highlight the respective communication endpoints, the destination address. Additionally, flow rules can influence network structure, by grouping devices in the same software-defined subnet together[9]. Figure 7b visualizes a smart home's network split into three segments facilitating SDN functionality: Six smart lights and six sensors are each attached to a separate OVS virtual switch.

In both physical and in the software-defined mode, the user can select each node (see Fig. 5). Further information like name, IP, and MAC address are displayed in a table and the user can initiate a dialog to isolate the device, *i.e.* instructing the SDN controller to disable its network communication by setting restrictive SDN flow rules.

3.2.2 Feature 2 - Dynamic Updates

While the concept behind this feature seems straightforward, its practical application poses challenges. Simply reloading the graph after a specified time inter-

[8] We currently hide broadcast communications and filter out flows with broadcast addresses.

[9] In order to make this distinction, a VLAN must be set up.

val is the obvious solution, but such an approach leads to the undesirable consequence of resetting the graph's structure on every update. Utilizing d3.js's physics engine allows nodes can be dragged around and automatically arrange themselves using the force-directed graph[10]. In future, we add a solution to preserve the positioning of the nodes during updates. For now, the basic update feature provides the user with the option to set a time interval, after which the visualization is reloaded. During an update, new API calls are made in the background and the SVG created by d3.js is re-rendered.

3.2.3 Feature 3 - Dynamic Network Flow Control

In the prototype the initial steps into network flow control can be directly influenced, giving dynamic control to the user by allowing them to interact with the SDN for completely isolating devices or re-joining them to the network. To do this, a device must first be selected in the visualization, as shown in Fig. 5. Following this, the user can press the *DELETE* key to trigger isolation. It is worth noting that switches, routers, and Home Assistant itself cannot be isolated due to potential unforeseen consequences for the network[11]. Before a device is isolated, a popup notifies the user about the upcoming action, ensuring they fully comprehend its implications. Once confirmed, the node changes to a gray color (by default), signifying that a flow rule has been dispatched to the OVS in the background. After a brief period, the rule is applied, restricting the isolated device's to solely communicate with the router.

To revoke this flow rule and reintegrate the device back into the network, the user needs to select the isolated device and press the *ENTER* key. Subsequently, the node resumes its original color, indicating its reactivated status. Isolated devices are consistently stored in the blacklist, located under `/data/` within the project's directory structure. This feature harbors significant potential for future work: It is conceivable to introduce subnetting features, more nuanced isolation options including port-blocking, and direct control of smart devices, such as activating lights, directly from the graph.

3.2.4 Feature 4 - Customization

Users' preferences for visualizations can often be subjective, making customization options crucial. Table 1 shows the various elements, including size, shape, color, and movement, to adjust the visualization using the interface shown in Fig. 8.

To elevate the user experience and streamline customization, color wheels and sliders have been integrated for certain parameters within the editor. These additions enhance the intuitive nature of the interface, making the customization process more user-friendly.

[10] Video of Viz[4]NetSec: https://youtu.be/Q57DYiuhmBY or https://henrich.poehls.com/papers/video_viz4netsec_interactions.mp4.

[11] The list of such protected devices can be configured.

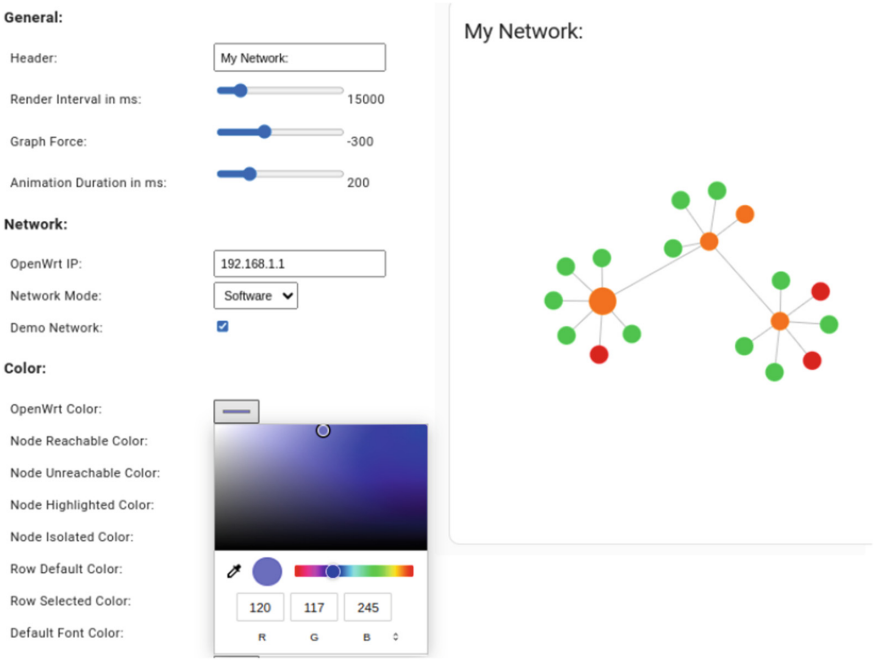


Fig. 8. Editor to change the visual appearance of the nodes in Viz[4]NetSec.

Table 1. Description of various parameters to customize Viz[4]NetSec.

General	
Header	Name of the network
Render Interval	Time until an update happens
Graph Force	Strength of the physics engine
Animation Duration	Animation time for events
Network	
OpenWrt IP	IP address of the source router
Network Mode	Network visualization mode
Demo Network	Demo network for development
Color	
Colors	Colors for fonts, nodes, and links in each state
Shape	
Shape	Size for nodes and links in each state

4 Discussion and Future Work

In this section, we discuss the different features of Viz^{4}NetSec and its integration within Home Assistant for smart home network use cases. We use the methodologies described by Staheli *et al.* when evaluating visualization tools designed for cyber security [58]. Their work provides a model for the different elements of a security visualization, so called *dimensions*, that can be evaluated. Using *critique* as the technique for evaluating Viz^{4}NetSec, we categorize our discussion within the following dimensions. For each evaluated dimension we briefly describe the scope but refer the reader to [58] for more details:

- *Algorithmic efficiency*: We provide empirical system performance measures, and an evaluation of scalability.
- *Usability and learnability*: How easily a user can use and learn the visual interface?
- *Component interoperability*: How well does Viz^{4}NetSec fit with already existing system?
- *Cognitive workload*: From a cognitive science perspective, how hard does the person have to think to accomplish their tasks while using the system?
- *Task performance*: How well does the user perform a predefined task using Viz^{4}NetSec?
- *Feature set utility*: How useful are the set of features available?

For each selected dimension, we describe how our approach to the design of Viz^{4}NetSec features (see Sect. 3) helps users to achieve the set goals: visualization for simple smart home tasks (see Sect. 1.1), and dynamicity of the exercised control (see Sect. 1.2). Moreover, we discuss limitations of Viz^{4}NetSec as well as potential future directions that current features could enable.

4.1 Algorithmic Efficiency

4.1.1 Time to Visualize the Force Directed Graph

In the relatively small network illustrated in Fig. 6 with eight active participants plus a computer used for performance measurement the average time to run the visualization is below 500 ms for page load and graph rendering without waiting for the arrangement of the graph. Specifically without waiting for the arrangement of all nodes in the graph to stabilize, the average is around 350 ms, the duration fluctuated between 250 ms and 475 ms across 50 measurements[12]. However, Viz^{4}NetSec editor features a force-directed graph whose time to stabilize varies based on the specified strength. To ensure graph readability, users can customize the strength that determines nodes proximity (see Fig. 8); a value

[12] Time measured over 50 runs in Google Chrome Version 116.0.5845.142 on Windows 10, Intel Core i7-4790k at 4.0 GHz and 16 GB of DDR3 RAM. One Raspberry Pi Model 4B ran Viz^{4}NetSec and sent the API calls, another one received them and ran SDN controller. Stabilization measured by event times logged to Chrome's Development Console.

of −300, which we chose as default, spaces the nodes further apart. On average, for this small-scale setup and using default settings, the graph requires between 600 ms and 825 ms to fully load and stabilize (see footnote 12).

To assess visualization time on a larger scale, demo networks with 20, 100, and 250 devices were constructed. To only measure the time to show the graph the API calls to the SDN controller are replaced with static data. With the default physics strength of −300, the stabilization times were slightly longer than the small-scale setup: around 0.653 s, 1.320 s, and 3.664 s, respectively (see footnote 12). However, it is worth noting that in larger networks, API calls might introduce additional delays, although graph rendering remains fairly consistent as long as the rendering hardware is not saturated.

4.1.2 Time to Isolate a Device from the Network

The time for an API call to the SDN controller that updates the ruleset governing the flow is almost immediate. However, for dynamicity the more important metric is the duration until the new rule effectively takes action. The interval until a device looses network connectivity, *i.e.*, is isolated due to the flow rule in effect[13], ranges from 3 s to 7 s. Reintegrating an isolated device back into the network takes slightly less time, with a duration between 2 s to 5 s.[14]

4.1.3 Network Delay

Upon measuring the performance of the network, no noticeable difference was observed between an OpenWRT router using its default configuration and the same router using OVS. Further testing, especially in larger-scale networks, is necessary to draw more comprehensive conclusions.

4.2 Usability and Learnability

Viz^{4}NetSec offers a visualization-based approach to facilitate the understanding of a complex concept that is the segmentation of a SDN architecture. The force-directed node-link diagram maps the logical layout of the system. Furthermore, Viz^{4}NetSec follows a network visualization approach that is well-studied in literature, broadly used in modern tools, and has been demonstrated to help user understand the underlying technical construct of a network. Learnability is supported as the dynamicity would allow a user to 'see' the effect on the networked device when interacting with the visualized node representing it, *e.g.* it is unreachable in the app once isolated and going gray in the visualization.

Future Work: It remains as future work to study the effects that logically separating networks from their physical reality of wired or wireless communication links using Software Defined Networking would have on the learnability. If users would have a mental model that wireless devices communicate directly with

[13] measured using a ping to Google from a laptop as the device to become isolated.

[14] Note: We assume the application of flow rules to become even faster if a dedicated OpenWRT router is used as hardware instead of a Raspberry Pi running OpenWRT.

one central wireless router in their network they would need to learn that an SDN-enabled router could logically group them into different networks.

4.3 Component Interoperability

We briefly explain that Viz^{4}NetSec was added into the existing smart home control software Home Assistant [16]. Not only does Viz^{4}NetSec integrate into the Home Assistant workflow, as a data visualization component, it also acts as an interface to the network flow controller (see Fig. 6). The software code, as well as step-by-step setup instructions are made available on the following repository https://github.com/pfeifer-j/visualization.

In general we observe benefits for interoperability if the management of smart home devices for home automation is grouped alongside the network communication management of the same networked devices. For example, with the current interoperability of Viz^{4}NetSec, the user could open the Viz^{4}NetSec card for troubleshooting and checking if the device is still online in the network, or if there would be a problem in switching a smart lamp on within the related task modeled elsewhere within Home Assistant.

Future Work: A more closely link of the node in the network visualization with the device's identification within Home Assistant would increase interoperability even further: on first instance, users could more easily re-identify known networked devices in the visual representation of the networks, e.g. by the same name, by a link to the device's configuration within Home Assistant's interface. Further, integration of the SDN data on which Viz^{4}NetSec works could allow to take SDN events (like node online or offline or a predefined inter-node communication) as triggers for Home Assistant's so called automations. Also vice versa, one could make an automation trigger certain flow rules, e.g. have a "privacy please" scene within Home Assistant that not only lowers the window shades but also isolates microphone based assistants from the Internet facilitating the SDN functionality.

4.4 Cognitive Workload and Task Performance

In its current prototype, Viz^{4}NetSec integrates within Home Assistant as an extension of the community-maintained tool. It complements the dashboard-like representation offered by the open-source integration for smart home. Viz^{4}NetSec configuration does not incur additional burden on the user beyond regular setup of add-on components to Home Assistant.

Future Work: To further understand the cognitive effort from using Viz^{4}NetSec, controlled user studies are required to measure task performance, success rate, satisfaction, and frustration.

4.5 Feature Set Utility

The visualization offered by Viz^{4}NetSec harmonizes the network configuration of the devices, regardless of their brands. The node-link diagram masks the complex configurations and segmentations offered by the SDN structure, and simplifies the representation of the network layout as flattened connections between switches, routers, and end-point devices. While in a traditional router interface, disconnecting devices might be a few clicks away, Viz^{4}NetSec users manipulate the interactive graphical representation to isolate or disconnect the device instantly.

Through the interactivity, controls, and visualizations offered by Viz^{4}NetSec, users no longer need to monitor what each smart home device is internally doing. Instead of having to disable communications to the outside from within vendor specific device interfaces and trust the device to adhere to that, the trust is shifted towards the network management system that gathers the data for the visual feedback and executes the user's commands given within the visualization. By that design, the visualization reflects information from the network interface. This goes as far as assisting the user in identifying devices that the user was not yet or no longer aware[15] of, detecting changed communication behavior[16] or that do not need a steady connectivity to the Internet.

Future Work: User studies are needed to identify new features, fine-grained controls that users deemed necessary to enhance their privacy and their control thereof, and to redefine the scope of potential future directions such that they align with real-life user (attack) scenarios.

Moreover, the aspect of dynamicity is generally applicable in both directions, *i.e.* on the one hand the visualization is directly influenced by the underlying network's activity and reflects its configuration. On the other hand, the interactions with the visualization get directly reflected in the network's configuration. While our prototype exposes initial work for both directions, future work is planned: We want to add a feature to visualize most current communications, alike the interface of EtherApe (see Fig. 4a). Furthermore, users should visually be able to replay past communication behavior, and review previous network communications. The latter may enable users to familiarize what the devices' communication behavior is[17] when they are not watching them. From a security point of view such a screen shot could be triggered if an additionally deployed intrusion detection system (IDS) would signal an alerting behavior.

[15] Ethnographic studies called it "lovingly neglected infrastructures" (in the German original its "liebevoll vernachlässigte Infrastrukturen" [12]).

[16] Just on 9th Jan. 2024 a user wondered "Why is my LG Washing Machine using 3.6 GB of data/day?" https://twitter.com/Johnie/status/1744556503183585471 (accessed 31.01.2024).

[17] This could enable them to find devices scanning the network using broadcasts or washing machines sending data to the Internet like discussed in footnote 16.

5 Conclusion

We present Viz^{4}NetSec, a network visualization for helping home users to dynamically configure their network and thus increase security and privacy in a smart home system context. As the number of connected smart home devices (SHD) continues to grow, techniques such as Software Defined Networking (SDN) have emerged as an effective way to help user reinforce or adjust their network policies. Viz^{4}NetSec implements node-link diagram to reflect complex concepts such as network microsegmentation [44] that a modern SDN offers — all within software so, without requiring the user to buy additional hardware switches or re-cable a network. It is integrated as a card to Home Assistant, a well-maintained open-source integration tool for smart homes. We choose Home Assistant because it nicely consolidates various smart home devices under a single interface regardless of their manufacturers [16], and Viz^{4}NetSec does the same offering a vendor-agnostic visualization and control of network communication of various devices without the need to understand vendor specific user interfaces.

Viz^{4}NetSec features a simple, customizable tree-like representation of the underlying logical network. Its interactivity enables home users to dynamically adjust their network policies (such as connectivity to the Internet) and receive instant visual feedback upon every update. All in all, our approach to Viz^{4}NetSec combines a visualization for simple smart home tasks with a support for dynamicity of the visually exercised controls.

The software code of the current prototype, as well as step-by-step setup instructions are made available online (https://github.com/pfeifer-j/visualization). In its current prototype version, Viz^{4}NetSec visualizes the network layout configured using an underlying SDN and empowers the user to control each device's connectivity within such a network. Already the current prototype stage's initial features helped us identify future research directions towards novel visualizations and dynamic interactions techniques to, on one hand, visualize the variety of information gathered from an SDN-based architecture, and on the other, hand ease the facilitation of its rules for fine-grained control.

Finally, one could add metrics based on an automated assessment of the security-posture or a privacy-increase. Hence, a given network configuration for a specific device or set of devices would get a score. This could lead to users sharing such "best" configurations leading to a gamification [10] of increasing the privacy and security of smart home networks.

Acknowledgments. The work of Pöhls and Pfeifer was funded by the Bavarian Ministry of Science and Arts (Germany) under the `ForDaySec.de` project.

References

1. Albany, M., Alsahafi, E., Alruwili, I., Elkhediri, S.: A review: secure internet of thing system for smart houses. Procedia Comput. Sci. **201**, 437–444 (2022)
2. Antonakakis, M., et al.: Understanding the Mirai Botnet. In: 26th USENIX Security Symposium, pp. 1093–1110. USENIX Association (2017)

3. Ball, R., Fink, G.A., North, C.: Home-centric visualization of network traffic for security administration. In: IEEE Symposium on Visualization for Cyber Security, pp. 55–64. ACM (2004)
4. Blue, R., Dunne, C., Fuchs, A., King, K., Schulman, A.: Visualizing real-time network resource usage. In: Goodall, J.R., Conti, G., Ma, K.-L. (eds.) VizSec 2008. LNCS, vol. 5210, pp. 119–135. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85933-8_12
5. Brady, F.: openwrt-luci-rpc documentation rel.1.1.16 (2023). https://readthedocs.org/projects/openwrt-luci-rpc/downloads/pdf/stable/. Accessed 25 Oct 2023
6. Buil-Gil, D., et al.: The digital harms of smart home devices: a systematic literature review. Comput. Hum. Behav. **145**, 107770 (2023)
7. Crabtree, A., Rodden, T., Hemmings, T., Benford, S.: Finding a place for UbiComp in the home. In: Dey, A.K., Schmidt, A., McCarthy, J.F. (eds.) UbiComp 2003. LNCS, vol. 2864, pp. 208–226. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39653-6_17
8. D3.js. D3.js - Data-Driven Documents v.7.8.5 (2023). https://d3js.org/. Accessed 01 Jan 2024
9. Danbatta, S., Varol, A.: Comparison of Zigbee, Z-wave, Wi-Fi, and bluetooth wireless technologies used in home automation (2019). Accessed 15 June 2023
10. Dicheva, D., Dichev, C., Agre, G., Angelova, G.: Gamification in education: a systematic mapping study. J. Educ. Technol. Soc. **18**(3), 75–88 (2015)
11. Dini, M.T., Sokolov, V.: Internet of Things security problems. arXiv preprint arXiv:1902.08597 (2019)
12. Eckhardt, D., Freiling, F., Herrmann, D., Katzenbeisser, S., Pöhls, H.C.: Sicherheit in der Digitalisierung des Alltags: Definition eines ethnografisch-informatischen Forschungsfeldes für die Lösung alltäglicher Sicherheitsprobleme. In: 13. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI Sicherheit 2024). LNI (2024)
13. Guimaraes, V.T., Freitas, C.M.D.S., Sadre, R., Tarouco, L.M.R., Granville, L.Z.: A survey on information visualization for network and service management. IEEE Commun. Surv. Tutor. **18**(1), 285–323 (2015)
14. Gupta, K., Pandey, A., Chan, L., Yadav, A., Staats, B., Borkin, M.A.: Portola: a hybrid tree and network visualization technique for network segmentation. In: IEEE Symposium on Visualization for Cyber Security, pp. 1–5 (2022)
15. Hnatyuk, K.: Internet of things (IoT) statistics: 2022/2023 (2023). https://marketsplash.com/internet-of-things-statistics. Accessed 30 Nov 2023
16. Home Assistant Developers. Home Assistant (ver. 2023.8) (2023). https://www.home-assistant.io/. Accessed 25 Oct 2023
17. Home Assistant Developers. Nmap tracker (2023). https://www.home-assistant.io/integrations/nmap_tracker/. Accessed 25 Oct 2023
18. Iqbal, W., et al.: ALAM: anonymous lightweight authentication mechanism for SDN-enabled smart homes. IEEE Internet Things J. **8**(12), 9622–9633 (2021)
19. Jeong, C.Y., Chang, B.H., Na, J.C.: A survey on visualization for wireless security. In: 4th International Conference on Networked Computing and Advanced Information Management, pp. 129–132 (2008)
20. Jianu, R., Rusu, A., Hu, Y., Taggart, D.: How to display group information on node-link diagrams: an evaluation. IEEE Trans. Visual Comput. Graphics **20**(11), 1530–1541 (2014)
21. Joshi, S.: 70 IoT statistics to unveil the past, present, and future of IoT (2023). https://learn.g2.com/IoT-statistics. Accessed 15 June 2023

22. Kan, Z., Hu, C., Wang, Z., Wang, G., Huang, X.: NetVis: a network security management visualization tool based on treemap. In: 2nd International Conference on Advanced Computer Control, pp. 18–21 (2010)
23. Klassen, C.: Lightbeam (2023). lightbeam.chikl.de. Accessed 11 Jan 2024
24. Kumar, S., Tiwari, P., Zymbler, M.: Internet of Things is a revolutionary approach for future technology enhancement: a review. J. Big data **6**(1), 1–21 (2019)
25. Ladefoged, J.: Device tracker - first home/last home (2017). https://community.home-assistant.io/t/device-tracker-first-home-last-home/30036. Accessed 30 Nov 2023
26. Lau, J., Zimmerman, B., Schaub, F.: Alexa, are you listening? Privacy perceptions, concerns and privacy-seeking behaviors with smart speakers. Proc. ACM Hum.-Comput. Interact. **2**(CSCW) (2018)
27. Lechtenbörger, J.: Das verworrene web (2019). www.informationelle-selbstbestimmung-im-internet.de/Collusion.html. Accessed 01 Jan 2024
28. Linux Foundation. Open vSwitch (v. 2.14.3) (2023). https://www.openvswitch.org/
29. Livnat, Y., Agutter, J., Moon, S., Erbacher, R.F., Foresti, S.: A visualization paradigm for network intrusion detection. In: 6thIEEE SMC Information Assurance Workshop, pp. 92–99. IEEE (2005)
30. Livnat, Y., Agutter, J., Moon, S., Foresti, S.: Visual correlation for situational awareness. In: 2005 IEEE Symposium on Information Visualization, INFOVIS 2005, pp. 95–102. IEEE (2005)
31. Raspberry Pi Ltd. Raspberry pi 4 model b (2023). https://www.raspberrypi.com/products/raspberry-pi-4-model-b/. Accessed 13 Oct 2023
32. Lima, M.: Visual complexity website (2024). http://www.visualcomplexity.com/vc/project.cfm?id=268. Accessed 14 Jan 2024
33. Masoudi, R., Ghaffari, A.: Software defined networks: a survey. J. Netw. Comput. Appl. **67**, 1–25 (2016). Accessed 25 Oct 2023
34. May, C.J., Hammerstein, J., Mattson, J., Rush, K.: Defense in depth: foundations for secure and resilient it enterprises. The Software Engineering Institute (2006)
35. McPherson, J., Ma, K.-L., Krystosk, P., Bartoletti, T., Christensen, M.: PortVis: a tool for port-based detection of security events. In: ACM Workshop on Visualization and Data Mining for Computer Security, pp. 73–81 (2004)
36. Neumann, P., Schlechtweg, S., Carpendale, S.: ArcTrees: visualizing relations in hierarchical data. In: EuroVis, pp. 53–60 (2005)
37. Nguyen-Ngoc, A., Lange, S., Geissler, S., Zinner, T., Tran-Gia, P.: Estimating the flow rule installation time of SDN switches when facing control plane delay. In: German, R., Hielscher, K.-S., Krieger, U.R. (eds.) MMB 2018. LNCS, vol. 10740, pp. 113–126. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-74947-1_8
38. Nobre, C., Meyer, M., Streit, M., Lex, A.: The state of the art in visualizing multivariate networks (2019). https://osf.io/upbm2
39. NSA. Manageable network plan (2015). https://nsarchive.gwu.edu/sites/default/files/documents/2725523/Document-2-11.pdf
40. Nunes, B.A.A., Mendonca, M., Nguyen, X.-N., et al.: A survey of software-defined networking: past, present, and future of programmable networks. IEEE Commun. Surv. Tutor. **16**(3), 1617–1634 (2014)
41. Open Networking Foundation. OpenFlow Switch Specification (ver. 1.5.1). https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf

42. OpenDaylight Project. Viewing network topology (2023). https://nexus.opendaylight.org/content/sites/site/org.opendaylight.docs/master/userguide/manuals/userguide/bk-user-guide/content/_viewing_network_topology.html. Accessed 30 Nov 2023
43. OpenWRT. OpenWRT Project (v. 21.02.3). https://openwrt.org/
44. Osman, A., Wasicek, A., Köpsell, S., Strufe, T.: Transparent microsegmentation in smart home IoT networks. In: 3rd USENIX Workshop on Hot Topics in Edge Computing (HotEdge). USENIX Association (2020)
45. Parulkar, G., Schmidt, D., Kraemer, E., Turner, J., Kantawala, A.: An architecture for monitoring, visualization, and control of gigabit networks. IEEE Netw. **11**(5), 34–43 (1997)
46. Pires, A.: Zigbee network map - red dashed lines (2020). https://community.home-assistant.io/t/zigbee-network-map-red-dashed-lines/216670. Accessed 30 Nov 2023
47. Plohmann, D., Enders, S.: Malpedia by Fraunhofer FKIE (2019). https://malpedia.caad.fkie.fraunhofer.de/details/elf.mirai. Accessed 20 Jan 2024
48. Plume. Plume IQ 1H 2022 smart home market report (2022)
49. Pöhls, H.C., Rakotondravony, N.: Dynamic consent: physical switches and feedback to adjust consent to IoT data collection. In: Streitz, N., Konomi, S. (eds.) HCII 2020. LNCS, vol. 12203, pp. 322–335. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-50344-4_23
50. Poole, E.S., Chetty, M., Grinter, R.E., Edwards,W.K.: More than meets the eye: transforming the user experience of home network management. In: 7th ACM Conference on Designing Interactive Systems, pp. 455–464 (2008)
51. Read the Docs. OpenWRT LuCI RPC Documentation (2023). https://readthedocs.org/projects/openwrt-luci-rpc/downloads/pdf/stable/. Accessed 13 Oct 2023
52. Ryu SDN Framework Community. Ryu SDN Framework v.4.34 (2023). https://ryu.readthedocs.io/en/latest/. Accessed 13 Oct 2023
53. Sarikaya, A., Correll, M., Bartram, L., Tory, M., Fisher, D.: What do we talk about when we talk about dashboards? IEEE Trans. Visual Comput. Graphics **25**(1), 682–692 (2018)
54. Schulz, H.-J.: Treevis.net: a tree visualization reference. IEEE Comput. Graphics Appl. **31**(6), 11–15 (2011)
55. Scott-Brown, J., Bach, B.: NetPanorama: a declarative grammar for network construction, transformation, and visualization (2023)
56. Shiravi, H., Shiravi, A., Ghorbani, A.A.: A survey of visualization systems for network security. IEEE Trans. Visual Comput. Graphics **18**(8), 1313–1329 (2011)
57. Spielvogel, K., Pöhls, H.C., Posegga, J.: TLS beyond the broker: enforcing fine-grained security and trust in publish/subscribe environments for IoT. In: Roman, R., Zhou, J. (eds.) STM 2021. LNCS, vol. 13075, pp. 145–162. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-91859-0_8
58. Staheli, D., et al.: Visualization evaluation for cyber security: trends and future directions. In: 11th Workshop on Visualization for Cyber Security, pp. 49–56 (2014)
59. Starks, J., Song, L., Allen, J.K., Mistree, F.: Integrating user preference into improved home appliance scheduling. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, vol. 85390, p. V03BT03A048. ASME (2021)
60. Stawowski, M.: Network security architecture. ISSA J. **7**, 34–38 (2009)
61. Toledo, J.: EtherApe: a live graphical network monitor tool (2000). http://etherape.sourceforge.net. Accessed 01 Jan 2024

62. Tolmie, P., Pycock, J., Diggins, T., MacLean, A., Karsenty, A.: Unremarkable computing. In: SIGCHI Conference on Human Factors in Computing Systems, pp. 399–406 (2002)
63. UlRehman, S., Manickam, S.: A study of smart home environment and its security threats. Int. J. Reliab. Qual. Saf. Eng. **23**(03), 1640005 (2016)
64. Wagner, N., et al.: Towards automated cyber decision support: a case study on network segmentation for security. In: IEEE Symposium Series on Computational Intelligence, pp. 1–10 (2016)
65. Yan, Z., Zhang, P., Vasilakos, A.V.: A survey on trust management for Internet of things. J. Netw. Comput. Appl. **42**, 120–134 (2014)
66. Yermalovich, P.: Dashboard visualization techniques in information security. In: International Symposium on Networks, Computers and Communications (ISNCC), pp. 1–6. IEEE (2020)
67. Zavalyshyn, I., Duarte, N.O., Santos, N.: HomePad: a privacy-aware smart hub for home environments. In: IEEE/ACM Symposium on Edge Computing (SEC), pp. 58–73 (2018)
68. zha ng. zha-map: a visualization tool for Zigbee Home Automation (2023). https://github.com/zha-ng/zha-map. Accessed 25 Oct 2023