Cybersecurity Vulnerabilities in Two Artificially Intelligent Humanoids on the Market

Matt Kinzler
ReliaQuest
Tampa, FL, USA

Justin Miller
Scanalytics Inc.
Milwaukee, WI, USA

Zhou Wu
MSCS
Marquette University
Milwaukee, WI, USA

Andrew B. Williams
HEIR Lab
The University of Kansas
Lawrence, KS, USA
andrew.williams@ku.edu

Debbie Perouli
MSCS
Marquette University
Milwaukee, WI, USA
debbie.perouli@mu.edu

***Abstract*—As artificially intelligent humanoids become increasingly prevalent in the home, it is imperative that we develop secure designs to guard against cyberattacks. The next evolution of AI-powered home devices, such as Alexa, is to create physical effectors to enable these devices to alter their environments. Current humanoids on the market, such as the EZ-Robot JD and NAO, are examples of artificially intelligent robots that may one day become common in home environments. If these humanoids are not designed to be safe against cybersecurity vulnerabilities, they may be used to cause harm to living spaces and possibly even the humans living in these spaces. This paper examines the cybersecurity of two humanoid robots and provides recommendations for future safe designs and protections in artificial intelligent social robots.**

***Index Terms*—humanoid, artificial intelligence, social robots, security vulnerabilities**

I. Introduction

The field of robotics is in an exciting position of rapid technological breakthroughs, while the barriers to entry are decreasing significantly. Social robots are becoming increasingly affordable and common not only in universities and the industry, but also in homes. Currently, there are about twenty robots that are marketed for households and other social environments. While there is abundant research in this booming field, there are few studies on the cybersecurity of these robots [1]–[4]. As robots become integrated in our daily lives, the amount of personal information they can access will increase. Not only will they hold valuable data like our personal computers, but they will also be able to manipulate the physical world, making them a useful vehicle to carry out cyber attacks with the potential to cause physical harm.

Humanoids interact with their surroundings and acquire large amounts of data through sensors and electrical components. If this data is intercepted or redirected to a malicious system, these systems will be receiving information including audio, video, actuator movement and interactions the robot experiences with its surroundings. This data can be used in countless ways of exploitation. Both robot manufacturers and code developers need to take into consideration cyber security and decide whether current levels of security are adequate for the use of their product.

In this paper, we analyze the cybersecurity vulnerabilities of two popular, artificially intelligent humanoids to propose a safer design. The humanoids we selected are the SoftBank Robotics NAO version 5 and the EZ-Robot JD. On many occasions, adding authentication mechanisms improves drastically the cybersecurity properties of the robot. We discuss the common design decisions that the manufacturers have taken and observe that, even when their decisions differ on implementation, cybersecurity does not seem to be a central part of the design process. We communicated the results of this study to the robots' manufacturers and have included their response.

II. User Perception of Cybersecurity Dangers

Several studies demonstrate a lack of user understanding of privacy and security risks associated with social robot or smartphone applications. Balebako *et al*. examines the user understanding of privacy leaks of smartphone applications and the users' reaction to being aware of the leakage by interview based focus groups and a data sharing detection app [5]. The authors find that users are not clear on how much data is shared through apps and are surprised by the actual data leakage. However, if the users are informed upon every time data is leaking, they feel annoyed, which leaves no simple solution to this issue. R. Kang *et al*. investigates the practices of security and privacy among people with different computer science backgrounds [6]. This study displays diverse mental models that the end user may have about how the Internet works and how their privacy might be at risk over the Internet. The important issue of representing the information to the end users in a way that makes sense to them remains open. A study of over 60 users found that people choose applications or services depending mostly on price, popularity, and recommendations from friends [7]. However, few people base their decisions on security and privacy policies, which implies implicit data sharing with third parties is unlikely to be noticed by end users.

A robot can have complex interactions with the human counterpart, and thus be able to serve attacks such as stealth and eavesdropping. Sometimes people over-trust robots. S.

The first two authors performed this research study while they were undergraduate students in the department of Mathematics, Statistics and Computer Science at Marquette University.

Booth *et al*. discovered this phenomenon under the context of university dormitory security. A large portion of the observed students were willing to help an unknown robot enter dormitory buildings [8].

III. Network and Cloud Attacks on Robots

Denning *et al*. [1] analyzes the security strength of commercial household robots that were available in 2009 and provides a systematic discussion regarding possible attacking interfaces and forms. Finnicum *et al* [9] proposes a robot model in which the robot functionality is augmented by a robot application market. Via this architecture, security and privacy are separated from the apps and managed by a security kernel.

A wide collection of research papers demonstrates the vulnerability of Cloud based services. Since social robots often rely on the Cloud in order to provide services such as emotion recognition, cloud attacks are relevant to them. For example, [10] discusses a side channel attack referring information via data access patterns; [11] demonstrates an attack in which the malicious party can disclosure data from encrypted MapReduce job traffic, if it has some control over the procedure of mapper and reducer. If an adversary can place a malicious virtual machine (VM) as resident within the same physical machine with the target guest VM, it is possible to break the memory isolation enforced by virtualization [12].

Hijacking robots through the local network is another common approach in cyber attacks. Attacks have already been demonstrated upon drones, or unmanned aerial systems(UAS) [13]. A study by J. Pleban *et al*. reveals that highly integrated complex systems with limited resources may suffer from poor implementation of best-practices. In their study, the Parrot AR.Drone 2.0 system is vulnerable to FTP and Telnet attacks [14]. Considering the potential cooperation between robot and smart things of IoT, more interfaces are open for malicious entities. The "Cross-device dependencies" illustrated in [15] demonstrates the possibility of altering the behavior of secured devices by manipulating compromised devices, which means that a security failure of any device within a system could have disastrous results.

IV. Selecting the NAO and JD Humanoids

We focused our study on the cybersecurity design decisions made by two companies that have been highly successful in manufacturing and selling humanoids to their respective audiences. One major criterion for our selection was that the humanoids should be programmable by their user through desktop applications instead of phone apps only, so that users can explore the robots at a greater depth.

The NAO robot was introduced to the market by its original company, Aldebaran Robotics, in 2008 and has since been used extensively in RoboCup competitions, research projects, education, and marketing campaigns globally [16]–[18]. SoftBank Robotics acquired Aldebaran in 2013, and with the sixth version of the NAO in 2018 has expanded the target audience to include the healthcare industry [19]. We studied the characteristics of NAO version five and we presented our first results in a late breaking report [2].

NAO's price at approximately 9000 USD made us consider whether there are more affordable options for programmable humanoids that can still provide a compelling experience to a more general audience. The JD Humanoid from EZ-Robot stood out at the impressive price of 430 USD. Although JD lacks the rich set of NAO's sensors, JD has sixteen degrees of freedom, carries a camera, is stable in its walking and dancing, and has an extensive set of capabilities due to leveraging Microsoft Cognitive Services, such as emotion and vision. Both humanoid designs allow the execution of demanding computations at a different device like a laptop. However, JD is completely dependent on such a device for its operation, which allows it to keep its price low, while its battery life is significantly smaller (approximately one hour long). The two humanoids also differ in height with the NAO (23 in/58 cm) being at least twice as tall as JD.

The EZ-B Wi-Fi enabled robot controller is the embedded processor at the heart of JD. Apart from being the brain in all of EZ-Robot's kits, it is also sold separately and marketed for do-it-yourself (DIY) robotic projects. According to the manufacturer's website, EZ-B is powering over 20,000 robots worldwide. Among the features that we found compelling in adopting JD, and its EZ-B controller, is the amount of high quality video tutorials that EZ-Robot provides as technical support together with its forum community.

V. NAO Architecture

The NAO robot is built to interact with its environment and as a result, there is a complex system of sensors, actuators, and control systems. Communicating with the CPU are a handful of microcontrollers which are used to distribute information to actuators and collect data from sensors. Ethernet, serial, and USB ports can be located on the back of the head. NAO is designed to use primarily the WiFi 802.11g protocol to communicate with network devices and other robots. NAO can gather and disseminate information through four microphones, two loudspeakers, two gyrometers, three accelerometers, two 30FPS video cameras, two ultrasonic sensors, and LEDS.

The NAO robot runs on a Gentoo Linux platform underneath a proprietary software framework called NAOqi. NAOqi handles tasks such as parallelism, resource management and synchronization so developers do not have to worry about such issues when writing applications for the robot [20]. NAOqi acts as a broker to organize and locate services so any module can access them. NAOqi has access to a list of modules, which are classes in a library to execute certain actions on the robot. By establishing a connection with a broker, a user gains access to all of the modules linked to the broker. This broker has root privileges and can access any hardware device.

Modules use a broker to interact with one another through proxies. A proxy represents a specific module and acts as a instance of the module. If proxies share a broker, they are called local modules, while modules with different brokers are called remote modules. Brokers can connect to remote

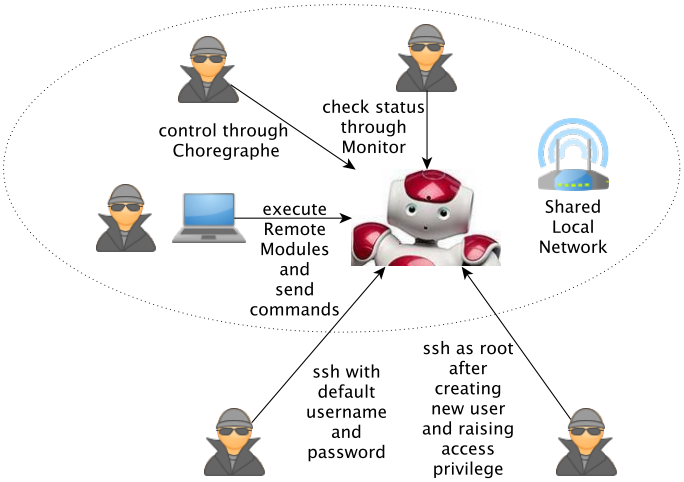

Fig. 1. Ways in which the NAO can be compromised.

modules and other brokers over a network, but they lack the speed of local modules. Custom modules can be added to the library the NAOqi broker manages.

Each NAO hosts a personal website on port 80 using nginx 1.3.14. This website prompts for a username and password, and once inside, a user can edit multiple settings on the NAO robot and view system information and current running processes. Other open ports and services include FTP on port 21, SSH on port 22, NAOqi on port 9559 and web services on 5222.

VI. Authentication Needs for NAO

In this Section, we uncover cybersecurity vulnerabilities related to the lack of authentication mechanisms in NAO. Figure 1 summarizes our findings.

A. Programming with Choregraphe

Choregraphe is the official Integrated Development Environemnt (IDE) provided by SoftBank Robotics. This application provides a simple way to write programs for the NAO robot either through drag-and-drop modules or by writing Python code. The software allows a user to upload programs to the NAO robot, which are then immediately executed. Programs can also be configured to run on startup.

Choregraphe does not conduct any authorization check when attempting to upload a program to any given NAO. The program even locates all NAO robots on the network and assuming port 9559 is listening, a program can be uploaded and override the robot's current task. The executable on the robot can issue a wide range of instructions including all available operating system calls.

The only limitation is that each robot can have only one connection to Choregraphe at any time. If many robots, with different owners, are connected to the same wireless network, then each owner needs to make sure that his/her application is connected to the respective robot at all times, so that there is no time window for someone else to connect to the robot and take control of it.

B. Remote Modules

The NAOqi interface allows other devices to access and instantiate modules on NAO over a network, and these are referred to as remote modules. In order to send commands to NAO, a Software Development Kit (SDK) to interface with the NAOqi software is needed. The SDK can be downloaded and installed after setting up a free developer account on the SoftBank Robotics website.

Remote modules bind with port 9559 on the NAO robot using a TCP connection. Instead of uploading an executable file to the robot, the remote module makes calls to the robot as needed and runs on the remote device, not the NAO. To test this, we ran a simple reverse shell python script, and when executing simple commands such as `ls`, the printed directory list was that of the remote computer. To analyze how the SDK communicated with the robot, we ran a program that sent five speech commands with a time delay of two seconds between each command and used the WireShark packet sniffer [21] to collect a TCP dump of the interaction. While our code ran, there was communication to establish a connection with the NAOqi architecture; then every two seconds the program would send a command from the remote computer to the robot, and the command would be executed. Once the program had completed, the remote socket was closed and the communication with the robot stopped.

Modules executed over a network require the parameters of the IP address of a NAO robot and the listening port. Assuming an attacker is on the same network, this information is easy to acquire, and with no authorization checks, nothing prevents one from running remote modules on any NAO within the same network. When modules are uploaded, they are executed concurrently with any program that is currently running on the NAO robot.

C. Monitor Status

Monitor is a program that can be downloaded from the SoftBank Robotics website and be exploited in a similar manner to Choregraphe. Monitor allows a user to connect to a NAO robot without any form of authentication and gain access to nearly every hardware feature in the robot. Using monitor, it is possible to:

- view a live video feed from the robot's cameras;
- view a list of currently running services;
- view current memory usage;
- access the NAO robot's log.

As the NAO robot is watching around the environment in which it resides, a malicious party can take advantage of Monitor and receive the video stream placing the NAO user's privacy at stake. Figure 2 shows a screenshot of the Monitor application including the NAO's camera view. Depending on the circumstances, the user's physical safety may also become at risk.

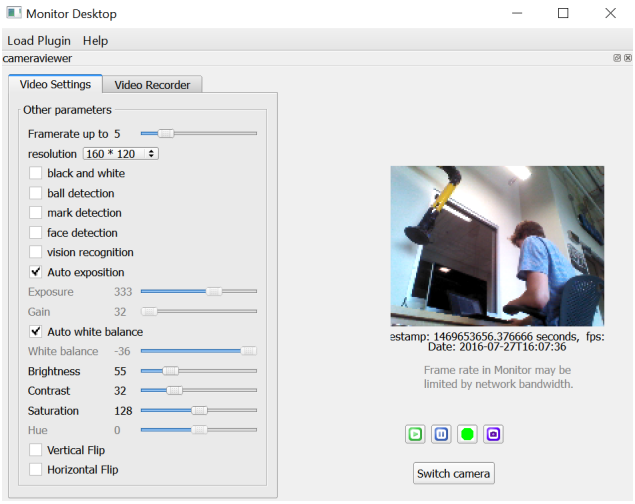

Fig. 2. Example of the NAO's camera view available through the Monitor.

D. Authentication Improvements for NAO

Independent of the mentioned mechanisms that allow a connection to NAO, a user can also connect to it through SSH. Every robot is shipped with the same (default) login name and password for communication through port 22 (SSH). The manufacturer provides a reasonable way to change the default password through the website hosted by the robot. Therefore, in this case, security is up to the prudence of the user.

However, even if the robot owners change the default password for the default user, they may not realize that an additional username can be used to access the robot: username `root`. The `root` user comes with its own default password. Since the robot owners will not necessarily be systems or cybersecurity experts, it is questionable whether they will ever think of changing the default password for `root`. By default, the `root` user cannot connect to the robot via SSH. Due to the lack of authentication mentioned in the previous Section though, one could execute a program on the robot that adds a new user, assigns root privileges to it, and enables access via SSH.

VII. The JD Operation Model

The JD humanoid's heart is the EZ-B robot controller. The EZ-B has two modes of operation: AP (access point) and client. In AP mode, the robot acts as an access point to which the user's device, such as a laptop, can be connected. The AP mode is the default. In client mode, the robot connects to a different access point like a home router.

The EZ-Robot company has designed the EZ-B protocol, which is used for communication between the robot and another user device. The programmer writes programs by installing the EZ-Builder IDE, which is available for Windows and for mobile devices. The programs are not compiled on the robot itself, but on the user's device (running Windows or a mobile OS). The translated commands are then sent to the robot for execution according to the proprietary EZ-B protocol.

A. Common Elements in Programming NAO and JD

In both the case of NAO and JD, the manufacturers provide more than one language for the users to choose and program the robot. They allow the use of at least one high level language, which is Python/C++/.NET/JAVA/Matlab/Urbi for NAO and EZ-Script for JD. According to the company, the EZ-Script language is similar to Basic and C++. They also provide the option to program in a drag and drop style that is more appealing to beginner programmers. In the case of JD, RoboScratch and Blockly are both available.

Another common characteristic is that both companies have designed their own IDE, which is the recommended way to program the robot. Choregraphe is NAO's IDE and EZ-Builder is JD's IDE. It should be noted that EZ-Builder can also be used to control platforms that do not have the EZ-B controller, such as a Sphero or an iRobot Roomba.

In order to minimize the load on the robot itself, the manufacturers suggest that users take advantage of the computing power of different devices in combination with the robot. The code can be executed remotely on a personal device such as a laptop or the manufacturer's cloud. Then, commands can be sent to the robot over the network, so that it exhibits the desired behavior. Remote Modules is the way through which this model is implemented on NAO, while JD leverages by design the computing power of another user device or the Microsoft Cognition Services. The Remote Modules model of NAO is the single model of operation for JD.

Finally, both NAO and JD have tools through which a user can check the robot status without using the IDE. In the case of NAO, this is done through the Monitor application and by remotely connecting through SSH. In the case of JD, the designers have used the Telnet protocol as we will describe in greater detail in Section VII-C.

B. Common Vulnerabilities Between NAO and JD

As explored in Section VI-A, Choregraphe does not perform any authorization check before allowing a user to connect and program the robot. It is similarly true for JD that any device running the EZ-Builder software can connect to any EZ-B controller running a robot from the EZ-Robots family. The methods through which a user can check and/or control the robot status outside the IDE do not rely on built-in authorization mechanisms. We have explored the authentication needs of the Monitor application for JD. In the next Section, we detail the lack of authentication and the degree of control one experiences with JD.

C. Controlling JD through Telnet

The most striking example of an EZ-B feature that demonstrates the company's design philosophy in regards to cybersecurity is the use of the Telnet protocol. The EZ-B processor runs by default a Telnet server allowing remote login from a machine that does not need to have any specialized software installed such as the EZ-Builder. The EZ-B design leverages Telnet as a tool to extensively monitor and control robot variables through the command line.

```
help: What you see now
version: Display hw/sw version
exit: CLI exit
scan: scan ap
wifistate: Show wifi state
ifconfig: Show IP address
arp: arp show/clean
ping: ping <ip>
dns: show/clean/<domain>
sockshow: Show all sockets
tasklist: List all thread name status
memshow: Print memory information
memdump: <addr> <length>
memset: <addr> <value 1> [<value 2> ... <value n>]
memp: Print memp list
wifidriver: Show wifi driver status
reboot: Reboot EZ-B
reset: Reset to default configuration
ugf: Start firmware upgrade
time: Show system time
flash: Flash memory map
identify: Identify EZ-B with flashing LED and Audio Beep
servo: Move a servo
servospeed: Set Servo Speed
set: Set digital port state
bs: Show Highest Buffer Sizes
```

Fig. 3. Control options for JD through Telnet.

```
#memp

Name             total used addr       size
-----------------------------------------------
RAW_PCB          4     0    0x20011760 28
UDP_PCB          8     3    0x200117d0 40
TCP_PCB          12    1    0x20011910 176
TCP_PCB_LISTEN   4     4    0x20012150 36
TCP_SEG          48    1    0x200121e0 20
NETBUF           16    0    0x200125a0 16
NETCONN          16    6    0x200126a0 52
TCPIP_MSG_API    8     0    0x200129e0 20
TCPIP_MSG_INPKT  28    0    0x20012a80 20
ARP_QUEUE        30    0    0x20012cb0 8
IGMP_GROUP       8     2    0x20012da0 20
SYS_TIMEOUT      10    6    0x20012e40 16
NETDB            1     0    0x20012ee0 308
PBUF_REF/ROM     16    0    0x20013014 16
PBUF_POOL_TX     2     0    0x20013114 1636
PBUF_POOL_RX     7     0    0x20013ddc 1636
```

Fig. 4. Heap allocation in JD's memory.

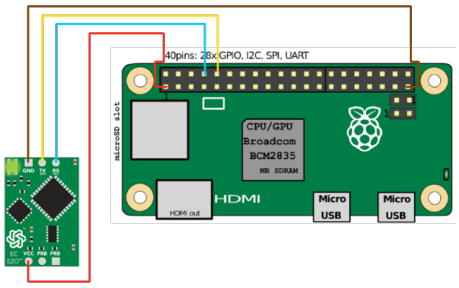

Fig. 5. Wiring Raspberry Pi Zero to the EZO Conductivity Sensor

Figure 3 lists the options available through Telnet. The user can change the position and speed of each of the robot's servos. The user can find out the IP and MAC addresses of the robot, observe and/or modify the IP address of the DNS server, scan for available access points within range, view the list of sockets. The `tasklist` command provides information about the threads running on the robot, such as the TCP server of the robot's camera. The available options also include important operations related to the robot's memory. Figure 4 shows the result of the `memp` command, which lists the amount of memory allocated on the heap for various dynamic structures together with the start address of the corresponding memory segment. Using the `memdump` option, the user can observe the contents of any memory address, while the user can also modify memory contents through `memset`. Apart from gaining physical control of the robot, the Telnet options provide a detailed picture of the robot's status and the ability to modify network and memory features.

The Telnet protocol is known for its lack of data encryption and has been widely replaced by SSH. More importantly, a user that has the right IP address and port number can connect to an EZ-B robot through Telnet without any authentication. Obtaining the robot's IP address and port number is not hard for a device attached to the same local network as the robot. For a device outside the robot's network, the only added complication is that the router attached to the robot's network will often perform Network Address Translation (NAT). However, this complication can be overcome with techniques that identify hosts behind NAT boxes [22].

VIII. Smart Homes

It is important to view the cybersecurity vulnerabilities in the design of humanoids and other robots in a broader context. In a DIY smart home setup, a robot enthusiast could use NAO, JD or an EZ-B controlled custom robot as the central controller of Internet of Things (IoT) enabled devices. As an example, we connected a Raspberry Pi Zero to an EZO conductivity sensor as shown in Figure 5 and then sent the sensor readings to the IoT cloud platform Ubidots. A robot such as JD, pictured in Figure 6, could be regularly downloading the readings from the cloud. In a different setup, the robot acting as a central controller of the smart home devices could be communicating directly with the IoT enabled sensors. In the event of abnormal sensor measurements, the robot could notify the user of the alarming situation (e.g. water in a house basement diagnosed by a conductivity sensor). The implications of weak cybersecurity in social robots can quickly escalate to impact the cybersecurity of systems with a vital role in a smart home.

IX. Industry Response

We communicated our concerns to both manufacturers and we have summarized their responses. The SoftBank engineer who responded to our inquiry suggested that NAO is an educational and research platform for which these cybersecurity limitations are known to the company. The engineer also suggested that the issues have been addressed in more

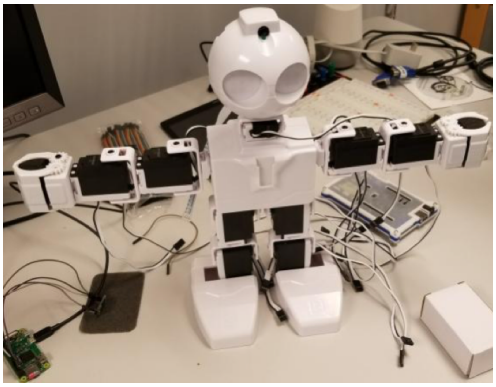

Fig. 6. JD next to the Raspberry Pi, which is connected to the conductivity sensor (bottom left).

advanced SoftBank robots. We have not had the opportunity to verify whether Pepper, the more advanced SoftBank robot, or even the sixth version of NAO have a different cybersecurity profile. We believe it is important to enhance the cybersecurity of these robots, since their use is being expanded in all sectors: education, research and industries such as healthcare.

The EZ-Robot founder and CEO responded that there are no security vulnerabilities in JD [23]. The rational behind his claim is that access to the robot is left completely open intentionally. If the design included authorization mechanisms, and those were easy to circumvent, then there would be potential vulnerabilities. However, since access to the robot is left open by design, the claim is that there are no cybersecurity vulnerabilities. We believe that deciding to leave the system open is by itself a cybersecurity vulnerability. Since EZ-Robot's target audience includes individuals who are likely to be unaware of cybersecurity risks, such as teenagers, we believe that the design should provide some default cybersecurity protections.

X. Conclusion

The balance between cybersecurity and usability when designing a product is not easy to achieve. However, the increasing attacks on IoT devices [24], [25] leave us no other choice for the safety and security of consumers. In this paper, we have analyzed the cybersecurity profile of two popular humanoids and made recommendations to guard against cybersecurity vulnerabilities in artificially intelligent humanoid robots.

Acknowledgment

This material is partially based upon work supported by the National Science Foundation under Grants No. ACI-1461264 and 1657548. Justin Miller was supported through a Marquette University Summer Honors Research Fellowship. We would also like to thank the reviewers and especially our shepherd, Vincent Toubiana, for their insightful comments.

References

[1] T. Denning, C. Matuszek, K. Koscher, J. R. Smith, and T. Kohno, "A spotlight on security and privacy risks with future household robots: Attacks and lessons," in *Proceedings of the 11th International Conference on Ubiquitous Computing*, ser. UbiComp '09. New York, NY, USA: ACM, 2009, pp. 105–114. [Online]. Available: http://doi.acm.org/10.1145/1620545.1620564

[2] J. Miller, A. B. Williams, and D. Perouli, "A Case Study on the Cybersecurity of Social Robots," in *Companion of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '18. New York, NY, USA: ACM, 2018, pp. 195–196. [Online]. Available: http://doi.acm.org/10.1145/3173386.3177078

[3] N. DeMarinis, S. Tellex, V. Kemerlis, G. Konidaris, and R. Fonseca, "Scanning the Internet for ROS: A View of Security in Robotics Research," 2018.

[4] Wired, Andy Greenberg, "Watch hackers hijack three robots for spying and sabotage," https://www.wired.com/story/watch-robot-hacks-spy-sabotage/ Last Accessed on November 4th, 2018.

[5] R. Balebako, J. Jung, W. Lu, L. F. Cranor, and C. Nguyen, "Little brothers watching you: Raising awareness of data leaks on smartphones," in *Proceedings of the Ninth Symposium on Usable Privacy and Security*. ACM, 2013, p. 12.

[6] R. Kang, L. Dabbish, N. Fruchter, and S. Kiesler, ""my data just goes everywhere:" user mental models of the internet and implications for privacy and security," in *Eleventh Symposium On Usable Privacy and Security (SOUPS 2015)*. Ottawa: USENIX Association, 2015, pp. 39–52. [Online]. Available: https://www.usenix.org/conference/soups2015/proceedings/presentation/kang

[7] E. Chin, A. P. Felt, V. Sekar, and D. Wagner, "Measuring user confidence in smartphone security and privacy," in *Proceedings of the eighth symposium on usable privacy and security*. ACM, 2012, p. 1.

[8] S. Booth, J. Tompkin, H. Pfister, J. Waldo, K. Gajos, and R. Nagpal, "Piggybacking robots: Human-robot overtrust in university dormitory security," in *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '17. New York, NY, USA: ACM, 2017, pp. 426–434. [Online]. Available: http://doi.acm.org/10.1145/2909824.3020211

[9] M. Finnicum and S. T. King, "Building secure robot applications," in *Proceedings of the 6th USENIX Conference on Hot Topics in Security*, ser. HotSec'11. Berkeley, CA, USA: USENIX Association, 2011, pp. 1–1. [Online]. Available: http://dl.acm.org/citation.cfm?id=2028040.2028041

[10] W. Zheng, A. Dave, J. G. Beekman, R. A. Popa, J. E. Gonzalez, and I. Stoica, "Opaque: An oblivious and encrypted distributed analytics platform." in *NSDI*, 2017, pp. 283–298.

[11] O. Ohrimenko, M. Costa, C. Fournet, C. Gkantsidis, M. Kohlweiss, and D. Sharma, "Observing and preventing leakage in mapreduce," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 1570–1581.

[12] Y. Xiao, X. Zhang, Y. Zhang, and R. Teodorescu, "One bit flips, one cloud flops: Cross-vm row hammer attacks and privilege escalation." in *USENIX Security Symposium*, 2016, pp. 19–35.

[13] M.-K. Yoon, B. Liu, N. Hovakimyan, and L. Sha, "Virtualdrone: virtual sensing, actuation, and communication for attack-resilient unmanned aerial systems," in *Proceedings of the 8th International Conference on Cyber-Physical Systems*. ACM, 2017, pp. 143–154.

[14] J.-S. Pleban, R. Band, and R. Creutzburg, "Hacking and securing the ar. drone 2.0 quadcopter: investigations for improving the security of a toy," in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 2014, pp. 90 300L–90 300L.

[15] T. Yu, V. Sekar, S. Seshan, Y. Agarwal, and C. Xu, "Handling a trillion (unfixable) flaws on a billion devices: Rethinking network security for the internet-of-things," in *Proceedings of the 14th ACM Workshop on Hot Topics in Networks*. ACM, 2015, p. 5.

[16] D. Albani, A. Youssef, V. Suriani, D. Nardi, and D. D. Bloisi, "A deep learning approach for object recognition with nao soccer robots," in *Robot World Cup*. Springer, 2016, pp. 392–403.

[17] I. Nuse. Humanoid robot takes over as a teacher. [Online]. Available: http://sciencenordic.com/humanoid-robot-takes-over-teacher

[18] J. Samaniego. 20 nao robots perform synchronized dance. [Online]. Available: https://www.pcworld.com/article/199835/sdfgh.html

[19] NAO v6, "https://www.softbankrobotics.com/emea/en/nao," Last Accessed on November 4th, 2018.

[20] The NAOqi Framework, "http://doc.aldebaran.com/2-1/dev/naoqi/index.html," Last Accessed on November 4th, 2018.
[21] Wireshark, "Network protocol analyzer," [Computer Software], 2018. [Online]. Available: https://www.wireshark.org
[22] H. Park, S. Shin, B. Roh, and C. Lee, "Identification of Hosts Behind a NAT Device Utilizing Multiple Fields of IP and TCP," in *2016 International Conference on Information and Communication Technology Convergence (ICTC)*, Oct 2016, pp. 484–486.
[23] EZRobotForum. Cybersecurity questions. [Online]. Available: http://www.ez-robot.com/Community/Forum/Thread?threadId=11737
[24] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the Mirai Botnet," in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, 2017, pp. 1093–1110. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis
[25] E. Ronen, A. Shamir, A. Weingarten, and C. O'Flynn, "IoT Goes Nuclear: Creating a ZigBee Chain Reaction," in *2017 IEEE Symposium on Security and Privacy (SP)*, May 2017, pp. 195–212.