# A Study in Securing the Internet of Things

Joey Tuttobene

Point Loma Nazarene University

Honors Research Project

Committee: Mike Leih, Ph.D. (Advisor); Macy Dennis; Benjamin Mood, Ph.D.

April 18, 2020

## Abstract

Current Internet of Things (IoT) cybersecurity threats were investigated using current literature and direct examination of IoT devices including the 3D printer control software OctoPrint, the autonomous vehicle middleware MOOS-IvP, and a common consumer smart lightbulb. This research found that most IoT devices transmit in plaintext and that software controls alone are insufficient in control systems applications. Recommended best practices for organizations include segregating IoT devices on a network, using non-default credentials, and keeping devices up to date. Manufacturers need to include robust encryption and authentication of messages with their offerings, as well as provide methods to update devices easily and safely. Finally, physical fail-safes can complement imperfect software in IoT control system applications.

*Keywords:* Internet of Things, IoT, cybersecurity, network, encryption, fail-safe

## Contents

# A Study in Securing the Internet of Things

## Introduction

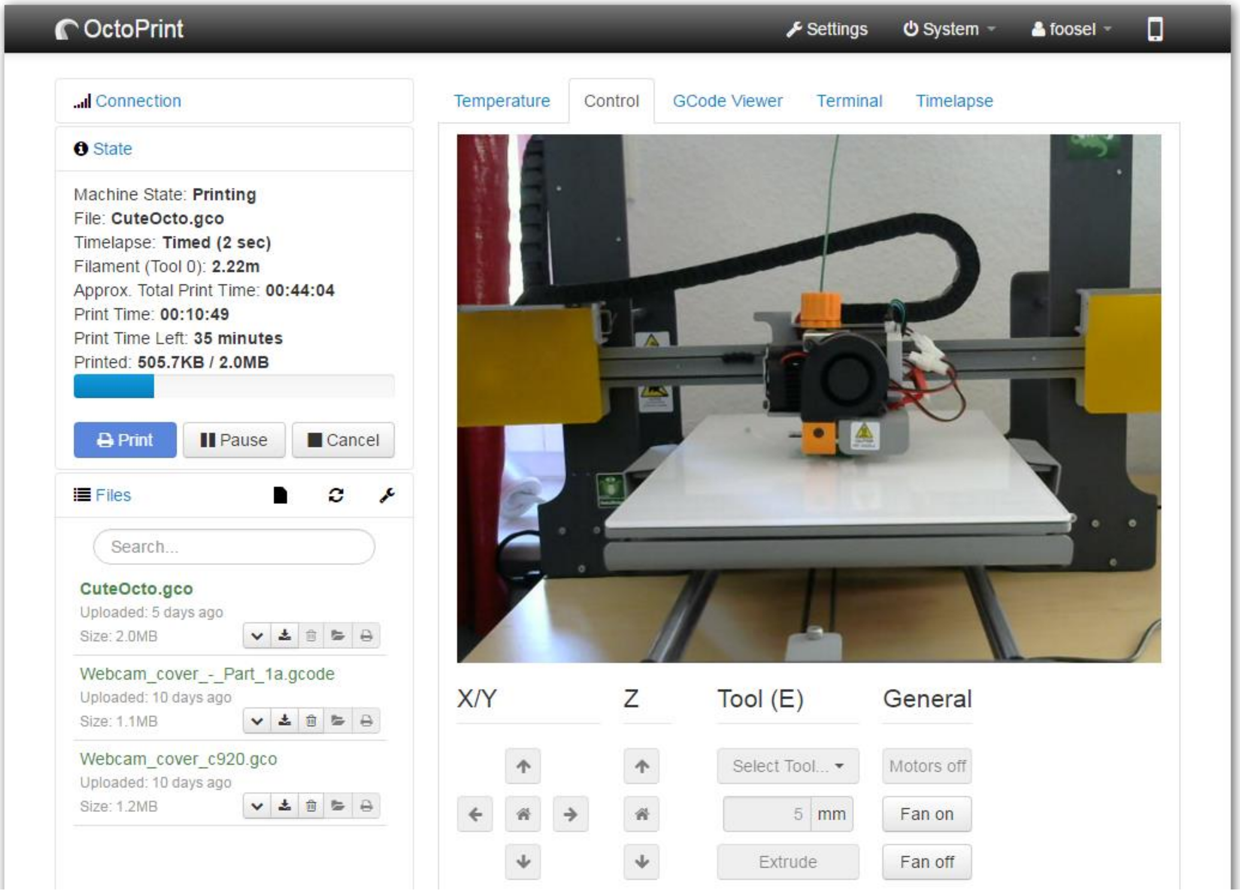
The Internet of Things is the networking of physical objects that are embedded with electronics, software, and sensors, which enables the collection and exchange of data between non-traditional computing devices (Feng, et al., 2019). Unlike a traditional computer, an Internet of Things (IoT) device is typically embedded into a physical object and interacts with the physical world through sensors and actuators that can be accessed over the internet. These devices can range from an internet-controlled thermostat to a self-driving car. Traditional computers and servers have received many security upgrades through decades of development and software improvements (Mohan, 2019). In contrast, IoT devices often do not receive security updates or have complicated patching processes (Feng, et al., 2019). IoT security still lags behind conventional computing system security (Feng, et al., 2019). Because the Internet of Things is both growing rapidly and relatively insecure, IoT devices provide a broad attack surface that cyber criminals can exploit to gain access to sensitive information, disrupt communications, or cause physical destruction (Huang, Cardenas, & Baldick, 2019). The field of cybersecurity aims to combat this threat by protecting IoT systems as best as possible without compromising their usefulness.

The security of the Internet of Things is improving, but IoT devices are still vulnerable, primarily in terms of data integrity and confidentiality. Organizations can protect their critical data by segregating IoT devices from their main networks and using network controls to enforce security policy. They can also make sure to update their fleet and use unique, non-default credentials. Manufacturers can improve their devices by providing a way to update and maintain devices as vulnerabilities are found. They need to encrypt traffic and verify the integrity and sender of communications. Physical devices should also incorporate physical fail-safes. Based on these aspects of IoT security needs, this paper will argue for the implementation of best practices to provide IoT device security updates, message encryption, and network segregation of IoT devices, as well as the integration of physical fail safes for IoT devices in control system applications.

## Research Goals and Methods

The goal of this research was to gain insight into current inadequacies in IoT devices and to investigate general best practices for securely using and architecting IoT systems. The intention was to outline and examine potential flaws with these systems and pose best practices and theoretical mitigations. This research was exploratory and aimed to further define the problem of IoT security using an experimental and deductive approach. This research methodology was appropriate because the intention was to investigate the problems surrounding IoT security using available literature and experimentation with a representative sample of IoT devices (Saunders, Lewis, & Thornhill, 2012). The open-source 3D printer control platform OctoPrint, the open-source robotics autonomy package MOOS-IvP, and the SAUDIO Smart LED Bulb were directly investigated. These systems make up a reasonably representative sample of the Internet of Things and provide security insight on robotics, autonomous vehicles, connected industrial systems, and consumer devices. Overall risks, including current practical and theoretical concerns, are categorized for each device class, and current best practices to mitigate these risks are discussed.

## OctoPrint



Machine State: **Printing**
File: **CuteOcto.gco**
Timelapse: **Timed (2 sec)**
Filament (Tool 0): **2.22m**
Approx. Total Print Time: **00:44:04**
Print Time: **00:10:49**
Print Time Left: **35 minutes**
Printed: **505.7KB / 2.0MB**

**CuteOcto.gco**
Uploaded: 5 days ago
Size: 2.0MB

Webcam_cover_-_Part_1a.gcode
Uploaded: 10 days ago
Size: 1.1MB

Webcam_cover_c920.gco
Uploaded: 10 days ago
Size: 1.2MB

OctoPrint is a full remote monitoring and control solution for 3D printers (Häußge , n.d.). OctoPrint was selected as it shares many similarities with connected industrial control systems in terms of capabilities. OctoPrint provides a web interface for a 3D printer backend, allowing a user to control a 3D printer remotely (Häußge , n.d.). This control includes uploading firmware, starting and stopping prints, viewing a webcam feed, and setting heating element temperatures (Häußge , n.d.). While this is intended to be used in a trusted local network setting, port forwarding can be used to make an OctoPrint instance accessible over the public internet. In August of 2018, the IoT search engine Shodan showed that over 3,700 OctoPrint instances were accessible over the Internet (Mertens, 2018).

The test setup was a Raspberry Pi 3B running OctoPi 0.17.0, which runs OctoPrint version 1.3.12. This instance was connected to RepRap Arduino Mega Pololu Shield (RAMPS) electronics scavenged from a Marlin-based printer, which is a fairly typical electronics and firmware setup for 3D printers. Traffic was captured within Wireshark, and CPU usage on the Pi was visually monitored with the htop utility. The webserver runs on port 80. Traffic is unencrypted and can be sniffed by others on the network. In versions prior to 1.3.6 there was a vulnerability in the access control login mechanism, but this has been fixed according to the developers (Häußge , n.d.). Passwords are now authenticated by a JavaScript script, and the username and password were not recovered in plaintext in any of the TCP or HTTP streams. When OctoPrint runs for the first time, it prompts the user to set up access controls, including creation of an account so that only authenticated users can access the printer. The access controls can be disabled, but users are first warned of major risks if the service is visible to untrusted networks such as the Internet.

One additional finding was that the service appears to run in a single thread. Because low-power computers such as the Raspberry Pi are popular in OctoPrint setups, additional traffic to the webserver can easily consume much of the CPU time. This could disrupt a print by preventing OctoPrint from sending new commands to the printer. Theoretically, continued heavy traffic could keep the heaters on longer than intended. This would be easily fixed by running printer communications in a separate thread from the web control interface.

OctoPrint and its developers seriously warn users of the risks of putting 3D printers on the open internet. Once an OctoPrint instance has been compromised, an attacker can steal intellectual property sitting on the printer and potentially start a fire with the heaters (Jubaleth, 2018). Alternative remote access solutions, such as VPNs, reverse proxies, and cloud-based plugins that typically have better security are suggested as better alternatives (Jubaleth, 2018). These measures are potentially inadequate. Once an attacker has admin access to an OctoPrint instance, they can bypass most of the software safety measures of the 3D printer. The attacker could upload firmware without heater control safeguards to the printer, which would allow them to run the printer's heating elements to high temperatures for a long time to start fires (Jubaleth, 2018). Besides fire, temperatures above 250 °C can cause the polytetrafluoroethylene (PTFE) commonly used in printer filament guide tubes to break down and release highly toxic gases, which can be deadly in an ill-ventilated area (Capricorn Bowden Tubes, 2018). This off-gassing effect only increases as temperature increases (Flouride Action Network).

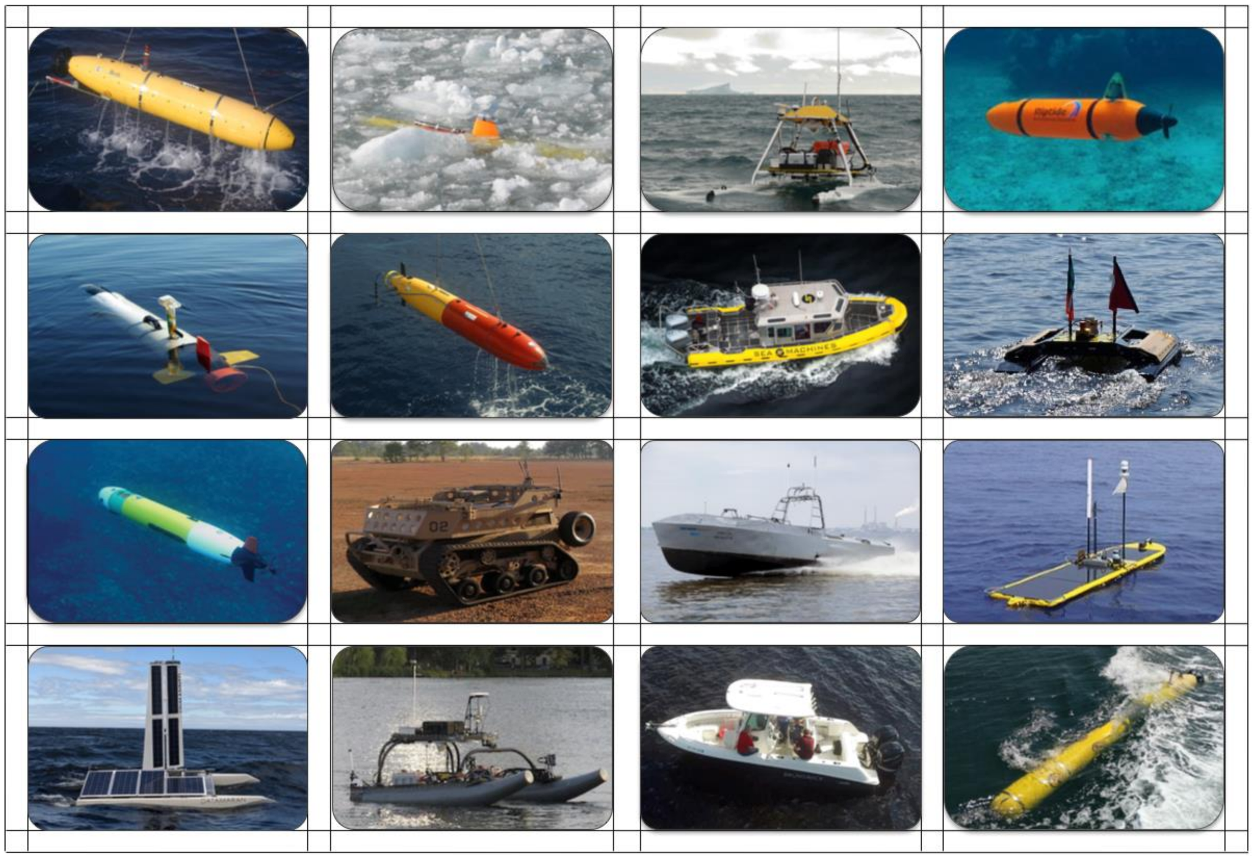
For increased safety, this paper proposes a multi-layered defense approach for OctoPrint users. Firstly, any OctoPrint-enabled 3D printer should not be publicly accessible over an insecure network, and it should instead be accessed through a VPN or a reverse proxy that can encrypt the traffic between the user and the printer. Secondly, a physical fail-safe needs to be added to the printer. This could be a fully analog thermostat that can interrupt power to the heater if a threshold is exceeded, or even a simple thermal fuse. Most 3D printer manufacturers do not include such safety features. MakerBot was a rare exception. In 2011, they created a normally-closed thermostat safety cutoff switch that sat between the heater and its power source to cut power in a failure scenario (MakerBot, 2011). Such a system is well worth the slightly added cost because it provides a second layer of physical safety that both mitigates the malicious firmware risk of a runaway heater and improves safety under normal conditions.

## Summary of Recommendations

- Add physical fail-safe mechanisms to potentially dangerous equipment
- Encrypt network traffic

- Make the webserver run on a separate thread from printer control
- Do not make devices visible to the open internet
- Enable access controls with unique, strong passwords

**MOOS-IvP**



MOOS-IvP (Mission Oriented Operating Suite – Interval Programming) is an open source, behavior-based autonomy framework built with a publish-subscribe model (Benjamin, Schmidt, & Newman, 2019). Essentially, this is C++ middleware intended for use in robotics and autonomous vehicles. MOOS-IvP includes a central database that various concurrent MOOS applications publish updates to and receive updates of subscribed variables. Autonomous decision-making is done by IvP Helm, which is a MOOS application that utilizes interval programming to select optimal behaviors based on mission parameters and current inputs

(Benjamin, Schmidt, & Newman, 2019). Groups of MOOS applications centered around one database are a MOOS community, and different MOOS communities can communicate with each other (Benjamin, Schmidt, & Newman, 2019).

MOOS-IvP was selected for its applications in autonomous vehicles and robotics. The similar open-source behavior-based framework Robot Operating System (ROS) has security analyses (Priyadarshini, 2017). However, there are no current literature analyses of MOOS-IvP's cyber security weaknesses. As robots and autonomous vehicles become more connected and pervasive, it is important to study and improve their security.

The MOOS-IvP test setup was a Kubuntu virtual machine with MOOS-IvP 17.7.2 and Wireshark. A virtual mission with two vehicles and a base station (mission M2 Berta) was run. In this mission, the vehicles are directed by the base station and share data with the base station. The resulting network traffic was captured and analyzed. All traffic sent between the MOOS applications was unencrypted TCP traffic. This includes loopback traffic sent within a single MOOS community as applications communicate with the central database, as well as traffic sent to different MOOS communities. While an adversary generally will not have access to loopback traffic, they can easily eavesdrop on messages between MOOS communities if they can get access to the network. An adversary may also be able to perform an injection attack once on the network because MOOS applications, including the MOOS database, do not verify message sources. This is partially by design for troubleshooting purposes, as a researcher can edit a control variable directly to see how a system responds, but this open architecture provides a potential vulnerability for attackers. Onboard the autonomous system, a MOOS mission is a text file that lists applications and application parameters to launch. Anyone who can edit this file can modify applications launched and application parameters. This is again by design, and the host OS needs to have proper file controls to prevent unauthorized changes.

MOOS-IvP effectively offloads security onto the users, but this allows for the users to consider their own threat models. The host OS needs to provide the necessary file protections to prevent an attacker from modifying the applications the robot runs, and it needs to properly protect traffic on the loopback adapter. The channels on which multiple MOOS communities communicate need to be encrypted in order to prevent eavesdropping and potential injection attacks. This is up to the users to properly configure. A Wi-Fi network with WPA2 partially protects the autonomous systems' communications from unauthorized access, but WPA2 itself is broken (Vanhoef & Piessens, 2017). MOOS-IvP is geared towards autonomous surface and underwater vehicles, and encryption in the acoustic networks such vehicles must use is harder to achieve due to limited bandwidth and signal strength. A lightweight cryptography and authentication system called Intervehicle Secure MOOS (IS-MOOS) encrypts and authenticates messages sent between vehicles over an acoustic network (Caiti, Calabro, Dini, Duca, & Munafo, 2012). Developers of autonomous systems are left to determine their own threat model and harden their systems accordingly. Major considerations for MOOS-IvP users include plaintext network communications, lack of message authentication and integrity, and proper host OS patching and configuration.

**Summary of Recommendations**

- Add mechanisms to easily verify which process sent which messages
- Add mechanisms to easily verify messages have not been modified
- Encrypt network traffic
- Properly configure permissions on the host OS and keep it up to date

## SAUDIO Smart Lightbulb

Smart household devices are increasingly common in the forms of smart speakers, doorbell cameras, and connected lights. The SAUDIO Smart Lightbulb is a generic 2.4GHz Wi-Fi enabled lightbulb that can be controlled from an app. This was selected because it is representative of common consumer-grade IoT devices. Additionally, other smart lightbulbs have been targeted before. The Philips Hue lightbulbs were targeted in a hack that exploited a vulnerability in the ZigBee wireless protocol and a weakness in Philips' encryption to create a malicious firmware payload (Ricker, 2016). In some cases, common smart lightbulbs including those made by LIFX and Philips can be used as a vector of information exfiltration, where a compromised device controls the smart light to send information out of the network (Maiti & Jadliwala, 2019).

Two SAUDIO lightbulbs were analyzed on a test Wi-Fi network secured with WPA2. NMAP was used to port scan the devices, and Wireshark was used for packet capture and inspection. Because the bulbs are controlled from an app (Smart Life), an Android device with both a packet capture app and Smart Life installed was used to investigate app traffic. Both lightbulbs were running Wi-Fi Module firmware version 1.5.0 and MCU Module version 1.5.0.

The SAUDIO lights performed well. Traffic appears to be properly encrypted with TLS from the app to the backend servers, and from the servers to the bulbs. The packets were different each time a repeated command was sent, suggesting that a nonce was used. The backend was a set of Amazon Web Services EC2 servers in the US West 2 region. The app itself provides a way to update the lightbulb's firmware. An NMAP scan showed that only two ports were not closed on a bulb: TCP port 6668 was open, and UDP port 49154 was open but filtered. The TCP port is used for control purposes. The UDP port is used to broadcast a heartbeat message which goes to the AWS backend. The heartbeat messages are unique to each bulb, but the messages are the same. It should be possible to spoof this heartbeat, but there is limited utility in doing so. Additionally, NMAP guesses that the OS is related to the Philips Bridge IwIP v1.4.0 with 93% confidence. These bulbs do not operate with a hub, but it is possible that its networking module and real-time OS are similar or identical to those in the hub that Philips makes to control the ZigBee-based Hue bulbs.

**Summary of Recommendations**

- Continue to keep bulb firmware, control app, and backend servers up to date

## Best Practices

Best practices for using and improving the Internet of Things vary, and many critical aspects are under vendor control. Based on this research, IoT devices often lack message encryption and authentication, which allows for eavesdropping and injection attacks. A 2019 study determined that only 18% of devices used SSL exclusively, while 41% of devices communicated entirely in plaintext; the remaining 41% encrypted some, but not all, transmissions (Zscaler ThreatLabZ, 2019). A staggering 91.5% of IoT traffic occurred in plaintext in 2019 (Zscaler ThreatLabZ, 2019). Proper implementation of robust encryption is necessary and should help maintain the integrity of messages while also verifying that they come from a trusted source.

Additionally, weak authentication mechanisms such as hardcoded passwords make IoT devices easy targets for botnet recruitment (Kan, 2016). Manufacturers should eliminate easily guessable default passwords as well as remove hardcoded passwords from devices like DVRs that do not need them. In addition to removing hardcoded passwords, manufacturers need to provide a safe way to update firmware when vulnerabilities are found (Mohan, 2019). They must take care to prevent downloading of malicious firmware from becoming another attack vector, as it was for Philips Hue bulbs (Ricker, 2016). Finally, many IoT devices are physically insecure if a Universal Asynchronous Receiver Transmitter (UART) is left enabled on the device. UARTs can potentially be used to create a serial connection with root access, modify and reverse-engineer the firmware, and dump the shadow file to crack default passwords (Heiland, 2018). From a manufacturer's point of view, their firmware is proprietary and should be protected (Heiland, 2018). Because a UART often provides the keys to the kingdom to those with physical access, UARTs should be disabled and removed in production.

Enterprises will want to manage their fleet of devices, and this can quickly become a significant amount of work. Over eighty different authentication schemes have been proposed or implemented for IoT devices (Lemos, 2020). Such a variety makes configuration mistakes easier and device management more difficult while also providing a wider attack surface (Lemos, 2020). Cloud services such as AWS, Azure, and GCP are attempting to solve that problem by creating standard processes for authenticating IoT devices (Lemos, 2020). Standardization of IoT authentication protocols could lead to easier and automated fleet management, reduce the overall attack surface, and reduce the burden on manufacturers to create their own authentication schemes.

While manufacturers need to improve the security of their products, organizations that utilize IoT technology can still take several steps to protect their devices and their data. Critically, because IoT devices are low-power and cannot support more traditional endpoint detection and response solutions, security must start with network controls (Das, 2019). Proper network segmentation is important because it makes it more difficult for adversaries to move laterally from a compromised IoT device into more sensitive parts of the network (Mohan, 2019). The network can enforce restrictions on inbound and outbound traffic to help hide IoT devices from hackers and limit compromised devices' capability to participate in botnets like Mirai (Mohan, 2019).

Beyond network controls, continuous monitoring of device behavior on the network is important. Organizations need to know what devices they have on their networks and what normal behavior looks like; this is a problem machine learning threat detection can help solve (Lanowitz, 2020). A zero-trust model of continuously monitoring IoT devices can help quickly identify compromised devices (Lanowitz, 2020). Finally, devices with out-of-date software are easy targets (Wallen, 2017). While downtime for updates can be expensive, downtime due to compromise can be more expensive. Organizations need to update IoT devices when security patches are released, while manufacturers need to provide safe and easy ways to patch vulnerabilities (Dunlap, 2019).

Finally, when possible, physical fail-safes can be added or enabled to make IoT devices in industrial applications more robust in the event of a cyberattack. In applications that control potentially dangerous or energy-intensive equipment, having fail-safes makes sense. Physical fail-safes can help in situations where software safeguards are insufficient. OctoPrint shows that software controls are possible to remotely circumvent. Attacks on industrial equipment like

Stuxnet can cause physical destruction (Wallen, 2017). Compromise of individual HVAC systems could be used to physically damage equipment or goods in temperature-critical rooms (Forescout, 2016). IoT HVAC equipment on a larger scale could also be used to attack power grids (Huang, Cardenas, & Baldick, 2019). Power grids have physical protections to help prevent damage to generation and transmission equipment without a major blackout (Huang, Cardenas, & Baldick, 2019). Physical fail-safes for HVAC applications include analog, hard-limited thermostats and thermal fuses. For industrial robots and industrial IoT, limit switches could be implemented in hardware rather than in software controls, and the electronics could enforce safe power limits in hardware to prevent physical damage. The inclusion of physical fail-safes in IoT devices provides a multi-layered security approach that can improve user safety in general and protect equipment from software-based attacks.

## Summary of Recommendations

- Encrypt network traffic
- Enforce strong, unique, and mutable passwords
- Provide a safe and easy way to patch devices as vulnerabilities are discovered
- Remove/disable unnecessary UARTs on production equipment
- Segment the network to separate IoT devices from the main network
- Use network controls to enforce policy regarding connections and traffic
- Inventory, monitor, and update IoT devices within the organization
- Use physical fail-safes with potentially dangerous or energy-intensive equipment

## Conclusion

OctoPrint’s developers take security seriously and attempt to advise and steer their users into secure usage. However, in a worst-case scenario, OctoPrint’s safety controls and 3D printers’ firmware safety controls are inadequate. A multi-layered security approach including physical fail-safe technology on the 3D printer, proper access controls in OctoPrint, and controlled access through a VPN or reverse proxy can mitigate this threat. In less extreme cases, an adversary can potentially eavesdrop on traffic to steal intellectual property or sabotage prints. Encrypting traffic and using separate threads to handle printing and networking are potential solutions.

MOOS-IvP is very open by design so that researchers can develop vehicle autonomy easily for a variety of vehicles and applications. However, that openness comes at a cost. Security is offloaded onto the user, who must run MOOS-IvP on an up-to-date, secure operating system, ensure proper authentication on the network, and provide their own encryption for communications between multiple vehicles.

Smart lightbulbs vary in security. In the case of the SAUDIO bulbs, connecting with a cloud backend individually over WPA2 Wi-Fi is fairly secure, especially when all commands are TLS-encrypted. The devices can be updated through the app, which also uses TLS to communicate with the backend. The bulbs do not work in a mesh network, protecting them from the sort of attacks that compromised Hue bulbs before (Ricker, 2016). While an adversary with persistent physical access may be able to compromise one bulb and steal Wi-Fi credentials, the devices themselves appear fairly robust against remote exploit compared to other smart lightbulbs.

Overall, this research has found that IoT data confidentiality is generally more limited than in traditional computing systems due to an acute lack of message encryption. Most of the systems tested sent traffic in cleartext, so properly encrypting traffic is an important next step in protecting the IoT. Proper network design to segregate these devices from sensitive networks and enforcing possible connections through network-based security controls are both vital because IoT devices cannot support robust endpoint protection. Additionally, many devices are at least theoretically vulnerable to injection attacks, which is a dangerous possibility for connected, self-driving cars or industrial equipment. Physical fail-safes in such scenarios could potentially mitigate the risk of catastrophic failure due to software hacks. These conclusions are not comprehensive of every IoT device, but they do highlight key factors for manufacturers and organizations to consider in creating and deploying IoT systems.

## References

Benjamin, M., Schmidt, H., & Newman, P. (2019, August 6). *An Overview of MOOS-IvP and a Users Guide to the IvP Helm - Release 19.8*. Retrieved from MOOS-IvP Documentation: https://oceanai.mit.edu/ivpman/pmwiki/pmwiki.php?n=Helm.Cover

Caiti, A., Calabro, V., Dini, G., Duca, A. L., & Munafo, A. (2012). MOOS MIDDLEWARE AND NODE ADAPTIVITY IN UNDERWATER SENSOR NETWORKS: RESULTS FROM THE UAN11 SEA TRIAL. *Proceedings of the 11th European Conference on Underwater Acoustics*.

Capricorn Bowden Tubes. (2018). *Safe Printing Temperatures.* Retrieved from Capricorn Premium PTFE Bowden Tubing: https://www.captubes.com/safety.html

Das, S. (2019, February 4). *IoT Security's Coming of Age is Overdue*. Retrieved from Dark Reading: https://www.darkreading.com/attacks-breaches/iot-securitys-coming-of-age-is-overdue/a/d-id/1333756

Dudek, W., & Szynkiewicz, W. (2019). Cyber-security for Mobile Service Robots - Challenges for Cyber-physical System Safety. *Journal of Telecommunications and Information Technology*, 32.

Dunlap, T. (2019). Unsecured IoT: 8 Ways Hackers Exploit Firmware Vulnerabilities. *Dark Reading*.

Feng, X., Liao, X., Wang, X., Wang, H., Li, Q., Yang, K., . . . Sun, L. (2019). Understanding and Securing Device Vulnerabilities through Automated Bug Report Analysis . *USENIX Security Symposium*.

Flouride Action Network. (n.d.). *Thermal Decomposition Products of Teflon.* Retrieved from Flouride Action Network: https://www.fluoridealert.org/wp-content/pesticides/teflon.decomposition.prod.htm

Forescout. (2016). *IoT Enterprise Risk Report.* Retrieved from Forescout: https://www.forescout.com/company/resources/iot-enterprise-risk-report/

Green, J. (2019). Why the Network Is Central to IoT Security. *Dark Reading*.

Häußge , G. (n.d.). *OctoPrint*. Retrieved from OctoPrint: https://octoprint.org/

Heiland, D. (2018, June 19). *Security Impact of Easily Accessible UART on IoT Technology*. Retrieved from Rapid7: https://blog.rapid7.com/2018/06/19/security-impact-of-easily-accessible-uart-on-iot-technology/

Huang, B., Cardenas, A. A., & Baldick, R. (2019). Not Everything is Dark and Gloomy: Power Grid Protections Against IoT Demand Attacks. *USENIX Security Symposium*.

Jubaleth. (2018, September 3). *A Guide To Safe Remote Access of OctoPrint*. Retrieved from OctoPrint.org: https://octoprint.org/blog/2018/09/03/safe-remote-access/

Kan, M. (2016, October 4). *IoT Botnet Highlights the Dangers of Default Passwords*. Retrieved from CSO: https://www.csoonline.com/article/3127263/iot-botnet-highlights-the-dangers-of-default-passwords.html

Lanowitz, T. (2020, January 30). *How To Secure Your IoT Ecosystem in the Age of 5G*. Retrieved from Dark Reading: https://www.darkreading.com/risk/how-to-secure-your-iot-ecosystem-in-the-age-of-5g/a/d-id/1336879

Lemos, R. (2020, February 2). *Babel of IoT Authentication Poses Security Challenges*. Retrieved from Dark Reading: https://www.darkreading.com/theedge/babel-of-iot-authentication-poses-security-challenges/b/d-id/1337049

Maiti, A., & Jadliwala, M. (2019). Light Ears: Information Leakage via Smart Lights. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*.

MakerBot. (2011, April 28). *MakerBot Safety Cutoff Switch Kit, Rev B.* Retrieved from Thingiverse: https://www.thingiverse.com/thing:8108

Mertens, X. (2018, August 30). *3D Printers in The Wild, What Can Go Wrong?* Retrieved from SANS ISC InfoSec Forums: https://isc.sans.edu/forums/diary/3D+Printers+in+The+Wild+What+Can+Go+Wrong/24044/

Mohan, P. (2019, August 12). *6 Security Considerations for Wrangling IoT*. Retrieved from https://www.darkreading.com/endpoint/6-security-considerations-for-wrangling-iot/a/d-id/1335411

Priyadarshini, I. (2017). Cybersecurity Risks in Robotics.

Ricker, T. (2016, November 3). *Watch a Drone Hack a Room Full of Smart Lightbulbs From Outside the Window*. Retrieved from The Verge: https://www.theverge.com/2016/11/3/13507126/iot-drone-hack

Saunders, M., Lewis, P., & Thornhill, A. (2012). *Research Methods for Business Students.* Pearson Education Limited.

Vanhoef, M., & Piessens, F. (2017). Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*.

Wallen, J. (2017, June 1). *Five nightmarish attacks that show the risks of IoT security*. Retrieved from ZDNet: https://www.zdnet.com/article/5-nightmarish-attacks-that-show-the-risks-of-iot-security/

Zscaler ThreatLabZ. (2019, May). *IoT in the Enterprise: An analysis of traffic and threats.* Retrieved from Zscaler: https://www.zscaler.com/resources/industry-reports/IoT-in-the-enterprise-2019.pdf