U.S. DEPARTMENT OF
ENERGY

Office of
ENERGY EFFICIENCY &
RENEWABLE ENERGY

An Authentication Vulnerability Assessment of Connected Lighting Systems

March 2020

(This page intentionally left blank)

An Authentication Vulnerability Assessment of Connected Lighting Systems

Michael Poplawski[1], Adam St. Lawrence[2], and Hung Ngo[1]

Pacific Northwest National Laboratory[1], Underwriters Laboratories[2]

March 2020

Produced for the U.S. Department of Energy, Energy Efficiency and Renewable Energy,
by the Pacific Northwest National Laboratory, Richland, Washington 99352

PNNL-28782

DISCLAIMER

PACIFIC NORTHWEST NATIONAL LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC05-76RL01830

Printed in the United States of America

Available to DOE and DOE contractors from the
Office of Scientific and Technical Information,
P.O. Box 62, Oak Ridge, TN 37831-0062;
ph: (865) 576-8401
fax: (865) 576-5728
email: reports@adonis.osti.gov

Available to the public from the National Technical Information Service
5301 Shawnee Rd., Alexandria, VA 22312
ph: (800) 553-NTIS (6847)
email: orders@ntis.gov <https://www.ntis.gov/about>
Online ordering: http://www.ntis.gov

Abstract

Emerging connected lighting systems (CLS) that incorporate distributed intelligence, network interfaces, and sensors can become data-collection platforms that enable a wide range of valuable new capabilities as well as greater energy savings in buildings and cities. However, CLS technology is currently at an early stage of development, and its increased connectivity introduces cybersecurity risks that are new to the lighting industry and that must be addressed for successful integration with other systems.

While a number of existing frameworks, guidelines, and tests for evaluating cybersecurity vulnerability may apply to CLS in whole or in part, there is currently no mandatory requirement for cybersecurity testing or certification. The lighting industry, including technology developers and specification organizations, is currently evaluating the suitability of existing frameworks and guidelines for CLS. To support these efforts, Pacific Northwest National Laboratory (PNNL) is conducting a series of studies intended to educate lighting-industry stakeholders on specific cybersecurity practices and characterize their implementation in commercially available CLS with varying system architectures, network-communication technologies, and degrees of maturity.

This first study explores authentication practices and their implementation in multiple CLS. A total of 18 tests were developed by Underwriters Laboratories (UL) and implemented in PNNL's Connected Lighting Test Bed (CLTB). The tests explore the implementation of basic authentication best practices as well as known technology-specific best practices. As a result, not all tests are applicable to all CLS.

A total of 40 out of 72 potential tests (four CLS, 18 potential tests each) were applicable for four evaluated CLS, and the CLS collectively passed 26 of the 40 tests (65%). While pass/fail ratio is a simple way of reporting test results, it is not an actionable metric. Cybersecurity vulnerability testing is a risk-analysis practice; the relevance of passing or failing a certain test is best evaluated in concert with an understanding of the risk associated with that vulnerability in a specific implementation. Nevertheless, pass/fail ratios give some indication of the range of performance found in market-available CLS.

This study demonstrates that tests for authentication vulnerabilities can be developed with objective pass/fall criteria, therby facilitating comparisons between CLS. Based on the limited results of this study, it appears that the CLS that are being brought to market have varying levels of authentication vulnerability. It is hoped that these evaluations will support and perhaps accelerate industry discussions on the risks of specific security vulnerabilities, what vulnerabilities should be addressed by in-development of future lighting-specific best practices, and whether any such practices should be included in voluntary lighting standards.

Introduction

Connected lighting systems (CLS) comprise an emerging class of lighting infrastructure that does more than just light spaces. Through the incorporation of distributed intelligence, network interfaces, and sensors, CLS become data-collection platforms that enable a wide range of valuable new capabilities as well as greater energy savings in buildings and cities. CLS may contain sensors intended to aid in the optimization of lighting service, as well as other sensors that might be used to optimize the performance of other connected building systems [Pandharipande 2018]. However, CLS technology is currently at an early stage of development, and many questions remain about how well it will work, whether it will actually save energy, how much measurable value is provided by new capabilities, and whether it will offer enough benefits and value-added features to justify the investment. Further, increased connectivity introduces cybersecurity risks that are new to the lighting industry and that must be addressed in order for successful integration with other systems.

The network integration of multiple systems, manufactured by different vendors, that manipulate aspects of, or otherwise interact with, the physical world typically results in a complex cyber-physical system (CPS). The

integration of CLS with other building-operation technology systems (e.g., HVAC) and/or building-information technology systems is one example of such a CPS. Complex CPS offer many demonstrated and perceived benefits; however, they also present greats risk to privacy and security protections [Yang 2017]. While many industries understand the value created by system integration, there is a recognition across industries that a common approach is needed to jointly address security, efficiency, privacy, and scalability [Siegel 2018]. In heterogeneous CPS environments such as might exist in buildings, perhaps the most important aspect of securing the integration of CLS and other building systems is implementing and maintaining a consistent security architecture, where secure components utilize secure communication protocols and secure access-control mechanisms. The heterogeneity of CPS components has been found to contribute significantly to many documented attacks [Humayed 2017].

While a number of different secure architectures that might be useful for such systems have begun to emerge, perhaps the biggest impediment to the widespread integration of such systems is the lack of one or more suitable, well-accepted system-development frameworks. In order for CLS that are integrated with other systems to achieve wide industry adoption and deliver upon their energy-savings potential, it is crucial to be able to identify common and likely systemwide attack vectors, protect the information and data flows between devices, and prevent system hijacking [Minoli 2017]. While zero-day exploits (e.g., WannaCry) and high-profile data breaches (e.g., Equifax) tend to garner the most attention by the press and general public, most security breaches come from known vulnerabilities that are not patched or secured. Industry experts expect that virtually all of the vulnerabilities exploited by the end of 2020 will continue to be ones known by security and IT professionals. [Moore 2017].

This study is the first in a series intended to educate lighting-industry stakeholders on specific cybersecurity practices and characterize the implementation of those practices in commercially available CLS with varying system architectures, network communication technologies, and degrees of maturity. This first study explores authentication practices and their implementation in multiple CLS. Future studies will address other practices and/or the characterization of additional CLS.

Background

Cybersecurity is a discipline focused on protecting data, resources, people, and organizations from attacks that may result in financial loss, loss of life, or other damage. Cybersecurity core functions involve managing risk – defined as "a measure of the extent to which an entity is threatened by a potential circumstance or event" and "a function of the adverse impacts that would arise if the circumstance or event occurs and the likelihood of its occurrence[1]" – by identifying potential threats, implementing appropriate security controls, testing security-control effectiveness, and responding to attacks that circumvent the security controls. As entities continually face new and evolving threats, these functions are part of a continuous process, with the goal of eliminating risk exposure or reducing it to manageable levels.

Identification of potential threats requires the implementation of a continuous threat-modeling process that attempts to account for any and all threats facing an entity, where a threat is defined as "any circumstance or event with the potential to adversely impact operations, assets, individuals, or other organizations, through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service[1]". An entity may be a hardware component, a software component, an application, a system, an environment, or an organization. The threat-modeling process generally results in a traceability matrix that maps identified threats to potential security controls and is often used as an input to the risk-management process.

[1] https://standardscatalog.ul.com/standards/en/standard_2900-1_1

Security controls are implemented to effectively address the identified threats based on the entity's goals and risk tolerance. Security controls are often categorized as management, operational, and technical, with examples including firewalls, cryptographic mechanisms, and access controls involving authentication and authorization mechanisms. Determining the effectiveness of implemented security controls is accomplished by testing, which often takes the form of vulnerability assessments, penetration testing, and/or red team operations. Testing attempts to determine whether security controls sufficiently address identified threats and risks in a methodical, repeatable manner.

Authentication

Authentication involves a class or family of security controls that constitute one or more processes to verify the identity claim. Defining the authentication process and needed mechanisms typically begins with the development of an appropriate threat model that defines trust boundaries and who or what may traverse them in order to interact with an entity of value (e.g., a system, network, or specific data). Once the identity of an entity is verified (or authenticated), the subsequent actions it may perform are determined by authorization. Authorization is the process of verifying that a requested action is permitted, and it is accomplished by assessing the privileges associated with the requesting entity. Changes to both authentication mechanisms and authorization privileges are typically audited by some defined means.

An identity is a unique representation of someone or something, such as an equipment user or a specific computer connected to a network. Verification of an identity is typically performed by submission of one or more secrets, which are compared with stored representations or copies of those secrets. Successful authentication provides reasonable assurance that the identity claim is legitimate and that the entity is thus entitled to perform actions permitted only to authenticated entities. The most common authentication mechanism is the traditional combination of a username and a secret password, which together are also referred to as a "credential set." Humans and machines are both capable of using credential sets as well as other authentication mechanisms, such as cryptographic keys and tokens.

An identity associated with an entity is typically established or enrolled at the time of manufacture in the case of hardware, or the time of release in the case of software, or on-demand. Entities typically enroll human and machine users in an on-demand manner, as it may be impossible to predetermine which users will need identities established beforehand. The enrollment process, which is sometimes referred to as "onboarding" or "provisioning," begins with a supplicant (e.g., a user, service, or system) requesting the establishment of an identity with the authenticator, or begins with the authenticator establishing an identity for the supplicant without an explicit request. The process then typically includes the authenticator employing some form of vetting (also known as "proofing") of evidence provided by the supplicant to establish trust. The process then typically concludes with the provision and storage of a secret (e.g., a password) known only to the supplicant and authenticator. Identities may also be established at the time of manufacture or release, with secrets being cryptographically stored within a secure hardware element, configuration file, or database. The identity of a machine entity is typically provisioned by an authenticator or by another device, such as a smartphone or tablet, and is established by sharing one or more cryptographic keys. Various forms of cryptographic keys exist, such as public/private key pairs, network keys, and application keys. An entity may possess one or more keys, with each key serving a different purpose.

The authentication process begins with a supplicant making an identity claim. An authenticator – typically, an entity that the supplicant desires access to – then verifies the identity claim. The supplicant provides a copy or representation of a secret (e.g., a password, a hash, a cryptographic key) that uniquely identifies the supplicant to the authenticator. The authenticator performs a comparison of the supplicant-provided secret and its copy or a representation of the secret and, based on the result of the comparison, confirms or rejects the identity claim.

Once authenticated, an entity generally seeks access to other local or remote entities (i.e., systems, networks, data). This subsequent access, referred to as "authorization," is controlled by an access-control policy that enumerates the permissions or rights associated with each entity and results in an access decision (i.e., allowed

or denied) based on a set of rules. Policies may be simple (e.g., a specific entity may access another specific entity) or complex (e.g., an entity with specific characteristics and conditions may access another entity with specific characteristics and conditions). The access-control model that's utilized determines the simplicity or complexity of authorization, and its implementation (e.g., through configuration) is another area of focus for security testing.

Authentication Vulnerabilities

Almost all identity-based attacks begin with attempts to gain a foothold in a system by manipulating or bypassing authentication mechanisms [Chen 2017]. Attackers looking to exploit authentication vulnerabilities target weaknesses in the communication medium used to transport secrets, the secrets-storage mechanism, or the authentication mechanism itself. Often system user-interfaces and their associated devices are deployed in poorly secured or publicly accessible spaces, which only increase the ability and opportunity for physical tampering as well as cloning attacks. In such deployments, traditional password-based authentication schemes may not be robust enough, as a skilled attacker with physical access to such systems can employ one or multiple techniques to overcome such schemes [Gope 2019].

Communication mediums (wired and wireless) may be susceptible to what are known as "sniffing attacks," in which an attacker is able to observe secrets in an intelligible manner, due to a lack of adequate cryptographic protections. Inadequacy may take the form of a failure to implement available cryptographic protections, insecure (i.e., misconfigured) implementation, or weakness in the cryptographic mechanism itself (e.g., weak or broken ciphers). The same cryptographic concepts apply to secrets storage. If secrets are stored without cryptographic protection, an attacker can observe intelligible secrets within the storage mechanism. Weaknesses within the authentication mechanism often involve the secrets themselves and typically take the form of easily guessed secrets, hard-coded secrets, or weak or brute-forcible secrets.

Easily guessed secrets are often the name of a product, person, company, location, or some combination thereof and may involve a single number, simple sequence of numbers (e.g., 1-2-3), or commonly used special characters (e.g., an exclamation point). Easily guessed secrets also include secrets that have been publicly disclosed as a result of prior breach. Hard-coded secrets are similar to easily guessed secrets but are set during the development or manufacturing process, or somewhere in the supply chain, whereas easily guessed secrets are often user-configured. Hard-coded secrets generally enable users to perform initial configuration and are often published in documentation or on a website, typically without any associated security controls. Published hard-coded secrets are easily obtained by attackers, as the information is public knowledge.

Provided enough time and resources, an attacker could use brute force to obtain any secret. A brute-force attack against a secret may involve attempts using all possible combinations that comprise the secret scheme (i.e., complexity requirements), or attempts to authenticate with secrets from one or more predefined dictionaries, until a successful authentication occurs. A brute-force attack may also involve attempts to reverse one or more cryptographic representations of the secret (e.g., cracking a password hash). The success of brute-force attacks is largely a function of the secret's strength (i.e., it's length, character set, and cryptographic protections).

Weaknesses in authentication can be identified and assessed using such methods as static and/or dynamic code analysis, software-composition analysis, vulnerability assessments, and penetration testing. Each method has its use cases and may be performed during development or post-release by staff or independent third parties. Static and/or dynamic code analysis involves evaluating application source code or runtime behavior for errors or conditions that may cause anomalous behavior. This is often done using automated tools but may also be performed manually using code reviews (e.g., pair programming or the review of pull requests). Static and/or dynamic code analysis is often performed by development teams, or by security teams supporting development, to catch as many potential vulnerabilities as possible prior to release – such as a debug option that was used during development and that, when set, would bypass authentication.

Software-composition analysis is a method of attempting to determine the software components used within an application and the known vulnerabilities associated with those components. For example, software-composition analysis may reveal that an application uses a specific software library (i.e., version) that's vulnerable to an authentication bypass as reported in the National Vulnerability Database. Software-composition analysis is typically an automated process that analyzes available sources and binaries.

Vulnerability assessments are typically performed using automated tools (scanners) to identify and classify vulnerabilities associated with a specific asset or set of assets (e.g., a group of servers). Vulnerability-assessment tools often rely on publicly disclosed weakness and vulnerability data (e.g., common vulnerabilities and exposures, also known as CVEs) to identify specific instances of vulnerabilities associated with an asset. Once a vulnerability has been identified, it is classified, either quantitatively (e.g., 9.2 on a scale of 1 to 10) or qualitatively (e.g., "critical"), and then prioritized among the other identified vulnerabilities. A report prioritizing each identified vulnerability to allow for effective risk management is often the end deliverable of a vulnerability assessment.

Penetration testing attempts to exploit identified vulnerabilities through automated and manual means, to demonstrate their risk. This differs considerably from a vulnerability assessment, which does not attempt exploitation. Penetration testing leverages expert understanding of the nuances of an asset or set of assets (e.g., processes, logic, relationships, dependencies) and, as a result, may generate many false positives (i.e., vulnerabilities that could not be exploited without the expert understanding of the asset or set of assets). Penetration testing attempts to exploit vulnerabilities discovered by vulnerability assessment (e.g., an insecurely configured or outdated component) as well as vulnerabilities that can be exposed by creatively combining information that would otherwise remain isolated in a manner that exploits one or more nuances. A report that describes exploited vulnerabilities in sufficient detail for each to be reproduced is often the end deliverable of a penetration test. Retesting takes place after risk-management actions have been taken (e.g., after the vulnerability is remediated or mitigated with a compensating control) to validate their effectiveness.

CLS Authentication Testing

CLS may utilize authentication mechanisms for a variety of purposes. Two of the most common purposes are to enable human users to access system-configuration or management software, and to enable machines to access application programming interfaces (APIs). System-configuration software might be utilized, for example, to set lighting levels and schedules, configure sensors, manually control or examine sensor readings from specific light sources, or create and review system reports generated by remote monitoring. System-configuration software might exist on a local computer or computer server or a mobile device, or be accessed via a web app that is served up from a local or cloud server. APIs are typically used to exchange data with other (lighting or nonlighting) systems to enable, for example, a heating, ventilation, and air conditioning system to have access to occupancy data collected by a lighting system. Unsanctioned access to CLS might enable an attacker to manually control or reconfigure lighting devices, obtain access to human user or historical system performance data, or interrupt or distort lighting or other system functionality that is dependent on data exchange between the systems.

Test Setup, Implementation, and Method

The test setup used to identify authentication vulnerabilities consisted of a user interface device with multiple operating systems, multiple web browsers, a login cracker, a web vulnerability scanner, a packet analyzer, and an over-the-air Zigbee packet sniffer (Figure 1). The test setup was implemented in the CLTB, as shown in Figure 2. Details about the hardware, software, and firmware comprising the test setup implementation are provided in Table 1. A total of 18 tests were developed by UL and implemented by PNNL to characterize the authentication vulnerability of CLS. The tests explore the implementation of basic authentication best practices (e.g., encrypting user credentials before transmitting them on the network) as well as known technology-specific best practices (e.g., the use of Zigbee default trust center, or the implementation of JSON Web Token).

As a result, not all tests are applicable to all CLS (i.e., not all CLS use Zigbee or JSON technology). Many (but not all) of the tests are described in, and derived from, the Open Web Application Security Project™ (OWASP) Testing Guide 4.0. Descriptions for each test method are provided in Table 2.

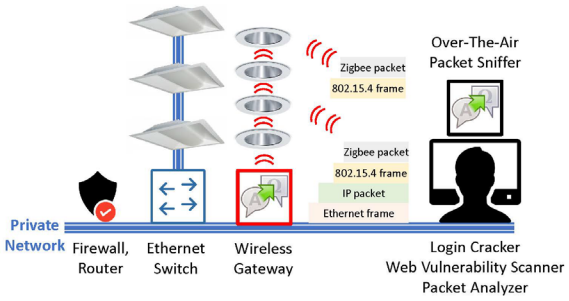

Figure 1. Authentication vulnerability test setup.

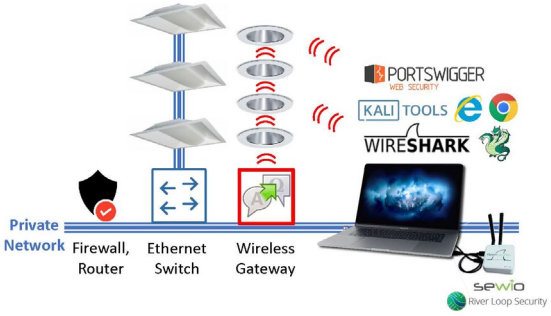

Figure 2. Authentication vulnerability test setup implementation in the CLTB.

Table 1. CLTB Equipment Used to Implement Authentication Tests.

Type/Description	Make	Model
User Interface	1) Apple 2) Dell	1) MacBook Pro (Retina, 15-inch, early 2013), macOS High Sierra 10.13.6 2) Latitude E6540, Windows 7
Web Browser	1) Google 2) Microsoft	1) Chrome 71.0.3578.98 2) Internet Explorer 11
Login Cracker	THC	Hydra 8.6.1
Web Vulnerability Scanner	PortSwigger Web Security	Burp Suite Community 1.7.36
Packet Analyzer	Wireshark	Wireshark 2.6.6
Over-The-Air Zigbee Packet Sniffer	1) Sewio 2) River Loop Security	1) Open Sniffer 3.0 (hardware) 2) KillerBee 2.0 (firmware)

Table 2. Authentication Test Descriptions

Test	Description
Test 1: Web Authentication Credentials Transported over an Unencrypted Channel	Determine whether authentication credentials (e.g., username, password) are protected (e.g., using HTTPS) in transit on the network. Additional detail is provided in the OWASP Testing Guide[2].
Test 2: Use of Default Web Credentials	Determine whether system has default accounts (e.g., admin) that, following installation, authenticate with a default account username/password. Additional detail is provided in the OWASP Testing Guide[3].
Test 3: Weak Web Lockout Mechanism	Determine whether a user account is locked after five or more failed authentication attempts, and whether the lockout time duration is less than 10 minutes. Additional detail is provided in the OWASP Testing Guide[4].
Test 4: Authentication Schema Bypass	Determine whether authentication credentials that are included in the request header after a successful login are invalidated after an authorized user logs out. Additional detail is provided in the OWASP Testing Guide[5].
Test 5: Insecure Authentication Credential Retention	Determine whether session cookies store authentication data in an insecure manner (e.g., clear-text, unencrypted). Additional detail is provided in the OWASP Testing Guide[6].
Test 6: Session Timeout	Determine whether a user is automatically logged out from an active session following a period of inactivity of more than 15 minutes. Additional detail is provided in the OWASP Testing Guide[7].
Test 7: Session Cookie Destruction	Determine whether session cookies are properly destroyed upon de-authentication or session termination due to inactivity. Additional detail is provided in the OWASP Testing Guide[8] and elsewhere[9].
Test 8: Renewed Authentication for Lost or Terminated SSH Sessions over a Remote Interface	Determine whether stored data from the previous SSH session can be used to bypass authentication mechanisms during a new session creation.
Test 9: Web Authentication Username Enumeration	Determine whether authentication error messages disclose authorized usernames, thereby facilitating brute-force attacks with known usernames. Additional detail is provided in the OWASP Testing Guide[10].
Test 10: Use of Zigbee Default Trust Center Link Key	Determine whether the publicly known Zigbee default trust center link key is used.
Test 11: JSON Web Token (JWT) "none" Algorithm Validation	Determine whether a JSON Web Token (JWT) may be used to bypass validation by utilizing "none" for the "alg" field.
Test 12: Weak JSON Web Token (JWT) HMAC SHA256 Secret	Determine whether the JSON Web Token (JWT) HMAC SHA256 secret can be obtained through a brute-force attack.

[2] https://www.owasp.org/index.php/Testing_for_Credentials_Transported_over_an_Encrypted_Channel_(OTG-AUTHN-001)
[3] https://www.owasp.org/index.php/Testing_for_default_credentials_(OTG-AUTHN-002)
[4] https://www.owasp.org/index.php/Testing_for_Weak_lock_out_mechanism_(OTG-AUTHN-003)
[5] https://www.owasp.org/index.php/Testing_for_Bypassing_Authentication_Schema_(OTG-AUTHN-004)
[6] https://www.owasp.org/index.php/Testing_for_Vulnerable_Remember_Password_(OTG-AUTHN-005)
[7] https://www.owasp.org/index.php/Test_Session_Timeout_(OTG-SESS-007)
[8] https://www.owasp.org/index.php/Testing_for_cookies_attributes_(OTG-SESS-002)
[9] https://www.vanstechelman.eu/content/cookie-replay-attacks-in-aspnet-when-using-forms-authentication
[10] https://www.owasp.org/index.php/Testing_for_User_Enumeration_and_Guessable_User_Account_(OWASP-AT-002)

Test 13: Missing JSON Web Token (JWT) “jti,” ”exp,” and ”iat” Claims	Determine whether JSON Web Token (JWT) replay protections have been implemented.
Test 14: Insecure Web-Based Credential Set Password Change	Determine whether a current password is required during a password change procedure.
Test 15: Assuming User Identity Through SAML Login	Determines whether an attacker can log in as a different user during SAML authentication, using an XML library vulnerability.
Test 16: MQTT Authentication Credentials	Determine whether the CONNECT packet sent from a MQTT client to a MQTT broker discloses authentication credentials.
Test 17: Bluetooth Replay and On-the-Fly Data Modification	Determine whether Bluetooth communications are encrypted and if data signing is implemented.
Test 18: Identifying Bluetooth Class of Device/Service	Determine whether Bluetooth devices broadcast Class of Device or Class of Service as part of their discovery beacons.

Test Units

CLS available on the market utilize a wide variety of system architectures (e.g., star, mesh, hybrid) and wireless (e.g., Zigbee, Bluetooth Mesh, Wi-Fi, cellular) or wired (e.g., DALI, Ethernet or Power-over-Ethernet) network communication technologies. An example of how varying system architectures might be integrated into a network is shown in Figure 3, which depicts a conceptual representation of four CLS, including a wireless system connected to a private network via a local gateway, a wireless system connected to a public network via a shared (e.g., cellular) gateway, a wired system that utilizes Ethernet- or Power over Ethernet-based communication and can be configured using a mobile device connected via a Wi-Fi gateway, and a wireless system that utilizes Bluetooth Mesh for both communication and configuration via a mobile device.

Five CLS, spanning a range of vintages, system architectures, network implementations, and other characteristics, were initially targeted for authentication testing (Table 3). The developed test-method suite was not suitable for one of the systems (CLS E) because it did not have an integral authentication mechanism and, for cybersecurity, relied on the mechanism implemented for the host computer. The test-method suite was run on the remaining four CLS.

Table 3. CLS Targeted for Authentication Testing.

	CLS A	CLS B	CLS C	CLS D	CLS E
Vintage	2015	2015	2019	2019	2018
System Architecture	Web app accessed via on-premise server; CLS devices connected via wireless gateway	Web app accessed via on-premise server; CLS devices connected via wireless gateway	Web app accessed via cloud server; CLS devices connected via wireless gateway	1) Web app accessed via cloud server and 2) iOS app; CLS devices connected via direct wireless	Web app accessed via on-premise server; CLS devices connected via wired switch
Network Connectivity	Wireless, Zigbee based Mesh	Wireless, Zigbee based Mesh	Wireless, 2G Cellular	Wireless, Bluetooth Mesh	Wired, Power over Ethernet (PoE)
Physical Layer Technology	IEEE 802.15.4	IEEE 802.15.4	GPRS	Bluetooth Low Energy	IEEE 802.3

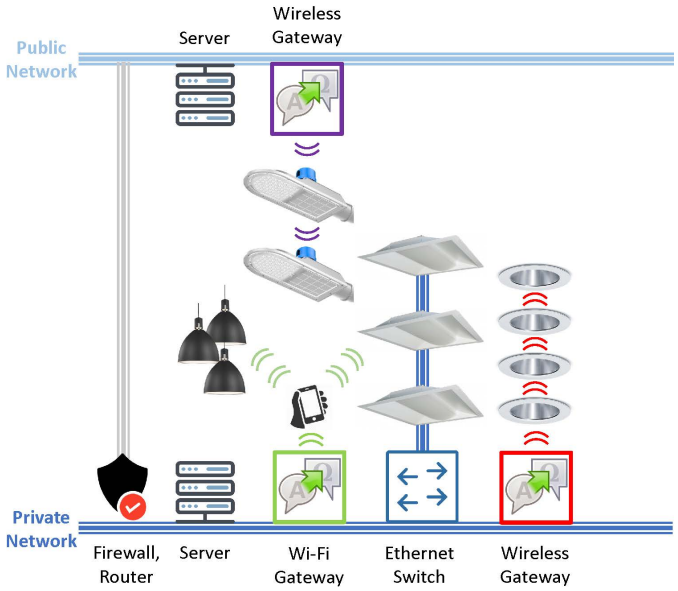

Figure 3. Conceptual representation of multiple CLS, showing common system architecture variations and technology implementations.

Test Results and Analysis

While 18 tests were defined, not all tests were applicable for every CLS. For example, while eight of the tests were applicable to all four CLS that were evaluated in this investigation, six of the tests were not applicable to any of the CLS, due to their non-use of the targeted technology (i.e., JSON Web Token, MQTT, Bluetooth). One of the tests could not be applied to one CLS, due to the unavailability of a sudo, or "superuser," password to access Secure Shell (SSH) on the local server. While CLS D was configurable or otherwise accessible via both a cloud interface and a mobile device, only the cloud interface was evaluated in this study, as the current test setup does not include an over-the-air Bluetooth packet sniffer. A total of 40 out of 72 possible tests (four

CLS, 18 tests each) were applicable for the four evaluated CLS, and the CLS collectively passed 26 of the 40 tests (65%). All four CLS passed two of the tests and failed one test. CLS B, the best statistical performer, passed nine out of a total of 11 applicable tests (82%); while CLS A, the worst statistical performer, only passed three out of a total of nine applicable tests (33%). The other two CLS passed seven out of a total of 10 applicable tests (70%). Test results are summarized in Table 4 and depicted graphically in Figures 4 (by CLS) and 5 (by authentication test number). While pass/fail ratio is a simple way of reporting test results, it is not really a relevant metric. Cybersecurity vulnerability testing is a risk-analysis practice; the relevance of passing or failing a specific test is best evaluated in concert with an understanding of the risk associated with that vulnerability in a specific implementation. Nevertheless, pass/fail ratios give some indication of the range of performance found in market-available CLS. Complete test results for each CLS are provided in Appendix A.

Table 4. Test Results Summary.

	Summary	CLS A	CLS B	CLS C	CLS D (*Cloud interface only)
Applicable	40	9	11	10	10
PASS	26 (63%)	3	9	7	7
FAIL	14 (37%)	6	2	3	3

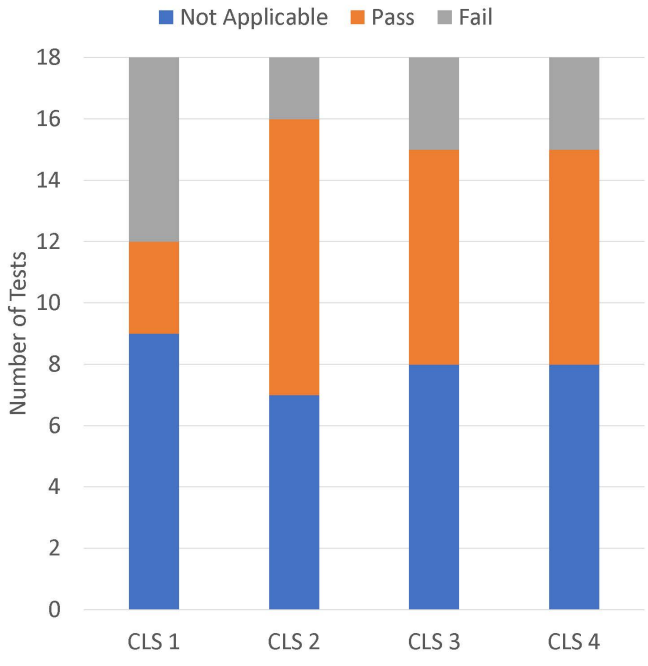

Figure 4. Authentication vulnerability testing results, by CLS.

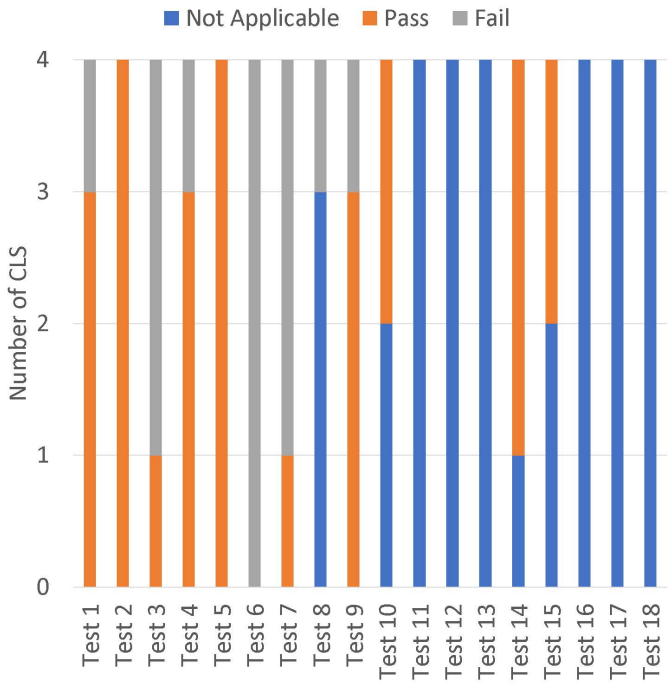

Figure 5. Authentication vulnerability testing results, by authentication test number.

Summary and Recommendations

There are numerous existing frameworks and guidelines for evaluating cybersecurity vulnerability, such as the National Institute of Standards and Technology (NIST) Cybersecurity Framework, the NIST 800 series comprising more than 150 resources, the International Electrotechnical Commission (IEC) 62443 series, International Organization for Standardization 27001 and 27002, Unified Facilities Criteria (UFC) 4-010-06, and UL 2900-1. Furthermore, a variety of testing resources are widely available, including the Open Web Application Security Project (OWASP) Testing Guide. While these frameworks, guidelines, and tests may apply to CLS in whole or in part, there is currently no mandatory requirement for cybersecurity testing or certification. The lighting industry, including technology developers and specification organizations, are evaluating the suitability of these frameworks and guidelines for CLS. This study demonstrates that tests for authentication vulnerabilities can be developed with objective pass/fall criteria, thereby facilitating comparisons between CLS. Based on the limited results of the present study, it appears that the CLS that are being brought to market have varying levels of authentication vulnerability.

Next Steps

PNNL plans to conduct more authentication testing (e.g., enhancing existing authentications tests, incorporating new authentication tests, or evaluating additional CLS), and to work with UL and other cybersecurity experts to develop tests for, and initiate an exploration into, authorization vulnerabilities. Authentication Test 1 will be enhanced to verify that Transport Layer Security is used to encrypt Hypertext Transfer Protocol Secure (HTTPS) traffic, as opposed to Secure Sockets Layer security, which has been deprecated.

Furthermore, PNNL will bring these results to the ANSI C137 Lighting Systems ad-hoc working group focusing on cybersecurity vulnerability, for consideration in the creation and development of new standards. It is hoped that these evaluations will support and perhaps accelerate industry discussions on the risks of specific

security vulnerabilities, what vulnerabilities should be addressed by in-development of future lighting-specific best practices, and whether any such practices should be included in voluntary lighting standards.

Recommendations

1. Lighting-industry stakeholders should provide feedback on how future cybersecurity studies could be modified or enhanced to provide greater industry value.

2. Cybersecurity vulnerability experts should provide feedback on the vulnerability tests that were utilized in the study, including whether important tests were missing and whether included tests are not particularly relevant to CLS.

3. Cybersecurity vulnerability experts should consider working with PNNL to develop new vulnerability tests that might be incorporated into future test suites.

4. CLS developers should consider supporting the integration of their CLS into the PNNL CLTB, where the CLS would be available for use in future cybersecurity vulnerability studies.

Appendix A: Authentication Test Results

Summary **Applicable: 40** **>PASS: 26 FAIL: 14** **Not applicable: 32**	**CLS A** **Applicable: 9** **>PASS: 3 FAIL: 6** **Not applicable: 9**	**CLS B** **Applicable: 11** **>PASS: 9 FAIL: 2** **Not applicable: 7**	**CLS C** **Applicable: 10** **>PASS: 7 FAIL: 3** **Not applicable: 8**	**CLS D (*Cloud interface only)** **Applicable: 10** **>PASS: 7 FAIL: 3** **Not applicable: 8**
Test 1 Applicable: 4 >PASS: 3 FAIL: 1 Not applicable: 0	FAIL: HTTPS is not used to securely transmit credentials	PASS	PASS	PASS
Test 2 Applicable: 4 >PASS: 4 FAIL: 0 Not applicable: 0	PASS	PASS	PASS	PASS
Test 3 Applicable: 4 >PASS: 1 FAIL: 3 Not applicable: 0	FAIL: No account lockout after six consecutive login failures	PASS	FAIL: No account lockout after six consecutive login failures	FAIL: No account lockout after six consecutive login failures
Test 4 Applicable: 4 >PASS: 3 FAIL: 1 Not applicable: 0	FAIL: Authentication token saved in cookie following logout	PASS	PASS	PASS
Test 5 Applicable: 4 >PASS: 4 FAIL: 0 Not applicable: 0	PASS	PASS	PASS	PASS
Test 6 Applicable: 4 >PASS: 0 FAIL: 4 Not applicable: 0	FAIL: No auto-logout after 15 minutes	FAIL: No auto-logout after 15 minutes	FAIL: No auto-logout after 15 minutes	FAIL: No auto-logout after 15 minutes
Test 7 Applicable: 4 >PASS: 1 FAIL: 3 Not applicable: 0	FAIL: Authentication token saved in cookie following session timeout	PASS	FAIL: Can re-authenticate following session termination using ←→ sequence	FAIL: Can re-authenticate following session termination using ←→ sequence
Test 8 Applicable: 2 >PASS: 0 FAIL: 1 Not applicable: 3	Not applicable: Sudo password to access SSH not available	FAIL: CLS does not request new authentication	Not applicable: CLS does not use SSH session	Not applicable: CLS does not use SSH session
Test 9 Applicable: 4 >PASS: 3 FAIL: 1 Not applicable: 0	FAIL: CLS discloses validity of username when a wrong password is entered	PASS	PASS	PASS

Summary Applicable: 40 >PASS: 26 FAIL: 14 Not applicable: 32	CLS A Applicable: 9 >PASS: 3 FAIL: 6 Not applicable: 9	CLS B Applicable: 11 >PASS: 9 FAIL: 2 Not applicable: 7	CLS C Applicable: 10 >PASS: 7 FAIL: 3 Not applicable: 8	CLS D (*Cloud interface only) Applicable: 10 >PASS: 7 FAIL: 3 Not applicable: 8
Test 10 Applicable: 2 >PASS: 2 FAIL: 0 Not applicable: 2	PASS	PASS	Not applicable: CLS does not use Zigbee	Not applicable: CLS does not use Zigbee
Test 11 Applicable: 0 >PASS: 0 FAIL: 0 Not applicable: 4	Not applicable: CLS does not use JWT	Not applicable: CLS does not use JWT	Not applicable: CLS does not use JWT	Not applicable: CLS does not use JWT
Test 12 Applicable: 0 >PASS: 0 FAIL: 0 Not applicable: 4	Not applicable: CLS does not use JWT	Not applicable: CLS does not use JWT	Not applicable: CLS does not use JWT	Not applicable: CLS does not use JWT
Test 13 Applicable: 0 >PASS: 0 FAIL: 0 Not applicable: 4	Not applicable: CLS does not use JWT	Not applicable: CLS does not use JWT	Not applicable: CLS does not use JWT	Not applicable: CLS does not use JWT
Test 14 Applicable: 3 >PASS: 3 FAIL: 0 Not applicable: 1	Not applicable: No password change option	PASS	PASS	PASS
Test 15 Applicable: 2 >PASS: 2 FAIL: 0 Not applicable: 2	Not applicable: CLS does not use SAML	Not applicable: CLS does not use SAML	PASS	PASS
Test 16 Applicable: 0 >PASS: 0 FAIL: 0 Not applicable: 4	Not applicable: CLS does not use MQTT	Not applicable: CLS does not use MQTT	Not applicable: CLS does not use MQTT	Not applicable: CLS does not use MQTT
Test 17 Applicable: 0 >PASS: 0 FAIL: Not applicable: 4	Not applicable: CLS does not utilize Bluetooth	Not applicable: CLS does not utilize Bluetooth	Not applicable: CLS does not utilize Bluetooth	Not applicable: CLS does not utilize Bluetooth*
Test 18 Applicable: 0 >PASS: 0 FAIL: 0 Not applicable: 4	Not applicable: CLS does not utilize Bluetooth	Not applicable: CLS does not utilize Bluetooth	Not applicable: CLS does not utilize Bluetooth	Not applicable: CLS does not utilize Bluetooth*

References

[Gope 2019] P. Gope and B. Sikdar, "Lightweight and Privacy-Preserving Two-Factor Authentication Scheme for IoT Devices," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 580-589, 2019.

[Pandharipande 2018] A. Pandharipande, M. Zhao, E. Frimout and P. Thijssen, "IoT lighting: Towards a connected building eco-system," *2018 IEEE 4th World Forum on Internet of Things* (WF-IoT), 2018.

[Siegel 2018] J. E. Siegel, S. Kumar and S. E. Sarma, "The Future Internet of Things: Secure, Efficient, and Model-Based*," IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2386-2398, 2018.

[Chen 2017] D. Chen, N. Zhang, Z. Qin, X. Mao, Z. Qin, X. Shen and X.-y. Li, "S2M: A Lightweight Acoustic Fingerprints-Based Wireless Device Authentication Protocol," *IEEE Internet of Things Journal* , vol. 4, no. 1, pp. 88-100, 2017.

[Humayed 2017] A. Humayed, J. Lin, F. Li and B. Luo, "Cyber-Physical Systems Security—A Survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1802-1831, 2017.

[Minoli 2017] D. Minoli, K. Sohraby and B. Occhiogrosso, "IoT Considerations, Requirements, and Architectures for Smart Buildings—Energy Optimization and Next-Generation Building Management Systems," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 269-283, 2017.

[Moore 2017] Moore, Susan. “Focus on the Biggest Security Threats, Not the Most Publicized.” *Smarter With Gartner*, November 2, 2017. https://www.gartner.com/smarterwithgartner/focus-on-the-biggest-security-threats-not-the-most-publicized/.

[Yang 2017] Y. Yang, L. Wu, G. Yin, L. Li and H. Zhao, "A Survey on Security and Privacy Issues in Internet-of-Things," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1250-1258, 2017.

(This page intentionally left blank)

U.S. DEPARTMENT OF
ENERGY

Office of
ENERGY EFFICIENCY &
RENEWABLE ENERGY

For more information, visit:
energy.gov/eere/ssl

PNNL-28782 • March 2020