VDOO

The Top 10 Requirements for Authentication in IoT

THE IMPORTANCE OF AUTHENTICATION

Many of the highly publicized cyber-attacks in IoT spread without using any sophisticated exploits, rather by entering devices through the unlocked front door – via a hard-coded administrative password put in by the device manufacturer, which was in many cases easy to guess (such as 'abcd1234').

This guide compiles the top ten requirements related to authentication for secure-by-design development of connected devices.

In it you will find:

Which are the most common authentication requirements for IoT?
Why each one is important?
What you can do to implement the requirements?

FORCE THE USER TO CHANGE THE DEFAULT PASSWORD ON FIRST LOGIN

VDOO

What does the requirement mean?

When users first use the device, they should be required to set a new password before using or configuring the device via the web-management interface.

Why is it important?

Default passwords are one of the most common attack vectors in IoT, especially for web-management interfaces, since they are exposed to remote access. Most users will keep the default password set by the vendor and never change it after logging in.

How can you implement it?

Before shipping the device to the customer, set the expiration time on all existing passwords to 0. Ensure that on password change, weak blank passwords will not be accepted.

ENFORCE PASSWORD COMPLEXITY

VDOO

What does the requirement mean?

This guidance comes from the National Institute of Standards and Technology (NIST) and notes the importance to allow users to create strong, memorable passwords. As such it is recommended to enforce a minimum length of 8 characters, and allow long passwords (up to 64 characters at least, and optionally more). It is also recommended to allow the use of wide character sets (including all printable ASCII, Unicode and spaces).

Why is it important?

Password complexity determines the effort an attacker must spend on guessing passwords (online) or cracking hashes (offline). Using blank or overly short passwords exposes the device to easy password guessing attacks.

How can you implement it?

There are many ways to enforce password complexity. Some of the most common ones are not requiring mixed case or special characters, not truncating passwords before verification and not allowing passwords containing the name of the associated user account. Furthermore you should count Unicode characters as one character per code point, and they should be normalized before verification.

Finally, to deliver a better experience, showing the user a strength meter on the password change UI is recommended.

ONLY ALLOW ROOT ACCESS TO PASSWORD STORAGE

VDOO

What does the requirement mean?

On Linux systems, there are several password files: /etc/passwd, /etc/shadow and /etc/gshadow. /etc/passwd contains only the names of accounts defined on the system, /etc/shadow contains the names of accounts defined on the system, along with salted password hashes. The same is true for /etc/gshadow, the groups password file. The regular user access to these files must be restricted.

Why is it important?

Allowing regular users access to password storage puts the system at risk: users with read access to password hashes could use them to crack weak or guessable passwords. Users with write access can install additional accounts or change account data to become privileged, or root users.

How can you implement it?

/etc/passwd must be defined with read permissions to all accounts, but write permissions to root only. /etc/shadow must be defined with read and write permissions to root only, with the same being true for /etc/gshadow, the groups password file.

ONLY STORE PASSWORDS AS SECURE, SALTED HASHES

VDOO

What does the requirement mean?

Key derivation functions take a password, a salt, and a cost factor as inputs then generate a password hash. Their purpose is to make each password guessing trial by an attacker who has obtained a password hash file expensive, and therefore, the cost of a guessing attack high or prohibitive. Examples of suitable key derivation functions include Password-based Key Derivation Function 2 (PBKDF2) [SP 800-132], Balloon, or Argon2.

Why is it important?

Storing passwords in clear text exposes them to risk of recovery by an attacker. Using weak algorithms for hashing could result in attackers cracking the hashes cryptographically. Omitting the use of salt exposes the hashes to the risk of cracking using rainbow tables and other pre-computation techniques. Using fewer iterations results in smaller computational effort for the attacker.

How can you implement it?

Password storage requirements should adhere to NIST SP800-63B, 5.1.1.2 Memorized Secret Verifiers. Additionally, you should use 10,000 iterations of a hash-based derivation function, such as PBKDF2, and a secure hash, such as HMAC-SHA-256 as the internal function. Finally, it is recommended to use unique salt values, at least 32 bits in size, produced using a strong RNG. The salt can be stored alongside the hash, or in hardware-secured storage if available.

USE DIFFERENT PASSWORDS FOR DIFFERENT FUNCTIONS

VDOO

What does the requirement mean?

Reusing operating system account passwords (i.e. the passwords defined for the Linux users) for other purposes, such as the device's web management interface, should be avoided.

Why is it important?

Password reuse is a common cause of accidental password disclosure, and threatens even strong and complex passwords. Unlike operating system accounts, which are stored using secure hashing techniques, passwords for other purposes are often stored in the clear in configuration files and databases. This enables attackers with access to device firmware to obtain the login credentials for the device, and potentially for the entire device model.

How can you implement it?

Prevent users from using the same password for both operating system and application logins, by removing password-based authentication for the operating system. Set the device up so that users have to log in remotely via a key-exchange or certificate method, and then delete the users’ passwords from the operating system.

Web application developers should design their applications in such a way that they will store user passwords in the form of a salted hash, instead of as plain text. Indeed, most programming languages for web app development, such as PHP, have their own libraries for this purpose.

STORE APPLICATION PASSWORDS IN ENCRYPTED FORM

VDOO

What does the requirement mean?

Encrypt any passwords stored by applications in files or databases, with the key stored separately.

Why is it important?

Passwords that appear in-clear in files can be obtained and abused by attackers that are able to disguise as the device identity.

How can you implement it?

If a hardware key management engine is available, use it as the highest priority. As a second priority, use operating system-provided secure key storage facilities such as Linux Key Retention Service, Windows DPAPI, Android KeyStore or iOS Keychain. Finally, the encryption key can be stored in a secure area of the firmware (on-chip Flash). If it is stored as a file on a filesystem, apply file system permissions to restrict access to the file.

AVOID DEFAULT PASSWORDS APPEAR IN KNOWN PUBLIC DICTIONARIES

VDOO

What does the requirement mean?

Check passwords for quality by using a password database. When defining a password on the device, check the password against an external password list that contains values known to be overly common or compromised, and reject the password if it is found. Inline APIs and downloadable database files are available for this purpose. When using an online API for password quality checking, make sure you do not compromise the password - use encrypted network communications, and submit only the password's hash, not the password itself.

Why is it important?

Week or known passwords are the most common and simple way to attack devices, an attacker who knows a password can compromise the device by performing operations and abusing the system on behalf of the user, reading their personal information, and finally, hijacking the device by replacing user's credentials with his own.

How can you implement it?

On Ubuntu, install the "libpam-pwquality" module. By default, this enforces a minimum password length of 8 characters, and checks against known weak passwords. Configure this module by editing /etc/security/pwquality.conf.

USE ONLINE SERVICES OR DATABASES TO VERIFY PASSWORD STRENGTH

VDOO

What does the requirement mean?

On password change, have the password management software check the prospective password against an externally managed list that contains values known to be commonly-used, expected, or compromised, and reject the password if it is found to be compromised.

Why is it important?

Using simple, guessable, or well-known passwords exposes the device to guessing attacks on its remote management interface (such as SSH), as well as offline password cracking if the device's firmware is available on the Internet.

How can you implement it?

Online APIs and downloadable databases are available for this purpose. If using an online service, use encrypted network communications and ensure that only the password hash is submitted, not the password itself.

An example of an online service offering an API is passwordping - www.passwordping.com

An open-source password list can be found at:

https://github.com/danielmiessler/SecLists/tree/master/Passwords/Common-Credentials

USE UNIQUE PER-DEVICE DEFAULT PASSWORDS

VDOO

What does the requirement mean?

Each device deployed in the field or shipped to users must have a unique password. The password must not be derived from visible, or freely accessible, device attributes. Examples of easily accessible attributes are a sticker with a serial number or barcode, the MAC address, device model number or device version number.

Why is it important?

All attributes that can be collected by an attacker using passive eavesdropping or actively connecting to the device or its web/cloud service, without performing actual attacks, must be excluded, as their knowledge allows easy password guessing.

How can you implement it?

Deriving the password from a publicly available device's attributes can be allowed if the derivation function uses a secret key (e.g. encryption with an organizational secret). The derivation should not happen on the device itself but at safe secured location.

PROVIDE PASSWORD RECOVERY MECHANISM WITHOUT REVEALING ACCOUNT INFO

VDOO

What does the requirement mean?

Password recovery or reset mechanisms shouldn’t provide information indicating the existence of a valid account, or any hints on its credentials. Additionally, the use of security questions should be avoided as they are typically weaker than strong passwords. Finally, do not output the new password in clear or send it over unsecured channels such as unencrypted email. Instead, send a link to the user containing authentication information, allowing the user to log in and receive or enter a new password.

The same guidelines should apply to key update and recovery mechanisms.

Why is it important?

Password reset flows often use weak authentication mechanisms such as security questions, or leak revealing information about the passwords using hint fields, or informative error messages. This allows attackers to enumerate accounts and recover passwords more easily than by brute force. Finally, new passwords are often sent by unsecured email, letting network eavesdroppers obtain the password in clear.

How can you implement it?

Designing a web page that users can use to retrieve or reset their passwords is the responsibility of web app developers. Most web app programming languages have available libraries and frameworks for this purpose. If a help desk exists, train help desk personnel to authenticate requests for password reset or recovery, and to recognize the signs of a “social engineering” attack.

ABOUT VDOO

VDOO provides an end-to-end web-based platform which helps IoT manufacturers understand what they need to do from a security standpoint to deploy a secure connected device providing guidance and solutions from development to post-deployment. The platform automatically performs device-specific analysis by focusing on the device's firmware only, thus not requiring any access to the physical device or to its source code.

The analysis service retrieves security requirements, tailored to the specific needs of the device, given its hardware and software components and their relative risk factors. It then provides an automated guidance to allow the manufacturer to properly implement security-essential building blocks and apply security by design.

Once a given product and firmware have undergone the full process of auto-analysis and security implementation, and the device's embedded-security was found to be complete, VDOO provides the manufacturer with an objective third-party security certificate.

Learn more at www.vdoo.com