HIDEEZ

September 2021

Hideez Group White Paper

Security Architecture

Hideez Enterprise Solution

Your Key to **Digital Safety**

Hideez Group Whitepaper

Hideez Enterprise Solution: Security Architecture

September 2021

Author

Denys Zaliznyak, CTO Hideez Group

Abstract

The purpose of this document is to explain how Hideez software and hardware products are designed and manufactured, what organizational and technical measures have been taken to ensure information security and what are the possible risks associated with them.

The document is intended for information security specialists and is meant to be a source of information to help them decide on the appropriateness of using Hideez products within a specific company. It does not provide a complete description of the functionality and benefits for users.

Table of contents

Terms and Definitions

- **Bluetooth LE, BLE** – Bluetooth Low Energy, a subset of the Bluetooth protocol that enables communication between devices in a low power mode;
- **MCU** – microcontroller unit, a microcircuit that consists of a central processor, RAM and flash memory, input-output ports;
- **MAC address** – a unique identifier assigned to each unit of active equipment or some of their interfaces in computer networks;
- **FW Master Key** – Root certificate for signing Hideez Key firmware (ECDSA);
- **Bond** – (bond file) – Bluetooth pairing information of the device including MAC address and AES-128 encryption key;
- **Device Master Key** – AES-128 encryption key to authorize and encrypt user data on the device;
- **MITM** – Man-in-the-middle attack;
- **AES** – Advanced Encryption Standard – symmetric encryption algorithm;
- **RSA** - Rivest–Shamir–Adleman – asymmetric encryption algorithm;
- **ECC** – Elliptic-curve cryptography – asymmetric encryption algorithm;
- **ECDSA** – Elliptic-curve Digital Signature Algorithm;
- **ECDH** – Elliptic-curve Diffie–Hellman;
- **OTP Secret** – a secret key for generating one-time passwords;
- **OTP** – one-time password or time-based one-time password (TOTP) based on RFC 6238 specification;
- **TPM** – Trusted Platform Module;
- **AD** – Microsoft Active Directory server (on premise);
- **FIDO** – Fast IDentity Online, the passwordless authentication standard and the eponymous alliance that develops and promotes this standard;
- **U2F** – Universal two-factor aithentication, an authentication standard using a hardware token as a second factor, developed by the FIDO alliance. FIDO2 - a passwordless hardware token or TPM authentication standard developed by the FIDO Alliance;
- **HSM** – Hardware Security Module, a physical computing device that safeguards and manages digital keys, performs encryption and decryption functions for digital signatures, strong authentication and other cryptographic functions.

Hideez Enterprise Solution Overview

> ⚠ **It is recommended to read the Hideez Enterprise Solution documentation to get a general idea of its functionality before reading this document.**

Hideez Key is a multifunctional hardware device with Bluetooth LE, NFC and USB interfaces designed for storing user credentials, generating one-time passwords, performing authentication according to FIDO2 and FIDO U2F standards, and generating digital signatures. The user documentation may also refer to it as Hardware Vault (see Hideez Key (Enterprise version) for details).

The device supports both proprietary and standard protocols (FIDO2 and FIDO U2F). And while support of the standard protocols is built into the modern operating systems, additional infrastructure is required for the rest of the functionality. The infranstructure consists of the following components:

- **HES** – Hideez Enterprise Server is a web server for centralized management of users, credentials, hardware devices and other elements of the Hideez infrastructure (see Hideez Enterprise Server).
- **Hideez Client** – a client application for the Windows platform which enables user interaction with the Hideez Key device and with the HES server (see Hideez Client App). This is an open source project (see our Github page for details).
- **Hideez Mobile Client** is a mobile application for Android and iOS designed to store a backup copy of user data and provide a backup method for logging into a user's computer.
- **Crypto.dll** – cross-platform C ++ library containing cryptographic AES-128, ECDH, ECDSA algorithms required to work with the Hideez Key. It also contains the FW Master Key public key.
- **Communication.dll** – cross-platform .Net library that contains low-level code allowing to work with Hideez Key devices.
- **Hideez Dongle** is a hardware USB adapter providing Bluetooth LE interface to connect with Hideez Key devices (see Hideez Dongle).
- **HLS** – Hideez License Server, a web server for end user license management. The server generates public key certificates for each device after ir is produced. The server uses the Azure Key Vault to store the FW Master Key and License Master Key private keys.
- **Azure Key Vault** – Microsoft Azure cloud service for storing encryption keys and performing cryptographic operations. We use keys stored in HSM (FIPS 140-2 Level

3 certified) with the private key export option disabled. This therefore ensures that the private key will not become known to third parties. In addition, Azure Key Vault makes it possible to log all operations with the key, which prevents unauthorized use.

- **Hideez Factory App** – a desktop application for testing and flashing devices at the factory. The application communicates with HLS via a web API to obtain serial numbers and public key certificates for devices.

- **Device Maintenance App** – a desktop application for updating firmware of the Hideez Key devices.

The entire infrastructure is built on the principle of **Security by Design** – any function in the system is primarily designed to be as secure as possible.

What critical data is stored inside Hideez products?

1. Hideez Key

The new Hideez Key, if not assigned to a user, contains the following critical data stored on the internal flash memory:

- Unique device certificate including ECC-256 public and private keys. The public key and serial number of the device are signed with the FW Master Key.
- Public part of the FW Master Key (ECC-256) to verify the firmware signature.
- Public part of the License Master Key (ECC-256) to verify the license signature.

Once the device is assigned to the user, the following data can be stored on the internal flash memory (in encrypted form):

- AES-128 standard encryption key for Bluetooth traffic (bond file).
- User passwords, logins, site addresses.
- Custom OTP secrets.
- Custom FIDO and U2F keys.
- User PIN to access the device.

The device's RAM stores:

- Device Master Key (AES-128) – data storage authorization and encryption key.
- Session keys of virtual communication channels (AES-128).

2. Hideez Enterprise Server

The server database stores:

- Unique Device Master Keys to access the encrypted storage of each Hideez Key.
- User data – name, email, phone, job title, etc.
- List of user accounts with information about login, site address or application name.
- User passwords saved as Shared Accounts.
- User passwords which are intended to be saved to user devices.

Server configuration files contain:

- Credentials for accessing the SMTP server.
- Credentials for accessing the Active Directory server (optional).
- Database encryption password (optional).

The server includes Crypto.dll which stores:

- FW Master Key public key for device authentication.

3. Workstatins on which Hideez Client is installed

The computer hard drive stores:

- AES-128 standard encryption key for Bluetooth traffic (bond file).

The main memory stores:

- Session keys of virtual communication channels.

The server includes Crypto.dll which stores:

- FW Master Key public key for device authentication.

4. Hideez Mobile Client

The mobile app stores an encrypted copy of all user data. The encryption key (AES-256) is generated on the phone using a cryptographically strong random number generation algorithm. The key is stored in the TPM, while access is protected by a biometric sensor (if available) or a PIN code.

5. HLS + Azure Key Vault

Server configuration files contain API keys for accessing Azure Key Vault, which in turn stores:

● Private part of the FW Master Key (ECC-256) for signing device certificates in production and for signing firmware.

● Private part of the License Master Key (ECC-256) for signing device licenses.

Both keys are generated within the HSM and can not be exported.

6. Hideez Factory App

This application does not contain any sensitive data. HLS access credentials are entered each time the application is launched.

What critical data is transferred between system elements?

Hideez Client ⟷ Hideez Key

- User logins, passwords, site addresses;
- User PIN code;
- Device Activation Code;

HES ⟷ Hideez Key

- User logins, passwords, website addresses;
- Device Master Key;

HES ⟷ Hideez Client

- User PIN code;

HES ⟷ Browser

Data that can be transmitted from the browser to the server:

- Credentials of the user who is authorized on the site;

- Personal data of users when they are edited or added;
- User accounts when they are edited or added;

Data that can be read from the server and displayed in the browser:

- Personal data of users;
- Metadata of user accounts (passwords and OTP secrets are not transferred to the browser);
- Device Activation Codes;

HES ⟶ Email, SMS, etc.

- Device Activation Code;
- Link to the password reset page which contains the authorization token;

HLS ⟷ Hideez Factory App

The following data is transmitted to the server:

- MAC addresses of devices;
- Public keys of devices;
- Results of device self-diagnostics, processor ID, version of an installed bootloader and a firmware;

The following data is transmitted from the server to the application:

- Serial numbers of devices;
- Signed public key certificates for each device that has been produced;

Hideez Factory App ⟷ Hideez Key

The following data is received from the device:

- Serial numbers of the device;
- Public key of the device;
- MAC address of the device, processor ID, self-test results;

The following data is transmitted to the device:

- Firmware, bootloader, self-diagnostic application;
- Serial number;
- Signed public key certificates;

Possible attack vectors

Having figured out where and how critical data is stored and what communication are used to transmit it between system components, we can assume the following options for cyberattacks:

Reading data from a device physically

After gaining physical access to a device that already contains user data, you can try to carry out the following attacks:

- Reading data through the microcontroller's wired programming interface (see Flash Read Protection).
- Reading data via Bluetooth / USB / NFC interfaces (see User authorization on the device).
- Invasive attacks which involve mechanical ways to open the MCU case (chemicals, laser or mechanical means) and further analysis of the data using a microscope or connecting to data lines. This is a technically complex attack that requires special expensive equipment, but is still theoretically possible (see Data Storage Encryption).

MITM attacks

Man-in-the-middle attacks are a type of attacks in which an attacker secretly relays and, if necessary, changes the connection between two parties who believe they are directly communicating with each other.

- Passively monitoring Bluetooth traffic or attempting to intercept encryption keys using a MITM attack (see Standard BLE Traffic Encryption, Virtual Secure Data Transmission Channels).
- Relay attacks which simulate the presence of a device near the receiver by reading its signal, transmitting it through any communication channel and reproducing it in the same form next to the receiving device (see Protection against Relay attacks).
- Interception of traffic between HES and client / browser (see Web traffic protection).
- Interception of data within a workstation (see Hideez Client Protection).
- Interception of data between the mobile application and the user's computer (see Mobile Client Protection).

Unauthorized access

- Unauthorized access to the HES administrator account (see Securing the HES Administrator Account).
- Unauthorized access to the HES database (see Database protection).
- Unauthorized access to the Hideez Key (see User authorization on the device).
- Unauthorized installation of the Hideez Client (see Hideez Client Verification).

Spoofing attacks

Spoofing covers a wide range of attacks such as:

- **Device spoofing** when an intruder tries to substitute a genuine Hideez device or Hideez software for one that does not differ from the original in appearance and functionality, but may contain secret methods of unauthorized access or unauthorized methods of transferring confidential data to third parties (see Hideez Key devices production process).

- **Firmware spoofing** – the same as imitating the entire device, but it is assumed that the attacker does not have direct physical contact with the device and needs to change the firmware. (see Hideez Key devices production process, Firmware update).

- **HES server spoofing** could allow data reading of all from devices remotely (see Installing a server in the company perimeter, User authorization on the Hideez Key device).

- **Hideez Client** application spoofing (see Hideez Client Protection, Open Source Code Development).

- **Device duplicating**, including serial number and MAC address for the purpose of illegal production. In general, this is not an attack on confidential data, but it can cause reputational and financial damage the company (see Hideez Key devices production process).

Hideez Key Device Protection

Data protection in the flash memory (Readback protection)

Many microcontrollers (MCUs) provide functionality known as **readback protection** or **readout protection**, which prohibits to read internal memory through debug or programming interfaces. The Nordic nRF52 processor used in the Hideez Key device is no exception. This protection is enabled at the manufacturing stage of the device once the firmware has been installed and a serial number has been assigned to it. This means that it is no longer possible to read anything from the device using such interfaces as JTAG or SWD.

But given that Nordic MCUs are general-purpose processors, and not specialized devices for protecting information, this protection is not enough and other protection methods, such as data and traffic encryption, must be applied.

Protection against Relay attacks

Hideez Key devices send data over a radio channel which theoretically allows an attacker with special skills and equipment to intercept the radio signal, save it, transmit it over a considerable distance as a digital file and reproduce it elsewhere.

For example, a user has configured his PC to be unlocked with the proximity of the Hideez Key. Let's assume that it is far from the PC, and the hackers know its location and also have physical access to the attacked PC. If you place the radio sniffer next to the user, you can record his Hideez Key's signal. Although the data is encrypted, it can be recorded and transferred for playback next to the attacked PC without modification.

If the Hideez security system was built on one-way data exchange (for example, on iBeacon technology), the attack could be successful. The Hideez system therefore requires a two-way Bluetooth connection and severe restrictions on the response time. If the delay in the response signal has the slightest deviation from the norm, the connection will be dropped.

Standard BLE traffic encryption

The BLE protocol has been providing traffic encryption according to the AES-128 standard since its first specification. Hideez Key devices have this encryption as well. The encryption key is generated randomly for each pair of devices (computer and Hideez Key). The encryption key is saved on both devices during the very first connection and is never broadcast over the radio channel afterwards. This process is called "bonding". This is a vulnerable stage, since the channel has not yet been encrypted, and the key can be intercepted by an attacker using special equipment, which can make it possible for the entire exchange to be intercepted and decrypted in the future. Also, encryption keys (bond files) stored on end devices, Hideez Key and the user's computer, become a soft spot. In this regard, we have implemented virtual channels for secure data transmission on top of the standard Bluetooth encryption.

Note: The bonding process in the BLE 5.0 specification already allows for ECDH key exchange which provides protection against passive eavesdropping, but it is still susceptible to MITM attacks. Our virtual channel algorithm (described below) provides the same mechanism based on ECDH, but with pre-signed public key certificates, which excludes an MITM attack if the attacker does not know the private FW Master Key.

Hideez Dongle

The next vulnerability after standard Bluetooth encryption is the implementation of the Bluetooth protocol in operating systems. The problem is that all connected Bluetooth devices are available to all applications in the system, and any of them can receive data from the device in clear text. A standard Bluetooth encryption is valid only up to the driver level, but not up to the user application level. In this case, the application does not require administrator rights, and even if the device is already opened by one of the applications, everyone else can connect and view the incoming traffic. The problem is eliminated by virtual channels of secure data transmission. And,

in addition, we use our own Bluetooth adapter and our own BLE stack implementation. The Hideez service launched on a client computer gets access to the Hideez Dongle in an exclusive mode which prevets other applications from monitoring traffic. Moreover, the dongle allows you to:

● *avoid dependence on errors in the drivers of various Bluetooth adapter manufacturers;*

● *avoid dependency on Microsoft's BLE stack implementation which may change when new versions are released (this has already happened many times);*

● *reduce the burden on the built-in Bluetooth and Wi-Fi adapter (it often contains a common antenna for both protocols).*

Virtual channel for secure data transmission

This is the most essential aspect of the Hideez Key device's security. Having established a connection with a device via Bluetooth, the client can initially execute only a few basic commands (e.g., obtain information about the device, serial number, hardware and software characteristics, etc.). To gain access to the device memory, you must pass authorization and establish an encrypted communication channel. The device can simultaneously establish up to six channels with different end clients and different encryption keys, so they are virtual. Moreover, they all work on top of one physical Bluetooth connection.

When a device is initially flashed at the factory, each device generates a set of keys for authorization and authentication. This procedure is performed only after the device is locked for reading, so the private keys of the device cannot be read. Public keys and device data (DeviceID, SerialNumber) are transmitted to the HLS server which signs them with a Hideez master key (ECDSA), so we receive public key certificates that are recorded back to the device.

When a virtual communication channel is established, the device is verified using a digital signature, an AES-128 session encryption key is generated, and an encryption key is exchanged via ECDH. The AES-CTR-128 method is used to encrypt messages. The message is aligned to a 16-byte boundary, and then encrypted using the EncryptionKey and Counter. The Counter is incremented for each package. The encrypted message is superimposed on CMAC-128 (16 bytes) with the MACKey key (added to the end of the message). If a packet size greater than 16 bytes is used, the MAC[1] is added to the end of the last packet. Until encryption is set, data is transmitted unencrypted, as MAC array 0x00.

A complete description of the procedure for personalizing the device, authentication and encryption of the communication channel is beyond the scope of this document. It should only be noted that the following advantages are achieved as a result of all these procedures:

● *The host can make sure that the connected device contains a valid Hideez signature and is a genuine device.*

● *The host can verify the signature of the serial number of the device, which prevents it from being tampered with.*

[1] In this context, MAC is the Message Authentication Code

● *Peer-To-Peer encryption is performed between host and device. For example, when HES opens a virtual communication channel, data is encrypted between the Hideez Key and the HES, and all intermediate devices and systems (network servers, user computers, Hideez Client application, etc.) are only used to transit data and cannot receive access to them.*

● *No keys, including Device Master Key and session key, are transmitted in plain text.*

User authorization process

Access to the device is limited due to the following security mechanisms:

● **Confirmation by pressing a button** allows you to exclude remote connection to the device, unbeknownst to its owner. It will not be connected without confirming the action by pressing the button.

● **PIN code**. After device has been connected, access to its memory can be obtained after entering the user's PIN code. This PIN is invented by the user at the moment of device activation and should be known only to him. In case of several unsuccessful attempts, the device is locked. You can unlock it only with the help of the HES administrator. The PIN code is stored on the flash memory of the device and is saved after a reboot. The length of the PIN code can be from 4 to 8 characters.

● **Device Master Key**. The master key is used to encrypt all user data on the flash memory. It is generated on the server using a cryptographically strong random number generation algorithm. Then the key is transmitted via an encrypted virtual channel to the device it never leaves the HES in cleartext and is not displayed to the user. On the device itself, the master key is stored only in the device's RAM. When rebooted (for example, when changing the battery), it is reset and needs to be re-entered. This requires a connection to the HES, since the keys are permanently stored only in the HES database, the user does not know them.

Device security profiles can be customized according to the needs of the client (see How to create and set device access profiles for more details).

Thus, one or more of the following actions may be required for user authentication, depending on the settings of the security profile and the state of the device (whether Device Master Key is loaded into RAM or not):

- Pressing a button on the device,
- Entering the PIN code,
- Server authorization for entering Device Master Key.

If the requested action cannot be performed, access to the device remains blocked. There are important differences between PIN and Device Master Key:

- The PIN code is known to the user;
- PIN is used to authenticate the user;
- The number of attempts to enter the PIN code is limited (3-32 attempts, after which the device is blocked);

● A locked device can be unlocked after the administrator has confirmed it and the user has changed the PIN;

● PIN code can be changed if necessary or in case of its loss;

● PIN is stored on the flash memory of the device itself;

● PIN is not stored on the server and is not known to anyone except the user.

Master key:

● Remains unknown to the user. It is generated by the server when the device is assigned to the user;

● Never leaves the server in cleartext;

● Is used for server authentication and for encrypting user data;

● If the master key is lost, it cannot be restored, the data encrypted with this key will be lost;

● Is active from the time of its generation until the key is untied from the user;

● Is stored on the server and in the device's RAM.

As a result of the above mentioned protection methods, the following advantages can be achieved;

● It is impossible to connect to the device without the user being aware (he is required to press a button);

● It is impossible to connect the device to a computer that does not belong to the company's network and is not confirmed by the system administrator (confirmation of the computer by the admin and entering the master key via HES is required);

● It is impossible to access the device without knowing the secret (PIN code);

● The device is blocked when attempting to hack it (guessing a PIN code);

● It is impossible to obtain the device memory decryption key from the device itself (the master key is stored only in the RAM and will be erased upon reboot / shutdown).

Storage encryption

Once the master key is transferred to the device, it is stored in RAM and is used not only for authorization, but also for encrypting user data on flash memory. Encryption is performed according to the AES-128 standard, since this algorithm has a hardware implementation inside the MCU. Data is encrypted "on the fly" in a way that is transparent to the user.

First of all, the master key must be transferred to the device. This can be done only through the HES, because the user does not know the master keys of his devices. To transfer the key, a virtual encrypted channel is organized, through which the server verifies the authenticity of the device and its serial number. Then the authorization command is executed, to which the master key is

passed. The device checks it and, if it is correct, allows access to the data over this channel. Other channels must be authorized separately, but the master key is saved in the device's RAM after it is transmitted over any of the channels. The server is always authorized with the master key, since it does not know the user's PIN. In this case, confirmation by pressing the button is not required. This allows the server to connect to the device at any time and gain full access without user involvement.

The channel authorization process for the user depends on the security profile settings and the presence of the master key in memory:

- User clicks a button, access is granted;
- The user enters a PIN code and / or presses a button;
- The user enters the PIN code and / or presses the button, and waits for server authorization via a separate channel.

Advantages:

- If the device is stolen / lost, the user data cannot be accessed as it is encrypted;
- The encryption key is not known to the user, does not leave the server in cleartext, and is not stored on the flash memory of the device.

Device activation

Before the user starts using the device, they must activate it. The activation code is generated on the server once the device is assigned to the user. It is unique for each operation. The activation code must be communicated to the user through any communication channel (e.g. by phone, SMS or email). The user must enter this code (it consists of 6 digits) into the Hideez Client application interface. Once the device is activated, the user goes through the authorization process.

If the activation code is entered incorrectly three times in a row, the device is blocked and the involvement of the system administrator is required. The administrator unlocks the device on the server. As a result, a new activation code is generated and sent to the user. If the device is blocked due to incorrect PIN code, it must be unlocked on the server and an activation code must be generated. After entering the activation code, the user comes up with a new PIN code.

Advantages:

- If the device falls into the wrong hands, one will not be able to use it. The activation code is communicated through a separate communication channel.
- If the PIN code is entered incorrectly, the functionality of the device can be restored without losing data.

Hideez Key production process

Our company completely controls information security in the device's production process. The production company (factory) does not receive any confidential information, so we can use different contractors without the threat of data leakage.

To this end, we have developed the Hideez Factory App which is connected via a web-API to our HLS server and is connected to the device programming machine through a wired interface.

The factory receives bootloader files before the production of each batch. They contain a public key for verifying the signature of the firmware and the firmware itself, which has already been signed and encrypted by us. The HLS server prepares information about the next batch (the initial serial number, and the planned number of devices).

Firmware flashing includes:

- Uploading test firmware and self-diagnostics of the device;
- Flashing the working firmware and bootloader;
- The Factory App transmits the MAC address of the new device to the HLS and receives the next serial number in response. The serial number is recorded in the device memory;
- The device is locked for reading, further communication is possible only via the Bluetooth interface;
- The device generates a set of ECC keys for authorization and authentication;
- Public keys together with device metadata (DeviceID, SerialNumber) are transferred to the Factory App, which in turn transfers them to HLS;
- HLS signs the received public keys of the device along with the metadata with its private key, and returns the received public key certificates written to the device.

Advantages:

- It is impossible to produce a device without us knowing about it;
- Each device has a unique serial number and unique ECC keys to verify its authenticity;
- Public key and serial number of the device are pre-signed by the Hideez master key, making it impossible to perform MITM attacks when exchanging keys via ECDH;
- The bootloader contains a public Hideez master key for checking firmware that will be downloaded via Bluetooth / USB in the future, which makes it impossible to replace the firmware.

Firmware update

Hideez Key software is constantly being improved. Therefore, we have provided a way to update the firmware via Bluetooth.

The device software consists of two parts – bootloader and firmware. A bootloader is a special application that updates the firmware inside the device. It contains a public key for verifying the firmware signature.

You need two parties to update the firmware. The transmitting party is the Hideez Maintenance application installed on any computer. The receiving one is the bootloader.

The update process consists of the following steps:

- Hideez Maintenance application switches the device to the Bootloader mode. To this end, a special command is sent after establishing a Bluetooth connection;
- Maintenance application uploads new firmware to the device;
- Once the firmware is uploaded, the bootloader checks the digital signature of the firmware;
- If the signature is valid, control is transferred to the new firmware. If not, the device remains in bootloader mode and waits for the firmware to be uploaded with the correct digital signature.

User data is not affected during the update.

The firmware file must be pre-signed. To this end, the HLS server also has the functionality of a build server. As soon as a Commit appears in the master branch of the firmware repository, it is automatically signed with the Azure Key Vault. This means that, firstly, the developers (company employees) cannot sign the firmware on their own. Secondly, the signed firmware can only contain the code that was uploaded to the GitHub repository. Each signature operation is logged by the Azure Key Vault service itself.

All these precautions are explained by the fact that knowing the master key for signing the firmware allows you to sign any firmware, and this firmware will be accepted by the Hideez Key device, which will allow you to get full control over it. The master key can be changed for each new batch of devices.

Reassigning devices between users

Hideez Key devices can be passed from one user to another because once the key is removed from the user, it is completely cleared.

- The administrator removes the device from the user.
- The device is marked as deactivated and requires cleaning.
- The next time the device is connected to any computer, the server establishes a connection, authorizes and performs cleaning.
- The device master key is deleted from the server database.

● The device is marked as free and assigned to another user; a new device master key is generated in the database.

● At the moment of the first connection, the master key is transmitted to the device for authorization and initialization of memory encryption.

Old devices can thus be reused without compromising security.

Hideez Client Protection

Each file in the Hideez Client installation package is necessarily signed with a certificate issued by Sectigo RSA Code Signing CA. The operating system verifies this certificate when installing the application. You can also develop the application yourself from the source codes by downloading them from the repository.

Data that is transferred from a device or to a device is encrypted twice – with standard BLE encryption and additionally using a virtual communication channel. Standard Bluetooth encryption is valid only up to the driver level, but not up to the user application level (see Hideez Dongle). The virtual communication channel decrypts data directly at the recipient. It can be the Hideez Client or HES UI application (when the Hideez Client acts as a relay). Critical data is not transmitted between computer processes in cleartext, as well as over a radio channel or network interfaces. Therefore, attackers will have to get them from RAM, which is much more difficult from a technical point of view.

If an attacker has access to and control of the user's computer, it is impossible to guarantee 100% protection of the data stored on the Hideez Key. Preventive measures can include an anti-virus program, timely updates, and correctly configured security policies.

HES protection

HES does not store user passwords, except while the password is pending delivery to the device and when Shared Accounts are being used. This significantly reduces its attractiveness to hackers, but it still remains a soft spot in the security system, as it stores access keys to all Hideez Key devices. Having received the keys and physical access to the device, an attacker may try to read user data.

If the device's master key is lost, the ability to decrypt the data is lost. The only thing you can do is to wipe the key and use it again. In this regard, a number of measures have been taken to ensure the security of the server.

Server deployment within the perimeter of the company

HES is not a cloud service, but a web application originally designed to work within a company's network. This allows you to isolate it from outside hacking attempts and exclude employee access to data by employees of cloud data centers or by Hideez employees. That is why we provide HES hosting services for demonstration purposes only. Even so, we believe that hosting HES in cloud services such as AWS or Azure is acceptable, provided that the following conditions are met:

- the hosting provider is reliable and has a good reputation;
- a commercial certificate is used to encrypt traffic (SSL, HTTPS);
- the Data Protection function is enabled, all recommendations for its configuration and use are followed;
- two-factor authentication is enabled for all HES users, recommendations on password complexity and storage are followed;
- access to a virtual machine with installed HES has a limited number of persons, two-factor authentication is enabled, recommendations on the complexity of passwords and their storage are followed;
- the virtual machine on which HES is installed receives all the necessary security updates in a timely manner.

HES Administrator account protection

Access to HES is carried out using a username / password and using a second factor according to the FIDO standard (optional). Originally, the server contains one administrator, but you can create an unlimited number of them. Recommendations for the complexity of the administrator password and its storage are generally accepted and are beyond the scope of this document.

Web traffic protection

HES should only be configured to use the HTTPS protocol. Use commercial (not self-signed) certificates to this end.

Database protection

Data Protection solves the problem of secure data storage in the database. First of all, there are Device Keys, encryption keys allowing to access the contents of Hideez Key devices. The database also temporarily stores passwords and OTP Secrets awaiting transmission to devices. If these are passwords for a Shared Account, they are permanently stored in the database. If Data Protection is not enabled, all passwords and keys are stored in clear text.

Therefore, it is necessary to assess which one of the employees has direct or potential access to the HES database. If there are those who should not have access to confidential data listed in the previous paragraph among them, Data Protection must be enabled. Please also take into account that data can be read physically from the HDD / SDD on which it is stored, i.e. you need to consider the possibility of both software and physical access to data.

Data Protection can be turned off in some cases, for example, if HES and the database server are running on the same physical server, access to which is restricted to trusted people. All confidential fields in the database are encrypted with the AES-256 algorithm. The encryption key (password) is entered manually by the administrator after starting the web server and is then stored only in the server's RAM.

If the server restarts, the encryption key will be erased and all administrators will receive an email asking them to enter the password from Data Protection. The server does not process incoming requests until the password is entered. Once the password is entered, the switches to the normal operating mode, and the data is decrypted on the fly.

The Data Protection password can be placed in the config file or Environment Variables on the server, so as not to enter them manually. In this case, the server can reboot and start working without administrator involvement. It is obvious that access to the server should be available only to authorized persons in this case.

The password for Data Protection can be changed. To do this, you need to enter the old password and the new password in the HES interface. All data in the database will be re-encrypted. The main risk when using Data Protection is the loss of the activation password. In this case, you lose access to encrypted data on the server, as well as access to all data on your Hideez Key devices. All devices will have to undergo a manual reset procedure with a complete memory wipe. Then you can reassign them to employees and re-add all accounts to devices. Based on this, the following recommendation can be made: the Data Protection password should be known to several employees. It should also be stored in a safe place on a physical medium.

In case of using Data Protection, there is also a risk that the HES server will be unavailable, in the event of an emergency restart, until the password for activating Data Protection is entered. But this should not affect the work of the company, since all accounts are stored on employees' devices (Hideez Keys) and can be used without having a connection with the HES server.

Changing the Data Protection password must be done in accordance with the company's information security regulations.

Hideez Client Verification

In most cases, it is required to prohibit the use of Hideez Key devices on computers that do not belong to the company's domain. To do this, each computer with the Hideez Client installed must be confirmed by the system administrator, all other computers will not be able to connect devices. This limitation cannot be circumvented by modifying the client application, since server authorization is required to connect the device, and the server will not execute it through an unverified computer.

The process of connecting a device to a computer for the first time (bonding) is as follows:

- If this is a connection to a new computer, the device does not yet have a bond for it and it is being created.

- The created bond is marked as unauthorized until authorization is made from this computer using the device master key.

- The master key is known only to the server, so the client sends an authorization request to the server and creates a virtual communication channel for this.

- The server checks that the given computer is authenticated and, if it is not, returns an error and drops the connection with the client.

- The server is authorized on the virtual channel, initiates encryption and transmits the device master key over the encrypted channel.

- The device verifies the key and authorizes this bond. After that Hideez Client can work on this computer without connection to the server.

Thus, it excludes the possibility of using the device on a home computer / any other computer or reading its memory.

HLS Protection

The same protection methods are applied to the HLS server as described for HES – two-factor authentication, Data Protection, restricted access to the server virtual machine. Unlike HES, the HLS server exists in a single copy. The database and the HLS web application are installed on the same physical server, access to the database is only allowed on the internal port.

The most critical information contained on this server is API keys for accessing the Azure Key Vault, where private keys are stored for manufacturing devices, signing firmware and generating licenses. The rest of the information is stored on the HLS server (list of clients, devices, licenses, etc.) and falls into the category of business data that is not critical from the point of view of clients' information security and no special protection measures are applied to it.

Open source development

Open source development is widely used for cybersecurity-related software products. First of all, it increases the credibility of the software product due to the ability of many developers and experts to audit the source code. Secondly, it gives the end user the opportunity to build the application on their own. This eliminates the embedding of unwanted or spyware functionality into it. This is especially critical for applications that are installed on users' computers, namely Hideez Client and Device Maintenance. We have opened the source codes for these applications, they can be viewed on the GitHub website (Hideez Client).

About Hideez

Hideez provides innovative IAM solutions ensuring both secure and usable authentication and eliminating the human factor risk.

Our main product, Hideez Key, is a multifunctional Bluetooth/NFC security key & password vault which can store and input login credentials at the push of the button, replace 2nd factor based on TOTP and FIDO U2F support, and help entities move beyond password-based authentication with FIDO2 passwordless standard. The device can lock employee workstations by proximity and unlock RFID door locks in the office, being a perfect fit for complex multi-user environments in any domain, including banking, finance, healthcare, education, IT, and other most vulnerable industries.

Employees can focus on their work, while Hideez takes care of their authentication and compliance with security protocols. Our IAM Solution integrates with multiple websites and apps, closing the gap of SSO adoption.

Find out more about us

www.hideez.com

Let's start a discussion together