Security Overview
Humly Room Display - HUM1001

Humly Room Display

System security

Security can not be added as an afterthought and it should not be seen as a hurdle to pass. Security is a continuous, never ending process that needs to be considered from the beginning. For us it is the very DNA that goes into the products already from the conceptual stage. For over a decade the biggest enterprises, banks, law firms and governments around the globe have been using devices and systems engineered by our team.

To meet the requirements of our most demanding customers we make sure we have full control of every single component that goes into the hardware and every line of code that makes up its software. The Humly Room Display system is custom design hardware running a tailored Linux based firmware with our own frontend and backend applications developed inhouse. For every update and improvement we put a lot of effort into staying one step ahead of hackers - and the competition.

Cadence of firmware and application upgrades

We continuously improve our products and provide up to three bigger updates per year and a few minor updates. In case of special urgency we support a release cadence as short as 2 weeks. As a part of our release management we apply a penetration testing regime where the application software is tested by an external team of penetration testers. In addition to a hacker approach they also have source code access to locate potential vulnerabilities that would beyond any scope for a regular penetration test[1]. Penetration tests are done on all major releases (**X**.Y.Y), and on all normal releases (Y.**X**.Y).

Processes, protocols and principles

Our secure product development lifecycle is based on threat assessment and processes from several sources, but the two main ones are ENISA *Hardware Threat Landscape and Good Practice Guide*[2], *and* CIS Security *Cybersecurity Best Practises*[3]

In this document we would like to highlight a few key principles that provide several layers of security for our products.

---

[1] White box testing in addition to black box testing
[2] https://www.enisa.europa.eu/publications/hardware-threat-landscape
[3] https://www.cisecurity.org/cybersecurity-best-practices/

Minimize exposed surface

The operating system is custom built based on the Yocto project with board support package from NXP/Freescale. It is a stripped down version that only includes services and functions that we actually use to make the attack surface as small as possible.

There is no way to leave the application layer and reach underlying systems and all I/O interfaces beyond the touch screen and RFID/NFC reader are disabled by default and only available for limited use cases initiated by an authenticated administrator[4]. Any tampering will lock the device out from the service and alerts the system administrator.

The devices only need a single port of encrypted communication on the network[5], effectively turning them into a dead end for any potential malicious user who tries to use the connection as an entry point.

Same policy also applies on the server side. In addition to the aforementioned traffic between device and server only one additional port is required for encrypted communication with the meeting booking or calendar system. However in a default setup we also have an internet connection over 443 for licensing, firmware upgrades and support[6].

Restricted access to functions and methods combined with input validation

All server and client side functions are restricted based on authentication level, and all external and internal input from authenticated users passes validation to avoid code or database injections.

Information protection

The devices only store information that is already available to display on the device. No customer credentials or other sensitive information is held in the device. Even the meeting details are stored in non-persistent memory in case a device would be stolen.

Server side we only handle the least amount of data required to operate the system so in the unlikely event of a breach there should be no sensitive information to access.

---

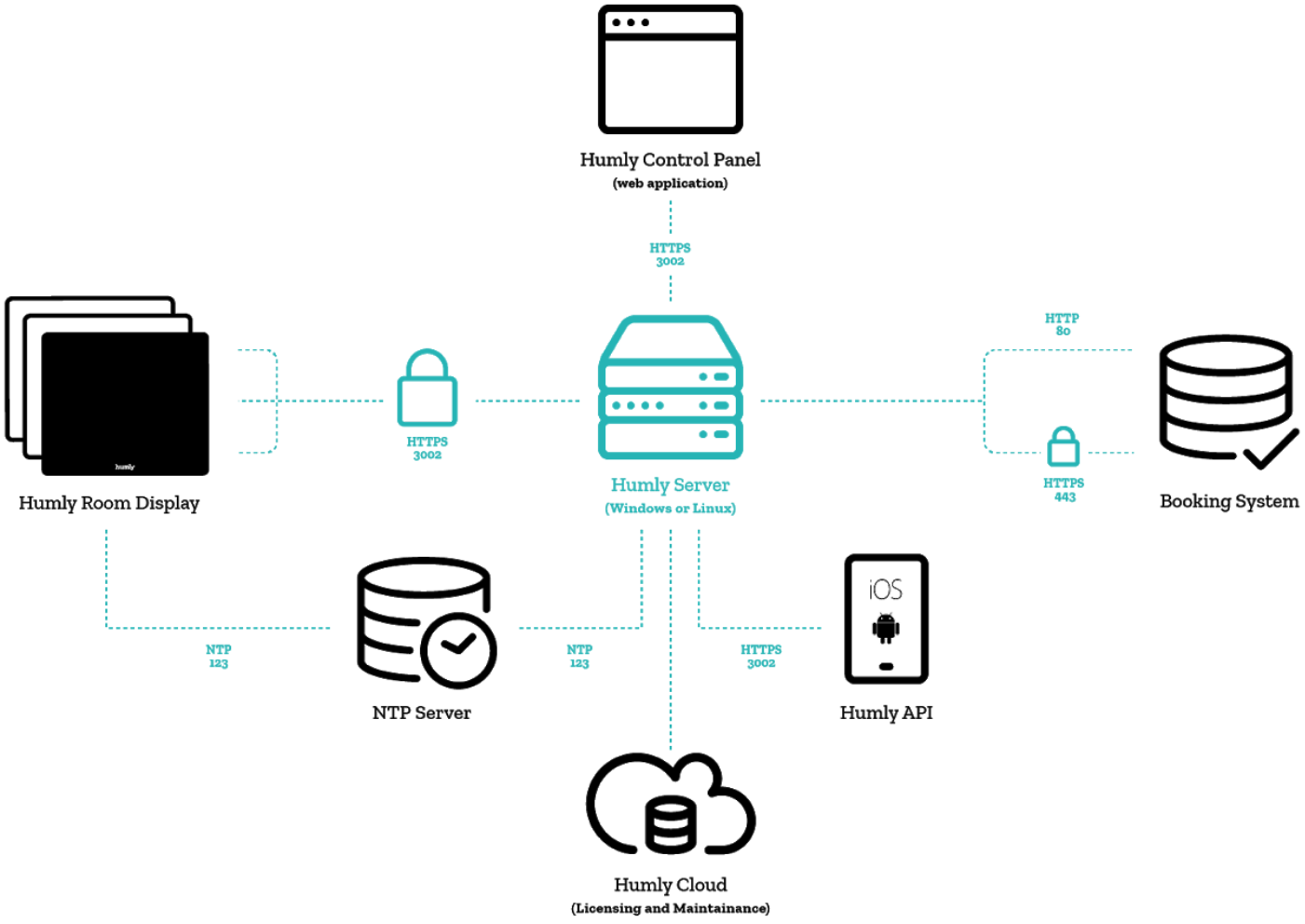
[4] Devices can be provided in a locked down version where wifi, bluetooth and USB port have been disabled on hardware layer.
[5] More detailed information on this in the System Architecture overview document
[6] This access can be done through a proxy that limits traffic to this specific purpose. It is also possible to set up the system without any outside connection, but it will require more manual work for licensing, firmware upgrades and support.

Network security

The overall system security is largely depending on the environment in which it lives. We recommend setting up the devices on a separate VLAN that is locked down to limit access to only the Humly Server and if needed a separate NTP server. The server on which Humly Server is installed must only be accessible by authorized administrators and passwords to both server and the Humly Control Panel must be secure and protected.

For customers interested in deploying the Humly Room Displays on an 802.1x protected network we provide basic support on a handful of configurations and are looking to expand this support together with customers on real pilot project

System overview



- The Humly Server communicates with Humly Room Displays via HTTPS over a configurable port (default 3002).
- The Humly Control Panel web application communicates with the Humly Server using configurable HTTP port during initial setup to configure SSL certificates. After SSL has been configured all communication uses HTTPS on the port configured during initial setup (default port 3002).
- The Humly Home server communicates with the booking system via either HTTP (80) or HTTPS (443)
- The system uses configurable NTP servers using UDP over port 123 to synchronize time and date.
- The system will connect to Humly Cloud using HTTPS over port 443 to validate and renew licenses and provide maintenance services like updates and error logging.
- Other devices and integrations can be allowed to communicate with the Humly Server over HTTPS using the same port that has been configured for the Humly (default 3002