THOR-NX-T/E LiftApp

Test, Lebenszyklus & Cyber Security

Dokument Information:

Eigenschaft	Kommentar
Projektname:	Test, Lebenszyklus & Cyber Security THOR LiftApp Software
Dokument Eigentümer:	Dipl.-Ing.(FH) Roy Schneider
Dateiname:	LiftApp_LifeCycle_Security_Deutsch.odt

Dokument Historie:

Name	Version	Änderung	Datum
Roy Schneider	1.2.3	Cloud-Diagramm aktualisiert	29.05.24
Roy Schneider	1.2.4	Begriffsklärung hinzugefügt.	03.05.24
Roy Schneider	1.2.5	Hinweis zu „Techniker vor Ort“ hinzugefügt.	17.05.24
Roy Schneider	1.2.6	Update zu MQTT und Compileroptimierungen.	06.06.24
Roy Schneider	1.2.7	Verantwortlichkeiten White-/Gray-/Black Tests	12.06.24
Roy Schneider	1.2.8	Hinweis in FR7 zur Sicherheitsschaltung.	03.07.24
Roy Schneider	1.2.9	Vorlage Vorfallbericht hinzugefügt.	17.09.24
Roy Schneider	1.3.0	Kapitel Software Update/Backup hinzugefügt.	20.09.24

Dieses Dokument verwendet die Schriftart „**OpenSans**“, die unter der Apache-Lizenz 2.0 lizenziert ist.

Icons und Symbole wurden ordnungsgemäß lizenziert von **Axialis IconWorkshop™**.

Erscheinungsdatum: 20.09.2024

THOR Engineering GmbH

Inhaltsverzeichnis

1 Normenverweise....5
2 Firma....6
3 Copyright....7
4 Fehlerreporte....7
5 Einleitung....8
6 Abkürzungen, Zeichen und Symbole....8
7 Zweck und Verwendung....8
8 Sicherheitsinformationen....8
9 Generell....9
9.1 Einleitung....9
9.2 Bedrohungsmodell....9
9.3 Begriffsklärung....10
9.4 Datenminimierung....11
9.5 Codeanalyse und automatische Dokumentationsunterstützung....11
10 Anforderungen an das Produkt....12
10.1 FR1 (SL-T2)....13
10.2 FR2 (SL-T2)....14
10.3 FR 3 (SL-T2)....15
Anwendung....15
Sicherheitskreis....16
10.4 FR 4 (SL-T2)....17
Anwendung....17
Sicherheitskreis....18
10.5 FR 5 (SL-T1)....18
10.6 FR 6 (SL-T1)....19
10.7 FR 7 (SL-T2)....20
Energieversorgung....20
Sicherheitskreis und Sicherheitsschaltung....20
CANbus Allgemein....20
Webserver und Cloud-Schnittstelle....22
MQTT-Schnittstelle....22
11 Entwicklungsumgebung und Datenschutz....25
11.1 Entwicklungsrechner....25
11.2 Datenschutz auf Routern, Switches und anderen Netzwerkgeräten....26
11.3 Datenschutz auf beteiligten NAS-Systemen....26
11.4 Datenschutz beim Generieren von Software-Releases....26
11.5 Datenschutz bezüglich E-Mail und externer Dateispeicherung....27
11.6 Datenschutz in Thor's Lift Cloud Interface....27
11.7 Verwendung der Cloud-API-Lösungen von Google und DeepL....27

11.8 Über Schwachstellen auf dem Laufenden bleiben....28
11.9 Workflow....30
12 Meldung von Vorfällen/Problemen....32
13 Aktualisierung und Pflege der Handbücher....35
14 Versionierung....36
14.1 Beispiel....36
14.2 Nummerierung....37
14.3 Versionskontrollsystem....37
15 Dokumentieren der Softwaretests/Fuzztests....38
16 LiftApp Update & Veröffentlichungsbenachrichtigung....39
16.1 Update Dokumentation....40
16.2 Update und Funktionale Sicherheit....40
17 Sicheres Aktualisieren der Firmware....41
17.1 Erstellen einer Sicherungskopie der aktuellen Software....42
17.2 Aktualisierung über SD-/USB-Massenspeicher....43
17.3 Aktualisierung über den Cloud-Dienst....45
18 Passwortsicherheit....48
19 Aufzug Parameter-Änderungsprotokoll....50
20 Netzwerkanschluss....51
20.1 Allgemein....51
20.2 Fuzzing der Schnittstellen....51
20.3 Offene Netzwerkports....52
21 USB/Micro-SD Karten Sicherheit....53
22 DEBUG Schnittstelle....53
23 Micro-USB-Anschluss....54
24 Sicherheitskreisabfrage....54
25 NeXt® Cloud Sicherheit....55
26 MQTT Schnittstelle Sicherheit....56
26.1 MQTT Einstellungen und Verbindungsstatus....57
26.2 MQTT Zugriff auf den Aufzug....57
27 Testen eines Release Kandidaten....58
27.1 Standard Testliste....58
27.2 Erweiterte Testliste (Neue Funktionen)....61
27.3 White-, Gray- und Black-Box-Tests....62
28 Checksumme und Software Version....63
29 Außerbetriebnahme....64
30 Programmierregeln....65
30.1 Auszug....65
30.2 Grundlegende und allgemeine Richtlinien....65
30.3 Regeln und Definitionen....67
Funktionen/Methoden und Attribute....67

Klassen ableiten....69
Sprünge....70
Typdefinitionen Aufzählungen/Bitfeldern....70
Switch/Case/Default Konstrukte....71
Längere if/else Konstrukte....71
Quellcode und Headerdateien....72
Klassische C-String Operationen....73
31 Code Analyse Werkzeuge....74
32 Riskante Compileroptimierungen....75
33 SHA-Implementierung....76

1 Normenverweise

/CiA 417-1..4 Version 2	CANopen-Anwendungsprofil für Aufzugsteuerungs systeme, Teil 4: Detaillierte Spezifikation der Anwendungsobjekte
/DIN EN 81-20:2020	Sicherheitsregeln für die Konstruktion und den Einbau von Aufzügen
/DIN EN 60664/	Isolationskoordination für elektrische Betriebsmittel in Niederspannungsanlagen
/DIN EN 60950/	Informationstechnische Ausrüstung und Sicherheit
/IEC 62443-4-1/2/	Technische Sicherheitsanforderungen an Komponenten industrieller Automatisierungssysteme
/ISO 8102-20/	Elektrische Anforderungen an Aufzüge, Fahrtreppen und Fahrsteige — Teil 20: Cybersecurity

2 Firma

Thor Engineering GmbH

Koblenzer Straße 96

53177 Bonn

Deutschland

E-Mail: hq@thor.engineering

https://www.thor.engineering/

Firmensitz: Koblenzer Straße 96, 53177 Bonn

Amtsgericht Bonn, HRB 21892

USt-IdNr.: DE304473775

Mitglied der NeXt Gruppe

https://next-group.org/

3 Copyright

Copyright © 2017-24 von THOR Engineering GmbH, Bonn

Viele der Bezeichnungen, die von Herstellern und Verkäufern verwendet werden, um ihre Produkte zu unterscheiden, werden als Marken beansprucht. Wo diese Bezeichnungen in diesem Dokument erscheinen und die THOR Engineering GmbH einen Markenanspruch erkannt hat, wurden die Bezeichnungen hervorgehoben gedruckt.

Alle Rechte vorbehalten.

WARNUNG: Die in diesem Dokument beschriebenen Informationen können Fehler oder Ungenauigkeiten enthalten. Alle Informationen können verbessert oder aktualisiert werden, einschließlich der Behebung von Fehlern und dem Hinzufügen von Funktionen. Wie bei allen Upgrades, kann volle Kompatibilität, auch wenn dies unser Ziel ist, nicht garantiert werden.

HAFTUNGSAUSSCHLUSS: Diese Informationen werden Ihnen „wie besehen“ ohne jegliche ausdrückliche oder stillschweigende Gewährleistung zur Verfügung gestellt. Das gesamte Risiko hinsichtlich der Nutzung der Informationen wird von Ihnen übernommen. Die THOR Engineering GmbH macht insbesondere keine Zusicherungen oder Garantie bezüglich der Verwendung der Ergebnisse oder der Ausführung der Informationen, einschließlich, aber nicht beschränkt auf ihre Angemessenheit, Genauigkeit, Zuverlässigkeit, Aktualität oder auf andere Weise. In keinem Fall haftet die THOR Engineering GmbH für direkte, indirekte, zufällige oder Folgeschäden, die auf einen Fehler in diesen Informationen zurückzuführen sind, auch wenn auf die Möglichkeit solcher Schäden hingewiesen wurde. In einigen Gesetzen ist der Ausschluss oder die Beschränkung stillschweigender Gewährleistungen oder Haftungen für zufällige Schäden oder Folgeschäden nicht zulässig, so dass die oben genannten Einschränkungen oder Ausschlüsse möglicherweise nicht gelten.

4 Fehlerreporte

In einem komplexen technischen Handbuch werden oft Fehler nach der Veröffentlichung gefunden. Wenn Fehler in diesem Handbuch gefunden werden, werden diese in einer späteren Version korrigiert. Updates werden über die Homepage des Unternehmens veröffentlicht. Fehlerberichte können per E-Mail an uns gesendet werden. Eingereichte Berichte müssen klar, vollständig und prägnant sein. Berichte müssen eine E-Mail-Adresse und ausreichende Informationen enthalten, damit der Fehler schnell aus dem Bericht verifiziert werden kann. Beschreiben Sie bitte den Fehler und die Schritte, die ihn erzeugen bzw. reproduzieren.

Für weitere Details lesen Sie Kapitel Meldung von Vorfällen/Problemen auf Seite 32.

5 Einleitung

Die THOR-Aufzugsteuerungen sind aufregende Hochleistungs-Mikrocomputer mit hervorragender Benutzeroberfläche und Multitasking-Fähigkeiten. Ihre technologisch fortschrittliche Hardware basiert auf einem modernen Embedded Linux®-System und einem ausgeklügelten Hardwaredesign. Die einzigartige Systemsoftware von Thor bietet Technikern beispiellose Leistung, Flexibilität und Komfort bei der Entwicklung hochmoderner Aufzuganwendungen.

6 Abkürzungen, Zeichen und Symbole

Die verwendeten Symbole wurden vom Axialis IconWorkshop ™ lizenziert.

- In diesem Dokument wird der Begriff "Aufzug" statt "Lift" verwendet.
- Der Begriff "LiftApp" bezieht sich auf die Anwendungssoftware für die Aufzugsteuerung.
- Der Begriff "OS" bezieht sich auf das Betriebssystem Embedded Linux®.
- Der Begriff "THOR NX-T/E" oder einfach "THOR" bezieht sich auf die Einheit, die aus dem Referenz-Hardware-Board und dem Referenz-Softwarepaket besteht.

7 Zweck und Verwendung

Das THOR NX-T/E Steuergerät ist speziell für den Einsatz in Aufzuganwendungen konzipiert. Um einen sicheren Betrieb zu gewährleisten, darf das Gerät nur entsprechend den gegebenen Anweisungen betrieben werden.

8 Sicherheitsinformationen

Lesen Sie vor der Inbetriebnahme, Montage und/oder Wartung dieses Gerätes die Sicherheitshinweise sorgfältig durch und achten Sie besonders auf Warnhinweise, die am Gehäuse oder an den Geräten selbst angebracht sind.

Vergewissern Sie sich, dass die Warnaufkleber nicht verdeckt oder beschädigt sind.

Ersetzen Sie jedes fehlende oder beschädigte Warnschild.

Unsere Aufzugsteuerung darf nur in Verbindung mit Ihrer Dokumentation installiert und betrieben werden. Inbetriebnahme, Installation und Betrieb des Gerätes dürfen nur von qualifiziertem Fachpersonal mit elektrotechnischer Qualifikation durchgeführt werden. Qualifizierte Mitarbeiter im Sinne der Sicherheitshinweise in dieser Dokumentation sind Personen, die nach den Normen der Elektrotechnik berechtigt sind, Geräte, Systeme und Stromkreise in Betrieb zu nehmen.

9 Generell

Software macht mindestens die Hälfte der Funktion von Aufzugsteuerungsprodukten aus. Sie ist so wichtig geworden wie die Hardware selbst, auf der sie läuft. Daher ist das Testen, Dokumentieren und Pflegen der Software so wichtig wie nie zuvor in der Technikgeschichte.

Die ISO 8102-20 beschreibt die Umsetzung der IEC 62443 - *Industrielle Kommunikationsnetze - IT-Sicherheit für Netze und Systeme* für Aufzüge.

9.1 Einleitung

Da die Software so leistungsfähig und komplex ist, ist das Testen und Warten zu einem wichtigen Teil des Lebenszyklus der Aufzugsteuerung geworden. Die Bereitstellung von Aktualisierungen in Bezug auf Normen, Fehlerbehebungen, Verbesserungen und **Cyber Security** sind Teil des Kundendienstes, der nach dem Verkauf des Geräts erbracht wird. Dies gilt auch für die Dokumentation, die ständig aktuell gehalten werden muss.

So wie Hardwareänderungen in Form von Schaltplänen dokumentiert werden, müssen Softwareänderungen dokumentiert werden, da die Software das Verhalten und die Funktion des Produkts maßgeblich beeinflusst.

9.2 Bedrohungsmodell

Ein Angreifer könnte ein Jugendlicher vor Ort sein, der das Hacken eines Aufzug einfach als spannende Aufgabe sieht. Aber auch Kriminelle, die sich durch Angreifen des Aufzuges Zugang zu Räumlichkeiten verschaffen wollen, sind denkbar. Allerdings benötigen diese beiden Arten von Angreifern lokalen Zugang, der Schaden ist auf die Anlage selber begrenzt und dies fällt eher unter Vandalismus (Cybervandalismus). Diese Angreifer setzen sich einem hohen persönlichen Risiko aus und es fällt eher in die Zuständigkeit der lokalen Sicherheitsorgane dies nachzuverfolgen.

Eine echte Bedrohung, die in Betracht gezogen werden sollte sind neben „*Hackern*“ auch „*Crawler*“, automatische Skripte, die nach offenen Netzwerkports suchen, um anzugreifen und Schaden anzurichten.

Wenn wir von Cybersecurity sprechen, so sind es eher flächige Angriffe auf die Transportinfrastruktur, die betrachtet werden sollen. Mit anderen Worten, Angriffe, die von Script-Kiddies, Hackern oder Crawlern aus der Ferne, aus persönlichem Stolz, zur Erpressung von Geld oder zur Schwächung einer Einrichtung, durch Stilllegung der Aufzüge, ausgeführt werden.

Angriffe über die Fernwartungsverbindungen, also die Verbindungen zwischen Aufzug

und einer Cloud sind hierfür am geeigneten, da sie den Angreifer in sicherer Entfernung zum Geschehen lassen und es ihm ermöglichen verdeckt zu operieren. Das Verhältnis zwischen Aufwand und erreichbarem Schaden ist für den Angreifer so sinnvoll genug, um einen Angriff auf diese Art auszuführen.

Wenn wir über die ISO 8102-20 für Aufzüge sprechen, müssen wir uns die Domäne „*Essential*“ ansehen, die wiederum die **SL(T)2** erfordert, für die folgendes definiert ist:

“Verhindern der nicht autorisierten Offenlegung von Informationen, an eine danach aktiv, mit einfachen Mitteln, bei geringen Aufwand, allgemeinen Fertigkeiten und geringer Motivation suchenden Einheit”.

9.3 Begriffsklärung

Spoofing (*Hochstaplerei*) – Vorgeben, etwas oder jemand anderes als man selbst zu sein. Die gewünschte Eigenschaft wäre Authentizität.

Tampering (*Manipulation*) – Veränderung von Parameter im Gerät, im Netzwerk, im Speicher oder anderswo. Die gewünschte Eigenschaft wäre die Integrität eines Systems.

Repudiation (*Zurückweisung*) – Behauptung, dass jemand etwas nicht getan haben oder nicht dafür verantwortlich waren. Die gewünschte Eigenschaft wäre, nicht für etwas verantwortlich zu sein, etwa für die Änderung eines Parameters, der einen Vorfall verursacht hat. Aus diesem Grund verfügt die Aufzugssteuerung über ein Parameteränderungsprotokoll.

Information disclosure (*Offenlegung von Informationen*) – Jemand erhält Informationen, zu deren Zugriff er nicht berechtigt ist. Die gewünschte Eigenschaft wäre hier einfach die Vertraulichkeit zu verletzen.

Denial of Service (*Dienststörung*) – Erschöpfung der für die Bereitstellung eines Dienstes erforderlichen Ressourcen oder Unbrauchbarkeit eines Systems. Die gewünschte Eigenschaft wäre die Verfügbarkeit Systems zu stören.

Elevation of privilege (Erhöhung/Anmaßung von Privilegien) – Jemandem unerwünscht oder unbemerkt erlauben, etwas zu tun, wozu er nicht berechtigt ist. Die gewünschte Eigenschaft wäre Autorisierung.

Wenn man die Anfangsbuchstaben der Begriffe aneinanderhängt, erhält man das Kunstwort STRIDE. STRIDE ist der Name eines von Loren Kohnfelder und Praerit Garg bei Microsoft entwickelten Modells zur Identifizierung von IT-Sicherheitsbedrohungen.

9.4 Datenminimierung

Generell speichern wir bei Thor Engineering nur das Minimum an erforderlichen Daten zur Planung, Entwicklung, Test, Konstruktion und E-Mail Konnektivität. Daten, die keinen direkten Bezug zu den Prozessen bei Thor Engineering haben, werden flüchtig gespeichert. Zu den gespeicherten personenbezogenen Daten können Name, Adresse, Anrede, E-Mail und Telefonnummer sowie die Art der Geschäftsbeziehung und Position im Unternehmen gehören. Dazu gehören Fehlerberichte, die sich auf Software- oder Hardwareprobleme beziehen, die auch die Kontaktperson sowie die zur Reproduktion des Problems aufgezeichneten Daten enthalten können.

9.5 Codeanalyse und automatische Dokumentationsunterstützung

Um Fehlern beim Schreiben des Codes vorzubeugen, werden die Entwickler von Code-Analyse-Tools unterstützt, wie etwa CDT-*Code-Analysis* von Eclipse und dem hervorragenden ***CPPCheck*** *Projekt*. Ein Beispiel wäre die Verwendung falscher Formatbezeichnern oder die Verwendung impliziter Umwandlungen oder das Belassen lokaler Variablen ohne Initialisierung oder die fehlenden Überprüfung von Übergabeparametern an Funktionen und Methoden. Jede Release der LiftApp (gerade Releasenummer) wurde zuvor von CPPCheck in einem 13 stündigen Durchlauf geprüft.

Alle Funktionen oder Methoden haben einen Kommentarkopf, der dem Syntaxstandard von DoxyGen entspricht. Das ermöglicht es anderen Entwicklern, sich mit diesem Tool einen Überblick über die Klassen oder Strukturen zu verschaffen.

Die automatische Prüfung und Dokumentation ist nur ein Helfer und kein Ersatz für die manuelle Prüfung und Dokumentation.

Für weitere Information, sei hier auf das Kapitel 'Code Analyse Werkzeuge' auf Seite 74 verwiesen.

10 Anforderungen an das Produkt

Die IEC-62443 definiert vier Hauptbereiche.

Bereich	Beschreibung	Beispiele
Essentiell	Wesentliche Funktionen, die für die Verfügbarkeit des Systems von entscheidender Bedeutung sind	Rufeingabe, Position, Richtungserkennung, Türbewegungen
Sicherheit	Funktionen mit zugeordnetem SIL Level	Überbrückung von Sperrmittelschaltern
Alarm	Notfallfunktionen	Notruf, Notlicht
Andere	Weitere Funktionen.	Infotainment, Musik

Jeder Funktionsbereich verfügt über ein Mindestsicherheitsniveau (SL-T), das erreicht werden soll. IEC-62443 definiert grundsätzlich fünf Ebenen.

Anforderung	Alarm	Essentiell	Sicherheit
FR 1 - Identifikation und Authentifizierung	2	2	3
FR 2 - Nutzungskontrolle	1	2	2
FR 3 - System Integrität	1	2	2
FR 4 - Datenvertraulichkeit	1	2	2
FR 5 – Beschränkung des Datenflusses	1	1	1
FR 6 - Rechtzeitige Reaktion auf Ereignisse	1	1	1
FR 7 - Ressourcenverfügbarkeit	1	2	2

Das Aufzugsteuerungsgerät an sich liegt im Funktionsbereich *Essentiell*. Die verbaute Sicherheitsschaltung ist eine elektromechanische Schaltung, die für sich betrachtet und als eigene Baugruppe integriert wurde. Ihre Aufgabe der Gleichlaufüberprüfung der Zonenkanäle erfolgt ohne Software. Das Steuergerät überwacht die Funktion nach jeder Fahrt, dies liegt jedoch im Bereich *Essentiell*, so wie die Schützüberwachung. Für beides ist nur funktionale Sicherheit und kein SIL Level definiert.

Für die betreffende Aufzugssteuerung wäre der SL-T gleich {2-2-2-2-1-1-2}.

10.1 FR1 (SL-T2)

Identifikation und Authentifizierung

Wenn Sie durch die Menüs blättern, finden Sie auf Menüpunkten gelbe oder rote Schlüsselsymbole. Diese zeigen an, dass Sie ein "Service" (gelb) oder "Setup" (rot) Passwort eingeben müssen, um die Einstellung zu ändern.

Bei Benutzerinaktivität muss die Passworteingabe wiederholt werden.

Abbildung 1: Menüpunkt, der eine Passwortberechtigung verlangt

Menüelement, für das die Berechtigung "Service" erforderlich ist.

Menüelement, für das die Berechtigung "Setup" erforderlich ist.

Das Handbuch sagt klar aus, dass der Errichter in Absprache mit dem Eigentümer, die wichtigen Einstellungen in der Steuerung durch die Festlegung eines „Setup"-Passworts schützen muss. Dies kann über die Benutzeroberfläche unter „System Menü → Sicherheit" erfolgen. Das Passwort sollte mindestens 6 Zeichen lang sein.

Für weitere Details folgen Sie bitte dem Kapitel 'Passwortsicherheit' auf Seite 48.

Passwörter werden grundsätzlich nicht im Speicher der Aufzugssteuerung gesichert. Stattdessen wird ein "*gesalzener*" (salted) SHA-1 (Hash) des Passworts gespeichert. Das bedeutet, dass die Aufzugssteuerung die Passworteingabe zwar sicher auf Echtheit prüfen kann, es aber nicht möglich ist, vom Hash auf die lesbare (sichtbare) Passwortzeichenfolge zurückzurechnen.

10.2 FR2 (SL-T2)

Nutzungskontrolle

Die Verwendung von Passwörtern wird in der „Historie“ (Ereignislogger) protokolliert. Alle Parameteränderungen werden im „Parameteränderungsprotokoll“ (Parameter Logger) aufgezeichnet. Das Parameteränderungsprotokoll ist ein Protokollsystem, das alle Änderungen speichert, die im Laufe der Zeit an den Parametern des Aufzugs vorgenommen wurden. Es speichert die letzten 256 Parameteränderungen lokal auf dem nicht flüchtigen Speicher der Steuerung.

Das Protokoll kann unter 'System Menü' → 'Sicherheit' → 'Parameter Änderungsprotokoll' eingesehen aber nicht gelöscht werden.

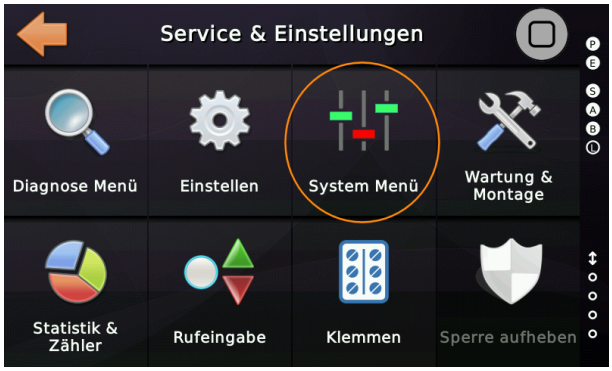


Abbildung 2: Parameter Änderungsprotokoll unter System Menü → Sicherheit

In diesem Protokoll werden folgende Informationen gespeichert:

- Welcher Parameter wurde geändert (Name/Hilfetext).
- Zu welchem Zeitpunkt wurde der Parameter geändert.
- Wie wurde der Parameter geändert.
- Lokal über die Benutzeroberfläche.
- Über das angebundene Bussystem.
- Per Fernzugriff (wenn erlaubt und möglich) über die Cloud Lösung.
- Welche Passwortberechtigung (Setup/Service/keine) war notwendig, um den Parameter zu ändern.
- Der alte und der neue Wert des Parameters, um die Änderung in einen Kontext zu bringen.

Für weitere Details folgen Sie bitte in diesem Dokument dem Kapitel 'Aufzug Parameter-Änderungsprotokoll' auf Seite 50.

10.3 FR 3 (SL-T2)

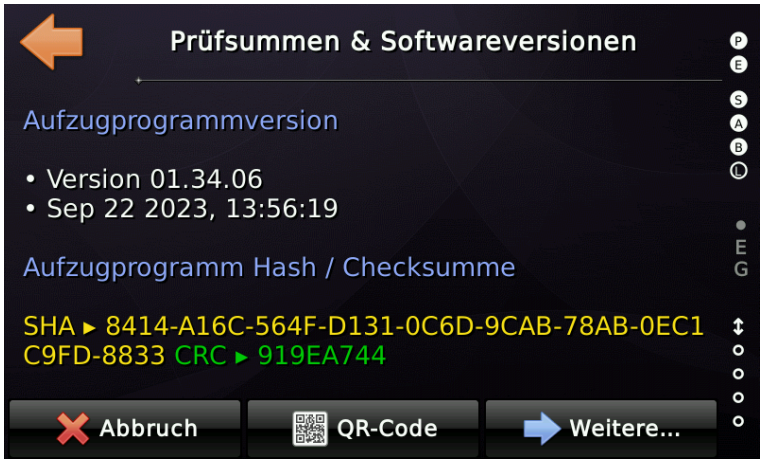
System Integrität

Anwendung

Bei jedem Anwendungsstart wird die Prüfsumme (CRC32) der Binärdatei überprüft, um sicherzustellen, dass sie sich nicht unbeaufsichtigt geändert hat. Das verwendete Dateisystem verwendet seinerseits ebenfalls Prüfsummen, um defekte Sektoren zu erkennen und keine Daten weiterzureichen, die nicht valide sind.

Bei der Installation eines Updates der Anwendung wird unabhängig von der Binärdatei selbst ein **SHA1** der Update-Binärdatei vom Hersteller bereitgestellt, damit der Techniker überprüfen kann, ob die Datei auf dem elektronischen Transportwege manipuliert wurde. Um diesen Vorgang zu vereinfachen, wird der **SHA1** des Updates von der laufenden Anwendung berechnet und dem Techniker angezeigt, der nun dafür verantwortlich ist, ihn mit dem zu vergleichen, den er per E-Mail erhalten hat. Darüber hinaus überprüft die Zugelassene Überwachungsstelle später in den wiederkehrenden Prüfungen die Prüfsumme, um sicherzustellen, dass sie über die Zeit erhalten bleibt.

Hinweis: Die Prüfsumme der Update-Binärdatei, die der Techniker per Mail erhält und die ihm auf dem Bildschirm angezeigt wird, kann sich von der Prüfsumme, die später auf dem Bildschirm für die Zugelassene Überwachungsstelle angezeigt wird, unterscheiden, da die Binärdatei bei der Installation auf dem Gerät einmalig „gestempelt“ wird, um es unmöglich zu machen, dass die installierte Binärdatei später auf einer anderen Gerät ausführbar wäre.



Für weitere Details folgen Sie bitte dem Kapitel 'Checksumme und Software Version' auf Seite 63.

Sicherheitskreis

Die Systemintegrität betrifft auch die Überwachung des Zustandes des Sicherheitskreises (ISO 8102-20 A.3.9.3). Der Sicherheitskreis wird mit einer Hardwarelösung mithilfe einer zertifizierten Abfrageschaltung erfasst, die die EN81-20/50-Normen erfüllt. Die Schaltung wurde vom Lift Instituut als benannter Stelle validiert und zertifiziert. Gegenstand der Prüfung war die Überprüfung, ob die Konformität mit der Aufzugrichtlinie 2014/33/EU auf Basis der harmonisierten Produktnormen EN 81-20 und EN 81-50 gegeben ist.

Wichtig bei der Prüfung ist, dass die Abfrageschaltung rückwirkungsfrei ist, also selbst den Zustand des Sicherheitskreises nicht beeinflussen kann.

Für weitere Details folgen Sie bitte dem Kapitel 'Sicherheitskreisabfrage' auf Seite 54.

10.4 FR 4 (SL-T2)

Datenvertraulichkeit/Datenverlässlichkeit

Anwendung

Die Aufzugssteuerung speichert Ereignisse im Zusammenhang mit dem Betrieb der Anlage, um die Fehlerverfolgung zu erleichtern und die Gesamtverfügbarkeit der Aufzugsanlage zu erhöhen. Zusammen mit erfassten statistischen Daten über Fahrten, Richtungsänderungen, Schützbetätigungen, Nachstellvorgängen und Türbewegungen, kann der Verschleiß von Komponenten durch das Wartungsunternehmen bewertet werden. Zusätzlich werden Parameteränderungen von der Aufzugssteuerung erfasst. Persönliche Daten, etwa „wer“ einen Parameter geändert hat oder „wer“ den Aufzug benutzt hat, werden keinesfalls erfasst. Dies gilt auch für die optionale Massenspeicherprotokollierung, die zusätzlich zur detaillierten Fehlerverfolgung genutzt werden kann.

Die Cloud-Lösung verfügt über Verschlüsselung (TLS) und zertifikatbasierte Handshakes, um sicherzustellen, dass keine Daten in die falschen Hände gelangen, z.B. durch Man-In-The-Middle Angriffe.

Dennoch ist es eine Anforderung, dass die Benutzer des Produktes und der Cloud-Lösung sicherstellen, dass Passwörter oder Anmeldeinformationen nicht nach außen dringen oder in die Hände Dritter geraten. Das gilt auch für Mitarbeiter, die das Unternehmen verlassen.

Wenn auch nur der geringste Verdacht auf ein Datenleck besteht, liegt es in ihrer Verantwortung, einzugreifen und Passwörter und Zugangsdaten zu ändern.

Wir empfehlen, dass es im Wartungsunternehmen eine benannte Person gibt, die Passwörter und Zugangsdaten zentral verwaltet und für deren Änderung verantwortlich ist. Wichtig ist auch dass immer nur das geringste Zugriffsniveau an die Mitarbeiter weitergegeben wird und nicht jeder *Admin* ist!

Das System speichert keine SETUP/SERVICE Passwörter in seinem nichtflüchtigen Speicher. Stattdessen wird nur der gesalzene SHA gespeichert, um eine Validierung des SETUP/SERVICE Passwortes zu ermöglichen.

Für weitere Details folgen Sie bitte den Kapiteln 'NeXt® Cloud Sicherheit' auf Seite 55 und 'Passwortsicherheit' auf Seite 48.

Sicherheitskreis

Die Vertraulichkeit bzw. Verlässlichkeit von Daten betrifft auch die Abfrage des Sicherheitskreises (ISO 8102-20 A.3.9.3). Der Sicherheitskreis wird in Hardware mithilfe einer zertifizierten Abfrageschaltung erfasst, die die EN81-20/50-Normen erfüllt. Der Zustand des Sicherheitskreises wird über kein Bussystem eingelesen/erfasst. Er wird direkt von der Hardware der Aufzugssteuerung über integrierte Komponenten elektrisch erfasst. Nachrichten, die die Signale des Sicherheitskreises widerspiegeln, werden nicht über das CANopen-System empfangen (gelesen), auch wenn CANopen CiA-417 solche Nachrichten definiert. Der Zustand des Sicherheitskreises wird lediglich auf dem Bus über Nachrichten zu Diagnosezwecken widergespiegelt (gesendet). Weitere Einzelheiten finden Sie im Kapitel 'Sicherheitskreisabfrage' auf Seite 54.

10.5 FR 5 (SL-T1)

Beschränkung des Datenflusses

Die Aufzugssteuerung tauscht Steuer- und Statuswörter mit den Peripheriegeräten über das CANopen-Bussystem, gemäß dem CiA-417-Profil für Aufzüge, aus. Der Prozessdatenfluss mit den Cloud-Lösungen erfolgt gemäß den Nutzungs-vereinbarungen der Cloud-Lösung.

Es ist wichtig, dass der Benutzer der Cloud-Lösung, in der Regel das Wartungsunternehmen, diese Art von Remote-Service mit dem Eigentümer des Aufzuges abstimmt, da der Eigentümer des Aufzuges nicht nur Eigentümer der Anlage, sondern auch der Daten ist, welche diese produziert.

Weitere Informationen zur **Cloud** finden Sie im Kapitel „NeXt® Cloud Sicherheit“ auf Seite 55 sowie im Kapitel „Passwortsicherheit“ auf Seite 48 in diesem Dokument.

Weitere Informationen zum **CANBus** finden Sie im Kapitel „CANbus Allgemein“ auf Seite 20 in diesem Dokument.

10.6 FR 6 (SL-T1)

Rechtzeitige Reaktion auf Ereignisse

Organisatorische Seite

Im Falle eines Vorfalls ist es wichtig, dass wir über eine strukturierte Vorgehensweise verfügen, um rechtzeitig darauf zu reagieren und gegebenenfalls unsere Kunden zeitnah zu informieren. Folgen Sie hierzu dem Kapitel Meldung von Vorfällen/Problemen auf Seite 32. Das Feedback aus dem Feld ist für die qualitative Weiterentwicklung des Produktes wichtig.

Technische Seite

Auf technischer Seite besitzt einen **Hardware Watchdog**, der sicherstellt, dass die benötigten Programmteile alle und immer abgearbeitet werden. Ist dies nicht der Fall werden alle Ausgänge abgeschaltet und das System sicher neu gestartet. Ein Systemstart wird im Ereignisspeicher des Systems aufgezeichnet.

Der Hardware Watchdog kann über das Menü Prüfungen getestet werden.

Zusätzlich gibt es zeitbasierende Warnungen/Störungen, die generiert werden, wenn:

- Das Zonensignal verspätet beim Verlassen der Etage zurückgesetzt wird.
- Das Einfahren mit Früh-Öffnenden Türen ungewöhnlich lange dauert.
- Das Öffnen oder Schließen der Türen ungewöhnlich lange dauert / fehlschlägt.
- Das Zeitverhalten des Antriebes ungewöhnlich ist, siehe Startkontrolle, Laufzeitkontrolle, Verzögerungskontrolle, Nachstellkontrolle, Aufsetzkontrolle.
- Die Fahrkorbbewegung während der Fahrt für eine Zeitspanne ausfällt, ohne das es einen ersichtlichen Grund gibt, z.B. durch ein Versagen des Drehgeberzahnriemens.
- Statusworte des Positionsgebers, Antriebes und der Fahrkorbbaugruppe werden bei der Übertragung über den Bus auf Ihr Zeitverhalten hin überwacht.

Prinzipiell verwendet CANopen CiA-417 Heartbeats (Herzschläge) mit denen permanent überprüft wird, dass Baugruppen, wie Antrieb, Positionsgeber und Lastmessung sich nicht vom Bus '*abmelden*'. Dabei überwachen die Baugruppen den Heartbeat der Steuerung und die Steuerung den Heartbeat der Baugruppen. Wenn sich IO-Baugruppen abmelden werden deren Eingänge/Ausgänge ausgeschaltet.

10.7 FR 7 (SL-T2)

Ressourcenverfügbarkeit

Energieversorgung

Das System überwacht direkt die Netzspannung (230 V AC oder 120 V AC), die das Netzteil zur Erzeugung der 24 V Gleichspannungsebene verwendet. Das bedeutet, dass die Steuerung weiß, dass die 24 V abfallen werden, bevor dies tatsächlich der Fall ist. Zusätzlich wird die Versorgungsspannung der internen Schaltkreise von einem speziellen Controller (PMIC) überwacht, der interne Spannungen aus den 24 V DC erzeugt. Die Spannung des Fahrkorblichtes wird auch direkt über einen 230 V AC / 120V AC Eingang überwacht. Zusätzlich verfügt das Notlicht-Batterieladegerät in der Regel über einen Fehlerausgang, der mit der Aufzugssteuerung verbunden ist.

Sicherheitskreis und Sicherheitsschaltung

Die Übertragung des Sicherheitskreises erfolgt nicht über das Bussystem. Der Anschluss erfolgt immer klassisch in Hardware, direkt an die zertifizierte Sicherheitsplatine. Wir unterstützen KEINE Encodersysteme, die den Sicherheitskreis über das Bussystem übertragen. Die Eingänge des Sicherheitskreises können nicht umprogrammiert werden. Weitere Einzelheiten finden Sie im Kapitel „Sicherheitskreisabfrage“ auf Seite 54.

Die verbaute Sicherheitsschaltung ist eine elektromechanische Schaltung, die für sich betrachtet und als eigene Baugruppe integriert wurde. Ihre Aufgabe der Gleichlauf-überprüfung der Zonenkanäle erfolgt ohne Software. Das Steuergerät überwacht die Funktion nach jeder Fahrt zusätzlich funktional über einen Rückmeldekontakt.

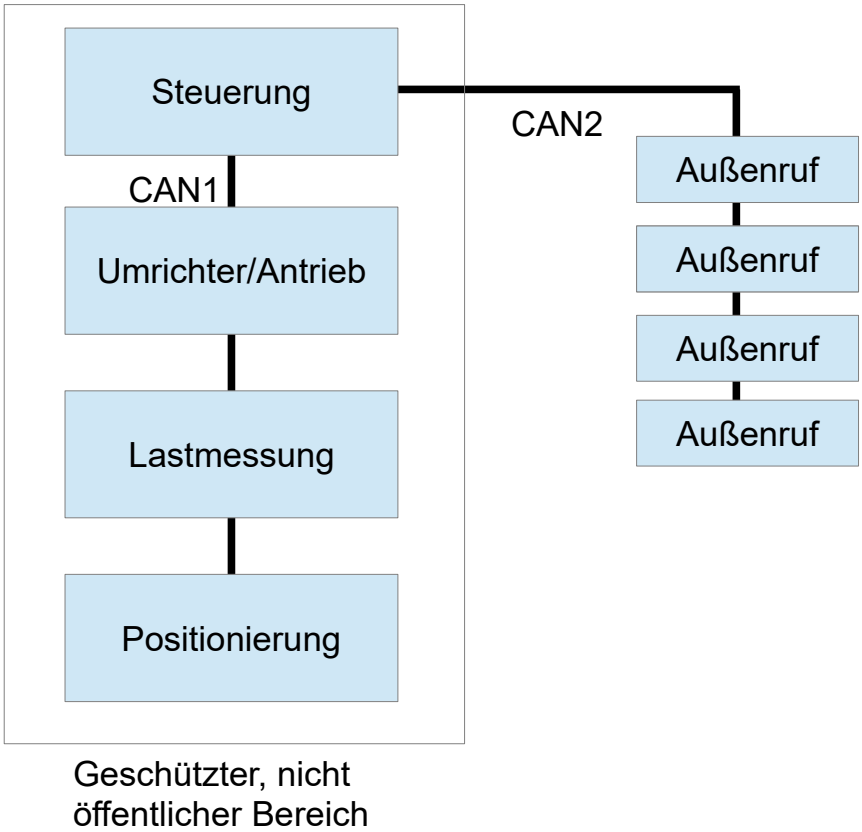
CANbus Allgemein

Die CANopen-Bus-Statuswörter von Peripheriegeräten, wie dem Positionsgeber oder der Antriebseinheit sind so parametriert, dass sie zyklisch gesendet werden, so dass Ausfälle oder verzögerte Übertragungen ordnungsgemäß erkannt werden können. Zusätzlich wurden Kontrollzeiten für Steuerwörter (Befehle) sinnvoll implementiert und getestet. Wenn ein Busfehler oder eine Bus-Off Situation eintritt, wird die Aufzugssteuerung vollständig zum Stillstand kommen und in einen sicheren Zustand wechseln. Die Türen bleiben außerhalb der Türzone geschlossen. Zusätzlich senden die Peripheriegeräte sogenannte CANopen Heartbeats, die von der Steuerung überwacht werden. Im Gegenzug überwachen auch die Peripheriegeräte, wie z. B. der Antrieb den Heartbeat (Herzschlag) der Steuerung und wechseln bei Ausfall nach '*pre-operational*', was bei einem Antrieb mit einem vollständigen Stopp verbunden ist.

Der CANbus verfügt über Identifikatoren (COB-IDs) für die Nachrichten, die bei

gleichzeitigem Schreibzugriff sicherstellen, dass die Nachrichten nach Ihrer Priorität (Wichtigkeit) gesendet werden. Die Hersteller von CANopen-Komponenten müssen zudem regelmäßig an Plug-Festen im CiA-Hauptsitz in Nürnberg teilnehmen, um die Interoperabilität der Komponenten und die Einhaltung des Busstandards sicherzustellen. Andernfalls ist es ihnen nicht gestattet, das CANopen-Logo auf ihren Produkten zu verwenden. **Wir empfehlen, nur Komponenten einzusetzen, die das CANopen Lift Logo mit *Stolz und zu Recht* tragen.**

Die Aufzugssteuerung selbst wird gemäß EN81-20/50 in einem speziellen und mit einem Schlüssel verschlossenen Raum installiert, der einen unbeaufsichtigten Zugriff auf die Hardware und Ausrüstung verhindern soll und üblicherweise als „*Maschinenraum*“ bezeichnet wird. Die CANbus-Schnittstellen und die entsprechende Verkabelung befinden sich in diesen geschlossenen und sicheren Räumen. Allerdings könnte der zweite CAN, also CAN2, der für die Außenrufe auf den Etagen verwendet wird, stärker exponiert sein. Daher sind alle Buskomponenten, die sich auf Positionierung, Antrieb, Lastmessung und Fahrkorb-I/O beziehen, nur an CAN1 angeschlossen. Die Schnittstellen CAN1 und CAN2 sind galvanisch und logisch getrennt. Die CAN1-Schnittstelle verläuft vom Steuerschrank direkt durch das Hängekabel und durch den Schacht in den Fahrkorb, wo die Paneele mit vandalensicheren Elementen verschlossen gehalten werden müssen.



Am CAN2 werden keine Statuswörter von Umrichter/Bremse, Positionsgeber, PSU oder Türen akzeptiert. Auch Steuerworte an diese Baugruppen können nicht gesendet

werden. Ein transparentes Routing von beliebigen Nachrichten zwischen CAN1 und CAN2 findet nicht statt.

Trotzdem kann man die Verfügbarkeit einer Anlage über geöffnete Außentableaus negativ beeinflussen, indem man z.B. die Quittungslampe gegen Masse kurzschließt.

Webserver und Cloud-Schnittstelle

Während der eingebaute Webserver nur für den temporären Einsatz zur Reparatur im lokalen Netzwerk gedacht ist, ist die Cloud-Lösung für den Remote-Einsatz über eine Internetverbindung gedacht. Es besteht die Möglichkeit, dass beide Schnittstellen angegriffen werden. Für den eingebauten Webserver könnte ein lokaler Angriff durch irgendein Script-Kiddy ein Szenario sein und für die Cloud-Schnittstelle stellen automatische Crawler eine echte Bedrohung dar.

MQTT-Schnittstelle

MQTT steht für „*Message Queuing Telemetry Transport*". Es ist ein offenes Nachrichten-protokoll. Es wird in der Regel für M2M (Maschine-zu-Maschine-Kommunikation), wie z.B. beim Internet der Dinge, eingesetzt. Die Schnittstelle ist ab Werk ausgeschaltet und kann nur direkt am Gerät (also nicht aus der Ferne) aktiviert werden.
Typischweise wird diese Schnittstelle mit unserer Aufzugsteuerung für AVG's *(Automated Guided Vehicle - Fahrerlose Transportfahrzeuge)* in Fabriken verwendet. Wir empfehlen die Verwendung des verschlüsselten TLS-Web-Socket-MQTT-Modus.
Nur in gesicherten Netzwerken, wie in Fabriken oder Krankenhausumgebungen, wo das Netzwerk für die Gebäudetechnik, separat und von außen nicht zugänglich ist, mag ein geringerer Verbindungsmodus gewählt werden. Prinzipiell ist es nicht möglich, über diese logische Schnittstelle Parameter zu ändern oder auf Elemente des Aufzuges zuzugreifen. Es ist aber durchaus möglich Rufe zu geben und Tür-Auf/Zu Tasten Anforderungen zu senden. Damit kann die Verfügbarkeit der Anlage beeinträchtigt werden.

Fuzzing Tests der Cloud JSON-REST-API und Webserver Schnittstelle

Um sicherzustellen, dass unsere Schnittstellen grundsolide und robust genug sind, um Angriffe abzuwehren, die mithilfe fehlerhafter HTTP-Header, fehlerhafter HTTP-Bodies, Web-Socket Datenfluten oder einfach durch Fuzzing der JSON-REST-API ausgeführt werden, mit dem Ziel das System zu brechen und den Aufzug außer Betrieb zu setzen, haben wir eine Reihe eigener Fuzzing- und Penetrationstests implementiert, wobei wir POSTMAN als Arbeitspferd verwendet haben.

POSTMAN ist ein weit verbreitetes Tool, das zur Durchführung von Fuzzing-Tests an Netzwerkschnittstellen verwendet wird.

https://www.postman.com/product/what-is-postman/

Um sicherzustellen, dass der Fuzz, der zum Testen der Schnittstelleneingabe verwendet wird, tatsächlich alle erdenklichen Daten enthält, aktualisieren wir unsere „*List of Naughty Strings*“ (Liste bösartiger Zeichenketten) aus dem „*minimaxir/big-list-of-naughty-strings*“-Repository und verwenden ein einfaches Shell Skript, um diese in eine CSV-Datei zu konvertieren und diese als Eingabe für POSTMAN zu verwenden.

Es ist besonders wichtig, diese „ungezogenen Zeichenfolgen“ für Eingaben der JSON-REST-API und der Webschnittstelle zu verwenden, um sicherzustellen, dass der Parsing-Code auf der LiftApp-Seite alle unerwarteten Daten verarbeitet und nicht ausfällt oder unbeaufsichtigt oder unerwünschte Abläufe ausführt.

Das POSTMAN-Tool wird auch verwendet, um zu Prüfen, ob es möglich ist, Probleme zu verursachen, indem die exponierten Netzwerkschnittstellen mit Datenrauschen geflutet werden. Außerdem können damit Port-Scans durchgeführt werden.

Das Tool POSTMAN wird auch verwendet, um diese Test zu dokumentieren, da es selber eine Historie über die Testausführung mitführt (Wann wurde was getestet).

Fuzzing Tests der MQTT-Schnittstelle

Zum Fuzz-Testen der MQTT-Schnittstelle haben wir eine eigene Testmethode entwickelt, die auf Basis von zufällig generierten, fehlerhaften MQTT Nachrichten (auch mit fehlerhaften Längenangaben) prüft, ob der MQTT-Parser alle denkbaren oder zufälligen Fehlerszenarien abfangen kann.

Der Test wird vor der Veröffentlichung einer Version ausgeführt und im Release-Dokument dokumentiert.

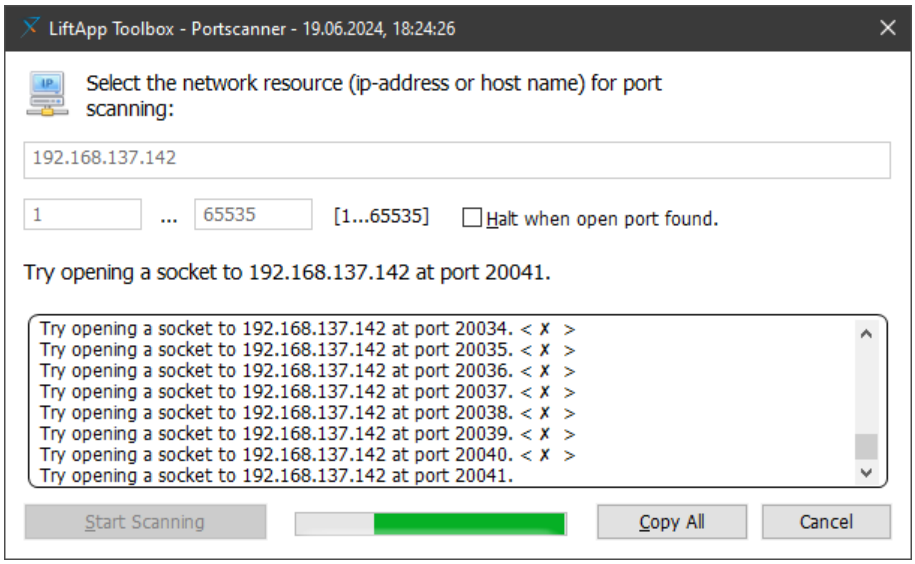
Protokol (*gekürzt*)

```
CAppMQTTClient::Do_Development_Fuzz_Testing() - 15.05.2024-13:21:35  Start Fuzzing the MQTT message parser,
using seed value of 4876620.

MQTT Fuzz 10 43 98 3D 3D 3D 85 2B 2B 2B CD 9D 9D 9D AF 2C 2C 2C 7E 51 51

CAppMQTTClient::Handle_Message_RX() Remaining length indication is larger than bytes received.

MQTT Fuzz 4A 5E 34 7F 7F 7F 1A DE DE DE 4E 35 35

CAppMQTTClient::Handle_Message_RX() Remaining length indication is larger than bytes received.

MQTT Fuzz 0C C9 19 CF CF CF AC 4E 4E 4E E1 C9 C9 C9

CAppMQTTClient::Handle_Message_RX() Remaining length indication is larger than bytes received.

MQTT Fuzz F1 94 31 D0 D0 D0 CC 23 23 23 4F 80 80 80 28 2A 2A 2A 98 26 26 26 D4 A9 A9 A9 36 97 97 97 51 1B 1B
1B 1B 1D 1D 1D F2 36 36 36 E1 56 56 56 CD 43

CAppMQTTClient::Handle_Message_RX() Remaining length indication is larger than bytes received.

MQTT Fuzz CF 14 68 BA BA BA 0F 61 61 61 7F 50 50 50 8E 37 37 37 CE 5D 5D 5D DC A0 A0 A0 A9 E3 E3 E3 D4 14 14
14 0B 74 74 74 E3 09 09 09 66 0A 0A 0A 18 E9

CAppMQTTClient::Handle_Message_RX() Unknown message type.

… <100 more>
```

Portscanning

Zur Überprüfung offener Ports nutzen wir unser eigenes Port-Scanner-Dienstprogramm, dass wir als Teil unserer LiftApp Toolbox auch veröffentlicht haben.

Dieses Tool kann von jedem kostenlos genutzt werden.

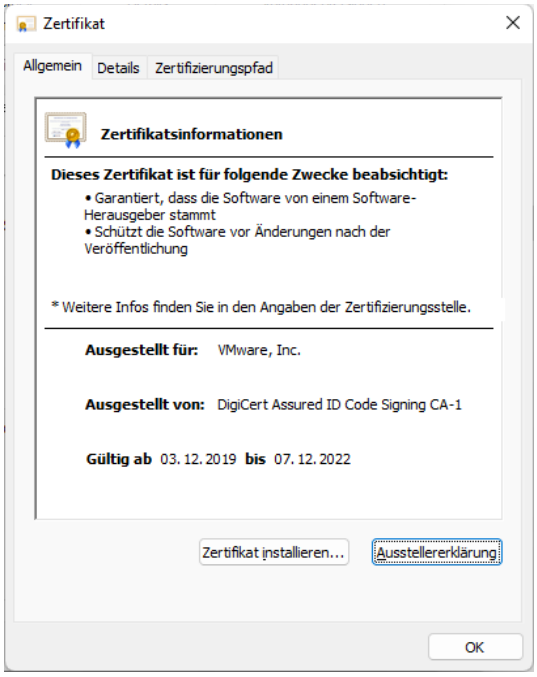


► Ab Werk haben unsere Steuerungen kein Netzwerkport offen!

11 Entwicklungsumgebung und Datenschutz

11.1 Entwicklungsrechner

Um das Risiko eines **Angriffs oder Eindringens** zu minimieren, erfolgt die Entwicklung der Aufzugsanwendung innerhalb einer **virtuellen Maschine**, die ausschließlich zum Schreiben, Kompilieren und Verknüpfen/Erstellen der Anwendung verwendet wird. Diese virtuelle Maschine wird nicht zum Surfen im Internet, für E-Mails oder andere Online-Aktivitäten verwendet, außer zum Abrufen von Updates, zum Einstellen der Uhr und zum Abrufen sicherer Zeitstempel für digitale Signaturen. Als Entwicklungsumgebungen für die Aufzuganwendung kommen Linux®-basierte Systeme zum Einsatz.



Die virtuelle Maschine wird über einen Vmware Player ausgeführt, der wiederum digital signiert ist, um sicherzustellen, dass es sich um die echte unmanipulierte Version vom Hersteller handelt.

Der Quellcode wird in einem Repository gespeichert. Das Passwort und der private Schlüssel, die zum Ein- und Auschecken, sowie zur Verwaltung verwendet werden, sind nur dem Entwickler bekannt. Es existiert eine Sicherungskopie auf einem physischen USB-Stick, der sich im Besitz des CEO des Unternehmens befindet. Die Schlüsseldateien sind aus offensichtlichen Gründen **nicht** Teil des Quellcode-Repositorys.

Die äußere Host-Maschine wird verwendet, um mit der Außenwelt zu kommunizieren. Dieser Host kann ein Windows®-Computer sein. Hier wird der Entwickler das Surfen und die E-Mail- oder Instant Messaging-Kommunikation durchführen. Dieser Host-Rechner wird durch eine aktuelle Antiviren-Software geschützt, die die Datei- und Netzwerkkommunikation in Echtzeit überwacht.

Alle lokalen PCs, Laptops und Workstations bei Thor Engineering sind mit einer aktuellen Virenscanner-Anwendung ausgestattet, die den Datei- und Netzwerk-I/O lokal überprüft. Diese ist auch dafür verantwortlich, eingehende E-Mails auf mögliche Bedrohungen zu scannen, basierend auf eingebetteten Skripten und feindlichen ausführbaren Anhängen.

Alle lokalen PCs, Laptops und Workstations bei Thor Engineering sind mit einer lokalen Firewall ausgestattet, um das Risiko eines Eindringens von innerhalb des Netzwerks zu minimieren.

Für die Linux®- und Windows®-Maschinen werden verfügbare Security-Updates täglich automatisch abgerufen.

Die Entwicklungsmaschinen werden abends und am Wochenende durch physische Trennung vom Stromnetz abgeschaltet. Die Funktion „*Power on over Network*“ ist im BIOS ebenfalls deaktiviert.

11.2 Datenschutz auf Routern, Switches und anderen Netzwerkgeräten

Die Firmware des Netzwerkrouters und der verwaltbaren Switches werden aktualisiert, sobald neue Firmware verfügbar ist. Die externe Verwaltungskonsole (WAN) solcher Geräte ist deaktiviert.

Wir verwenden keine anonymen Dateifreigaben oder unverschlüsseltes FTP über unsere Netzwerke.

Alle drahtlosen Netzwerke sind durch einen 16-stelligen Schlüssel mit WPA2 gesichert.

Gäste verwenden nur das drahtlose „Gast“-Konto, das keinen Zugriff auf unsere lokalen Computersysteme oder Netzwerkdateispeicher gewährt, die möglicherweise vertrauliche Daten enthalten.

11.3 Datenschutz auf beteiligten NAS-Systemen

Die Firmware der NAS-Systeme wird aktualisiert, sobald neue Firmware verfügbar ist. Die externe Verwaltungskonsole der Geräte (WAN) ist deaktiviert. Der Zugriff auf das NAS ist durch einen Benutzernamen und ein Passwort gesichert, das der Passwortrichtlinie des Unternehmens entspricht.

11.4 Datenschutz beim Generieren von Software-Releases

Software, die von Thor Engineering veröffentlicht wird, wird während des Kompilierens/Verknüpfens auf Malware überprüft. Freigegebene Software ist entweder digital signiert (Authenticode-Zertifikat) oder (Embedded Software) mit einem zusätzlichen SHA1-Hash versehen, um anschließend zu überprüfen und sicherzustellen, dass die Software-Binärdatei bis zur Installation intakt und authentisch ist. Unser Partner für Authenticode-Zertifikate ist die Symantec Corporation. Wir entwickeln vorzugsweise in virtuellen Linux®-Umgebungen, die nur zum Bearbeiten von Quellcode, Kompilieren und Linken verwendet werden. In diesen virtuellen Maschinen gibt es keine E-Mail-Kommunikation und keine Verwendung von Webbrowsern.

11.5 Datenschutz bezüglich E-Mail und externer Dateispeicherung

Auf alle bei Thor Engineering verwendeten E-Mail-Konten wird nur mit TLS-Verschlüsselung zugegriffen. Unser vertrauenswürdiger E-Mail-Anbieter (One.com) hat seinerseits die Einhaltung der DSGVO sicherzustellen.

https://help.one.com/hc/en-us/articles/360000253649-How-does-One-com-comply-with-the-GDPR-

Unser Anbieter für externe Dateispeicherung ist Dropbox Business, das angegeben hat, die DSGVO auf seiner eigenen Seite zu erfüllen, indem es nach ISO 27018, dem internationalen Standard für Praktiken in Cloud-Privatsphäre und Datenschutz, zertifiziert ist.

https://www.dropbox.com/en_GB/security/gdpr

11.6 Datenschutz in Thor's Lift Cloud Interface

Die Cloud-Schnittstelle von Thor wird verwendet, um Aufzugsteuerungen mit einer Cloud zu verbinden, die hauptsächlich für vorausschauende Wartung verwendet wird. Personenbezogene Daten stehen bei dieser Lösung nicht im Vordergrund. Um sicherzustellen, dass es keinen unbeaufsichtigten Zugriff auf den Aufzug über die Cloud-Lösung gibt, der dazu führen könnte, dass der Aufzug angegriffen oder Daten gestohlen werden, bietet die Cloud-Lösung von Thor standardmäßig **TLS-Verschlüsselung** und **zertifikatbasierte Serverauthentifizierung** ohne Kompromisse. Der Cloud-Anbieter, der wiederrum den Endpunkt Zugang ermöglicht, muss seinerseits durch Einhaltung der Normen, wie der ISO 27017 „*Informationssicherheit im Cloud Computing* “ und ISO 27018 *„Datenschutz für Cloud Services“* sicherstellen, dass Zugangsdaten nicht in falsche Hände fallen.

11.7 Verwendung der Cloud-API-Lösungen von Google und DeepL

Derzeit verwendet Thor Engineering die „Übersetzungs-API“ der genannten Cloud-Dienste. Diese Dienste werden derzeit nur von Thors **Entwicklungsrechnern** (PC's) verwendet und **nicht von der Aufzugsteuerung selber**.

11.8 Über Schwachstellen auf dem Laufenden bleiben

Wir verfolgen hier einen hybriden Ansatz. Wir lesen regelmäßig die Branchennachrichten und Anbieterquellen für unsere Anwendungen, die wir auf unseren Systemen verwenden. Darüber hinaus besuchen wir regelmäßig die BitDefender Lab News und die DigiCert News, da wir deren Produkte (Virenscanner und Code Signing) hausintern auf unseren Computern verwenden.

https://www.bitdefender.com/blog/labs

https://www.digicert.com/news

Darüber hinaus kommunizieren wir regelmäßig mit den Webentwicklern von MASORA AG (Schweiz) und Code Ink (Niederlande), die ihre Versionen der Next® Cloud API implementieren. Auf diese Weise stellen wir sicher, dass wir Neuigkeiten, Wissen und technikbezogene Neuigkeiten teilen.

Nationale Schwachstellendatenbank (NVD)

Wir fragen die NVD ab, die National Vulnerability Datenbank (NVD). Dabei handelt es sich um das Repository der US-Regierung für auf Standards basierende Daten zum Schwachstellenmanagement. Sie verwenden eine CVSS-Metrik, um die Sicherheitsprobleme zu messen und es für uns transparenter zu machen, wie wir reagieren müssen.

https://nvd.nist.gov/

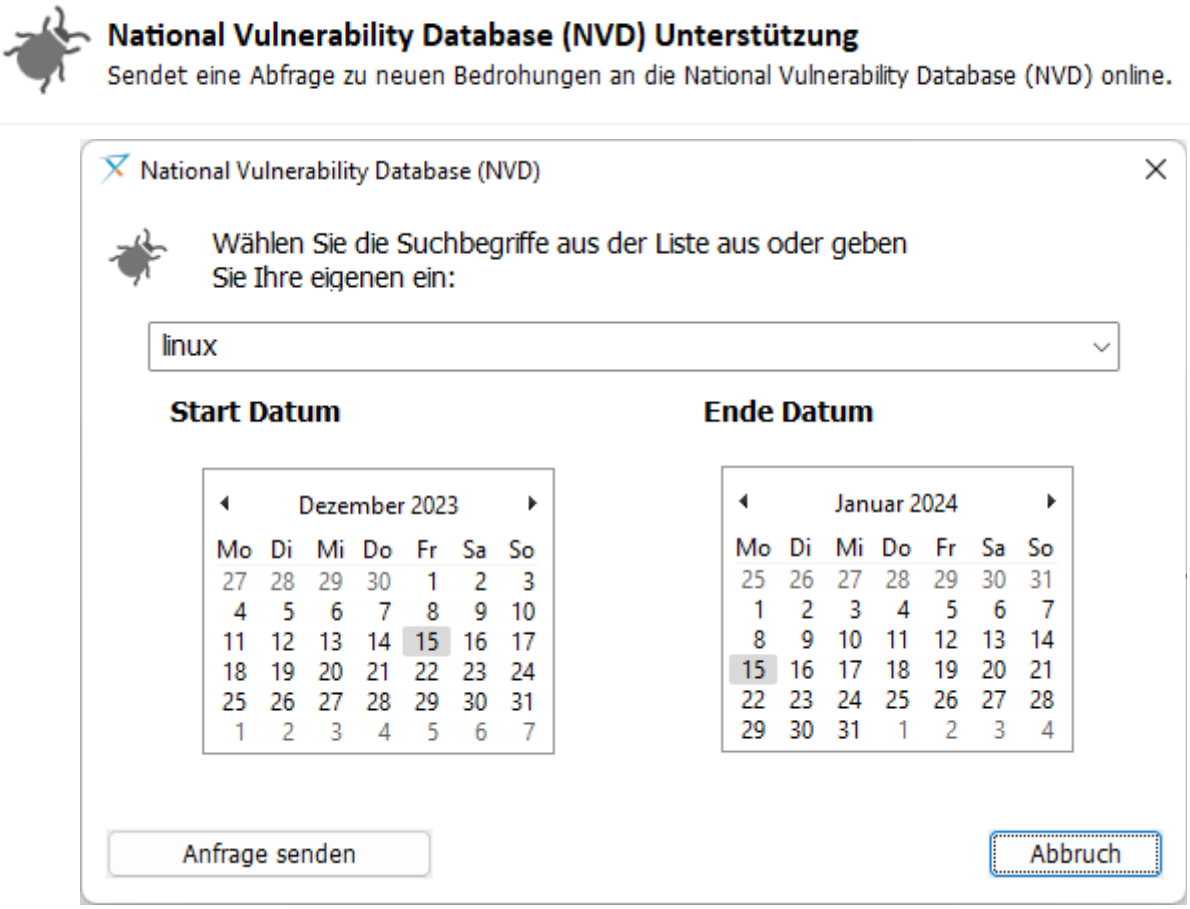
Um dies effizienter zu gestalten, haben wir beim NVD einen API-Schlüssel angefordert, um deren REST-API zu verwenden und Linux®- und Windows®-bezogene Sicherheitsnachrichten über REST-API-Anfragen zu erhalten, die direkt über den Browser gesendet werden.

Beispiel zum Abrufen Linux-basierter NVD-Datensätze für die erste Dezemberwoche 2023 über den Browser:

https://services.nvd.nist.gov/rest/json/cves/2.0/?noRejected&pubStartDate=2023-12-01T00:00:00.000&pubEndDate=2023-12-08T00:00:00.000&keywordSearch=Linux

Das JSON basierte Ergebnis muss dann auf die Übertragbarkeit auf unsere Systeme überprüft werden.

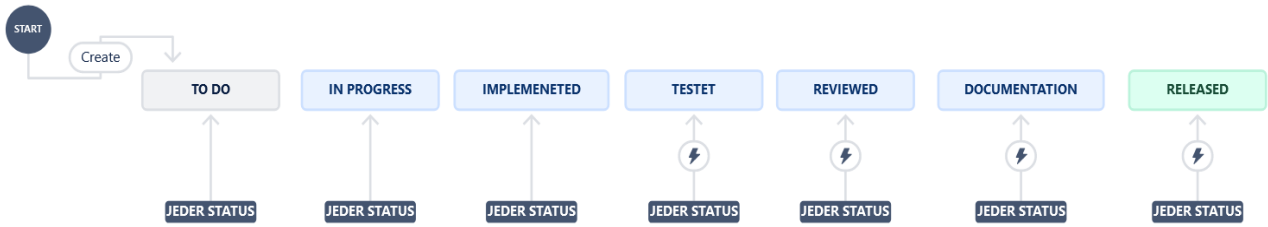
Um den Vorgang zu vereinfachen und ihn zu einer Click-Once-Funktion zu machen, haben wir ein Dienstprogramm erstellt, das in unsere LiftApp Toolbox integriert ist:



Für die Linux®- und Windows®-Maschinen werden verfügbare Updates täglich automatisch über den dedizierten Update-Daemon oder -Dienst abgerufen.

11.9 Workflow

Um dem Team klar zu machen, was die aktuellen Aufgaben sind und welchen Status jede Aufgabe hat, und um sicherzustellen, dass eine Aufgabe nur dann in den endgültigen „Release“-Status verschoben werden kann, wenn sie zuvor die erforderlichen Status durchlaufen hat, wie TESTED und REVIEWED, verwendet das THOR-Team ein KAN-Board. Der Arbeitsablauf kann wie folgt vereinfacht werden:



Der Status RELEASED kann nur erreicht werden, wenn das Team die Aufgaben TESTED, DOCUMENTATION und REVIEWED bearbeitet und bestanden hat. Das folgende RACI-Diagramm zeigt, wer in diesem Arbeitsablauf für was verantwortlich ist.

	Lars Gustafsson	Roy Schneider	Thomas Reul
Kundenfeedback (START)			R
Details einholen/prüfen (CREATE)		I	R
Prüfen/Validieren (TO DO)		I	R
Testumgebung erstellen (IMPLEMENTATION)		R	I
Problem lösen (IMPLEMENTATION)		R	I
Lösung prüfen (TESTING / REVIEW)		I	R
Dokumentation updaten (REVIEW)	I	R	C
Kunden benachrichtigen (RELEASE)	A		R

R – Verantworlich (Responsible)

A – Rechenschaftspflichtig (Accountable)

C – Beratend (Consulted)

I – Informiert (Informed)

Die zu erledigenden Aufgaben und die Verfolgung, welche aktuelle Aufgabe oder welches Problem sich in welchem Entwicklungs- oder Dokumentationsstand befindet, bildet das THOR-Team mit einem KAN-Board ab.

Dadurch wird sichergestellt, dass Softwareänderungen nicht veröffentlicht werden, bevor sie finalisiert wurden.

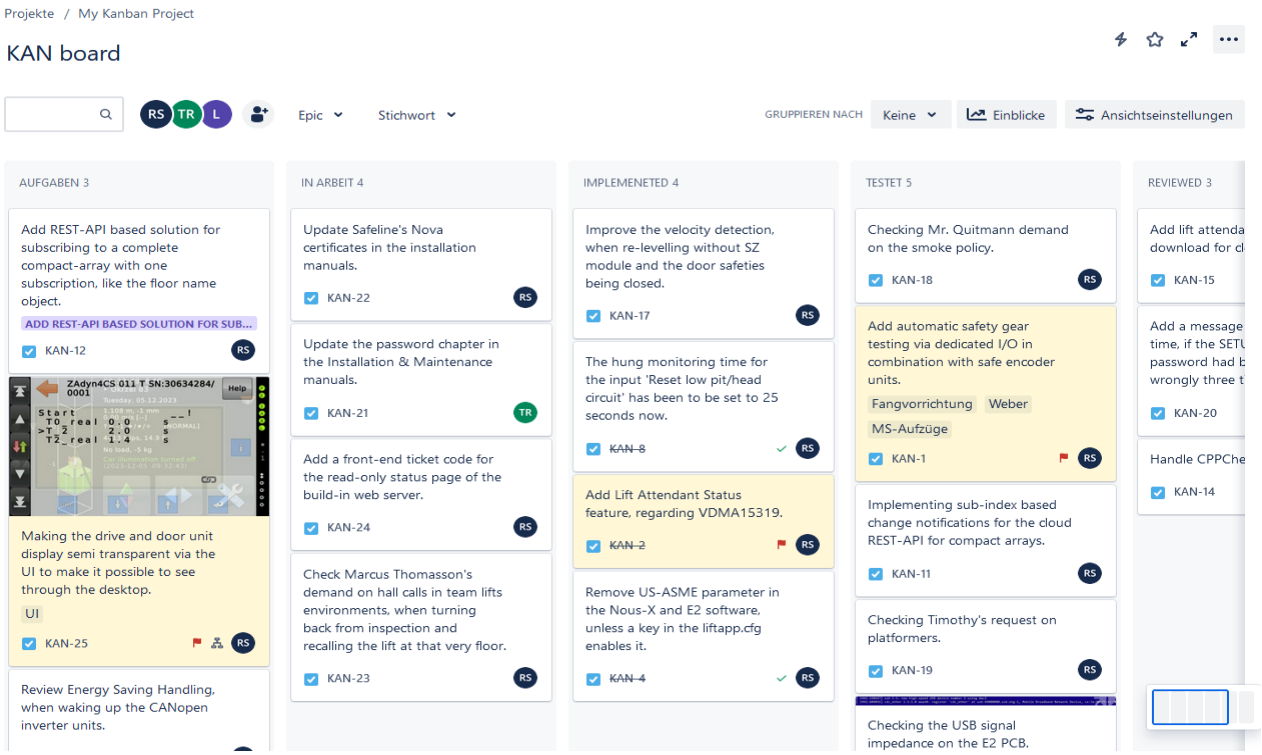


Abbildung 3: Jira Workflow - KAN Board (Kanban)

Kanban ist eine Methode bei der der bestehende Prozess in kleinen Schritten verbessert wird, statt in großen Sprüngen. Indem kleine Änderungen durchgeführt werden, wird das Fehlerrisiko für jede Release verringert. Die Kombination des KAN-Boards, in dem jede Aufgabe mit ihrem aktuellen Status und zugehörigen Daten und Dokumenten gespeichert ist, mit einem Gantt-Diagramm, das die Zeitleiste abbildet, mit der das Team arbeitet, bietet einen guten Überblick und Planungsmöglichkeiten.

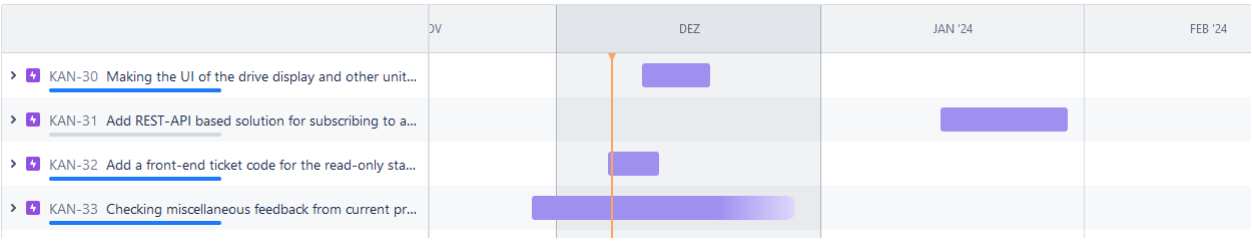


Abbildung 4: Jira Workflow - Gantt Diagramm

Dadurch wird sichergestellt, dass die Aufgaben nicht kollidieren und Kunden frühzeitig informiert werden können, wenn es zu Verzögerungen kommt, beispielsweise aufgrund einer Erkrankung von Entwicklern.

12 Meldung von Vorfällen/Problemen

Im Falle eines Vorfalls ist es wichtig, dass wir über eine strukturierte Vorgehensweise verfügen, um zu reagieren. Bug-/Fehler-/Sicherheits-/Problem-/Vorfallberichte können per E-Mail direkt an uns gesendet werden:

hq@thor.engineering

Eingereichte Berichte müssen klar, vollständig und prägnant sein. Berichte müssen einen Namen, Aufzug-/Steuerungsnummer, eine E-Mail-Adresse und genügend Informationen enthalten, damit der Fehler anhand des Berichts schnell überprüft werden kann. Beschreiben Sie also bitte das Problem und die Schritte, die es verursachen so genau wie es Ihnen möglich ist. Ein Vorfallbericht ist eine Datei, die alle Ereignisse dokumentiert, die möglicherweise Funktions- oder Sicherheitsprobleme eines Produkts verursacht haben. Er wird verwendet, um festzuhalten, was passiert ist, und normalerweise auch die Schritte, die erforderlich sind, um das Problem, Sicherheitsverletzungen und Fehlverhalten am Installationsort zu reproduzieren. Dies hilft dem Management auch dabei, neue Richtlinien zu implementieren, um die Wirksamkeit dieser Änderungen hinsichtlich Sicherheit und Qualität zu bestimmen. Unsere Vorfallberichts-Vorlagendatei auf dem Unternehmensserver sieht folgendermaßen aus (04.09.2024):

```
THOR Engineering Vorfallbericht
===============================

Ein Vorfallbericht ist eine Datei, die alle Ereignisse dokumentiert, die möglicherweise Funktions-
oder Sicherheitsprobleme eines Produkts verursacht haben.

Thema: /* Geben Sie dem Vorfall eine Überschrift! */
Datum: dd.mm.yyyy
Aktualisiert: --.--.----

Einleitung - Der erste Teil des Vorfallberichts behandelt das Wer/Was/Wann und Wo des Vorfalls.
-

Text - Beschreiben Sie den Vorfall von Anfang bis Ende und achten Sie darauf, dass die Details in
chronologischer Reihenfolge aufgeführt werden, um Verwirrung zu vermeiden. Fügen Sie, wenn möglich,
die Details des Technikers hinzu, die zur Reproduktion des Problems erforderlich sind. Dadurch wird
der Behebungsprozess beschleunigt.
-

Einstellungen oder Umgebung - Legen Sie dem Bericht bitte den Parameterausdruck der Steuerung bei.
-

Betroffene Installation
-

Durch den Vorfall verursachte Probleme
-

Zeugen/Berichterstatter
-

Schnelle Lösung / Temporäre / Kurzzeitlösung am Installationsort
-

---
Unterzeichnet von: /* Geben Sie den vollständigen Namen und die Unterschrift des Autors des
Vorfallberichts an, damit dieser die Verantwortung übernimmt und die Aufzeichnungen führt. */
```

Die Verantwortlichkeiten für die Behandlung von Vorfallberichten innerhalb der Organisation beschreibt dieses RACI Diagramm.

R – Verantworlich
(Responsible)

A – Rechenschaftspflichtig
(Accountable)

C – Beratend
(Consulted)

I – Informiert
(Informed)

	Lars Gustafsson	Roy Schneider	Thomas Reul
Kundenfeedback (START)	I*		R
Details einholen/prüfen (CREATE)		I	R
Prüfen/Validieren (TO DO)		I	R
Testumgebung erstellen (IMPLEMENTATION)		R	I
Problem lösen (IMPLEMENTATION)		R	I
Lösung prüfen (TESTING / REVIEW)		I	R
Dokumentation updaten (REVIEW)	I	R	C
Kunden benachrichtigen (RELEASE)	A		R

*) Wenn der Vorfall sicherheitsrelevant ist.

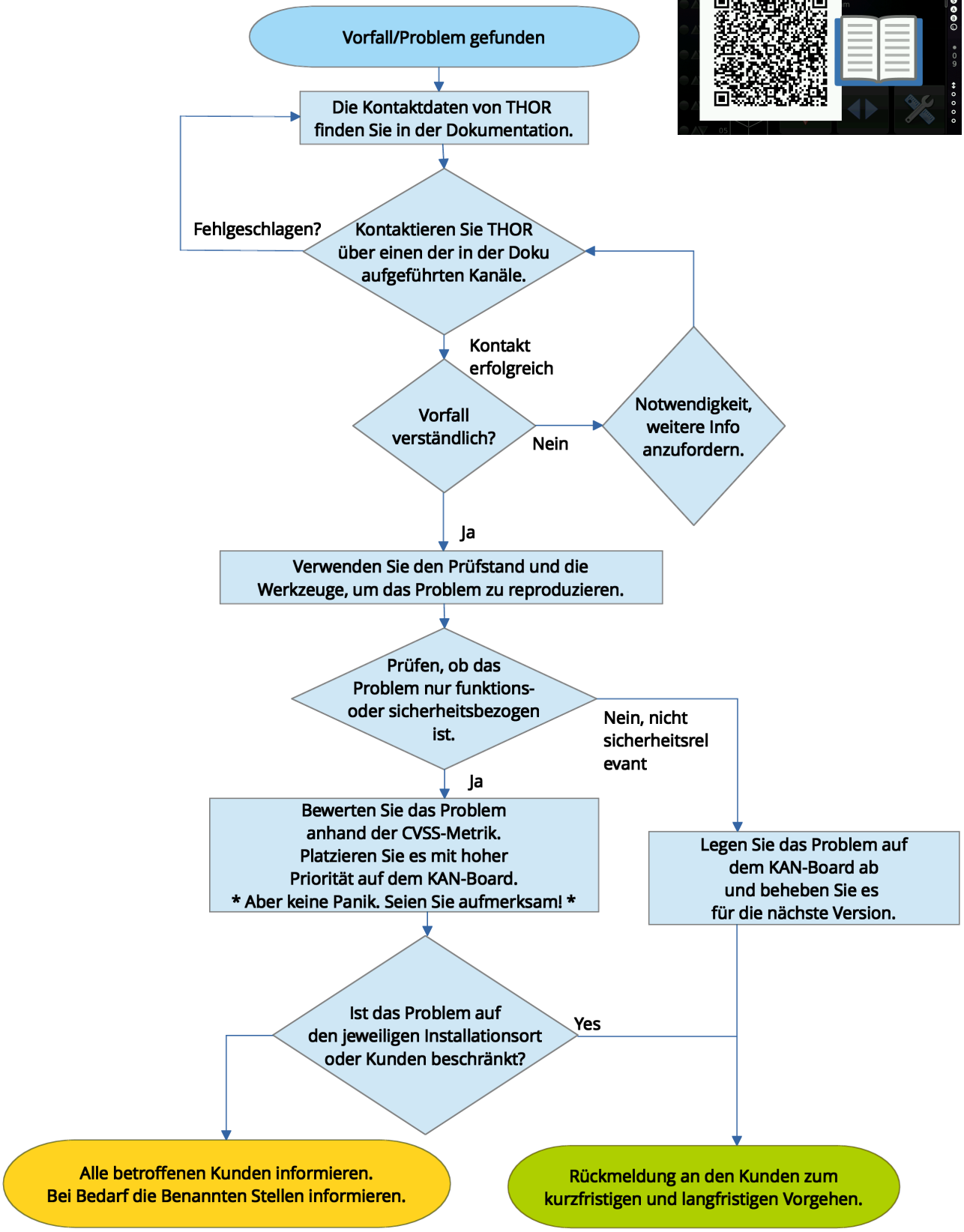


Abbildung 5: Flußdiagramm Meldung von Vorfällen

13 Aktualisierung und Pflege der Handbücher

Es gibt drei wichtige Handbücher für die Software der Aufzugsteuerung:

- Das Software-Referenzhandbuch mit seinen rund 600 Seiten, das alle Funktionen der Aufzugsteuerung widerspiegelt. **Dieses Handbuch wird mit jeder veröffentlichten Softwareversion in englischer und deutscher Sprache aktualisiert.** Die anderen Übersetzungen, wie Französisch oder Niederländisch, werden extern übersetzt und daher bei Bedarf aktualisiert, jedoch mindestens zweimal im Jahr.
- Das Wartungs- und Montagehandbuch wird aktualisiert, wenn die neue Funktion oder das neue Merkmal die Installation oder Wartung des Aufzuges direkt beeinflussen, jedoch mindestens zweimal jährlich.
- Die kleine Hardware-Broschüre wird nur aktualisiert, wenn die Hardware aktualisiert wird oder wenn Fehler oder Probleme gefunden wurden.

Fazit:

Das wichtigste Dokument ist das Software-Referenzhandbuch, da dies die Grundlage ist, von der alle anderen Handbücher und Dokumentationen abgeleitet sind. Es muss mit jedem neuen Software-Release aktualisiert werden und spiegelt immer wieder, für welches Software-Release das Handbuch gültig ist:

Dokumentenhistorie:

Name	Version	Änderung/Kommentar	Datum	LiftApp
rsc	2.1.9	Hinweis zu polumschaltbaren Antrieben ergänzt.	28.06.22	1.30.02
rsc	2.1.10	Neue MODbus Register hinzugefügt.	01.07.22	1.30.02
rsc	2.1.11	Automobiltransportkapitel aktualisiert.	08.09.22	1.30.08
rsc	2.1.12	Signaltabelle für den AZRS Block aktualisiert.	27.09.22	1.30.10
rsc	2.1.13	Anhänge aktualisiert.	13.10.22	1.30.12
rsc	2.1.14	Funktion '*Sichere Türöffnung*' hinzugefügt.	20.10.22	1.30.14
rsc	2.1.15	Parameter '*Notstrom Maximale Fahrten*'.	17.11.22	1.31.02
rsc	2.1.16	CANopen Monitor hinzugefügt.	13.12.22	1.31.04
rsc	2.1.17	Kapitel zum Türentriegelungsmotor hinzugefügt.	10.01.23	1.31.08
rsc	2.1.18	Kapitel '*Absinkverhinderung*' aktualisiert.	19.01.23	1.31.10

Neue Software bedeutet aktualisiertes Handbuch!

14 Versionierung

Generell haben alle für den Kunden freigegebenen Versionen einen manuell erstellten Eintrag im Dokument „*LiftApp_Version_History.pdf*“. Der Kunde kann anhand des Dokuments jederzeit überprüfen, welche Funktionen, Optionen und Features hinzugefügt, geändert oder behoben wurden.

Einen Beispieleintrag finden Sie hier. Der Eintrag wird in einer Form und Formulierung erstellt, die für den Techniker des Endkunden verständlich ist. Der Eintrag enthält die Versionsnummer, das Datum und welche Funktionen hinzugefügt, verbessert und/oder behoben wurden. Der Secure Hash (SHA) der Veröffentlichung ist nicht Teil dieses Dokuments. **Der Hash (SHA) wird mit der Freigabebenachrichtigung direkt an den Kunden versendet.** Im Dokument werden die gleichen Symbole wie in der Benutzeroberfläche verwendet, wenn auf die betreffenden Funktionen verwiesen wird, um es dem Kunden visuell einfacher zu machen, den Kontext der Eingabe zu erfassen.

Der Eintrag muss mögliche Sicherheitsprobleme oder Risiken hervorheben.

14.1 Beispiel

V1.20.14 (03-2020)

Neue Features/Funktionen

- Grundlegende Unterstützung für die kommenden intelligenten Netzteile hinzugefügt. Dies wird sicherlich in Zukunft verbessert werden, da wir statistische Daten und dergleichen haben möchten. Derzeit werden die Einheiten erkannt, in Betrieb genommen und eine einfache Statusseite ist verfügbar. Weitere werden folgen...

Verbesserungen

- Bei Verwendung der Codeeingabe des Innenrufs über die Tasten im Panel in der Kabine blinkt jetzt der Behindertenruf, der eine Codeeingabe erfordert, solange die Codeeingabe aktiv ist und auf die Eingabe der Nummern wartet.

14.2 Nummerierung

Wurden neue Funktionen hinzugefügt, wird die Minor-Versionsnummer erhöht, so dass z.B. aus einer V1.21.16 eine V1.22.02 wird. Bei jeder Art von Verbesserung oder Fehlerbehebung wird die Releasenummer (die letzten beiden Ziffern) erhöht. Diese Ziffern werden immer erhöht, sobald eine Version veröffentlicht wurde. Ungerade Veröffentlichungsnummern weisen immer auf eine Version „In Making“ hin. Eine gerade Versionsnummer zeigt eine Version an, die für den Kunden freigegeben wurde. Nachdem die Version hier und von unseren Testpartnern getestet wurde, wird die Erweiterung '_stable' hinzugefügt. Eine solche Version ist dann fertig zum Versand an die Kunden. Der Dateiname sieht dann so aus:

liftapp_01_26_04_stable

14.3 Versionskontrollsystem

Nachdem eine Version erstellt und zum Testen freigegeben wurde, soll über das Versionskontrollsystem (Git) ein Tag erstellt werden, um später eine Version wiederherstellen zu können. Da auch das Software-Referenzhandbuch Teil desselben Repositories ist, würde das neu erstellte Handbuch dann mit der markierten Version übereinstimmen.

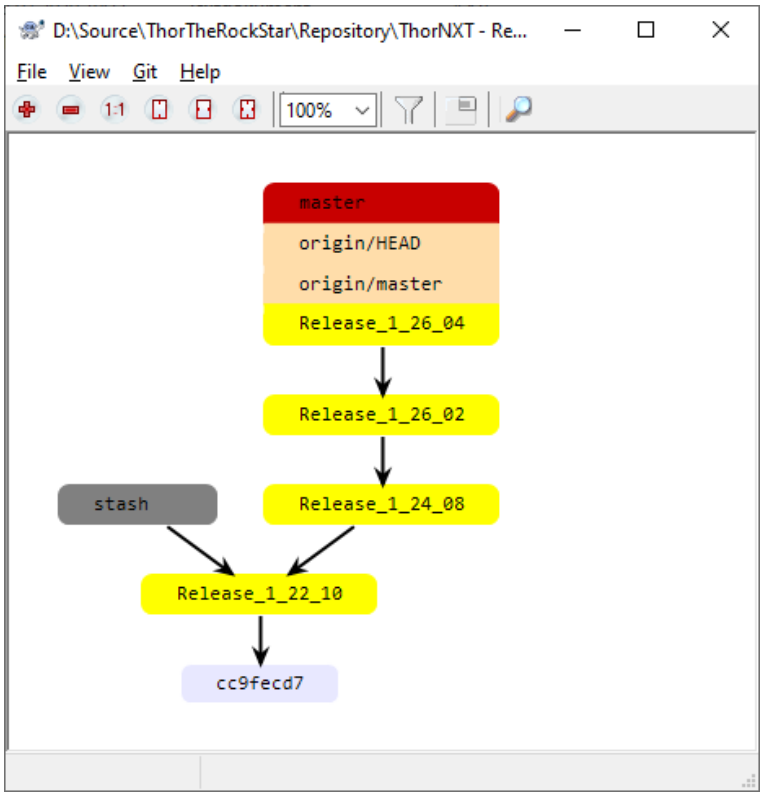


Abbildung 6: Versionskontrollsystem

15 Dokumentieren der Softwaretests/Fuzztests

Um zu dokumentieren, dass alle anwendbaren, vom Standard geforderten sicherheitsrelevanten Prozesse vor der Produktfreigabe abgeschlossen wurden und dass die notwendigen Aufzeichnungen zum Abschluss der einzelnen Prozesse verfügbar sind, gehen wir wie folgt vor.

Wenn wir eine neue Version für die Veröffentlichung vorbereiten, checken wir die letzten Testprotokolle in das Repository ein, nachdem wir das GIT-Repository getaggt haben, wenn die Firmware endgültig freigegeben wird.

Wir tun dies in Form eines Dokuments der Form „**LiftApp_Release_Version_x_yy_zz.pdf**“. Es enthält, was seit der letzten Version an der Software geändert wurde, und die Ergebnisse unserer Testverfahren, wie Fuzzing und Port-Scanner-Tests, aber auch die Tests, die wir zur Kennwortsicherheit und zu aufzug-spezifischen Funktionen durchgeführt haben. Jeder Eintrag enthält Angaben dazu, „wer“ und „wann“ der betreffende Test durchgeführt wurde. Dieses Dokument kann später im Falle eines Vorfalls überprüft werden, um festzustellen, ob ein Prozess verbessert werden muss oder versäumt wurde.

R – Verantwortlich (Responsible)

A – Rechenschaftspflichtig (Accountable)

C – Beratend (Consulted)

I – Informiert (Informed)

	Lars Gustafsson	Roy Schneider	Thomas Reul
(START)		R	I
(CREATE)		R	I
(TESTING)		R	R
(REVIEW)		R	A
(RELEASE)	A	R	I

16 LiftApp Update & Veröffentlichungsbenachrichtigung

Der Kunde wird immer über jede Veröffentlichung der LiftApp per E-Mail informiert.

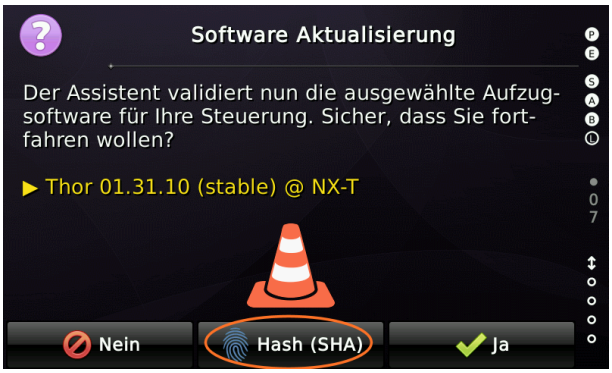
Dazu gehören hauptsächlich unsere OEM-Partner, die Steuerungsschränke bauen und an ihre Endkunden verkaufen. Dies kann auch Ingenieur- oder Planungsbüros oder Endkunden umfassen, die eine Benachrichtigung über eine neue Softwareversion beantragt haben. Dazu gehören auch unsere Entwicklungspartner in der Aufzugindustrie, wie zum Beispiel Antriebs-, Tür- und Positionsgeberhersteller.

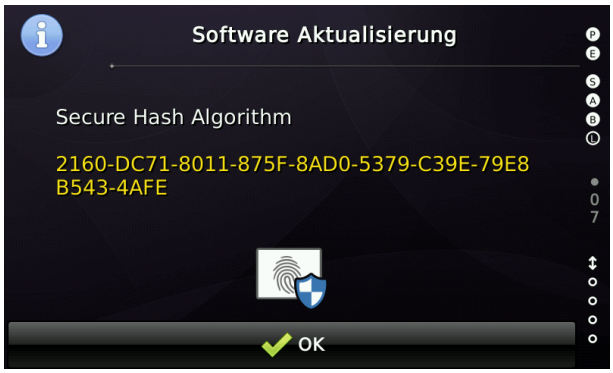
Diese Versionshinweise enthalten den Versionierungseintrag plus den sicheren Hash, den die installierbare Anwendung haben soll.

Beachten Sie immer, dass der Endkunde den **SHA überprüfen** muss, wenn er das Update durchführt, um sicherzustellen, dass die zu installierende Software seit ihrer Veröffentlichung nicht manipuliert wurde. Dieser SHA gibt Ihnen zusätzliche Sicherheit zum **automatisierten CRC32**-Validierungsprozess.

Ein Update ist generell nur möglich, wenn vor Ort der Aufzug auf Inspektion, Rückholsteuerung oder Nothalt geschaltet wurde. Das Setup-Passwort (🔑) muss korrekt lokal eingegeben werden und der automatische Validierungsprozess muss erfolgreich durchlaufen. Trotzdem wird immer empfohlen, den SHA manuell zu überprüfen. Sehen Sie dazu auch das Flussdiagramm auf Seite 47.

Erfolgt der Download des Updates nicht von einem lokalen USB/SD Massenspeicher, sondern direkt von der Cloud, so zwingen wir den Techniker, die letzten 4 Ziffern des Hashs vor der Anzeige des Selbigen am Bildschirm manuell einzugeben, um sicherzustellen, dass er/sie die E-Mail zuvor auch gelesen haben.

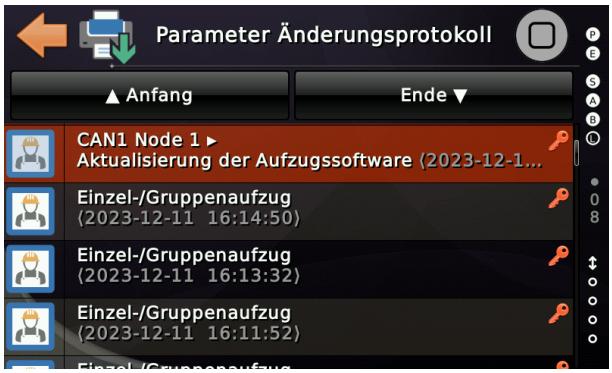




Das Update kann von einem USB-Massenspeicher, einer Micro-SD-Karte oder einer Cloudverbindung manuell heruntergeladen werden. Ein PUSH eines Updates auf die Steuergeräte ist nicht vorgesehen und wird von uns als Sicherheitsrisiko eingestuft, da wir der Meinung sind, dass der Techniker nach einem Update immer die funktionale Sicherheit der Anlage, sowie die korrekte Funktion von Elementen mit denen die Passagiere direkt in Interaktion treten, wie Türen und Lichtgitter u.ä. prüfen muss.

16.1 Update Dokumentation

In der Aufzugssteuerung wird das Software-Update dokumentiert, einschließlich Uhrzeit und Datum, der alten Version und der neuen Version, die installiert wurde. Dieses Änderungsprotokoll ist 256 Einträge groß und kann nicht gelöscht werden. Sind 256 Einträge erreicht, wird bei einem neuen Eintrag der älteste Eintrag verworfen.



Das Aufzugsparameter-Änderungsprotokoll kann aufgerufen werden, indem zunächst auf die Schaltfläche „Favoriten“ ( ) getippt und dann dem Pfad „System Menü“ → „Sicherheit“ → „Parameter-Änderungsprotokoll“ gefolgt wird.

16.2 Update und Funktionale Sicherheit

Wird ein Update an einer Aufzuganlage ausgeführt, so berührt dieses Update keine SIL-3 relevanten Funktionen. Solche Funktionen sind entweder in Hardware oder durch externe Komponenten, wie einen sicheren Geber (Positionsüberwachungs-einheit) realisiert.

Das Softwareupdate verändert selbstständig keine Betriebsparameter der Steuerung.

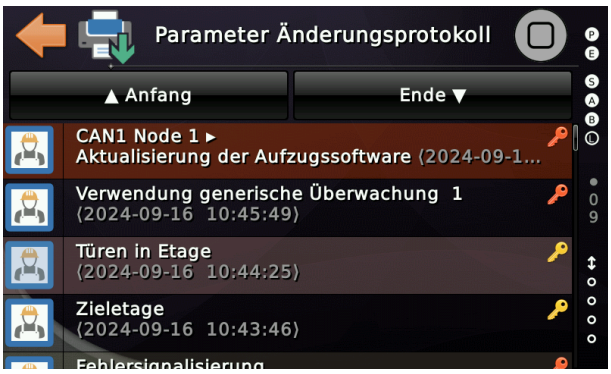
Wir empfehlen, zunächst eine Sicherungskopie der Software zu erstellen und dann das Update zu installieren, um im Falle eines unwahrscheinlichen und unvorhergesehenen Fehlers eine Rückkehr zur vorherigen Programmversion und damit die Betriebsbereitschaft des Lifts sicherzustellen. Vor der Aktualisierung der Software empfiehlt es sich, eine Softwaresicherung durchzuführen.

17 Sicheres Aktualisieren der Firmware

Zum **Aktualisieren der Firmware** können USB- und SD-Karten-Massenspeicher verwendet werden. Es ist auch möglich, eine neue Firmware manuell über die Cloud-Verbindung herunterzuladen, anstatt die Aktualisierungsdatei von einem USB-/SD-Massenspeicher zu lesen.

Die Aktualisierung der Software kann **nur manuell** und **nur lokal** auf dem Gerät erfolgen, wenn...

- Der Aufzug muss auf **Inspektion, Rückholsteuerung oder Not-Aus** geschaltet werden.
- Um die Firmware zu aktualisieren, muss der Setup-Code vor Ort eingegeben werden.
- Die Firmware-Datei wird von der Aufzugssteuerung validiert. Dazu werden der ELF-Token, der eingebaute CRC32 der Datei, die Hersteller-ID und der Produktcode überprüft.
- Zusätzlich muss der Techniker vor Ort den SHA der zuvor in den Versionshinweisen angegebenen Datei überprüfen. Die Versionshinweise wurden dem Techniker zuvor gesendet, normalerweise per E-Mail, also nicht auf die gleiche Weise wie die Datei, die normalerweise über einen Filesharing-Dienst übertragen wird. Die vorhandene Aufzugsoftware berechnet den SHA der angeforderten Datei vom USB-Stick/Micro-SD-Karte und zeigt ihn auf dem Bildschirm in einer leicht lesbaren Weise an.
- Nur wenn alle Anforderungen erfüllt sind, kann die Software aktualisiert werden.
- Jede Aktualisierung der Software wird auch im Parameteränderungsprotokoll aufgezeichnet, das vom Techniker nicht gelöscht werden kann.

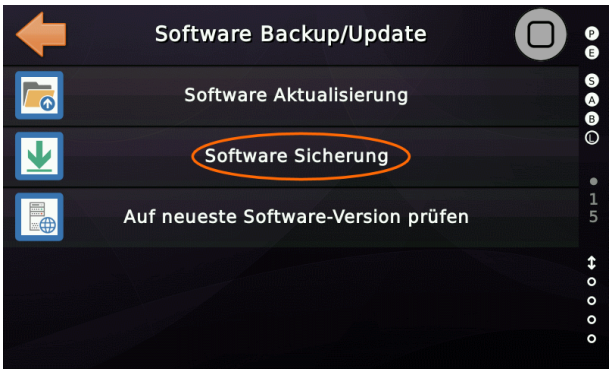


17.1 Erstellen einer Sicherungskopie der aktuellen Software

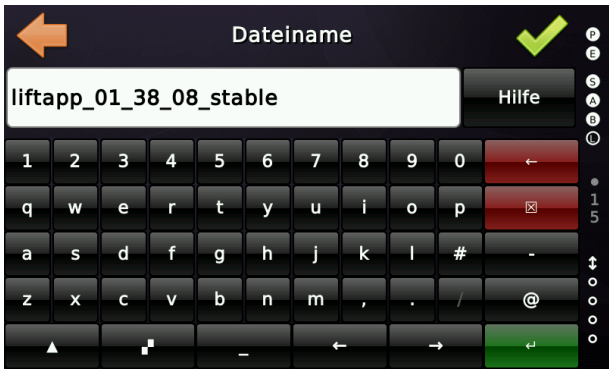
Um sicherzustellen dass im Falle eines unwahrscheinlichen und unvorhergesehenen Fehlers, die Rückkehr zur vorherigen Programmversion möglich ist und der Aufzug somit betriebsbereit bleibt, wird empfohlen vor dem Softwareupdate ein Softwarebackup auszuführen.

Den entsprechenden Parameter finden Sie, indem Sie zuerst auf „*Home*“ und dann auf die Hardware-Taste „*Favoriten*“ drücken und anschließend auf „*Systemmenü*“ → „*System*“ → „*Software-Backup/Update*“ gehen.

Beachten Sie, dass Sie den **SETUP**-Code eingeben und die Inspektion, die Rückholsteuerung oder den Not-Aus aktiviert haben müssen, um eine Sicherung (Backup) durchführen zu können.



Um eine Sicherung auf einen Massenspeicher (USB/SD) durchzuführen, wählen Sie „*Software Sicherung*“. Sie werden nun aufgefordert, den Namen und Speicherort manuell auszuwählen, wie im Dialog gezeigt.



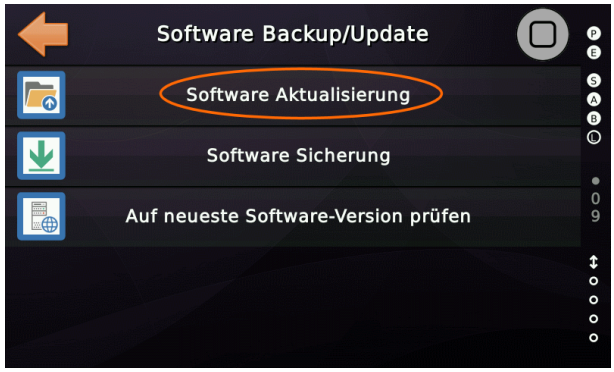
Um die Datei in einen anderen Ordner zu speichern, muss dieser doppelt angetippt werden, so dass in den Order gewechselt wird.

Die Sicherungsdatei kann nur verwendet werden, um dieselbe Steuerung erneut wiederherzustellen. Sie ist nicht zum Aktualisieren anderer Steuerungen geeignet.

17.2 Aktualisierung über SD-/USB-Massenspeicher

Den entsprechenden Parameter finden Sie, indem Sie zuerst auf „*Home*“ und dann auf die Hardware-Taste „*Favoriten*“ drücken und anschließend auf „*Systemmenü*“ → „*System*“ → „*Software-Backup/Update*“ gehen.

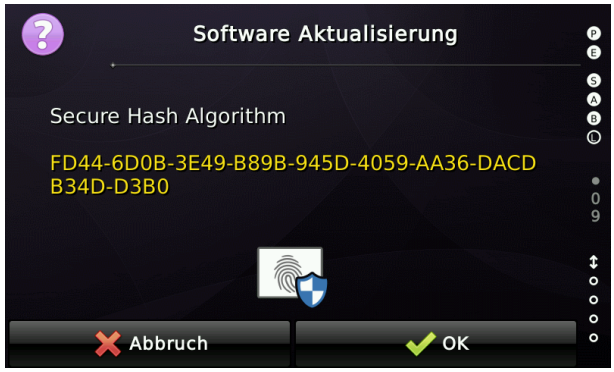
Beachten Sie, dass Sie den SETUP-Code eingeben und die Inspektion, die Rückholsteuerung oder den Not-Aus aktiviert haben müssen, um ein Update durchführen zu können.



Um ein Update über den Massenspeicher (USB/SD) durchzuführen, wählen Sie „*Software Aktualisierung*“. Sie werden nun aufgefordert, die Update-Datei manuell auszuwählen, wie im Dialog gezeigt.

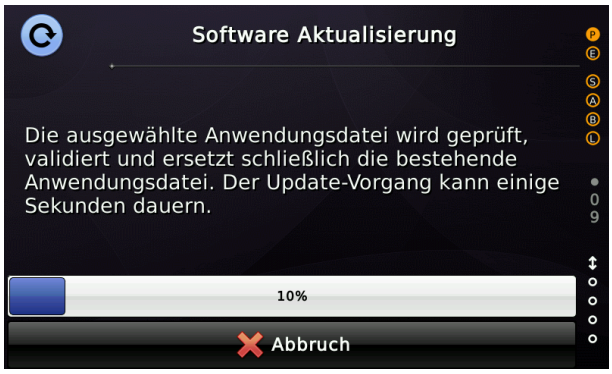
Der SHA der Update-Datei wird nun berechnet und auf dem Bildschirm angezeigt.

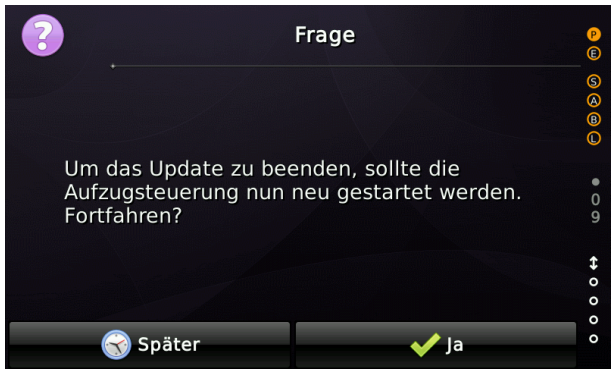
Die Aufgabe des Technikers besteht nun darin, den angezeigten HASH mit dem Hash aus der Release Note zu vergleichen, die er/sie zuvor per E-Mail erhalten hat.



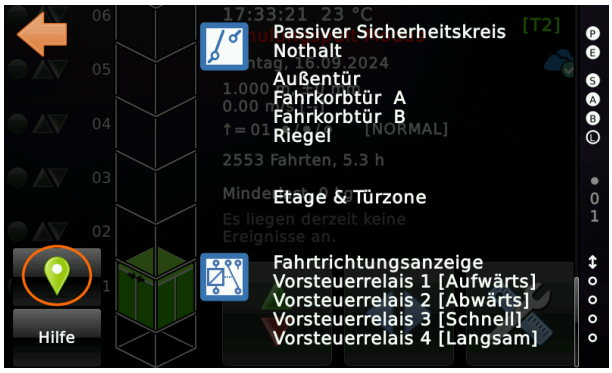
Nur wenn diese Hash-Nummer identisch ist, wird die Software durch Drücken von „OK“ installiert.

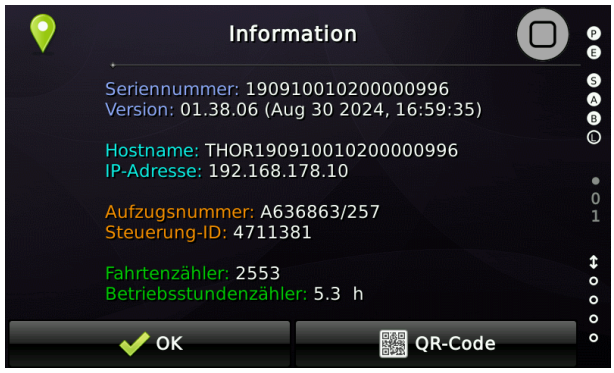
Wenn die SHA-Nummer übereinstimmt, drücken Sie „OK“, andernfalls „Abbrechen“.



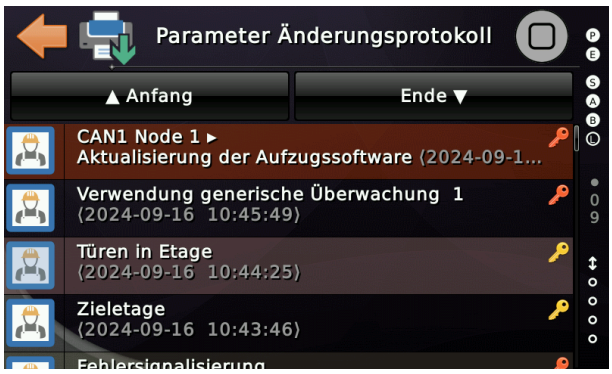


Nach dem Neustart der Aufzugsteuerungssoftware können Sie die Aktualisierung der Software prüfen, indem Sie auf dem Hauptbildschirm einmal nach unten wischen und anschließend über das grüne Symbol die Systeminformationen anzeigen.





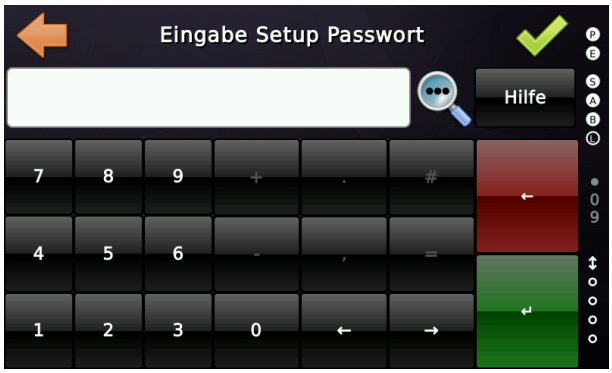
Alle Softwareaktualisierungen werden außerdem im Parameteränderungsprotokoll aufgezeichnet und können vom Techniker nicht gelöscht werden.

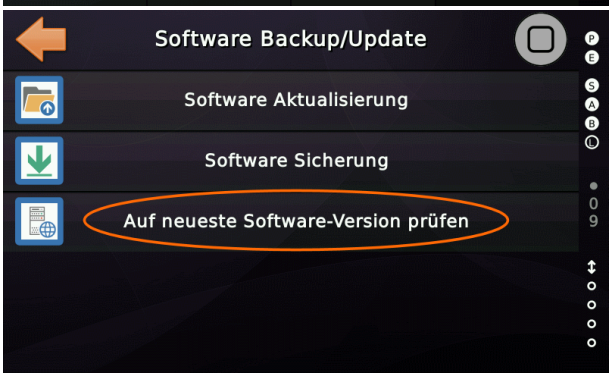


17.3 Aktualisierung über den Cloud-Dienst

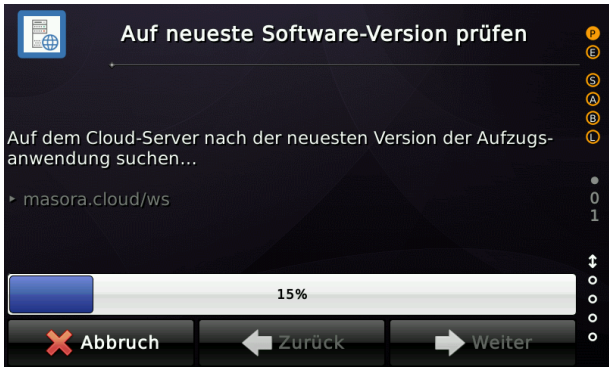
Den entsprechenden Parameter finden Sie, indem Sie zuerst auf „*Home*“ und dann auf die Hardware-Taste „*Favoriten*“ drücken und anschließend auf „*Systemmenü*“ → „*System*“ → „*Software-Backup/Update*“ gehen.

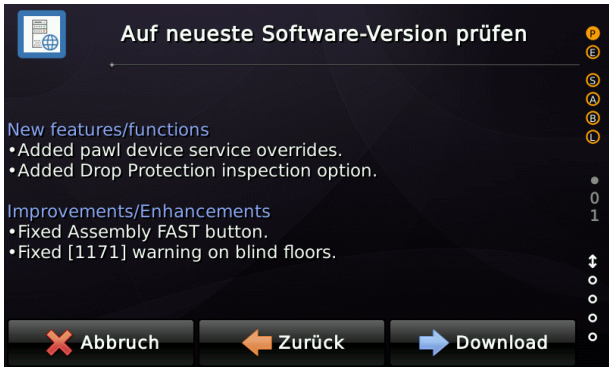
Beachten Sie, dass Sie den SETUP-Code eingeben und die Inspektion, die Rückholsteuerung oder den Not-Aus aktiviert haben müssen, um ein Update durchzuführen.

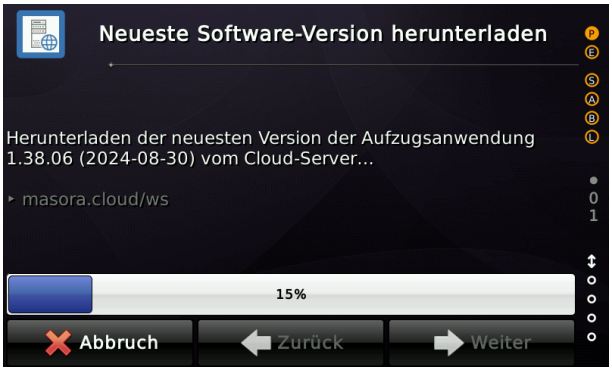


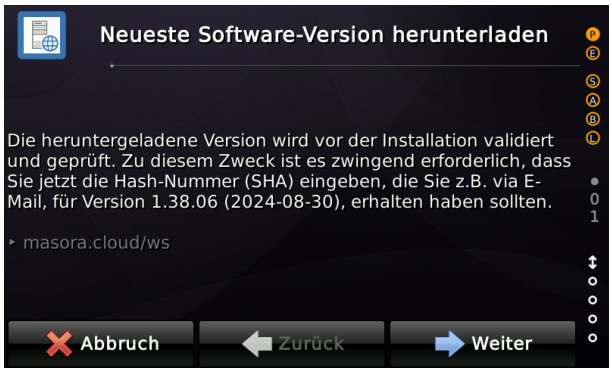


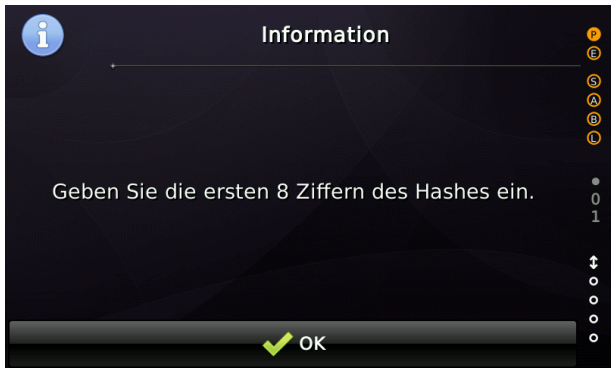
Um eine Update-Datei über den Cloud-Dienst zu laden, wählen Sie „*Nach neuester Softwareversion suchen*“. Wenn die Verbindung erfolgreich ist, wird Ihnen die aktuell installierte Version und die angezeigt, die als Update heruntergeladen werden kann.



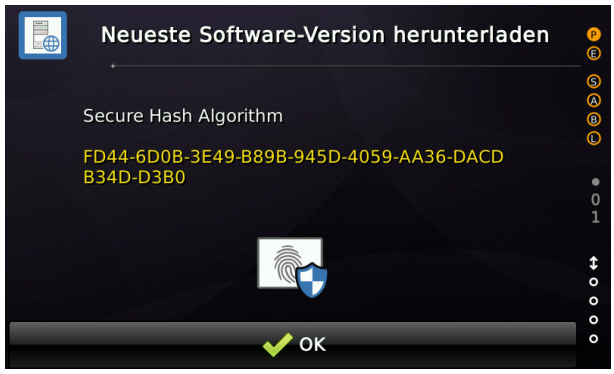




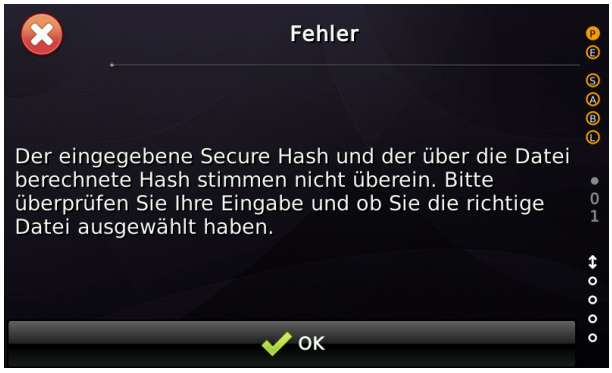


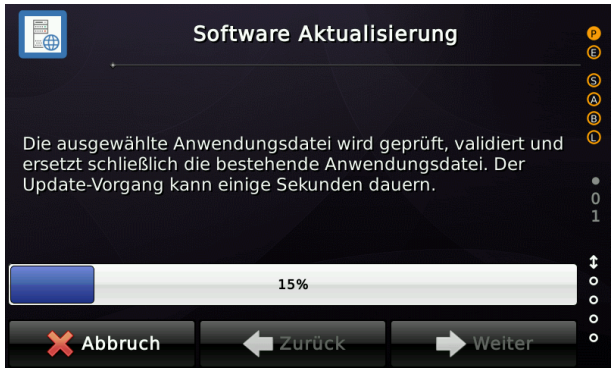


Der SHA der Update-Datei wird nun berechnet, aber **nicht** auf dem Bildschirm angezeigt. **Stattdessen muss der Techniker nun die ersten acht Ziffern des HASH aus der Release-Note eingeben, die er/sie zuvor per E-Mail erhalten hat.**

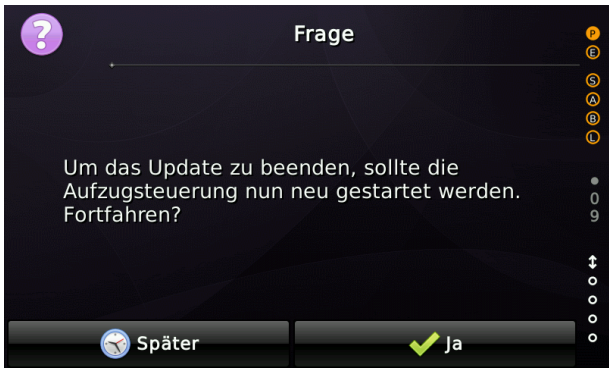


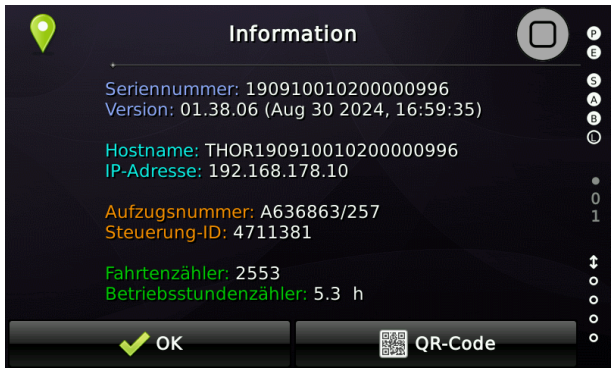
Sollte der Hash nicht übereinstimmen oder falsch eingegeben worden sein, kann der Updatevorgang nicht durchgeführt werden. Andernfalls wird das Update eingespielt.





Nach dem Neustart können Sie sehen, dass die Software aktualisiert wurde.





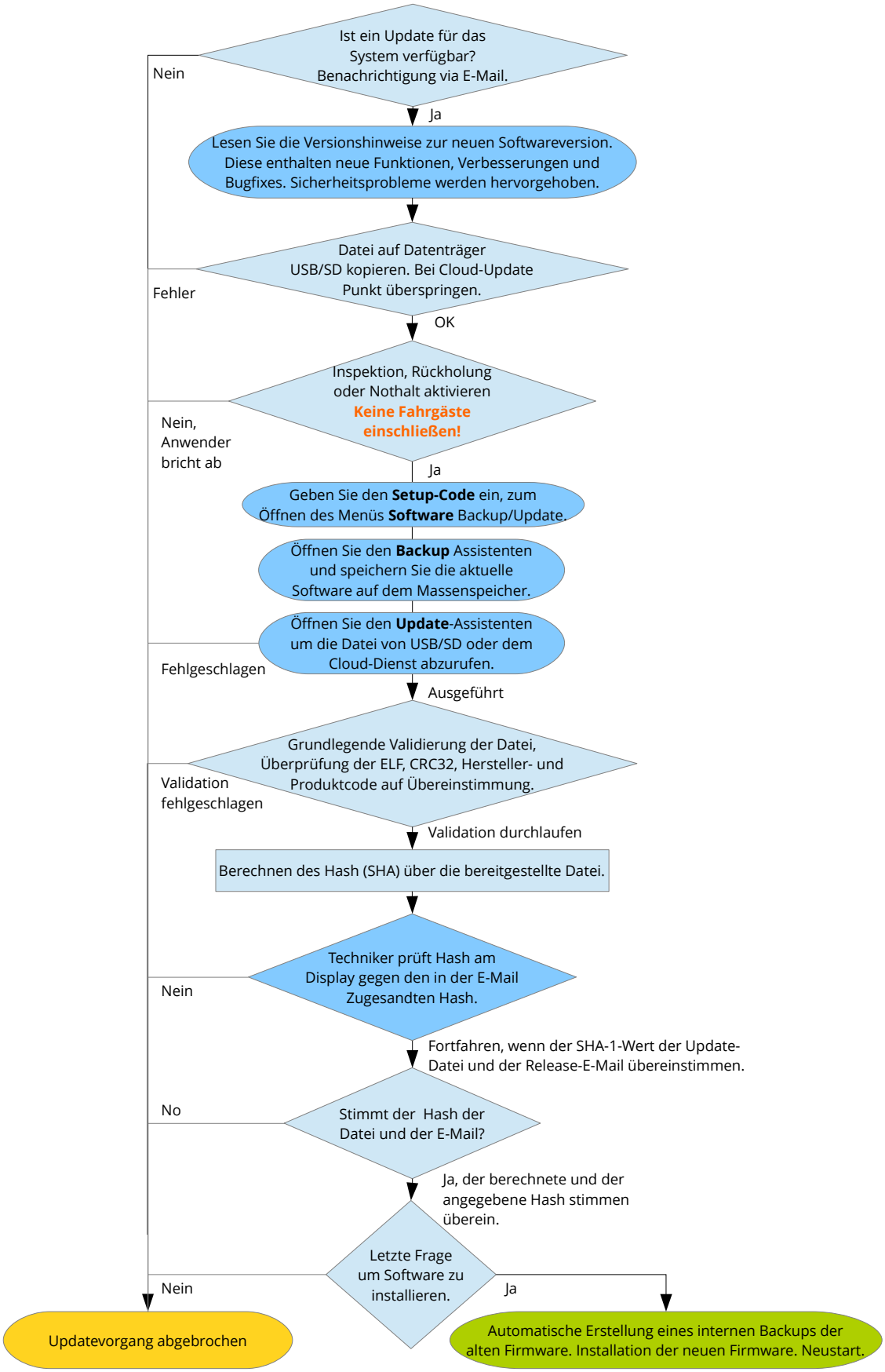
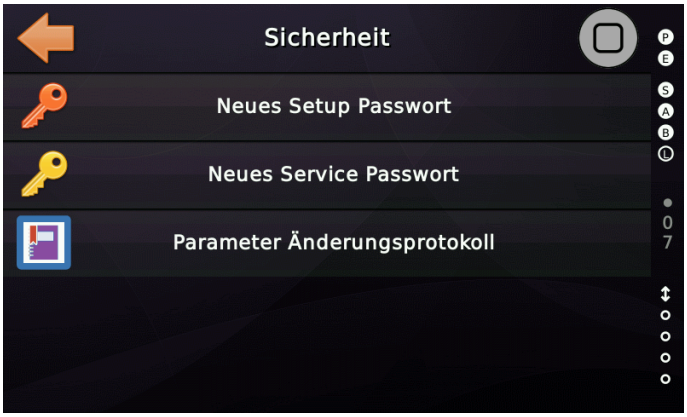


Abbildung 7: Flußdiagramm Software Backup/Update

18 Passwortsicherheit

Wir empfehlen dringend, dass die Kunden einen **Setup Code** und einen **Service Code** einrichten, um die unbeaufsichtigte Nutzung der Benutzeroberfläche der Aufzugssteuerung zu vermeiden.



Das Setup- und Service-Passwort sollte 6-stellig sein und Buchstaben und Zahlen enthalten.

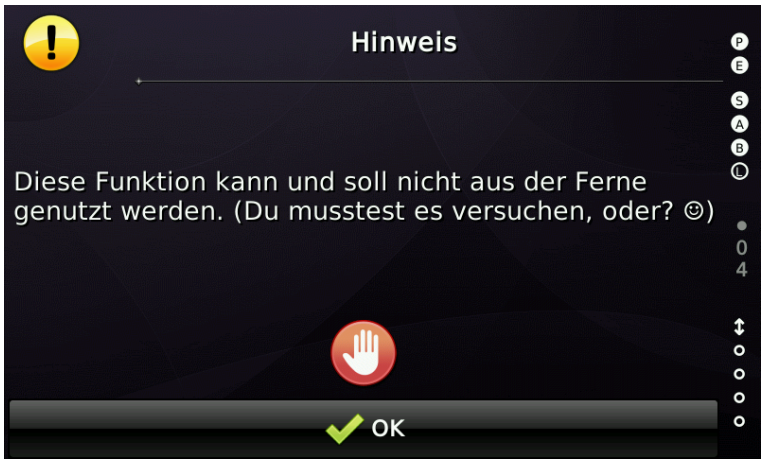
Das Setup Passwort (roter Schlüssel 🔑) sichert Parameter, wie die Zeiten der Schützüberwachung oder die Orientierung des verwendeten Positionsgebers. Das Service Passwort (gelber Schlüssel 🔑) sichert Einstellungen wie Parkzeiten oder Einstellungen zur Weiterfahrtanzeige.

Während das Servicepasswort innerhalb eines Wartungsunternehmens gleich sein kann, empfehlen wir für das **Setup**-Passwort ein **anlagenspezifisches Passwort**!

SETUP/SERVICE Passwörter werden grundsätzlich **nicht** im Speicher der Aufzugssteuerung gesichert. Stattdessen wird ein "*gesalzener*" (salted) SHA-1 (Hash) des Passworts gespeichert. Das bedeutet, dass die Aufzugssteuerung die Passworteingabe zwar sicher auf Echtheit prüfen kann, es aber nicht möglich ist, vom Hash auf die lesbare (sichtbare) Passwortzeichenfolge zurückzurechnen.

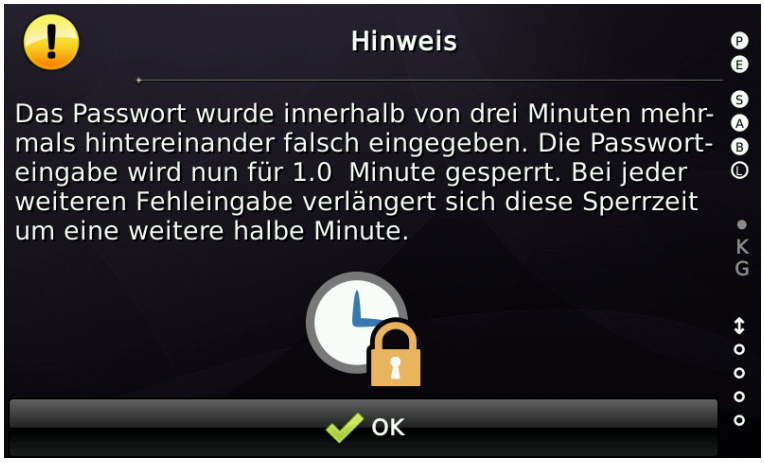
Bei Benutzerinaktivität muss die Passworteingabe wiederholt werden.

Einige Parameter und Assistenten sind mit dem Attribut „*Techniker vor Ort*“ gekennzeichnet und können nicht aus der Ferne verwendet werden. Diese sind im Menüausdruck gekennzeichnet.



Ab Version V1.24.18 (12-2023)

Wenn das Passwort, in einem Zeitraum von drei Minuten, dreimal hintereinander falsch eingegeben wurde, so wird die Passworteingabe für eine Minute gesperrt. Bei jeder weiteren Fehleingabe erhöht sich diese Sperrzeit um eine weitere halbe Minute. Wenn seit dem letzten Fehlversuch fünfzehn Minuten vergangen sind, werden die internen Zähler zurückgesetzt und es werden erneut drei Fehlversuche gewährt.

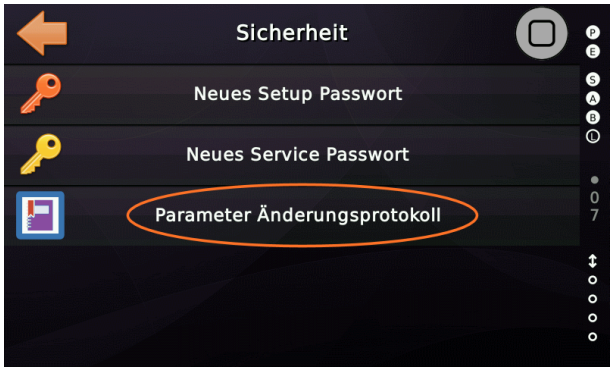


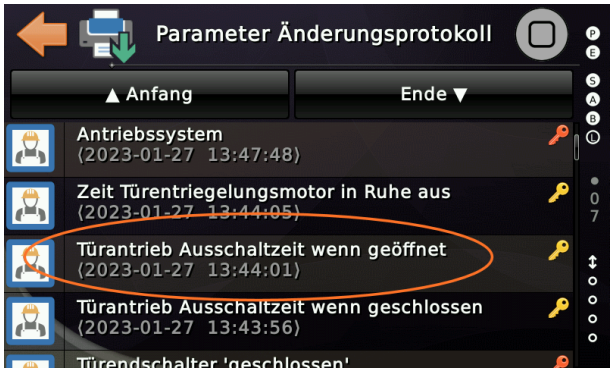
Ab Version V1.34.02 (09-2023)

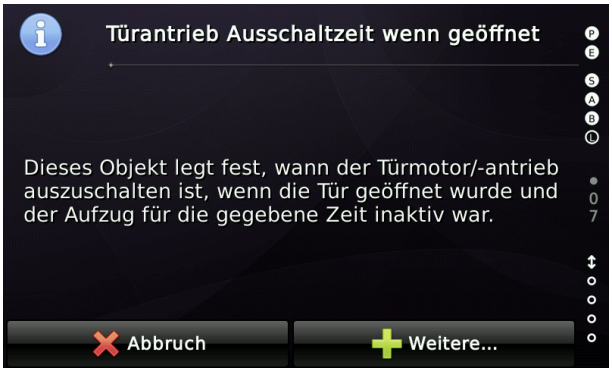
Nach Abschluss der Einstellfahrt wird der Techniker aufgefordert zumindest ein
SETUP Passwort anzulegen.

19 Aufzug Parameter-Änderungsprotokoll

Um alle an der Aufzugssteuerung vorgenommenen Parameteränderungen aufzuzeichnen, werden Änderungen im Gerät nichtflüchtig aufgezeichnet. Dieses Protokoll kann vom Endkunden nicht gelöscht werden. Es kann nur vom Hersteller zurückgesetzt werden. Es wird der geänderte Parameter, der alte Parameterwert, der neue Parameterwert und der Zeitpunkt der Änderung gespeichert.







Das Symbol eines Eintrages signalisiert, ob der Parameter vor Ort, durch das Bussystem, über einen temporär angebundenen WiFi Zugang, die Cloud oder einen Assistenten (z.B. Bremswege einlernen) geändert wurde.

20 Netzwerkanschluss

20.1 Allgemein

Die Geräte besitzen keine Mobilfunktechnik oder andere drahtlose Kommunikation an Board.

Über den integrierten kabelgebundenen Netzwerkanschluss (RJ-45) kann das Gerät an einen Router angebunden werden, der wiederum eine Netzwerkanbindung bereitstellt. Sollte es sich um eine drahtlose Anbindung handeln, ist es zwingend erforderlich dieses temporäre Netzwerk mit WPA2-PSK zu verschlüsseln und den Zugriff über ein sicheres (8 stelliges) Passwort zu schützen. Aus Sicherheitsgründen verwenden die Steuergeräte standardmäßig eine randomisierte MAC-Adresse. Dies kann jedoch bei Einbindung in dauerhafte Netzwerke (s.u.) umgestellt werden.

Bei Anbindung an ein Gebäudenetzwerk, wie es zum Beispiel bei Krankenhäusern der Fall ist, empfehlen wir für Haustechnik, wie Aufzüge, Klima, Beleuchtung usw. ein eigenes VLAN zu verwenden. Ein sogenanntes VLAN ist ein Virtual Local Area Network, also ein logisches Teilnetzwerk eines physischen Netzwerkes Local Area Network (LANs). Das Virtual Local Area Network bildet ein eigenes Netzwerksegment und eine eigene Broadcast-Domäne.

Wir raten davon ab, die Aufzugsteuerung an das gleiche logische Netzwerk zu hängen, wie Drucker, Büro-PC's und ähnliches Equipment, da deren physikalischen Zugänge oft leicht zu erreichen sind.

Bei dauerhafter lokaler Netzwerkanbindung empfehlen wir außerdem, die Verwendung gemanagter Switche, bei denen die MAC des Teilnehmers an einem Netzwerkanschluss vorgegeben werden kann. Dazu kann die randomisierte MAC-Adresse in der Steuerung durch eine feste (vom Endkunden vorgebbare) MAC-Adresse ersetzt werden.

Wir raten davon ab, die Aufzugsteuerung an ein Gebäude WiFi® zu hängen, auch wenn dieses geschützt ist.

20.2 Fuzzing der Schnittstellen

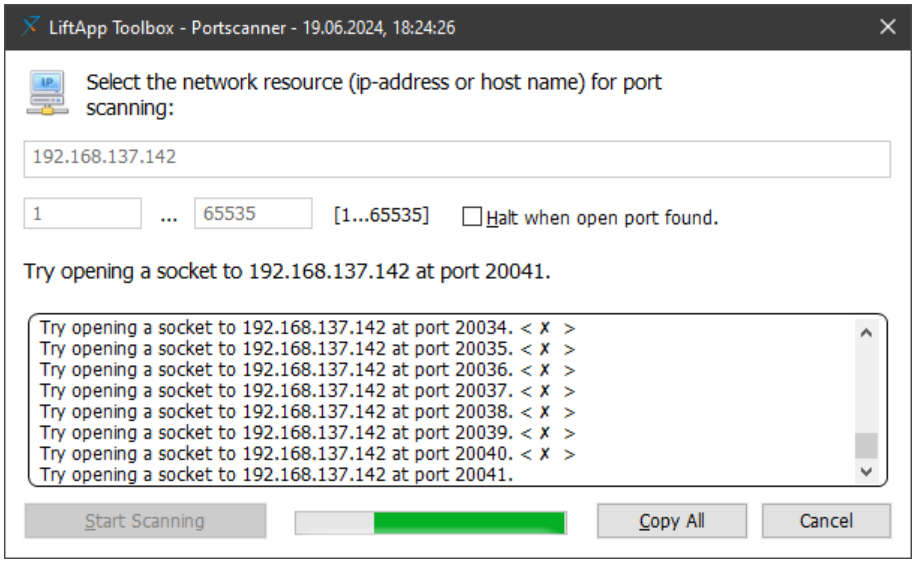
Die Netzwerkschnittstellen werden vor der Freigabe einer neuen Release mithilfe des Tools POSTMAN unter Verwendung von '*Naughty Strings*' gefuzzt. Diese **Fuzzing-Tests** beinhalten falsch geformte HTTP-Headers, HTTP-Bodies und fehlerhafte Eingaben. Außerdem wird geprüft, ob durch massenweises Fluten der Schnittstellen das Aufzugprogramm in seiner Performance negativ beeinflusst werden kann.

► Siehe auch Webserver und Cloud-Schnittstelle auf Seite 22.

20.3 Offene Netzwerkports

Standardmäßig ist **kein Netzwerkport** geöffnet. Der Kunde kann jedoch einen Port öffnen, indem er einen für das Projekt erforderlichen Dienst wie den Webserver aktiviert. Damit wir überprüfen können, ob standardmäßig kein Netzwerkport geöffnet ist, haben wir unser eigenes Port-Scanner-Tool erstellt, das in unsere LiftApp Toolbox integriert ist.
Bevor eine neue Software veröffentlicht wird, wird mit diesem Scanner überprüft, ob in der neuen Version ein Port unbeabsichtigt geöffnet wäre, beispielsweise wenn der Softwareentwickler einfach vergessen hätte, einen zu Debug Zwecken geöffneten Port abzuschalten.

Netwerk Portscanner
Diese Utility versucht ein offenes Port auf der Netzwerkresource zu finden und liefert das Ergebnis zurück.



Das Tool liefert bei der Veröffentlichung einer Firmware ein Protokoll, welches wir archivieren.

```
LiftApp Toolbox - Portscanner - 19.06.2024, 18:24:26

19-06-2024, 06:24PM

Try opening a socket to 192.168.178.10 at port 1.
Try opening a socket to 192.168.178.10 at port 2.
Try opening a socket to 192.168.178.10 at port 3.
…
Try opening a socket to 192.168.178.10 at port 65535.
```

21 USB/Micro-SD Karten Sicherheit

Das Gerät ist mit USB-Host-Anschlüssen und einer Micro-SD-Karte ausgestattet, die Massenspeicher für Datei-Ein/Ausgabe unterstützt. Der USB- und der SD-Kartensteckplatz kann verwendet werden, um Textausdrucke, wie die Fehlerhistorie, das Parameteränderungsprotokoll, den Parametersatz, den Statistikausdruck zu speichern. Auch das Loggen auf einen Massenspeicher zur Fehlersuche wird unterstützt. Neuere Geräte unterstützen auch den zeitweiligen Anschluss von USB-Routern, die die USB-CDC Klasse verwenden.

Der USB- und SD-Karten-Massenspeicher kann jedoch auch zum **Aktualisieren der Firmware** verwendet werden. Eine Beschreibung zur Durchführung einer Softwaresicherung oder -aktualisierung finden Sie im Kapitel LiftApp Update & Veröffentlichungsbenachrichtigung auf Seite 39.

22 DEBUG Schnittstelle

Die Geräte haben eine nicht bestückte TTL-3.3V-UART DEBUG-Schnittstelle, die weder über einen Stecker, noch einen Sockel auf der Platine erreicht werden kann, weil diese **nicht bestückt** sind. Wenn jedoch jemand einen passenden Steckverbinder einlötet, die Pinbelegung durch Reverse-Engineering ermitteln würde und dann einen speziellen Hardware-Adapter verwendet, erhält der Eindringling das Boot-Protokoll auf dieser UART. Dieses enthält keine sicherheitsrelevanten Daten, auch keine Seriennummern, Passwörter oder ähnliches. Sobald das Aufzugprogramm startet, wird die Schnittstelle dann vollständig funktionsunfähig. Um diese Schnittstelle nutzbar zu machen, muss zunächst eine spezielle Entwicklungsversion der LiftApp durch einen Techniker des Herstellers installiert werden.

23 Micro-USB-Anschluss

Auf einigen Geräten ist ein USB-Micro-Anschluss bestückt. Dieser ist nicht für die Verwendung durch den Kunden gedacht, sondern nur für den Hersteller zu Reparaturzwecken verfügbar. Um diesen Anschluss nutzen zu können ist dieser Anschluss mit einem 8-stelligen alphanumerischen und **zufälligen** Passwort geschützt. Alleine der Hersteller hat eine Tabelle mit der Zuordnung zwischen Seriennummer und Passwort. Mit dieser Schnittstelle ist zum Beispiel ein Zurücksetzen des Parameteränderungsaufzeichnung durch den Hersteller möglich, wenn ein Gerät zur Reparatur eingesendet und nicht wieder an den Kunden zurückgeht. Einen '*Generalschlüssel*' gibt es nicht und wird es auch nicht geben!

24 Sicherheitskreisabfrage

Die Abfrage der Sperrmittelschalter erfolgt über unsere baumustergeprüfte Abfrageschaltung hardwarebasierend. Eine softwarebasierende Übertragung der Zustände der Fahrkorbtüren, der Schachttüren oder der Türverriegelungen, über das Bussystem wird von uns **nicht** unterstützt. Diese Signale müssen drahtbasierend verlegt werden und werden von uns unmittelbar an der Steuerung verarbeitet. Das Umprogrammieren der Funktionen dieser Klemmen ist weder lokal noch aus der Ferne möglich – auch nicht über das Bussystem.

Der Versuch des Zugriffs auf diese Klemmen über das Bussystem wird nicht ausgeführt und mit einem Abortcode beantwortet:

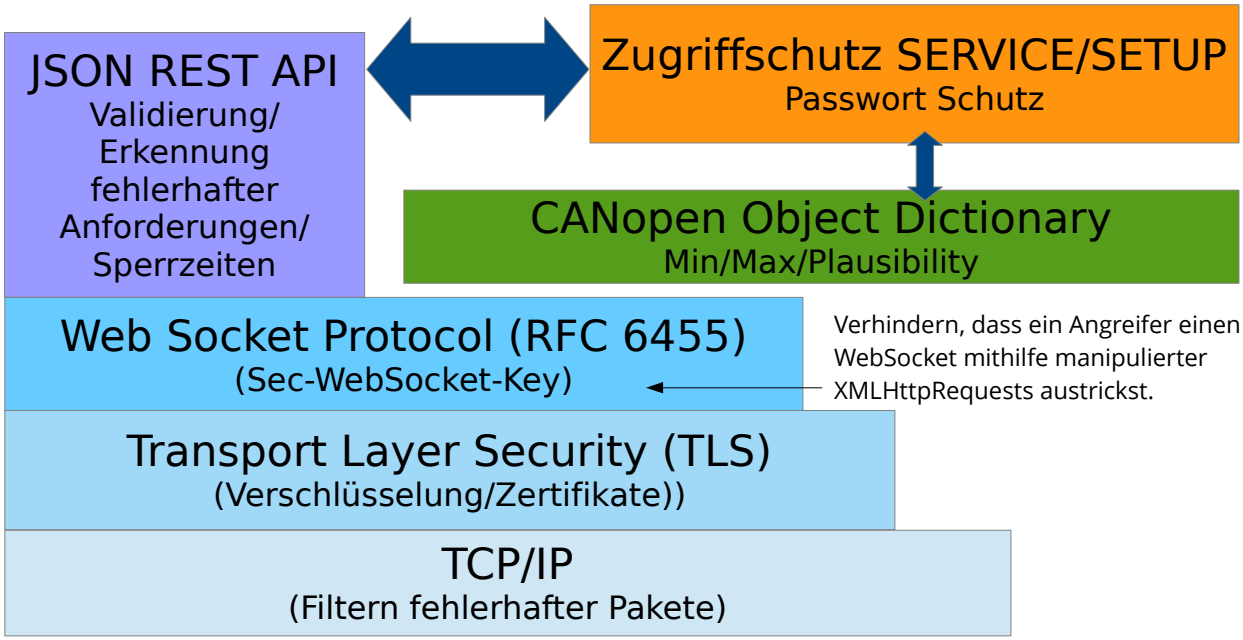
```
Node    1, ▸ RSDO initiate download 0x6100:0x13, number of bytes 6
Node    1, ▸ TSDO abort 0x6100:0x13 code 0x08000020 'Data cannot be transferred or stored to the application.'
```

Ein manipulieren der Signalzustände der Sperrmittelkette ist weder lokal, noch über das Bussystem, noch aus der Ferne möglich.

Die in CANopen definierten Eingangsfunktionen der Sicherheitskreissignale werden von uns nicht unterstützt, da wir die Übertragung der Sicherheitskreissignale über den Bus nie unterstützt haben.

25 NeXt® Cloud Sicherheit

Um sicherzustellen, dass es keinen unbeaufsichtigten Zugriff auf den Aufzug über die Cloud-Lösung gibt, der dazu führen könnte, dass der Aufzug angegriffen wird, verfügt die Cloud-Lösung von Thor standardmäßig über TLS-Verschlüsselung und eine zertifikatbasierte Serverauthentifizierung ohne Kompromisse. Das Transport Layer Security (TLS) ist der Nachfolger des älteren und inzwischen veralteten Secure Sockets Layer (SSL). Es ist ein kryptografisches Protokoll, das entwickelt wurde, um Kommunikationssicherheit über ein Computernetzwerk bereitzustellen. Dieses Protokoll ist bereits weit verbreitet in Anwendungen wie E-Mail, Online-Banking und Instant Messaging. Diese Protokollschicht umfasst Verschlüsselung und einen zertifikatbasierten Handshake, um zu überprüfen, ob der Cloud-Server wirklich derjenige ist, mit dem sich der Aufzug verbinden möchte, und nicht eine „Fälschung“, die durch einen DNS-Angriff erstellt wurde. Das verwendete Serverzertifikat enthält den Servernamen und die vertrauenswürdige Zertifizierungsstelle (CA), die das Domänenzertifikat ausgestellt hat. Es enthält auch den öffentlichen Schlüssel des Servers, der zum Verschlüsseln der Nutzdaten verwendet wird.



Der Fernzugriff auf den Bildschirm ist mit Einschränkungen verbunden. Funktionen und Parameter, die eindeutig nicht für die Remote-Nutzung vorgesehen sind, sind als „Techniker vor Ort“ gekennzeichnet und können nicht remote bedient werden. Der Schutz der Parameter (SETUP & SERVICE) durch Passwörter ist derselbe.

Wir empfehlen dringend, die Aufzugssteuerung bei der Verbindung mit der Cloud-Lösung mit einem SETUP & SERVICE-Passwort zu schützen. Diese Passwörter sollten auf keinen Fall in den „Notizen“ zum Aufzug in der Cloud selber abgelegt werden.

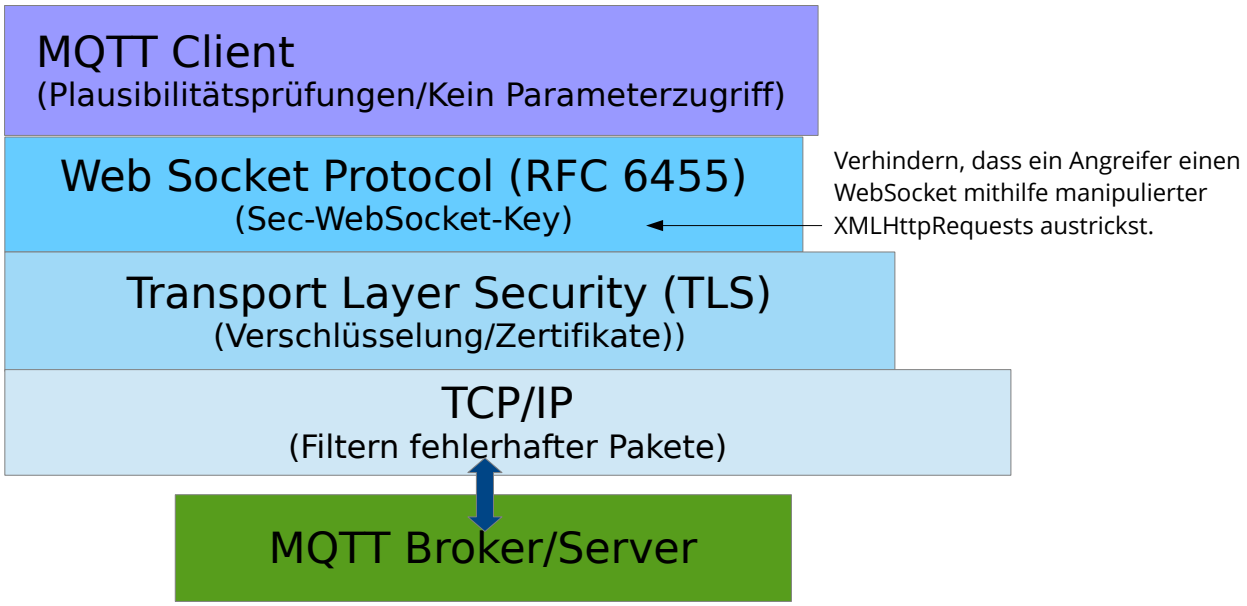
26 MQTT Schnittstelle Sicherheit

MQTT steht für „*Message Queuing Telemetry Transport*“. Es ist ein offenes Nachrichtenprotokoll. Es wird in der Regel für M2M (Maschine-zu-Maschine-Kommunikation), wie z.B. beim Internet der Dinge, eingesetzt.

Die Aufzugsteuerungssoftware LiftApp stellt einen MQTT-Client zur Verfügung, der im eigenen Haus mit dem Fokus auf Robustheit und Sicherheit entwickelt wurde. Diese Schnittstelle muss explizit lokal am Gerät aktiviert werden. Per Fernzugriff ist dies nicht möglich. Neben dem Schreiben von robustem Code und der Sicherstellung, dass eine ungültige MQTT-Nachricht das System nicht beschädigt oder zum Absturz bringt, ist es wichtig, dass auch das MQTT-Broker-System auf Kundenseite robust und ordnungsgemäß vor unbeaufsichtigtem Zugriff geschützt ist.

Der Kunde muss sicherstellen, dass Sicherheitsupdates zeitnah installiert werden und sein MQTT-System vor unbefugten, äußeren Zugriff geschützt bleibt.

Um sicherzustellen, dass auch die Verbindung und der Transport der MQTT-Nachricht von und zum Broker/Dienst sicher ist, empfehlen wir die Verwendung der integrierten TLS-Unterstützung. Das bedeutet, dass die MQTT-Nachricht über eine TLS-verschlüsselte WebSocket-Verbindung transportiert wird.



Bei der Verbindung über das Internet ist der Secure Socket (TLS) der definitiv bevorzugte Verbindungsmodus. Wenn Sie das System in einer lokalen Fabrik- oder Krankenhausumgebung betreiben und dort ein sicheres und gekapseltes Netzwerk für technische Einrichtungen wie Aufzüge verwendet wird, können Sie sich für die einfacheren Verbindungsmodi entscheiden.

Wir testen die MQTT-Schnittstelle regelmäßig mithilfe einer **Fuzzing Testmethode**, die tausende von fehlerhaften MQTT-Nachrichten erzeugt, die ungültige Nachrichten-

typen, ungültige Restlängenindikatoren, ungültige Header und Nutzdaten und auch weißes Rauschen enthalten. Diese Nachrichten werden dann an den MQTT-Nachrichtenparser gesendet, um zu überprüfen, ob alle denkbaren und unerwarteten Fehlerfälle ordnungsgemäß behandelt werden.

Informationen zum Fuzzing-Test, die an der MQTT-Schnittstelle durchgeführt werden, finden Sie hier unter Fuzzing Tests der MQTT-Schnittstelle auf Seite 23.

26.1 MQTT Einstellungen und Verbindungsstatus

Die notwendigen Einstellungen finden Sie durch Drücken der Hardwaretaste 'Favoriten' um dann nach 'System Menü' → 'Netzwerk' → 'Weitere...' → 'Noch mehr' → 'Viel mehr' → 'MQTT-Unterstützung' zu verzweigen. Auf der letzten Seite der MQTT-Einstellungen, finden Sie den Verbindungsstatus . In diesem Beispiel wird eine verschlüsselte TLS-WebSocket verwendet, um eine Verbindung zu einem Broker mit QoS-Level 1 (Quality of Service) herzustellen.

Abbildung 8: MQTT Verbindungsstatus

26.2 MQTT Zugriff auf den Aufzug

Der Zugriff auf den Aufzug per MQTT ist eingeschränkt. Es ist nur möglich Rufe zu geben, den Taster-Tür-Auf/Zu zu betätigen und spezielle Feldbusklemmen per MQTT zu schalten. Gedacht ist die Schnittstelle primär für Fabrikumgebungen, in denen automatisierte Fahrzeuge den Aufzug verwenden. Aber auch im Bereich der Gebäudeautomation ersetzt MQTT stetig ältere Feldbussysteme, z.B. in Krankenhäusern.

27 Testen eines Release Kandidaten

Bevor eine Version als stabil gekennzeichnet und schließlich für die OEM-Partner freigegeben wird, muss sie mehrere Testverfahren bestehen. Dazu gehört der Funktionstest durch den Autor der Software sowie ein „Black/Gray“-Test durch unsere Servicekollegen, der nicht in den Entwicklungsprozess einbezogen wurde.

27.1 Standard Testliste

- *Überprüfen, ob die neue Version durch die vorherige Version ersetzt werden kann. Es wäre fatal, wenn eine ausgelieferte Software nicht mehr aktualisiert werden kann, ohne dass das Gerät ins Werk zurückgeschickt wird.*

- *Testen der Warmstart- und Kaltstartfunktion. Das Ändern bestimmter Parameter erfordert einen Neustart des Geräts. Um das so schnell wie möglich zu machen, unterstützt die Software intern einen schnellen Warmstart, bei dem der POSIX-Prozess am Leben erhalten wird.*

- *Überprüfen der Funktion zum Zurücksetzen auf Werkseinstellungen und zum Zurücksetzen der Onboard-Anschlüsse. Wenn Platinen ersetzt oder ausgetauscht werden, ist ein Zurücksetzen auf die Werkseinstellungen oder ein Zurücksetzen der On-Board-Anschlüsse erforderlich. Da diese Funktion nicht so oft verwendet wird, würde ein Fehler nicht sofort erkannt. Es ist also wichtig, es hier zu testen.*

- *Überprüfung der Funktionen zum Ausdrucken der Parameter, der Historie, des Parameteränderungsprotokolls und der Mengenliste der Störungen.*
  Dieser Test wird in zwei Sprachen, Deutsch und Englisch, durchgeführt, um sicherzustellen, dass wir Probleme mit Umlauten erkennen.

- *Überprüfung der normalen Fahrt im Betriebsmodus Positionsprofil und Geschwindigkeitsprofil. Bei diesem Test ist es sinnvoll, auch die Schnellstartfunktion in beiden Betriebsarten zu testen, da das im Ablauf einen Unterschied macht.*

- *Prüfung der Inspektion, der Rückholsteuerung und der gleichzeitigen Betätigung beider Inspektionssteuerungen. Prüfung, ob die Inspektion Vorrang vor der Rückholsteuerung hat.*

- *Testen des Reset-Betriebs der Grubeninspektion über einen klassische Eingang, über*

das Display (Schweden) und über die alternative Eingabemethode unter Verwendung eines Impulscodes.

- *Überprüfung der Nachregulierung und SZ-Fehlererkennung sowie Generierung von Warnungen bei verspätetem Wegfall des Zonensignals.*

- *Überprüfung der Not-Aus-Funktionen, Außerbetriebnahme, Wartungsmodus und Montagebetriebsmodus.*

- *Löschen aller Etagenpositionen und Betreiben des Geräts im Montagemodus über die Rückholsteuerung.*

- *Durchführung einer manuellen und automatischen Lernfahrt mit einem einfachen Positionsgeber (Wachendorff, ELGO 2M) und einem sicheren Positionsgeber ELGO33CP und Safe ANTS.*

- *Verwendung des UCM-Testassistenten, des Endschalter-Testassistenten, des Puffertest-Assistenten, des Laufzeittest-Assistenten und des Sicherheitskreisbrücken-Testassistenten.*

- *Testen der Überwachungsfunktionen Schützüberwachung und Bremskontaktüberwachung.*

- *Testen der Steuerspannungsausfall- und Fahrkorblichtspannungsausfallerkennung. Testen der Phasenausfallerkennung.*

- *Prüfung der nichtflüchtigen Blockierung für den passive Sicherheitskreis. Testen der Außensteuerung-Aus-Funktion und des Innenvorzuges.*

- *Testen der hydraulischen Rücksendefahrt, der Startüberwachung, der Laufzeitüberwachung und des Verzögerungsüberwachungstimers.*

- *Prüfung der Drehrichtungsüberwachung und Überwachung der Fahrkorbbewegung.*

- *Testen der Cloud-Schnittstelle.*

- *Testen der Brückenerkennung des Sicherheitskreises.*
- *Testen der Minderlast, Volllast und Überlastfunktion.*
- *Testen der Energiespar- und Standby-Funktionen. Dazu gehören auch die Timer für die Etagendisplays.*
- *Testen der Sammel- und SFR-Rufverarbeitung.*
- *Brandfall testen (einfacher, dynamischer und Feuermeldezentralen-Modus).*
- *Testen von Feuerwehrfahrt in den Varianten EN81 und US-ASME.*
- *Testen der Notstromfunktion. Dieser Test umfasst die Überprüfung der Signale, die für die Notstromversorgung mehrerer Aufzüge in Folge verwendet werden.*
- *Testen der Notbefreiungsfunktion.*
- *Testen der Lösung für niedrige Grube und Schachtkopf, einschließlich Test einer Barriere auf dem Fahrkorb.*
- *Testen der benutzerdefinierten Temperaturschwelleneingänge und auch der Erfassung der Umgebungstemperatur, wenn die in den Normen festgelegten maximal zulässigen Werte überschritten werden.*
- *Testen der IO-Klemmen des NX-T2/3/E, des M18 und der Nous-Boards.*
- *Testen von gegenseitig verriegelten Türen.*
- *Testen der zusätzlichen Türüberwachung.*
- *Prüfung der Trenntürüberwachung.*
- *Prüfung von Automatiktüren, Drehtüren mit automatischer Fahrkorbabschlusstür und reinen Drehtüren mit Sicherheitslichtgittern.*

- *Testen der Einfahrt mit früh öffnenden Türen, einschließlich des Abfallens der Türzone, des Fehlens der Türzone und des Klemmens/Hängens der Türzone.*

- *Überprüfung ob der Sperrvorgang nichtflüchtig gespeichert wird, also ob ein elektrischer Neustart die Sperre nicht automatisch aufhebt.*

- *Überprüfen der Funktion des Drehtüröffners.*

- *Überprüfen der Erkennung eines dauerhaft unterbrochenen Lichtvorhangs.*

- *Prüfen, ob die 'Endschalter «geschlossen» Brücken-/Hängererkennung' funktioniert.*

- *Testen des QR-Code-Generators.*

27.2 Erweiterte Testliste (Neue Funktionen)

- *Ein ausführlicher Test der neuen Funktionen auf der Release-Liste.*

- *Ein ausführlicher Test der aktualisierten Funktionen auf der Release-Liste.*

- *Ein Test der neuen und/oder aktualisierten Funktionen durch ein Servicekollegen, der das neu oder aktualisierte Kapitel des Software-Referenzhandbuchs nimmt und dann einfach ohne weitere Anweisungen versucht, die Funktion ordnungsgemäß zum Laufen zu bringen. Dies kann dazu führen, dass das entsprechende Kapitel des Handbuchs erneut aktualisiert wird, bevor die Software aktualisiert wurde.*

- *Fuzz-Testen der Netzwerkschnittstelle (eingebauter Webserver und JSON-REST-API) unter Verwendung von POSTMAN und unseren Testsammlungen, wobei eine aktuelle „Liste unartiger Zeichenfolgen“ verwendet wird, die auf die Eingabefelder abgefeuert und zur Erstellung fehlerhafter Blöcke, wird defekter HTTP-Header, fehlerhafter HTTP-Bodies und ungültiger JSON-Requests und gültiger JSON-Requests, die jedoch ungültige oder unerwartete Daten enthalten, verwendet wird.*

- *Fuzz-Testen der MQTT Schnittstelle über eine eigens dafür geschriebene Testmethode, die zufällige fehlerhafte MQTT-Nachrichten (unter anderem mit fehlerhaften Längenkennzeichnungen) erzeugt.*

27.3 White-, Gray- und Black-Box-Tests

Bevor eine neue Softwareversion der LiftApp veröffentlicht wird, durchläuft die Software den Testplan (*siehe Standard Testliste auf Seite 58 und folgende*), bei dem der Softwareentwickler und der Servicetechniker neben den vorhandenen Funktionen auch die neu implementierten Funktionen testen.

R – Durchführend A - Verantwortlich C – Beratend I – Informiert	Lars Gustafsson	Roy Schneider	Thomas Reul
Erstellen und Verwalten des Testdokuments	I	R	I
White-Tests der vorhandenen Funktionen.	-	R	R
Ausführen der Gray-/Black-Tests	-	I	R
Überprüfung Vollständigkeit der Prüfdokumentation.	A	R	C

Während der Softwareentwickler genau weiß, wie die Funktionen funktionieren sollen (White-Test), und sicherstellt, dass sie primär tatsächlich auch das tun, was sie tun sollen, wird der Servicetechniker das entsprechende Kapitel im Software-Referenzhandbuch zur Hand nehmen und die Funktionen gemäß der Dokumentation testen. Dabei führt er einen Black-Test oder zumindest einen Gray-Test durch, wenn man berücksichtigt, dass es sich bei der neuen Funktion möglicherweise um eine Erweiterung einer vorhandenen Funktion handelt und Wissen und Erfahrung aus vorherigen Tests dabei unvermeidlich und gegebenenfalls sogar erforderlich sind.

28 Checksumme und Software Version

Die Aufzugssteuerung bietet eine einfache Möglichkeit, zu überprüfen, welche Programmversion in der Aufzugssteuerung ausgeführt wird, sowie die Prüfsumme der aktuell ausgeführten Anwendung.

Sie finden diese Seite hier:

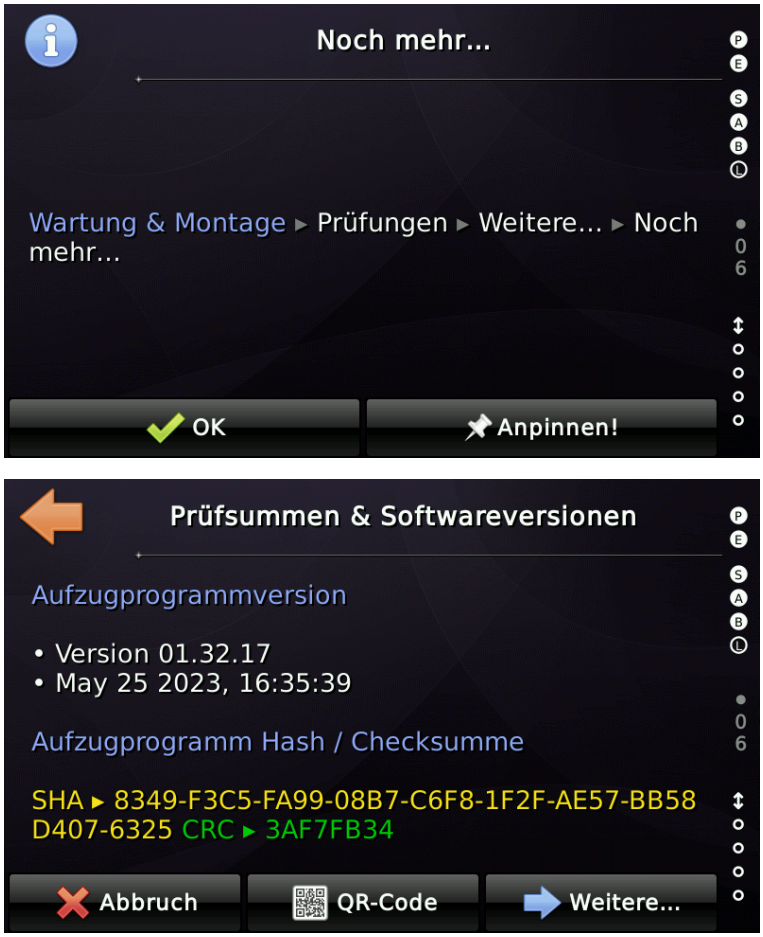
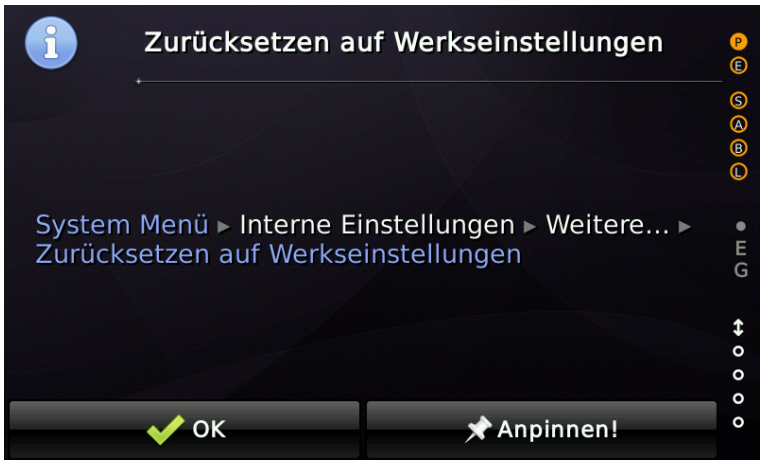


Abbildung 9: Prüfsummen & Softwareversion

Beim Start der Aufzugsanwendung wird die Integrität und die Checksumme der Applikation nachgerechnet und nur bei Übereinstimmung gestartet. Änderungen der Applikation durch Hardwareversagen wird so verhindert. Das verwendete Dateisystem verwendet seinerseits ebenfalls Prüfsummen, um defekte Sektoren zu erkennen und keine Daten weiterzureichen, die nicht valide sind.

29 Außerbetriebnahme

Wird das Gerät, zum Beispiel im Zuge einer Modernisierung, ausgebaut, so sollte es vor der Entsorgung auf Werkseinstellungen zurückgesetzt werden. Dies stellt sicher, dass Aufzugnummer und Steuerungsnummer aus dem Gerät entfernt werden. Personenbezogene Daten werden auf dem Gerät generell nicht gespeichert und bedürfen deshalb auch keiner Löschung. Das Gerät sollte dann sachgerecht und den lokalen Richtlinien folgend entsorgt werden.



Wurde das Gerät zur Betriebszeit mit einer Micro-SD-Speicherkarte ausgestatet, die der Datensicherung oder Sprachansage dient, ist diese nach Außerbetriebnahme sicher zu Löschen (vollständig Formatieren) und dann gemäß der Elektronikschrottverordnung zu entsorgen. In der DIN-Norm 66399 ist die sichere Vernichtung von „*Datenträgern aus der Büro- und Datentechnik*“ gesetzlich geregelt. Diese sollte dann Anwendung finden.

30 Programmierregeln

Die Verwendung gemeinsamer Programmierregeln innerhalb des Entwicklungsteams stellt sicher, dass der Code von Kollegen gegengelesen und überprüft werden kann, wodurch es wahrscheinlicher wird, dass Probleme, Bugs und nicht abgefangene Fehlerbedingungen gefunden werden können. Diese Regeln sind immer wieder Gegenstand von Diskussionen und werden Schritt für Schritt verbessert.

30.1 Auszug

Die folgenden Programmierregeln sollen eine Ressource zum Schreiben von gutem, zuverlässigem und lesbarem Code sein, die es dem Team und den Programmierern des Projekts erleichtern, den vorhandenen Code zu überprüfen, zu erweitern, zu verstehen und daraus zu lernen. Diese Regel soll auch sicherstellen, dass der Code robust ist und weniger fehleranfällig.

Gut lesbarer und leicht verständlicher Code ist eine gute Voraussetzung um Cyber Security Angriffslücken von vorn herein zu vermeiden.

30.2 Grundlegende und allgemeine Richtlinien

- Greifen Sie niemals direkt auf Datenstrukturen zu, die über Threads geteilt werden, ohne den richtigen gegenseitigen Ausschluss (gesperrt durch ein Semaphore, Mutex). Denken Sie daran, dass andere Threads möglicherweise im selben Moment auf dieselben Strukturen zugreifen.

- Bevorzugen Sie Ereignisbenachrichtigungen gegenüber Abfragemethoden.

- Geben Sie einen sinnvolle Rückgabewert, wenn eine Ressource nicht verfügbar ist, wenn die Anwendung sie benötigt.

- Binden Sie niemals System- oder Anwendungsressourcen, es sei denn, es ist absolut notwendig.

- Halten Sie sich immer an die einfachen Strukturkonventionen!

- Alle reservierten oder derzeit nicht verwendeten Felder/Elemente sollten initialisiert werden.

- Verwenden Sie keine vorzeichenbehafteten Variablen oder vorzeichenbehaftete Mathematik für Speicheradressen.

- Vermeiden Sie nach Möglichkeit eine tiefe Verschachtelung von Code. Dadurch wird es für andere besser lesbar und das bedeutet, dass der Code leichter verstanden und auf Fehler überprüft werden kann.

- Wiederholen Sie den Code nicht noch einmal. Erstellen Sie stattdessen eine Funktion oder Methode. Wenn die Leistung ein entscheidender Faktor ist, machen Sie diese Funktion oder Methode „inline“, um unnötigen Overhead zu vermeiden, indem Sie in diese Unterfunktion oder Methode springen und von dort zurückkehren.

- Vermeiden Sie es, Code in Makros zu packen. Es kann manchmal erforderlich sein, aber verwenden Sie diese Methode mit Vorsicht, da sie die Überprüfung des Codes erschwert.

- Wenn lokale Variablen konstante Werte enthalten, deklarieren Sie sie als „const“, was es unmöglich macht, die Werte versehentlich zu ändern oder diese versehentlich als „temporäre“ Variablen für Berechnungen zu benutzen, die später im Lebenszyklus der Software hinzugefügt wurden.

- Verwenden Sie niemals CPU-Verzögerungsschleifen. Verwenden Sie stattdessen die Timer-Funktionen 'addclock/diffclock' oder die 'Board_Sleep'-Funktion.

- Unter Linux® sollte die Verwendung von 'nanosleep()' den älteren 'msleep()- und usleep()'-Funktionen vorgezogen werden, da diese Funktionen möglicherweise nicht an die MONOTONIC-Uhr gebunden sind und Probleme verursachen, wenn die Systemzeit gestellt wird.

30.3 Regeln und Definitionen

Funktionen/Methoden und Attribute

Der Funktions-/Methodenkommentarblock enthält die „DoxyGen“-üblichen reservierten Schlüsselwörter.

Methode ohne Parameter (Definition)

```
/**
 * Gibt zurück, ob sich der Aufzug derzeit im Prioritätsruf-Betriebsmodus
 * befindet.
 *
 * @return   TRUE/FALSE
 */

int CliftPilot::Is_Prio_Call_Operation(void) const
{
    return(m_priority_call_state ? TRUE : FALSE);
}
```

Methode ohne Parameter (Deklaration)

```
public:

    /* Gibt zurück, ob sich der Aufzug derzeit im Prioritätsruf-Betriebsmodus
     * befindet. */

    int Is_Prio_Call_Operation(void) const;
```

- Eine Funktion ohne Parameter wird in der Deklaration/Definition explizit als „void“ definiert.

- Die Art und Weise, wie die Klammern gesetzt werden, wird nicht streng definiert, aber zur besseren Lesbarkeit sollte ein Projekt nur ein Klammerschema verwenden.

- Verwenden Sie immer Klammern um Return-Anweisungen und in „if“-Klauseln.

```
if ((a == 3) && (b == 5))
{
    ...
}
return(a + b);
```

- Begrenzen Sie den Variablenbereich innerhalb von Funktionen und Methoden. Wenn möglich, machen Sie Variablen konstant, um sicherzustellen, dass sie nicht durch Code geändert werden, der später von jemand anderem hinzugefügt wird.

- Aufzählungs-/Typdefinitionen haben das Wort „Enum“ oder „Typ“ am Ende ihres Namens.

- Attribute haben ein „m_“ am Anfang ihres Namens.

- Globale Variablen werden selten verwendet und haben ein „g_“ am Anfang ihres Namens.

- Alle Klassennamen beginnen mit einem großen „C“ oder „Q“ für Qt®-Klassen.

- Funktionen, die eine „Getter“-Funktionalität bereitstellen, sollten mit „Get“, „Is“ oder „Are“ beginnen.

- Funktionen, die eine „Setter“-Funktionalität bereitstellen, sollten mit „Set“ beginnen.

- Handler (zyklisch/ereignisgesteuert) sollten mit „Handle“ oder „Do“ beginnen.

Klassen ableiten

- Bei der Ableitung von Klassen sollte Polymorphismus vermieden werden.
- Der abgeleitete Klassenname kann den Namen seiner Oberklasse enthalten.
- Wenn die Superklasse die Basisklasse selbst ist, sollte das Wort „Base“ aus dem Namen der abgeleiteten Klasse entfernt werden.

Klassen ableiten (Deklaration)

```
/**
 * Türklasse für eine typische automatische Kabinen-/Schachttürkombination.
 */

class CLiftDoorCntrlAuto : public CLiftDoorCntrlBase
{

};
```

Klassen ableiten (Deklaration)

```
/**
 * Türklasse für eine Kombination aus automatischer Fahrkorbtür und manueller
 * (Dreh-)Schachttür.
 */

class CLiftDoorCntrlAutoSwing : public CLiftDoorCntrlAuto
{

};
```

Sprünge

- Absolute Sprünge wie "`goto xyz`" sind auf Anwendungsebene verboten.

- Die Verwendung von "`continue;`" sollte mit Vorsicht verwendet werden. Ein häufiger Fehler besteht darin, einen Zeiger am Ende einer Schleife zu inkrementieren und diese inkrementelle Anweisung versehentlich zu umgehen, weil eine Verzweigung ein Continue in einem switch/case-Konstrukt verwendet.

Typdefinitionen Aufzählungen/Bitfeldern

Deklaration von Aufzählungen

```
/**
 * Lift-Boy Zustände
 */

typedef enum LiftBoyOperationStateType
{
    LIFT_BOY_OPERATION_STATE_OFF = 0,       // aus
    LIFT_BOY_OPERATION_STATE_ON = 1,        // ein
    LIFT_BOY_OPERATION_STATE_START = 2,     // ein, warten auf START
    LIFT_BOY_OPERATION_STATE_RUN = 3,       // Fahren zum nächsten Halt
    LIFT_BOY_OPERATION_STATE_ERROR = 4      // Fehler

} LiftBoyOperationStateType;
```

Deklaration von Bitfeldern

```
/**
 * Struktur zum Speichern anstehender Rufquittungen.
 */

typedef struct CallAckCancellationType
{
    uint8_t floor;                 // Etage des Rufes

    struct
    {
        uint8_t call_type : 4;    // Ruftyp
        uint8_t door_mask : 4;    // Türmaske

    } info;

    uint8_t count;                // Zeitgesteuerter Zähler

} CallAckCancellationType;
```

Die Organisation von Bitfeldern hängt von der Big/Little-Endian Architektur ab.

Switch/Case/Default Konstrukte

Ein Switch/Case Konstrukt sollte immer einen Standardpfad haben, auch wenn er leer ist oder nur eine Debug-Meldung enthält, wie folgt:

Switch/Case/Default

```
    /* Filter Türendschalter */

    switch (sigid)
    {
    case APP_LIFT_DOOR_INPUT_CLOSED:

        /* Signale für etagen-/türselektive Türendschalter weiterleiten. */

        if ((floor == APP_LIFT_FLOOR_ALL) || (!floor))
        {
              <snip>
        }
        break;

  <snip>

    default:
        GURU0(("Door %d: Unknown signal %d passed. ", m_door_id, sigid));
        break;
    }
```

Sogenannte „Fall Thru“-Konstrukte innerhalb eines Switch-Case sind unerwünscht, da sie leicht zu Fehlern und Irrtümern führen können.

Längere if/else Konstrukte

Um lange if/else-Konstrukte besser lesbar zu machen, ist es eine gute Idee, auch die leeren else-Pfade hinzuzufügen. Dies zeigt auch an, dass der 'else' Pfad nicht vergessen wurde.

If/else

```
    /* Door lock rule on Safety light curtains. */

    if (data->idoor_count)
    {

        /* Some (very) long code... */

    }
    else
    {
        GURU0(("CUnitBase: Should never be executed."));
    }

<snip>
```

Quellcode und Headerdateien

Um das Verständnis für den Zweck von Klassen und Dateien klar und einfach zu machen, muss jede Quell- und Headerdatei einen anfänglichen Kommentarblock enthalten, der grundlegende Informationen zu Funktion, Zweck, Sprache, Toolchain, Originaldateiname, Projektname und Autor(en) enthält. Das zusätzliche Datum ist eigentlich überflüssig, da das Repositorysystem (GIT) dieses mitverfolgt. Dennoch hat es sich in der Praxis als sinnvoll erwiesen, das Datum auch manuell zu aktualisieren.

Source/Header initial comment block

```
/**
 *  Copyright © 2016 Thor Engineering GmbH
 *
 *  liftpilot.cpp
 *
 *  Implementation of the basic states, the lift passes through while
 *  processing calls - or better destinations, which are defined by
 *  one or more calls. Each destination is a 3-tuple of a floor, a
 *  door-mask (containing one or more doors attached) and a call type.
 *  By reaching a destination one or more of these 3-tuples will be
 *  canceled. The main goal of the "LiftPilot" class is to finish
 *  all destinations in the shortest time possible.
 *
 *  Project:            LiftApp for the NeXt project
 *
 *  Programmer:         Roy Schneider
 *  Last Change:        18.05.2016
 *
 *  Language:           C/C++
 *  Toolchain:          GCC/GNU-Make
 */

#include "../main.h"

#include "liftdata.h"
#include "liftparam.h"
#include "liftapp.h"
#include "liftpilot.h"

#include "../../logfile/logfile.h"

<snip>
```

Klassische C-String Operationen

- Wenn möglich und sinnvoll, verwenden Sie eine String-Klasse anstelle klassischer String-Operationen.
- Die Verwendung veralteter String-Funktionen wie strcpy, strcat, strlen oder sprintf sollte vermieden werden. Stattdessen sollten die Varianten mit Ziellängenangabe wie snprintf, strncpy, strncat oder strnlen verwendet werden.
- Überprüfen Sie gründlich, ob die angegebene maximale Zeichenanzahl mit der realen (Rest-)Zielpuffergröße übereinstimmt.
- Verwenden Sie beim Bestimmen der String-Puffergrößen immer das _countof()-Makro anstelle des sizeof()-Makros, um sicherzustellen, dass die Puffergröße korrekt berechnet wird, auch wenn es sich um wchar_t (Multi-Byte-Chars) anstelle von einfachen Chars handelt.
- Stellen Sie sicher, dass die String-Puffer immer nullterminiert sind. Beachten Sie bei der Verwendung von strncpy, dass der Zielpuffer nicht mit einer Null beendet wird, wenn die maximale Zeichenanzahl erreicht ist. Dies ist ein anderes Verhalten als beispielsweise snprintf.
- Überprüfen Sie beim Anhängen von Zeichenfolgen an Zeichenfolgen den verbleibenden Pufferplatz.

String Operationen

```
char sztemp[32];

/* Stellen Sie immer sicher, dass die Zielzeichenfolge nullterminiert ist
 * und dass der Zielpuffer nicht überläuft. Beachten Sie, dass strncpy die
 * Zeichenfolge nicht mit einer Null abschließt, wenn maxlen erreicht wurde! */

if (pstr)
{
    strncpy(sztemp, pstr, _countof(sztemp) - 1);
    sztemp[_countof(sztemp) - 1] = 0;
}

/* Stellen Sie immer sicher, dass die Zielzeichenfolge nullterminiert ist
 * und dass der Zielpuffer nicht überläuft. */

static const char s_fmtin[] = "Debug: %d";

snprintf(sztemp, _countof(sztemp), s_fmtin, ival);

/* Überprüfen Sie beim Anhängen von Zeichenfolgen den verbleibenden Puffer. */

size_t slen = strnlen(sztemp, _countof(sztemp) - 1);

strncat(sztemp, "TEST", _countof(sztemp) - 1 - slen);
sztemp[_countof(sztemp) - 1] = 0;
```

31 Code Analyse Werkzeuge

Um uns die Arbeit beim Auffinden potenzieller und tatsächlicher Fehler und Probleme verschiedener Art zu erleichtern, nutzen wir auf dem Markt erhältliche statische Analysetools, die dabei helfen, den Code während der Entwicklung zu analysieren und schwerwiegende Fehler frühzeitig in der Entwicklungsphase zu erkennen.

Solche Mängel können beseitigt werden, bevor der Code tatsächlich funktionsfähig ist. Ein später festgestellter Mangel ist immer teuer zu beheben.

Wir verwenden aktiv...

- **CDT-Codeanalyse**, die im Hintergrund läuft, während der Entwickler tippt und schreibt. Dieses Tool erkennt beispielsweise nicht initialisierte Variablen direkt beim Eingeben des Codes.

- **GNU-Code-Diagnose**, die eine ganze Reihe von Problemen direkt beim Kompilieren erkennt, wie z. B. Nichtübereinstimmungen zwischen Datentypen von Variablen und den Formatspezifizierern in C-Format-Strings.

- Das **CPPCheck Static Source Code Analysis** Tool ist ein Open-Source-Analysetool für C- und C++-Code. Es bietet eine einzigartige Codeanalyse zur Erkennung von Fehlern und konzentriert sich auf die Erkennung von undefiniertem Verhalten und gefährlichen Codierungskonstrukten. Ziel ist es, nur echte Fehler im Code zu erkennen und möglichst wenige Fehlwarnungen zu generieren.

- **DoxyGen** ist eigentlich kein Code-Diagnosetool, entdeckt aber bei der Überprüfung der Dokumentations-Tags auch Nichtübereinstimmungen, wie zum Beispiel Abweichungen bei der Variablenbenennung in Deklarationen und Definitionen. Außerdem werden zusätzlich Warnungen vor unklarer Methodenüberladung ausgegeben.

32 Riskante Compileroptimierungen

Ein sehr gutes Beispiel für riskante Compileroptimierungen, die Sicherheitsprobleme verursachen können, ist CVE-2009-1897, das von RedHat dokumentiert wurde. Hier hätte der Compiler eine Nullzeigerprüfung entfernt, da die Variable versehentlich im zuvor dereferenzierten Code enthalten war, sodass die Logik des Compilers die Null-Prüfung entfernen würde. Das Codeanalysetool CPPCheck durchsucht den Code tatsächlich nach Problemen wie diesen:

Risky Compiler Optimizations

```
/* Hier hätte der Compiler eine Nullzeigerprüfung entfernt, da die Variable
 * im zuvor dereferenzierten Code versehentlich vorhanden war. */

mystruct_t * pstr = Get_Pointer();
const int32_t ival = pstr->value;

if (!pstr)
{
    GURU0((“GURU MEDITATION 00000003”))
}
```

Die LiftApp und der Linux-Kernel verwenden **-fno-delete-null-pointer-checks** von GCC, um solche Optimierungen zu deaktivieren.

Anwendungen müssen häufig vertrauliche Daten, wie Passworteingaben oder Schlüsseldaten lesen. Der für diese vertraulichen Daten verwendete Speicher (Stack oder Heap) muss ordnungsgemäß bereinigt werden, indem sein Inhalt überschrieben wird, bevor die Funktion zum Aufrufer zurückkehrt. Compiler Optimierungen entfernen Variablenschreibzugriffe, wenn diese Variablen später in der Funktion nicht mehr verwendet werden. Dies führt dazu, dass vertrauliche Informationen nach ihrer Verwendung im Speicher verbleiben können. Dies kann verhindert werden, indem eine Funktion zum Löschen des Speichers aufgerufen wird, so dass aus Sicht des Compilers eine Verwendung der Variablen stattfindet. Zusätzliche kann '*volatile*' verwendet werden, um eine solche Compiler-Optimierung zu verhindern.

Risky Compiler Optimizations

```
/* Um sicherzustellen, dass Code, der zum Löschen vertraulicher Daten aus dem
 * Speicher verwendet wird, nicht optimiert wird, stellen wir sicher, dass Sie
 * eine dedizierte Methode zum Löschen aufrufen. */

volatile char szpwddata[64];

Get_Pwd_Data(&szpwddata, sizeof(szpwddata));

… … Verwendung der Daten… … und bereinigen indem eine Funktion aufrufen wird.

Scrub_Sensitive_Data(&szpwddata, sizeof(szpwddata));
```

33 SHA-Implementierung

Die folgenden Codes spiegeln die Implementierung des sicheren Hash-Algorithmus wider, wie er in der Aufzugsteuerungsanwendung von uns verwendet wird.

```
/**
 * Copyright (c) 2017-19 Thor Engineering GmbH
 *
 * sha.cpp       The "Secure Hash Algorithm" SHA1 implementation.
 *
 * Project:      LiftApp for the NeXt project
 *
 * Programmer:   Roy Schneider
 * Last Change:  19.08.2019
 *
 * Language:     C/C++
 * Toolchain:    GCC/GNU-Make
 *
 * NOTE:
 * This implementation in C++ was inspired by the published work of
 * John Halleck (University of Utah).
 */

#include "../main.h"
#include "../base/base_types.h"
#include "bitutils.h"
#include "sha1.h"

/**
 * Constructor
 */

CSHA1Provider::CSHA1Provider()
{
    memset(&m_context, 0, sizeof(m_context));
}

/**
 * Destructor
 */

CSHA1Provider::~CSHA1Provider()
{
}

/**
 * Initialize the SHA provider instance.
 *
 * @return OK/ERROR
 */

int CSHA1Provider::Init (void)
{
    /* Init */

    memset(&m_context, 0, sizeof(m_context));
```

```
    register unsigned long *_pd = m_context.cprocess;

    *_pd++ = 0x67452301;
    *_pd++ = 0xEFCDAB89;
    *_pd++ = 0x98BADCFE;
    *_pd++ = 0x10325476;
    *_pd   = 0xC3D2E1F0;

    /* Return */

    return(OK);
}

/**
 * Execute the SHA rounds and transform the data.
 */

inline void CSHA1Provider::Transform (void)
{
    int ival;
    CryptSHA1ContextType *pc;
    unsigned long dwval, *pdw;
    unsigned long dwA, dwB, dwC, dwD, dwE;
    unsigned long dw[128];

    /* Init */

    pc = &m_context;

    /* Check */

    /* Init */

    register unsigned long *_ps = pc->cprocess;

    dwA = *_ps++;
    dwB = *_ps++;
    dwC = *_ps++;
    dwD = *_ps++;
    dwE = *_ps;

    ival = APP_SHA_1_BLOCKWORDSIZE;
    pdw  = dw;

    register unsigned long *_pd = pc->ldata;

    while(likely(ival--))
    {
        *pdw++ = *_pd;
        *_pd++ = 0;
    }

    ival = 16;

    while(likely(ival < 80))
    {
        _pd = dw + ival;

        *_pd = *(_pd - 3) ^ *(_pd - 8 ) ^ *(_pd - 14) ^ *(_pd - 16);
```

```
        *_pd = ROTINT32(1, *_pd);

        ival++;
    }

    ival = 0;
    _pd  = dw;

    while(likely(ival < 20))
    {
        dwval = (*_pd++) + ROTINT32(5, dwA) + dwE + 0x5A827999L + \
                ((dwB & dwC) | (~dwB & dwD));

        dwE = dwD;
        dwD = dwC;
        dwC = ROTINT32(30, dwB);
        dwB = dwA;
        dwA = dwval;

        ival++;
    }

    while(likely(ival < 40))
    {
        dwval = (*_pd++) + ROTINT32(5, dwA) + dwE + 0x6ED9EBA1L + \
                (dwB ^ dwC ^ dwD);

        dwE = dwD;
        dwD = dwC;
        dwC = ROTINT32(30, dwB);
        dwB = dwA;
        dwA = dwval;

        ival++;
    }

    while(likely(ival < 60))
    {
        dwval = (*_pd++) + ROTINT32(5, dwA) + dwE + \
                 0x8F1BBCDCL + ((dwB & dwC) | (dwB & dwD) | (dwC & dwD));

        dwE = dwD;
        dwD = dwC;
        dwC = ROTINT32(30, dwB);
        dwB = dwA;
        dwA = dwval;

        ival++;
    }

    while(likely(ival < 80))
    {
        dwval = (*_pd++) + ROTINT32(5, dwA) + dwE + 0xCA62C1D6L + \
                (dwB ^ dwC ^ dwD);

        dwE = dwD;
        dwD = dwC;
        dwC = ROTINT32(30, dwB);
        dwB = dwA;
```

```
        dwA = dwval;

        ival++;
    }

    _pd = pc->cprocess;
    register unsigned long lmsk = 0xFFFFFFFF;

    *_pd   += dwA;
    *_pd++ &= lmsk;
    *_pd   += dwB;
    *_pd++ &= lmsk;
    *_pd   += dwC;
    *_pd++ &= lmsk;
    *_pd   += dwD;
    *_pd++ &= lmsk;
    *_pd   += dwE;
    *_pd   &= lmsk;

    pc->iword = 0;
    pc->ibyte = 0;
}

/**
 * Update the SHA context with the given string.
 *
 * @param pbuf  Pointer to the data buffer used to update the hash.
 * @param icnt  Length (or count of) bytes in the buffer given by 'pbuf'.
 *
 * @return OK/ERROR
 */

int CSHA1Provider::Update(const unsigned char *pbuf, int icnt)
{
    int ierr;
    int iword;
    CryptSHA1ContextType *pc;
    unsigned long dwval, dwmask;

    /* Init */

    pc = &m_context;

    /* Check */

    if (likely((pc) && (pbuf) && (icnt > 0)))
    {
        /* Init */

        dwmask = 0x1FFFFFFF; // 29 bit mask

        pc->lcount_hi  += icnt >> 29;
        pc->lcount_low += icnt & dwmask;
        pc->lcount_hi  += pc->lcount_low >> 29;
        pc->lcount_low &= dwmask;

        iword = pc->iword;
        dwval = pc->ldata[iword];
```

```
        while(likely(icnt--))
        {
            dwval = (*pbuf++) | (dwval << 8);

            pc->ibyte++;

            if (unlikely(pc->ibyte >= 4 /*32 bit*/))
            {
                pc->ldata[iword++] = dwval;

                dwval = 0;

                if (unlikely(iword >= APP_SHA_1_BLOCKWORDSIZE))
                {
                    Transform();

                    iword = 0;
                }

                pc->ibyte = 0;
            }
        }

        pc->iword = iword;
        pc->ldata[iword] = dwval;

        ierr = OK;
    }
    else
    {
        ierr = ERROR;
    }

    /* Return */

    return(ierr);
}

/**
 * Pad (fill up) the SHA buffer.
 */

void CSHA1Provider::Pad(void)
{
    CryptSHA1ContextType *pc;
    int ival;

    /* Init */

    pc = &m_context;

    /* Init */

    unsigned long *_pd = pc->ldata + pc->iword;

    *_pd <<= 8;
    *_pd  |= BIT7;

    switch(pc->ibyte)
```

```
    {
    case 2:
        *_pd <<= 8;
        break;
    case 1:
        *_pd <<= 16;
        break;
    case 0:
        *_pd <<= 24;
        break;
    default:
        break;
    }

    ival = pc->iword + 1;
    _pd  = pc->ldata + ival;

    while(likely(ival < APP_SHA_1_BLOCKWORDSIZE))
    {
        *_pd++ = 0;

        ival++;
    }

    pc->iword = 0;
    pc->ibyte = 0;
}

/**
 * Return the SHA bytes in the given buffer.
 *
 * @param phash  Hash result buffer. See CryptSHA1Digest8Type for details.
 *
 * @return OK/ERROR
 */

int CSHA1Provider::GetBytes(CryptSHA1Digest8Type phash)
{
    CryptSHA1ContextType *pc;
    int ival;
    int ierr;
    unsigned long cval;

    /* Init */

    pc = &m_context;

    /* Check */

    if (likely((pc) && (phash)))
    {
        /* Init */

        ival = 0;

        unsigned char *ph = phash;

        while(likely(ival < APP_SHA_1_DIGESTWORDSIZE))
        {
```

```
            cval = pc->cprocess[ival];

            *ph++ = (unsigned char) LOINT8((cval >> 24));
            *ph++ = (unsigned char) LOINT8((cval >> 16));
            *ph++ = (unsigned char) LOINT8((cval >>  8));
            *ph++ = (unsigned char) LOINT8((cval));

            ival++;
        }

        ierr = OK;
    }
    else
    {
        ierr = ERROR;
    }

    /* Return */

    return(ierr);
}

/**
 * Finalize the SHA hash.
 *
 * @param phash  Hash result buffer. See CryptSHA1Digest8Type for details.
 *
 * @return OK/ERROR
 */

int CSHA1Provider::Final(CryptSHA1Digest8Type phash)
{
    CryptSHA1ContextType *pc;
    int ival;
    int n;
    int ierr;

    /* Init */

    pc = &m_context;

    /* Check */

    if (likely((pc) && (phash)))
    {
        /* Init */

        ival = (pc->iword << 2) + pc->ibyte + 1;

        Pad();

        if (unlikely(ival > APP_SHA_1_PADDING_REMAINDER))
        {
            Transform();

            unsigned long * _pl = pc->ldata;

            n = (APP_SHA_1_BLOCKWORDSIZE - APP_SHA_1_PADDING_WORDS);
```

```
            while(likely(n--))
            {
                *_pl++ = 0;
            }

            pc->iword = APP_SHA_1_BLOCKWORDSIZE;
            pc->ibyte = 0;
        }

        unsigned long * _pl = &pc->ldata[14];

        *_pl++ = pc->lcount_hi;
        *_pl   = pc->lcount_low << 3;

        Transform();

        ierr = GetBytes(phash);
    }
    else
    {
        ierr = ERROR;
    }

    return(ierr);
}

/**
 * Create a SHA hash from the given string.
 *
 * @param phash    Hash result buffer. See CryptSHA1Digest8Type for details.
 * @param pbuf     Pointer to the buffer containing the string to hash.
 * @param icnt     Count of characters in the buffer containing the string
 *                 to hash.
 *
 * @return OK/ERROR
 */

int CSHA1Provider::HashIt(CryptSHA1Digest8Type *phash, \
                          const unsigned char *pbuf, int icnt)
{
    int ierr;

    /* Init */

    memset(&m_context, 0, sizeof(m_context));
    memset(phash, 0, sizeof(CryptSHA1Digest8Type));

    /* Check */

    ierr = ERROR;

    if (likely((pbuf) && (phash) && (icnt > 0)))
    {
        if (likely(Init() == OK))
        {
            if (likely(Update(pbuf, icnt) == OK))
            {
                ierr = Final(*phash);
            }
```

```
        }
    }

    return(ierr);
}

/**
 * Verify (compare) two SHA1 hashs.
 *
 * @param pplain1 [in] First plain SHA hash.
 * @param pplain2 [in] Second plain SHA hash used to verify (compare) the first.
 *
 * @return OK/ERROR
 */

int CSHA1Provider::VerifyIt(CryptSHA1Digest8Type pplain1, \
                            CryptSHA1Digest8Type pplain2)
{
    int ierr;

    /* Init */

    ierr = ERROR;

    if (likely((pplain1) && (pplain2)))
    {
        if (likely(!memcmp(pplain1, pplain2, sizeof(CryptSHA1Digest8Type))))
        {
            ierr = OK;
        }
    }

    return(ierr);
}

/**
 * Calculate SHA1 of the given file stream and finally return it to the caller.
 *
 * <Beware that this code is optimized for 32/64 bit memory
 *  block alignment for little endian processors.>
 *
 * @param pfile  Pointer to the file, returned by fopen().
 * @param phash  Hash result buffer. See CryptSHA1Digest8Type for details.
 *
 * @return OK/ERROR
 */

int CSHA1Provider::CalculateSHAFile(FILE *pfile, CryptSHA1Digest8Type *phash)
{
    int ierr;
    size_t ilen;
    unsigned char *ptmp;

    /* Init */

    ierr = ERROR;

    memset(&m_context, 0, sizeof(m_context));
```

```
    if (likely(phash))
    {
        memset(phash, 0, sizeof(CryptSHA1Digest8Type));

        /* Check */

        if (likely(pfile))
        {
            /* Alloc */

            ilen = 16 * 1024;
            ptmp = new unsigned char[ilen];

            if (likely(ptmp))
            {
                if (likely(Init() == OK))
                {
                    if (likely(!fseek(pfile, 0, SEEK_SET)))
                    {
                        unsigned long ulread = 0;

                        do
                        {
                            ulread = fread(ptmp, 1 /*byte*/, ilen, pfile);

                            if (likely(ulread))
                            {
                                ierr = Update(ptmp, (int) ulread);

                                if (unlikely(ierr != OK))
                                {
                                    ierr = ERROR;
                                    break;
                                }
                            }
                        }
                        while(likely(ulread));

                        /* Finalize */

                        if (likely(ierr != ERROR))
                        {
                            ierr = Final(*phash);
                        }
                    }
                }

                /* Free memory */

                delete(ptmp);

            } // if (likely(ptmp))
        }
    }

    /* Return */

    return(ierr);
}
```

```
/**
 * Calculate the SHA of the given file name and finally return it to the caller.
 *
 * <Beware that this code is optimized for 32/64 bit memory
 *  block alignment for little endian processors.>
 *
 * @param pfilename  Pointer to the file, returned by fopen().
 * @param phash      Hash result buffer. See CryptSHA1Digest8Type for details.
 *
 * @return OK/ERROR
 */

int CSHA1Provider::CalculateSHAFile(const char *pfilename, \
                                    CryptSHA1Digest8Type *phash)
{
    int ierr;

    /* Init */

    ierr = ERROR;

    /* The file exists, open it as a binary. */

    if (likely(pfilename))
    {
        if (likely(*pfilename))
        {
            FILE *pfile = fopen (pfilename, "rb");

            if (likely(pfile))
            {
                ierr = CalculateSHAFile(pfile, phash);

                fclose(pfile);
            }
        }
    }

    return(ierr);
}

/* sha1.cpp */
```

Abbildungen

Abbildung 1: Menüpunkt, der eine Passwortberechtigung verlangt.....13
Abbildung 2: Parameter Änderungsprotokoll unter System Menü → Sicherheit.....14
Abbildung 3: Jira Workflow - KAN Board (Kanban).....31
Abbildung 4: Jira Workflow - Gantt Diagramm.....31
Abbildung 5: Flußdiagramm Meldung von Vorfällen.....34
Abbildung 6: Versionskontrollsystem.....37
Abbildung 7: Flußdiagramm Software Backup/Update.....47
Abbildung 8: MQTT Verbindungsstatus.....57
Abbildung 9: Prüfsummen & Softwareversion.....63