



6 | 2026

Juni
www.eurailpress.de/sd
118. Jahrgang

SIGNAL+DRAHT

SIGNALLING & DATACOMMUNICATION



06 **Paperless Onsite Activities:
Digitalisierung von
Baustellentätigkeiten**
Paperless Onsite Activities:
the digitalisation of
onsite activities

37 **Test und Nachweis
bei GoA 3+: KI-basierte
Perzeptionssysteme**
Testing and validating
in GoA 3+: AI-based
perception systems

45 **Automatisierte Abnahme-
prüfung – Ergebnisse des
Anwendungstests**
Application test result
of the automated
acceptance test

DeployME – Patching unter rollendem Rad

DeployME – patching with the wheels still turning

Viktor Locher

Heutige vornehmlich auf Software basierende Automatisierungslösungen für den Bahnbetrieb erfordern regelmäßig das Einbringen von aktualisierter Software (Patches) auf allen Rechnern in einem verteilten System. Bisher sind für diesen Vorgang die Rechner außer Betrieb zu nehmen, mit den entsprechenden Einschränkungen für den Bahnbetrieb. Die TMS-Lösung Controlguide Itlis N von Siemens verfügt über „Deployment Management Environment“ (DeployME), ein System, welches unterbrechungsfreies Patchen möglich macht.

Itlis N mit seiner hochverfügbaren Systemarchitektur unterstützt das Patchen von landesweiten Betriebsleitsystemen ohne Stehzeiten – oder bildlich gesprochen: „unter rollendem Rad“. Damit wird ein zeitnahes Patchen (im Wesentlichen das Aufspielen von Security-Fixes für das Betriebssystem) in einer safety-relevanten Umgebung ermöglicht, was in Zeiten zunehmender Sicherheitsbedrohungen unerlässlich ist.

DeployME ist für die automatisierte Verteilung und Installation der Patches zuständig. Es basiert auf einer weit verbreiteten Automatisierungslösung für Data Center (Ansible AWX) und erfüllt hohe Anforderungen an eine safety-relevante Umgebung. Es wurde entwickelt, um Security-Patches automatisiert auf das Betriebssystem von „Itlis N“ aufzuspielen.

DeployME

- basiert auf einem modernen Cloud-Technologie-Stack mit Kubernetes als Container-Deployment-Plattform,
- kann sowohl in der Cloud als auch „On-Premises“ betrieben werden,
- unterstützt die automatische Air-Gapped-Installation für „On-Premises“-Einrichtungen,
- bietet Benutzerführung durch ein grafisches „Patch-Cockpit“,
- braucht keine separaten Software-Agenten auf dem Zielrechner,

um hier nur die wichtigsten Funktionen zu nennen.

Dank seines modularen Aufbaus kann DeployME leicht für andere Anwendungen erweitert werden (Bild 1).

Die Interaktion der Zielsysteme mit Ansible AWX wird über eine REST-Schnittstelle sichergestellt; so kann der Start der Installation eines Updates durch den Betreiber/End-User der Systeme (z. B. Itlis N) ausgelöst werden, was eine verpflichtende Kundenanforderung war.

1 Zusammenspiel mehrerer Akteure

Das „sichere Ausrollen“ von Security-Patches auf Kritischer Infrastruktur ohne Impact auf den Betrieb und ohne Ausfallzeit des Zielsystems basiert auf drei Haupt-Akteuren:

- die Automatisierungslösung DeployME
- QoS („Quality of Service“ stellt sicher, dass kritischer Netzverkehr immer genügend Bandbreite erhält) auf dem betroffenen Netzwerk
- die Redundanz des Systems (resp. der Software), welches gepatcht wird.

Modern automation solutions for railway operations, which are mainly software-based, regularly require software updates (patches) to be applied to all the computers in a distributed system. Until now, this patching process has meant taking the computers concerned out of service with obvious negative consequences for railway operations. The Siemens Controlguide Itlis N TMS solution includes the “deployment management environment” (DeployME), a system that enables patches to be applied without interrupting ongoing operations.

Itlis N and its high-availability system architecture support the patching of national operations control systems “on the fly” with no system downtime required. This makes it possible to apply patches (primarily security fixes for the operating system) promptly in a safety-relevant environment; an essential capability at a time of escalating security threats.

DeployME takes care of automated patch distribution and installation. Developed to automatically install security patches for the Itlis N operating system, it is based on Ansible AWX (an automation solution widely used in data centres) and complies with the stringent requirements for use in safety-relevant settings.

DeployME

- is based on a modern cloud technology stack with Kubernetes as the container deployment platform,
- can be operated both in the cloud and on-premises,
- supports automated air-gapped installation for on-premises equipment,
- includes a graphic Patch Cockpit for user guidance and
- does not require any separate software agents on the target computer,

... to highlight just a few of its most important features.

Thanks to its modular design, DeployME can easily be extended to cover other use cases as well (fig.1).

The interaction between the target systems and Ansible AWX is ensured via a REST interface. This means that the installation of an update can be triggered by the system operator/end user (for example Itlis N), which was an essential customer requirement.

1 The interplay of multiple factors

The “safe rollout” of security patches on critical infrastructure without impacting ongoing operations and with no target system downtime relies on three principal factors:

- the DeployME automation solution
- QoS (quality of service) on the network concerned so as to ensure there is always sufficient bandwidth available for any critical network traffic
- the redundancy of the system (or software) to be patched

DeployME ensures that the rollout and installation of patches follow a defined workflow with no scope for operator error (hence “safe rollout”). QoS on the network concerned prevents any adverse ef-

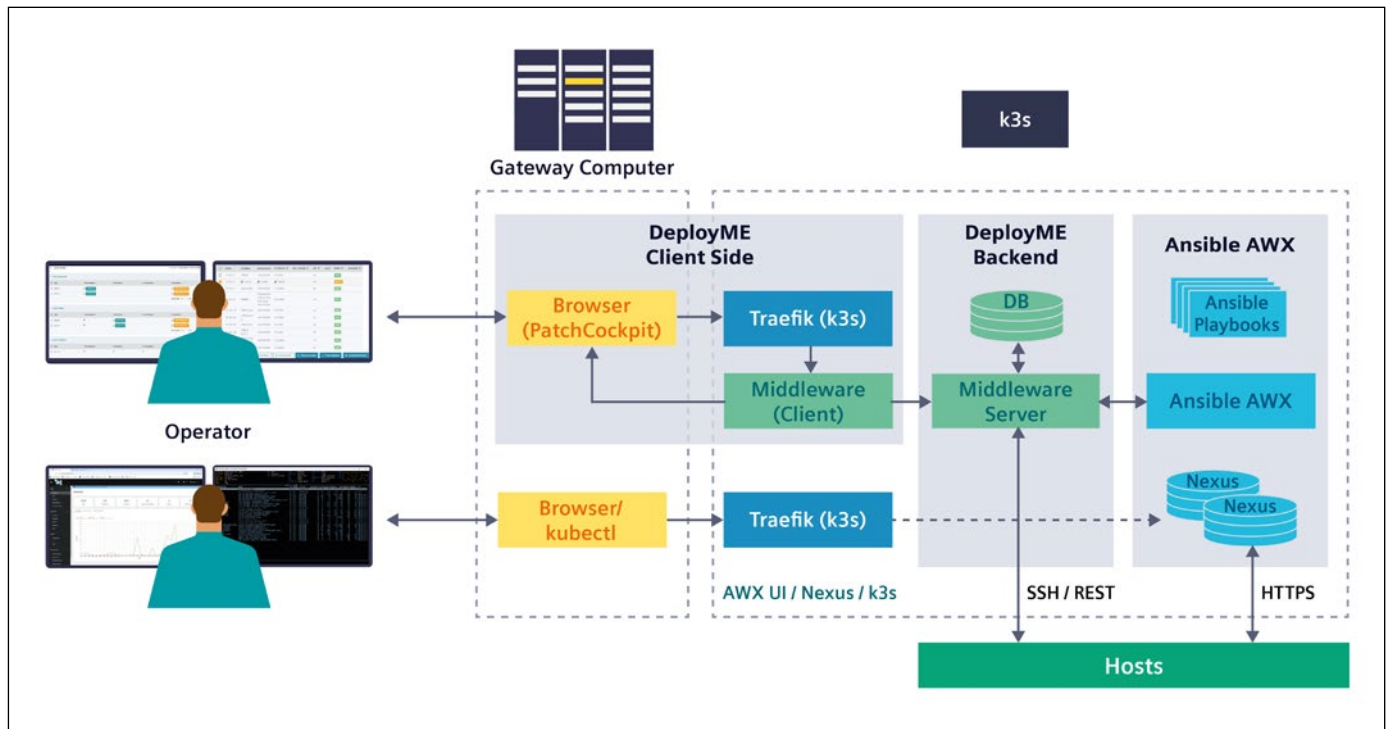


Bild 1: DeployME Prinzip-Aufbau

Fig. 1: The basic outline of DeployME

Quelle aller Bilder / Source of all figures: eigene Darstellung / own illustration

DeployME stellt sicher, dass das Ausrollen und Installieren von Patches einem festgelegten Workflow folgt und Bedienfehler ausgeschlossen werden (deshalb „sicheres Ausrollen“). QoS auf dem betroffenen Netzwerk verhindert negativen Einfluss auf den operativen Datenverkehr während des Deployments, und die redundante Architektur des Zielsystems ermöglicht das Patchen ohne betriebliche Ausfallzeiten. Dazu wird der Teil des Systems, welcher im passiven Betriebszustand ist, abgeschaltet und gepatcht. Nach einer Redundanzumschaltung auf den gerade aktualisierten Teil kann der Patch-Vorgang auf der anderen Systemkomponente vorgenommen werden.

2 Ansible AWX als Automatisierungslösung

Das Open-Source-Produkt Ansible AWX ist die Automatisierungslösung, welche als Basis für das Deployment zum Einsatz kommt. Über sogenannte Playbooks werden die einzelnen Aufgaben – „Deployen“ (Verteilen des Patches), „Releasen“ (Hinweis an die Bedien-Person, dass der Patch jetzt bereitsteht) und „Installieren“ – automatisiert. Ansible AWX kommt ohne Agenten auf den Zielsystemen aus und kommuniziert über SSH (SSH, „Secure Shell“, ist ein Netzwerkprotokoll, das durch Verschlüsselung eine sichere Verbindung zwischen zwei Computern über ein unsicheres Netzwerk ermöglicht). Eine REST-Schnittstelle erlaubt es, die eigentliche Installation eines Patches auf dem eigenen System zu einem Zeitpunkt anzustoßen, welcher gemäß Einschätzung der Bedien-Person des Systems der geeignete ist (aus Safety-Gründen darf die Applikation während der Patch-Installation nicht laufen). Die REST-API („Representational State Transfer Application Programming Interface“) ist eine Programmierschnittstelle, die auf der REST-Architektur basiert und es Systemen ermöglicht, über ein Netzwerk mit einem Server zu kommunizieren und sowohl Daten abzufragen als auch Befehle auf dem Server abzusetzen.

ffects on operations data traffic during deployment, while the redundant architecture of the target system makes it possible to install patches with no operating downtime. First, the part of the system that is in a passive operating state is shut down and patched. A redundancy switchover to this newly updated part is then affected so that the patching process can be completed for the other part.

2 Ansible AWX as an automation solution

The open-source Ansible AWX product is the automation solution used as the basis for deployment. The various tasks – deploy (patch distribution), release (notification of the operator that the patch is now ready) and install – are automated using what are referred to as playbooks. Ansible AWX, which does not need any agents on the target systems, communicates via SSH (Secure Shell), a network protocol that creates a secure connection between two computers over an unsecure network using encryption. A REST interface enables operators to initiate the actual installation of a patch on the system for which they are responsible at a time they consider suitable (for safety reasons, the application must not be running during patch installation). The REST API (Representational State Transfer Application Programming Interface), which is based on the REST architecture, enables systems to communicate with a server over a network and to request data and execute commands on the server.

3 Enforcement of a defined workflow

Users are guided by a Patch Cockpit, which removes the possibility of operator error and enforces the defined workflow agreed with the rail customer. The Patch Cockpit is the browser-based frontend of a middleware component developed as a layer between the frontend and Ansible AWX. The logic behind the Patch Cockpit checks the prerequisites for every step to ensure the workflow is completed as defined (fig. 2).

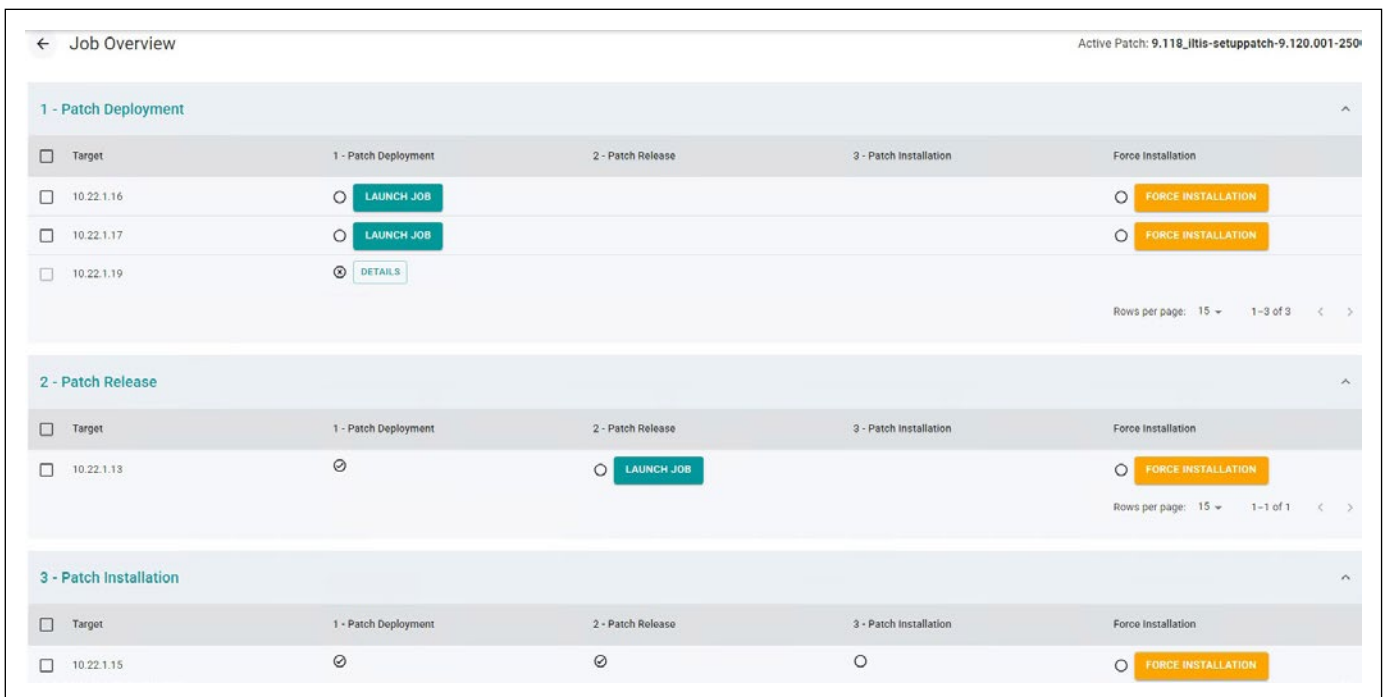


Bild 2: DeployME Patch-Cockpit
Fig. 2: The DeployME Patch Cockpit

3 Durchsetzen eines vorgegebenen Workflows

Als Benutzerführung dient ein Patch-Cockpit, welches Bedienfehler verhindert und den vorgegebenen, mit dem Kunden abgestimmten Workflow durchsetzt. Das Patch-Cockpit ist das Browser-basierte Frontend einer Middleware, welche als Layer zwischen Frontend und Ansible AWX entwickelt wurde. Die Logik hinter dem Patch-Cockpit prüft für jeden Schritt die Voraussetzungen und stellt so sicher, dass der Workflow gemäß Vorgabe durchlaufen wird (Bild 2).

4 Middleware als Abstraktions-Layer

Die Middleware abstrahiert die Kommunikation zwischen Frontend (Patch-Cockpit) und Ansible AWX und speichert die Parameter von DeployME, die Status der Zielsysteme und alle Deployment-Vorgänge in einer Datenbank. Dieser Zwischenlayer ermöglicht durch die Abstraktion von Ansible AWX einen modularen Aufbau, wodurch eine Erweiterung der Funktionalität von DeployME vereinfacht wird:

- Unterstützung weiterer Use-Cases wie z.B. das Patchen von Netzwerk-Komponenten und anderer Betriebssysteme
- das Ausrollen von Applikationen anderer Hersteller
- Anpassen, respektive Erweitern des Frontends usw.

5 Robustheit der Lösung

DeployME muss mit unterschiedlichen Anomalien umgehen können: Netzwerk-Unterbrüche oder -Verzögerungen, Nicht-Verfügbarkeit von Zielsystemen, Unterbruch oder gar Crash eines Zielsystems während des Patch-Vorgangs, Fehlbedienungen des Bedien-Personals etc.

Die Lösung für diese Herausforderungen basiert auf folgenden Pfeilern:

- Idempotenz aller Aktionen; d.h., wenn eine Aktion schiefgeht, kann sie einfach wiederholt werden. Die Aktion führt immer zum gleichen Ergebnis, unabhängig, wie oft sie wiederholt wird.

4 Middleware as an abstraction layer

The middleware abstracts the communication between the front end (Patch Cockpit) and Ansible AWX and saves the DeployME parameters, the status of the target systems and all the deployment processes in a database. Abstracting Ansible AWX means that this intermediate layer opens up the possibility of modular structures, which makes it easier to extend the functionality of DeployME with

- support for additional use cases such as patching network components and other operating systems,
- the rollout of applications from other vendors,
- the modification / extension of the frontend etc.

5 The robustness of the solution

DeployME must be able to cope with a wide range of anomalies: network interruptions or delays, the non-availability of target systems, the interruption or even a crash of a target system during the patching process, mistakes by the operator etc.

The solution to these challenges is based on the following pillars:

- the idempotency of all actions – if an action fails, it can simply be repeated. The action always leads to the same result irrespective of how often it is repeated.
- this applies for all aspects of the deployment steps:
 - if the network is interrupted, the patch can be distributed again or distribution can resume from the point at which it was interrupted.
 - if certain target systems are unavailable, the patch process continues and the target systems concerned are patched later.
 - if patch installation is interrupted, the process can start from the beginning again with the system identifying where the interruption occurred and continuing from the appropriate point.
 - any operator errors and / or duplicated inputs are intercepted.

- Dies gilt für alle Aspekte der Deployment-Schritte:
 - bei einem Netzwerk-Unterbruch kann ein Patch erneut verteilt werden, resp. die Verteilung fährt dort weiter, wo der Unterbruch stattfand;
 - bei Nicht-Verfügbarkeit von Zielsystemen wird der Patch-Prozess nicht unterbrochen; nicht erreichbare Zielsysteme werden im Nachhinein wieder „eingefangen“;
 - bei Unterbrechung der Patch-Installation kann wieder von vorne begonnen werden, das System eruiert, wo die Unterbrechung stattfand, und fährt an der korrekten Stelle weiter;
 - Fehlbedienungen und Mehrfachbedienungen werden abgefangen.

• Redundanz von DeployME: DeployME kann mehrfach instanziiert werden; sollte eine Instanz crashen, kann eine weitere deren Aufgabe übernehmen und dort weiterfahren, wo die nicht mehr verfügbare Instanz abgebrochen hat. Der Kunde entscheidet das gewünschte Maß an Redundanz und bestellt entsprechend beim Lieferanten.

Da der Kunde seine Zielsysteme selbst in Betrieb nehmen, resp. auswechseln kann, kommt eine weitere Herausforderung hinzu: DeployME muss feststellen können, dass hinter einer IP-Adresse auch wirklich das erwartete System steckt. Dies wird gelöst, indem zu den eigentlichen Aufgaben wie „Deployen“, „Releasen“ und „Installieren“ eine ganze Zahl weiterer Aufgaben hinzukommen, welche vor und nach den Haupt-Aufgaben durchgeführt werden:

- Prüfen der Identität des Zielsystems mittels einer UUID. Ein „Universally Unique Identifier“, UUID, ist eine 128-Bit-Zahl, die verwendet wird, um Informationen in verteilten Systemen eindeutig zu identifizieren.
- Prüfen verschiedener Voraussetzungen, die das Zielsystem erfüllen muss (z. B. korrekte Setup-Version, korrekter Setup-Typ, verfügbarer Disk-Space für eine Patch-Installation),
- Prüfen des Status des Zielsystems:
 - welche Applikation darf laufen oder muss gestoppt sein
 - ist eine Bedien-Person gerade am Arbeiten

- the redundancy of DeployME: DeployME can be instantiated multiple times so that if one instance crashes, another instance can take over its tasks and continue from where it stopped. The customer decides on the level of redundancy required and orders a corresponding specification from Siemens.

The fact that customers can commission and swap out the target systems themselves creates a further challenge: DeployME must be able to establish that the system at a particular IP address is actually the system it expects. This problem is solved by adding a whole range of additional tasks that are completed before and after the main tasks:

- a check of the target system's identity based on a UUID (Universally Unique Identifier), a 128-bit number used for the definitive identification of information in distributed systems.
- a check of various necessary conditions that have to be satisfied by the target system (for example, the correct setup version, the correct setup type, the disk space available for patch installation)
- a check of the target system's status:
 - which application is allowed to be run or needs to be stopped, is there currently an operator at work,
 - what point in the deployment process has the target system now reached, etc.

6 The cloud-native technology stack

The DeployME components run as pods (containers) on a Kubernetes single-node cluster (on just one host) and can be operated in the cloud or on-premises (fig. 3).

Kubernetes itself runs on Linux, which is operated virtually.

Developed on GitLab¹, various pipelines alongside the actual build pipeline execute multiple actions prior to release. Examples include:

- installation in a container on Amazon AWS for smoke testing to ensure that the installation completes without any errors.

¹ GitLab is an all-in-one platform providing everything teams need to develop and deliver software, from code management (Git) and CI/CD (continuous integration / continuous delivery) pipelines to issue tracking, security scans and monitoring.

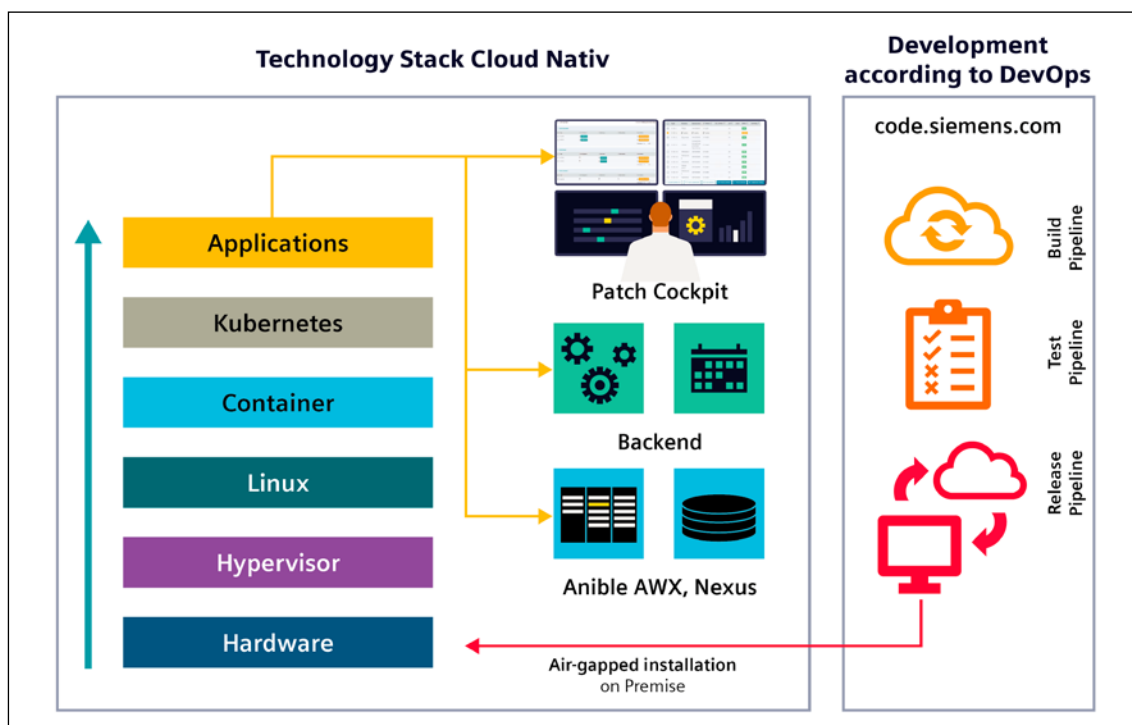


Bild 3: DeployME Cloud-Stack

Fig. 3: The DeployME cloud stack

- an welcher Stelle im Deployment-Prozess steht das Zielsystem aktuell usw.

6 Cloud-native Technologie-Stack

Die Komponenten von DeployME laufen als Pods (resp. Container) auf einem Kubernetes SingleNode-Cluster – also nur auf einem Host („Node“) – und können „in the cloud“ oder „on premises“ betrieben werden (Bild 3).

Kubernetes selbst läuft auf Linux, welches virtuell betrieben wird.

Auf GitLab¹ entwickelt, führen verschiedene Pipelines nebst dem eigentlichen „Build“ vor dem Release mehrere Aktionen aus (nicht abschließend):

- Eine Installation in einem Container auf Amazon AWP stellt als Smoke-Test sicher, dass die Installation grundsätzlich fehlerfrei durchläuft.
- Eine weitere Pipeline scannt alle Packages auf Security-Schwachstellen und erstellt einen Report.
- Für das OSS-Clearing werden automatisch entsprechende BOM-Files („Bill Of Material“) erzeugt und an den zuständigen Siemens-Server geschickt.

7 Air-gapped Installation

Da es aus Security-Gründen keine direkte Verbindung aus dem Iltis N-Netz ins Internet geben darf, muss die Installation ohne Internet-Verbindung möglich sein. Das Paketierungs-Framework „Zarf“ übernimmt diese Arbeit: Die gesamte Installation wird in sogenannte Zarf-packages gepackt und kann mit einem Script automatisiert und air-gapped installiert werden. Auch diese Zarf-packages werden automatisch in einer Build-Pipeline auf Git erstellt.

8 Integrität der Daten und des Codes

DeployME ist eine mächtige Lösung, welche falsch eingesetzt gravierenden Schaden anrichten kann: Die Automatisierung läuft auf den Zielsystemen mit Administrator-Rechten, was z. B. für die Installation eines Security-Patches des Betriebssystems Windows unumgänglich, aber auch ein potenzielles Einfallstor für Schadsoftware ist.

Es muss deshalb sichergestellt sein, dass sowohl der Code, welcher ausgeführt wird, als auch die Patches im Ablauf des ganzen Deployment-Prozesses nicht verändert wurden.

Dies geschieht durch kryptographische Hashes, welche während der ganzen Kette an Deployment-Schritten regelmäßig geprüft werden: ausgehend vom Git-Repository → auf die Kunden-Anlage mit der DeployME-Installation → bis zu den Patches auf den Zielsystemen. Auch das Sicherstellen von Integrität mit Zertifikaten kann unterstützt werden, ist in der aktuellen Version aber nicht umgesetzt.

Wie schon erwähnt, findet die Kommunikation zwischen Ansible AWP und Zielsystemen über eine verschlüsselte Verbindung (SSH) statt, die Browser-Anbindung läuft ausschließlich verschlüsselt (über HTTPS). Als Patch-Repository kommt Nexus zum Einsatz, welches die Patches ebenfalls über HTTPS verteilt.

¹ GitLab ist eine All-in-One-Plattform, die von der Code-Verwaltung (Git) über CI/CD-Pipelines („Continuous Integration/Continuous Delivery“, Kontinuierliche Integration/Kontinuierliche Auslieferung), Issue-Tracking, Sicherheits-Scans bis hin zum Monitoring alles bietet, was Teams für die Entwicklung und Bereitstellung von Software benötigen.

Sicher. Modular. Und für die digitale Schiene.

Unser sicheres und modulares Bahnsteuerungssystem PSSrail revolutioniert die Bahnindustrie und ermöglicht eine digitale Transformation. Von der Überwachung und Steuerung von Bahnübergängen bis hin zur elektrisch ortsgesteuerten Weiche (EOW) sowie dem EULYNX Object Controller bieten wir eine flexible Plattform, die sich Ihren individuellen Anforderungen anpassen lässt. Investieren Sie jetzt in die zukünftige Automatisierung der Bahn und profitieren Sie von innovativen, sicheren und digitalen Lösungen. Wir unterstützen Sie bei den Anforderungen, die die Digitalisierung der Bahnindustrie mit sich bringt.



Jetzt mehr erfahren!

PILZ

THE SPIRIT OF SAFETY

Pilz GmbH & Co. KG
Tel.: 0711 3409-0, info@pilz.de, www.pilz.de

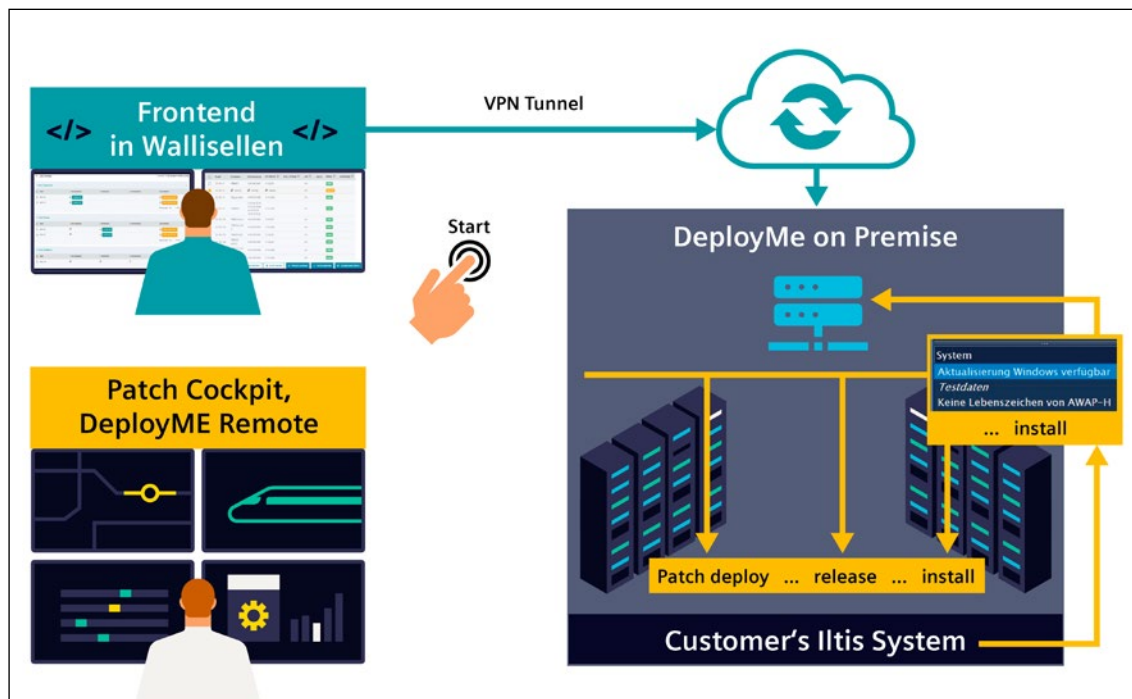


Bild 4: DeployME Frontend <> Backend
Fig. 4: The DeployME frontend <> backend

9 Testing

Bevor ein neuer Release von der Git-Umgebung heruntergeladen und im Test-Lab für die finalen Tests installiert wird, können erste Smoke-Tests (schnelle, grundlegende Tests, welche überprüfen, ob die wichtigsten Funktionen gegeben sind) durchgeführt werden:

- automatisiert im Rahmen einer Pipeline (siehe weiter oben); dies ist fester Bestandteil des Build-Prozesses;
- auf der lokalen Umgebung auf dem eigenen Laptop.

Auf dem eigenen (Siemens-)Laptop wird die Entwicklungsumgebung in einer WSL2-Installation („Windows Subsystem for Linux“) aufgesetzt. Für erste Tests kann DeployME in einer k3d Kubernetes-Umgebung aufgesetzt werden. K3d kommt anstelle von k3s zum Einsatz und setzt den Kubernetes-Cluster in einem Docker-Container auf. Auf diese Weise kann nicht nur das Setup auf korrekte Funktion geprüft werden, sondern die vollständige Applikation mit Backend und Frontend kann lokal gestartet und geprüft werden. Das PatchCockpit läuft dann im lokalen Browser und verbindet sich über ein Port-Mapping auf den lokalen Kubernetes-Cluster, der als Container im WSL2-Linux läuft.

10 Setup

Das DeployME-Setup ist vollständig automatisiert. Das Ergebnis der Build-Pipeline sind verschiedene tar-Files und PowerShell-Skripts, welche der Reihe nach auf einem Windows-Rechner, der über Netzwerk mit dem DeployME-Rechner verbunden ist, ausgeführt werden. Außer der Angabe der Passwörter für den Zugriff auf Ansible AWX und Nexus müssen keine Eingaben getätigt werden. Es braucht für diese Aktion keinen Internet-Zugriff („air-gapped installation“) (Bild 4).

Beim DeployME-Setup wird ein Intermediate-Zertifikat, welches von einem Root-Zertifikat signiert ist, übergeben. Aus diesem Intermediate-Zertifikat erzeugt Kubernetes dann ein entsprechendes HTTPS-Zertifikat. Auf diese Weise kann die DeployME-Infrastruktur in eine PKI integriert werden, und alle Verbindungen finden automatisch gesichert statt.

- another pipeline scans all the packages for security vulnerabilities and generates a report.
- appropriate BOM (bill of material) files are generated and automatically sent to the relevant Siemens server for OSS clearing.

7 Air-gapped installation

Security requirements dictate that the Iltis N network cannot be directly connected to the internet, so it must be possible to complete the installation with no internet connection. This capability is provided using the Zarf packaging framework: the entire installation is stored in Zarf packages and can be installed automatically with a script, thereby maintaining the air-gap. These Zarf packages are also created automatically in a built-pipeline on Git.

8 Data and code integrity

DeployME is a powerful solution with the capacity to cause chaos if used incorrectly: the automated process runs on the target systems with administrator privileges, which is essential when installing security patches for the Windows operating system, for example, but also presents a potential point of entry for malicious software. It is therefore essential to ensure that neither the executed code nor the actual patches are modified at any point in the deployment process.

This is achieved using cryptographic hashes that are checked regularly all along the chain of deployment steps from the Git repository → to the customer system with the DeployME installation → to the patches on the target systems. Integrity verification using certificates can also be supported but is not implemented in the current version.

As already mentioned, Ansible AWX and the target systems communicate over an encrypted connection (SSH). The browser connection is always encrypted (using HTTPS) and the used patch repository is Nexus, which also distributes the patches via HTTPS.

Auch das Setup des Linux-Betriebssystems (virtuelle Maschine), auf welchem DeployME läuft, ist automatisiert. Über ein Cloud-Init-File wird Linux vollständig parametrisiert, anschließend läuft die Linux-Installation ohne User-Input durch. Im Cloud-Init-File werden die Parameter für die IP-Adresse, den Rechnernamen und das Root-Passwort je Kunde angepasst. Das Cloud-Init-File wird in Form eines ISO-Files als zusätzliches CDROM-Laufwerk auf der virtuellen Maschine gemountet und durch das Linux-Setup automatisch eingelesen.

11 Fazit

DeployME ist eine zukunftssträchtige Lösung. Aktuell wird es für das Patchen von Windows-Betriebssystemen eingesetzt, kann aber auch Linux patchen und wird schon bald für Netzwerk-Komponenten einsetzbar sein. Grundsätzlich sind aber beliebige weitere Deployment-Aufgaben denkbar:

- DeployME ist modular aufgebaut und kann durch sogenannte „AddOns“ auf einfache Weise erweitert werden.
- Durch das Erweitern mit eigenen Scripts kann Ansible AWX grundsätzlich fast alles automatisieren.
- Durch ein Browser-basiertes Frontend ist auch das Benutzer-Interface beliebig anpassbar oder erweiterbar.

Auch an die Skalierbarkeit ist gedacht: Durch die Lösung mit Kubernetes ist es ohne großen Aufwand möglich, die aktuelle Single-Node-Umgebung als Multi-Node-Cluster aufzubauen, um so Ausfallsicherheit und Skalierung auf Ebene Kubernetes zu erreichen. Somit ist DeployME bestens gerüstet für weitere Herausforderungen. ■

9 Testing

Initial smoke tests (quick basic tests to verify whether key functions are working) can be performed before a new release is downloaded from the Git environment and installed in the test lab for final testing. These smoke tests can be

- a) completed automatically as part of a pipeline (see above); this is an integral part of the build process.
- b) completed on a laptop in the local environment.

The development environment is provided in a WSL2 (Windows Subsystem for Linux) installation on a (Siemens) laptop. DeployME can be provided in a k3d Kubernetes environment for initial testing. K3d is used instead of k3s and provides the Kubernetes cluster in a Docker container. This method makes it possible not only to check that the setup is working properly, but the complete application with the backend and frontend can also be locally started and tested. The Patch Cockpit runs in the local browser and connects via port mapping to the local Kubernetes cluster running as a container in WSL2-Linux.

10 Setup

The DeployME setup is fully automated. The built pipeline outputs various tar files and PowerShell scripts that are executed in order on a Windows computer connected over a network with the DeployME computer. The only required inputs are the passwords for access to Ansible AWX and Nexus. Internet access is not required for this action (air-gapped installation) (fig. 4).

An intermediate certificate signed by a root certificate is transferred during DeployME setup. Kubernetes then creates a corresponding HTTPS certificate from this intermediate certificate. This makes it possible to integrate the DeployME infrastructure into a PKI, with all the automatically established connections being secure.

The setup of the Linux operating system (virtual machine) on which DeployME runs is also automated. Linux is fully parameterised via a cloud-init file and the Linux installation then completes with no user input required. The IP address, computer name and root password parameters for each customer are configured in the cloud-init file. The cloud-init file in the form of an ISO file is mounted on the virtual machine as an additional CD-ROM drive and read automatically in the Linux setup process.

11 Conclusion

The DeployME solution is fit for the future. It is currently used for patching Windows operating systems but can patch Linux too and will soon also be suitable for use with network components. It is capable in principle of being adapted for any number of additional deployment tasks:

- DeployME is modular in design and can be easily extended with add-ons.
- Ansible AWX has the potential in principle to automate practically anything if extended with specific scripts.
- A browser-based frontend means that the user interface can also be adapted and extended as required.

The scalability aspect has been covered as well: the fact that the solution is based on Kubernetes means that the current single-node environment could be expanded to a multi-node cluster relatively easily for fault tolerance and scaling at the Kubernetes level. DeployME is thus ideally equipped to meet future challenges. ■

AUTOR | AUTHOR

Viktor Locher

Senior System Engineer

Siemens Mobility AG

Anschrift / Address: Hammerweg 1, CH-8304 Wallisellen

E-Mail: viktor.locher.ext@siemens.com