indeni

How Indeni Works

The World's Best Practices, Automated

White Paper

Executive Summary

Visibility is Key

Indeni provides security infrastructure automation with unprecedented visibility that's up and running in minutes. We've automated the world's best practices to deliver predictive, prioritized, and actionable insights that help you prevent costly disruptions. We dramatically reduce chaos and risk to improve agility, giving you the confidence to accelerate mission-critical projects that drive new business.

This white paper describes the technical underpinnings of the Indeni platform, focusing on:

- the device Knowledge that provides Indeni with high accuracy in device stability;
- the information flow between device onboarding, Auto-Detect, and Auto-Triage;
- and the internal software components of the Indeni platform.

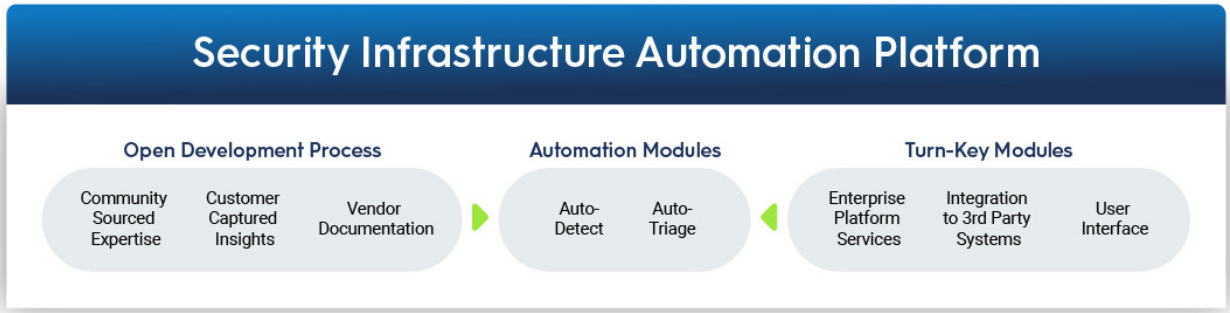
Who should read

This document is intended to provide a high-level technical overview of the Indeni platform for solution architects and customer deployment engineers. Its focus is on the concepts unique to Indeni, rather than the specifics of application service containers and databases.

Table of Contents

Production Ready Knowledge

Overview of Automation Types



The foundation of the Indeni Security Infrastructure Automation Platform is Knowledge: vendor-specific machine instructions and business logic for multi-factor detection and investigation. There are two types of Knowledge: Auto-Detect Elements (ADEs) are used for high-confidence detection of issues, and Auto-Triage Elements (ATEs) are used to annotate and enrich issues with automated best practice investigation.

For Indeni to be useful for Operations teams, we realized that users would need to be able to trust our findings, which means using auditable interfaces. Therefore, the knowledge and code for each of our automation modules are published in an Open Source manner on our website, so it's available for customer analysis, auditing, and improvement.

Auto-Detect

ADEs are the set of native device commands and business logic to retrieve and interpret information about device configuration, state, and metrics. These automation modules are composed of Scripts and Rules.

When Indeni first started crowd-sourcing knowledge, our research showed that administrators seldom relied on the output of a single command when troubleshooting their devices, or, if they did, it was because they knew the specifics for how the device was configured. It showed us that real experts are not only familiar with the commands for their equipment, but also the power of multiple commands in specific combinations. We therefore designed our rule architecture to provide automated conclusions based on multiple contextual inputs.

Scripts contain the specific CLI commands, API calls, and SNMP OIDs to extract information from remote devices, as well as what measurements to extract. Each script also lists its own analysis interval, which ranges from running once per minute up to once per hour. While scripts typically contains a single command or call, they often return multiple measurements, since many device commands return multiple data items in a table or similar human-readable form.

Rules contain the business logic to evaluate one or more measurements to detect an issue. While the simplest form of a rule would be a single measurement exceeding a threshold (e.g. average CPU usage on a core exceeding 60%), a more complex rule could check whether clustering is enabled, then compare interfaces between the cluster peers, then parse the configuration to determine whether a standby peer should have interfaces up or down, and then compare

that calculated expectation to the actual device state. Rules typically make use of multiple measurements, and each measurment can be referenced by multiple rules.

Example ADEs

These two ADEs use the same rule, but different scripts. Consequently, they have the same business logic, but different remediation instructions. Readers interested in the underlying differences between ADEs targeting different vendors are encouraged to read the “Script(s)” and “Rule” section of each ADE to compare and contrast the XML/API and SSH CLI.

Palo Alto: Communication between management server and specific devices not working
Check Point: Communication between management server and specific devices not working

Auto-Triage

ATEs are the set of automation commands to perform best practice diagnostic and troubleshooting steps. Whereas ADEs are restricted to gathering information, ATEs can perform no-risk actions, such as sending a ping from the remote device, or even from the Indeni platform. These Automation Modules are powered by Indeni contributed or publically available open source Ansible Modules and Ansible Playbooks.

Indeni chose to embed Ansible for Auto-Triage because Ansible’s hierarchical model provides for abstraction of business logic, which allows for readability by device experts without requiring a background in reading code. Indeni recognizes that domain expertise can be rare, and has attempted to follow the design concept of providing information rather than requiring it. Additionally, Ansible is an Open Source project, and all of the Modules that Indeni is using are either Open Source or internally developed and publicly posted. That allows customers and prospects to audit the automation code and establish their own trust level.

Ansible Modules are code libraries, usually Python, that contain the details for interacting with specific objects or devices, and which provide a high-level abstract interface. For example, if the preferred native interface to a device were CLI over SSH, the Ansible Module would handle the details of establishing the session, authenticating, forwarding the designated commands, and parsing and returning the result(s). Modules are a useful way to hide the device interaction details from the business logic for a particular task.

Ansible Playbooks are collections of Plays, usually Tasks. A Task is a simple list of the inputs and output(s) provided by a Module, so the intended interaction with the remote devices is easy to read. To support this readability, Playbooks use the YAML syntax, which is based primarily on whitespace rather than special characters. The result is that each Play reads like a bullet-point slide rather than a computer program, but runs like a well trained engineer focused on understanding the severity and resolution for each issue.

Example ATEs

These two ATEs use the same ADE rule, but target different devices. Consequently, they have the same detection business logic, but different Auto-Triage automation. Readers interested in the underlying differences between ATEs targeting different vendors are encouraged to read the “Auto-Triage” section of each ATE to compare and contrast the playbook for each device.

Palo Alto: Communication between management server and specific devices not working
Check Point: Communication between management server and specific devices not working

Internal Process Flow

How Knowledge is Applied Through Enterprise Platform

Information processing in Indeni has both periodic and event driven elements. The periodic elements continuously recur on set frequencies, and the event driven elements only occur when triggered by certain circumstances. Together, Indeni combines these two patterns to provide high visibility with low impact.

Device Onboarding

Device Onboarding is the event driven process of adding a remote device to Indeni, interrogating it to determine its characteristics, and bringing it under analysis. The full onboarding process is triggered only by an Indeni administrator adding a remote device, but there are additional occasions (listed later in this subsection) when it is necessary to re-interrogate it.

Interrogating a device is the interaction to determine the specific characteristics which will specify which Knowledge should be applied. For example, a vendor may release security infrastructure products with different operating systems, or may have different products for enforcement and for management. Interrogation pre-populates Indeni with the information about which Auto-Detect scripts and which Auto-Triage modules to use for the specifics of device interaction, and which Auto-Detect rules and Auto-Triage playbooks to use for the business logic of decision making.

It is sometimes necessary to re-Interrogate a device to recalibrate the set of Knowledge that should be used, e.g. the device's version is upgraded and the vendor has changed the commands and/or information provided. While the Indeni platform normally is able to detect these situations itself, an administrator can manually trigger re-Interrogation by manually disabling a device, then re-enabling it. That assures that the Indeni platform can always apply the right Knowledge to remote devices to reach accurate and actionable conclusions.

Auto-Detect

Auto-Detect is the periodic process of querying remote devices to detect issues. Based on the characteristics determined during Interrogation, Indeni pre-tunes itself for each remote device to determine which Auto-Detect Knowledge to use. Different products offer different features and have variable levels of complexity, e.g. a small number of fixed-purpose interfaces within a single domain vs a large number of multi-VLAN multi-zone interfaces with internal software-defined connections to discrete virtual firewall instances.

Each Auto-Detect script lists its own automation interval. Depending on the script, the interval may need to be short to react to measurements that may indicate urgent action or change frequently, or much less frequent for variables that are relatively static or for operations that have a higher impact on the device. The interval is used to build an internal schedule of when each script will trigger on each device.

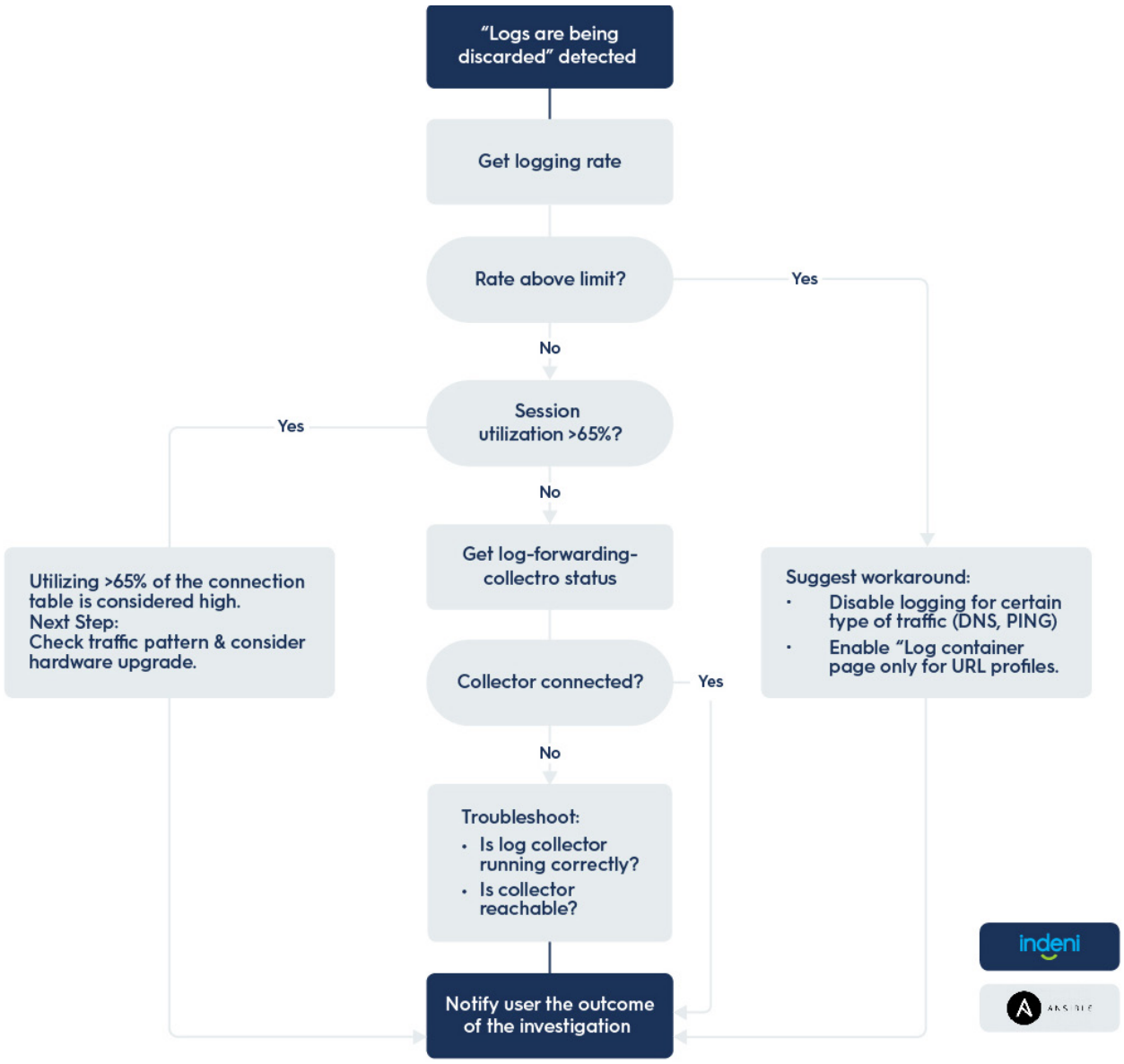
Once each minute, all scheduled scripts are executed with optimization to prevent thundering herd situations on devices. All raw script output is remotely collected, then locally parsed and processed to create unique measurements. Each measurement is identified by name based on its definition in the script, and is stored in a round-robin database (RRDB) for processing.

Once the scripts complete, all rules are executed on the new measurements. Each rule has access to all measurements for any given device, to allow fully contextual situational processing while maintaining a low information-gathering impact.

Each rule has one primary output: does an issue exist? If so, the additional information is populated, such as the issue specifics, description, remediation, and detail log. Starting in Indeni 7.0, when a rule detects an issue, it can also trigger an Auto-Triage playbook.

Auto-Triage

Auto-Triage is the event driven process of interacting with remote devices to gather information and perform no-risk diagnostic actions from either the device or the Indeni platform itself. Playbooks are explicitly triggered by an Auto-Detect rule that has detected an issue on a device.



A useful capability of Auto-Triage is branching logic, which allows for multiple if/then program forks. That capability provides for straightforward mapping of best practice troubleshooting and investigation flowcharts into unattended diagnostic automation. Additionally, Auto-Triage can identify what information is relevant based on previous results, so it can retrieve additional relevant measurements and test results.

Since Auto-Triage is triggered by issue detection, its response time is very close to an on-device event-driven reaction, so all of the information that is captured during the automated incident investigation is captured either during or very shortly after the time the issue occurred. That rapid response is a powerful counter to the dreaded scenario of "wait and see if it happens again."

Once the Auto-Triage phase finishes, all investigation findings are appended to the Issue listing in the GUI, as well as pushed to any downstream integrations such as SNMP, syslog, or ServiceNow. This automatic enrichment of the trouble ticket elevates Indeni's findings with the sensation that another engineer has already performed several troubleshooting steps and documented their findings. In the event that the troubleshooting resolved the issue, Indeni will apply its internal Issue lifecycle management, and mark the Issue as closed once it is detected as resolved.

Platform Architecture

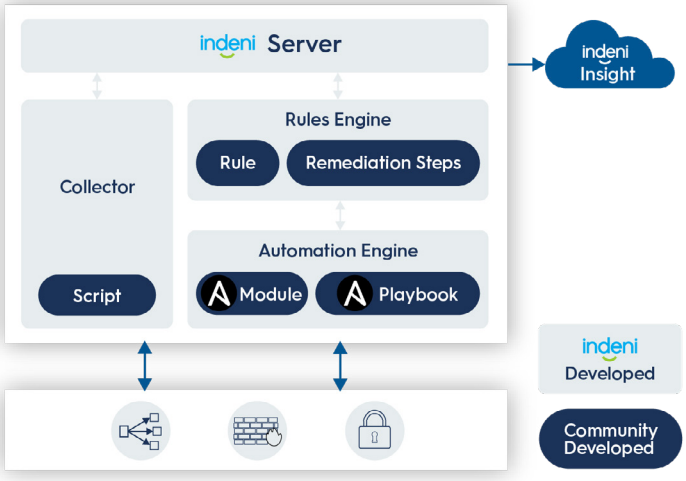
Indeni provides production-ready knowledge curated from vetted, community-sourced experience. Our automation tackles tedious tasks with turn-key integration with your existing processes. It's certified automation, with control, so you can focus on mission-critical projects that drive new business.

Key Benefits:

- Automatically detect issues with greater accuracy
- Triage issues immediately, without human intervention
- Resolve complicated issues quickly before they become problems.

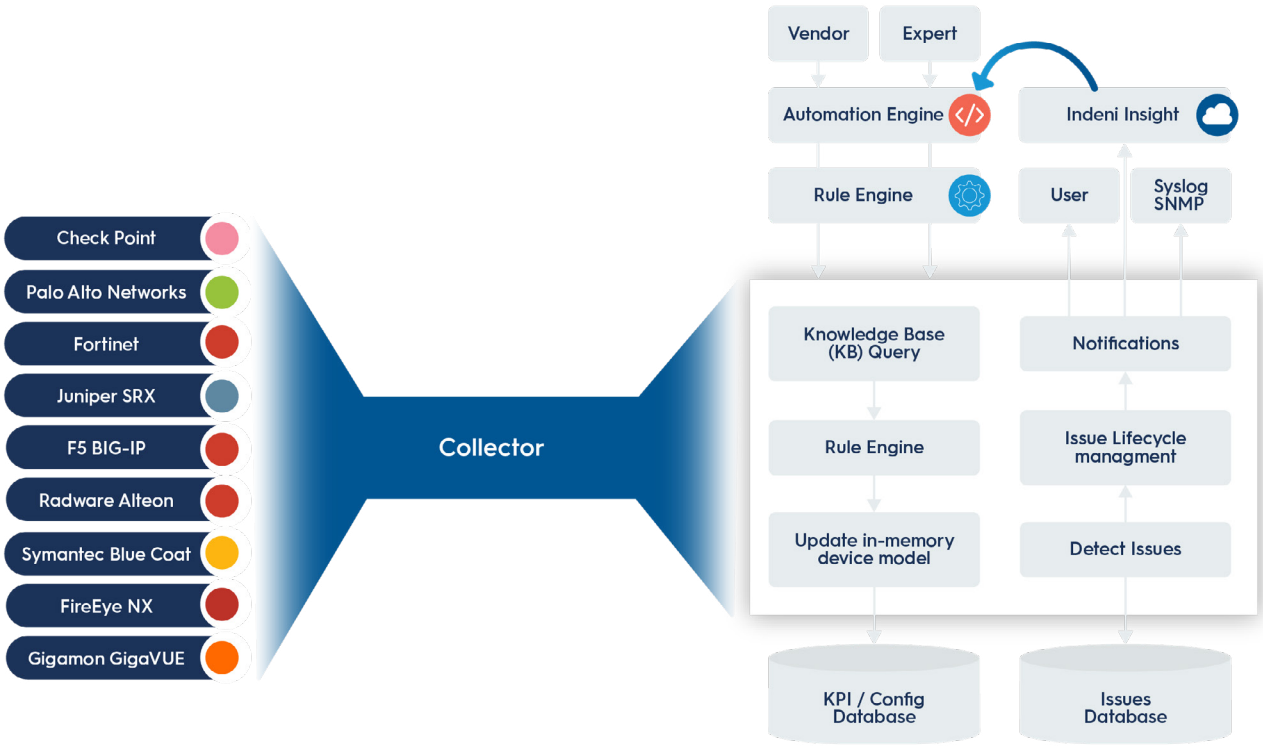
Technical Overview

The Indeni platform is comprised of many components including the Collector, Server, Automation Engine, Automation Modules, User Interface, and Indeni Insight.



Collector

The Collector is a fairly lightweight component whose sole purpose is to interact with network and security devices through various protocols, issue CLI commands and API calls, and collect data. It is the component that executes the Auto-Detect scripts to retrieve measurements from devices. While many measurements are simple elements like single numbers, they can be as large as the whole set of configuration data collected from a device.



Server

The Server is where the data processing actually occurs. The Server contains multiple types of databases, all of them on the same virtual or physical machine at the customer site. The Server instructs the Collector which devices to connect to, what credentials to use, and which script to execute, and in turn it receives the measurements for that script for that device. Most of those measurements are stored in a round-robin database (RRDB), although other measurements may be stored in a config backup database, KPI database, etc.

The Server also contains the Rules Engine, which executes the Auto-Detect rules. The rules process the script measurements and reach conclusions about Issues. For any rules that trigger Auto-Triage Knowledge, the server instructs the Automation Engine which device to connect to, what credentials to use, and which playbook to execute.

Automation Engine

The Automation Engine is the framework which integrates the Auto-Triage playbooks and modules, then executes the resulting instructions, usually on the remote device, but potentially on the Indeni platform itself if there are connectivity or HA issues.

The Automation Engine uses Ansible Runner to process the ATEs, which are a combination of modules and playbooks from Ansible Galaxy, specific security infrastructure vendors, and Indeni and the Indeni Community.

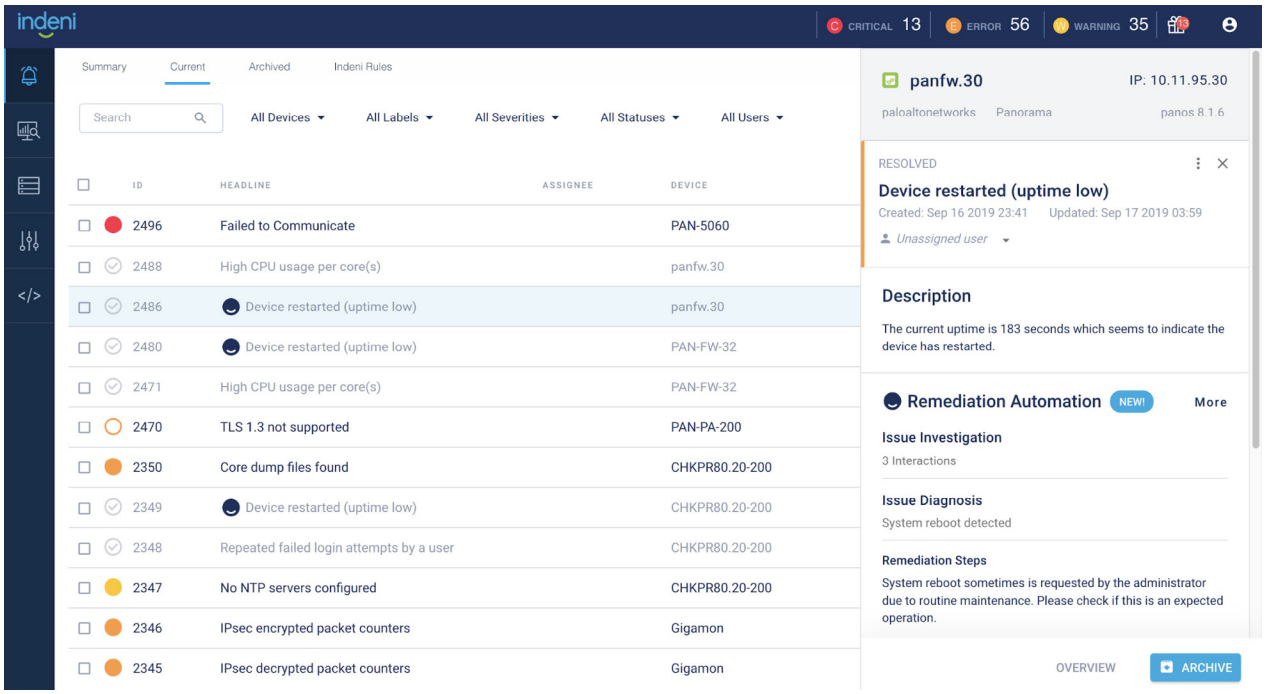
Automation Elements

"Automation Elements" is the umbrella term for Auto-Detect Elements (ADEs) and Auto-Triage Elements (ATEs). The world's best practices are captured in these two types of automation modules, both of which can be examined not only inside the Indeni platform, but also on the Indeni Knowledge Explorer web site.

User Interface

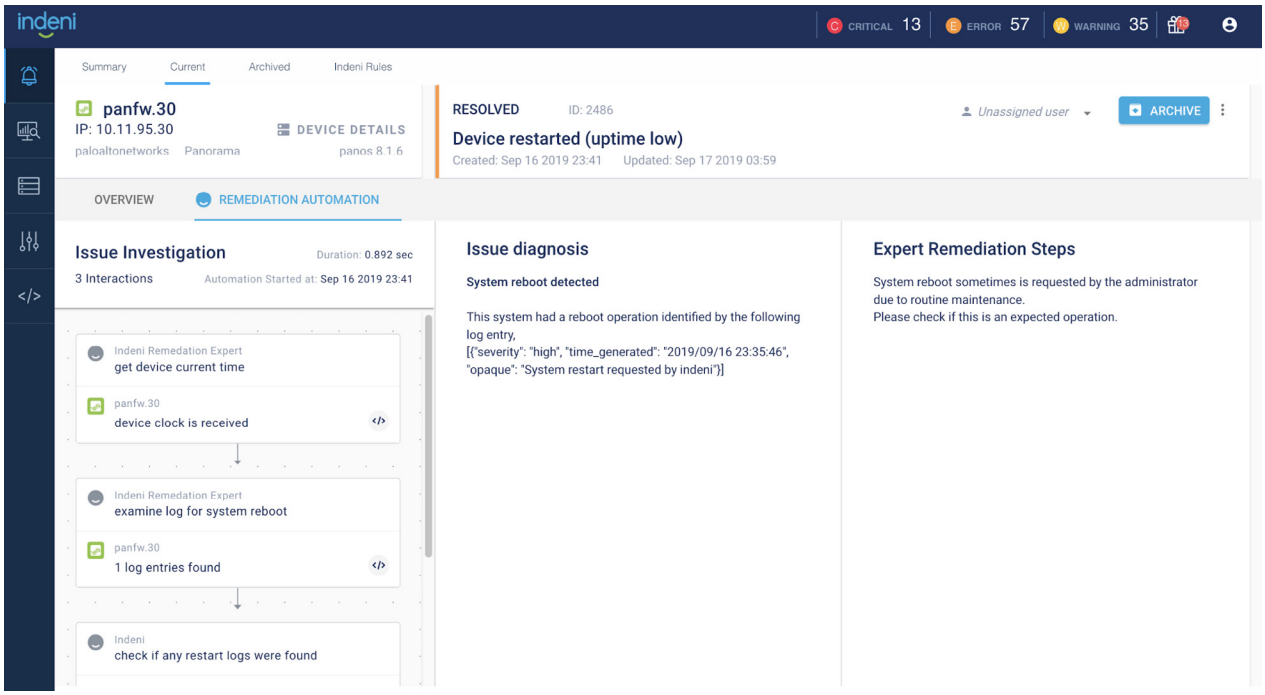
The Indeni web user interface (UI) is based on standard Operations tool design paradigms to ensure that it is immediately usable and useful. It uses a self-signed certificate for TLS to protect sensitive data in transit, and enforces Role-Based Access Control (RBAC) to limit action permissions to authorized users.

UI Summary Tab



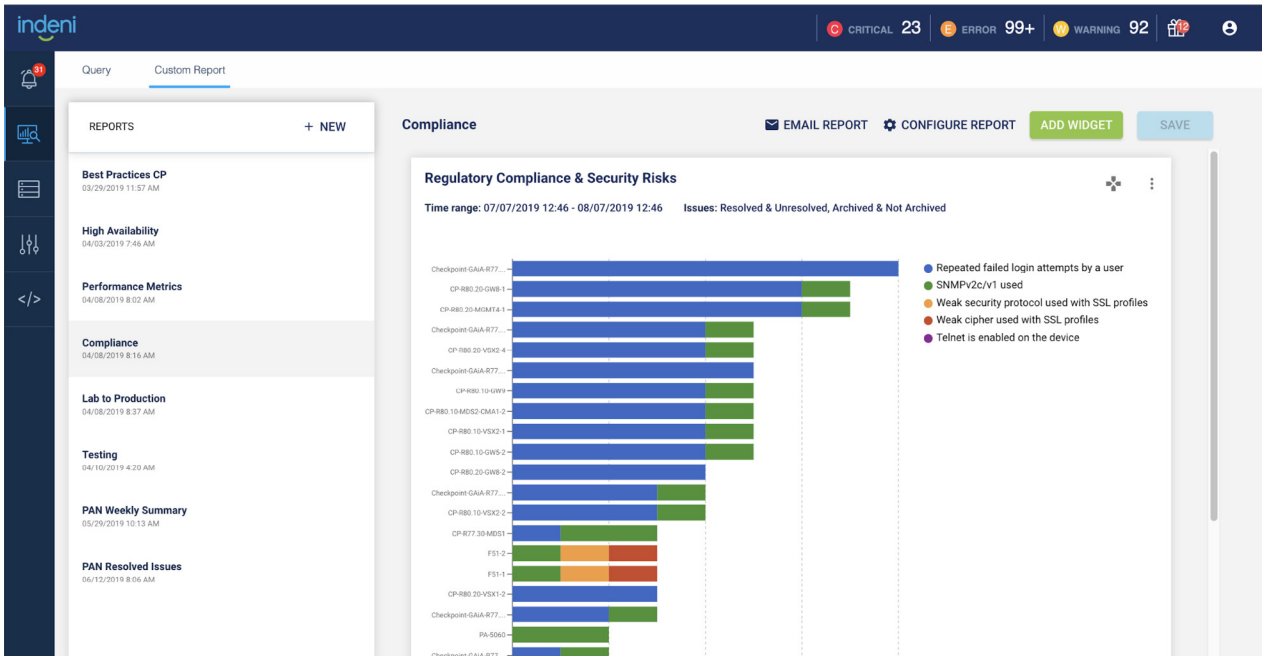
The UI Summary tab is the default landing page, immediately presenting the list of both active and recently remediated issues. Each issue is based on an ADE conclusion, with select issues enhanced by ATE investigation analysis. All issues include remediation steps either to accelerate TTR or avoid an outage altogether.

UI Analysis Tab



The UI Analysis tab provides a deeper dive into the measurements and history of each issue, with either a history graph of the most relevant measurement, or a logic chain to audit any ATE investigation actions. Indeni boosts team productivity by combining per issue measurements, diagnosis, and remediation steps, mixed together in production-ready knowledge and automation.

UI Reports Tab



The UI Reports are clear visualizations of actionable conclusions, either scheduled or on-demand. The underlying widgets allow quick setup and customization to provide focus on key areas like High Availability readiness, Best Practices compliance, and devices currently at High Risk of an outage due to resource exhaustion.

Indeni Insight

Indeni Insight is the set of services including Indeni Proactive Support and Indeni Continuous Automation Improvement.

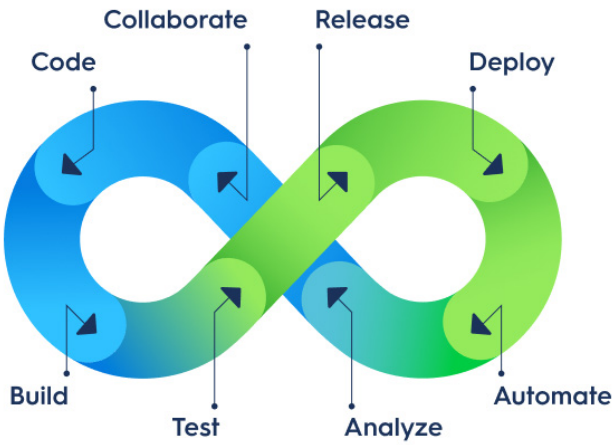
Indeni Proactive Support has especially been popular with customers, as it provides "Indeni for Indeni", an advance notice from our support that the Indeni platform at the customer site is exhibiting signs of instability, so there's time to take corrective action before any potential outage.

Indeni Continuous Automation Improvement is a service which leverages our back-end data lake to analyze the performance of deployed and potential automation modules. With the feedback from real world platforms, Indeni can detect when individual ADEs and ATEs are potential candidates for code updates, based on variances in such aspects as detect rates, recurrence, and resolution times. This service helps Indeni maintain the trustworthiness and top performance of its automation at customer sites.

Indeni In Action

Retail Customer Case Study

In December 2017, an Indeni customer experienced an issue with one of their Palo Alto Networks devices. Specifically, the device had just one power supply installed in it and that power supply died. Once that happened, the device shut down and a painful chain reaction occurred in the network. The root cause analysis pointed to the power supply and the conclusion was that all of their thousands of devices need to be audited to ensure they each have two working power supplies installed. You can imagine how much manual work that would be.



That was on a Thursday. The customer immediately reached out to the Indeni community and asked for help. Specifically, they needed a script that will let them know if any of their devices has just one working power supply - whether it's because the other power supply in the same device is down, or if it doesn't have one at all.

On Friday, one of the members of the community, who is a Palo Alto Networks expert, prepared a short document describing the data that needs to be collected and how it should be analyzed. On Saturday, another community member (a developer) wrote the automation script. On Sunday, a third community member reviewed and approved the script and on Monday it was deployed in the customer environment.

Three people, two in the US and one in Lithuania, spent a few hours and together helped save hundreds of hours for a group of people they've never even met.

At A Glance

- **Collaborate -** Customer reached out to Indeni Community via Indeni Crowd
- **Code -** Core team of developer, device subject matter expert and mentor assembled
- **Build -** Core team built Knowledge script and reviewed with customer
- **Test -** Developer tested code manually with Indeni Development tools. Once complete Indeni validated through automated internal testing
- **Release -** Indeni approved release and Indeni software was updated
- **Deploy -** Customer updated their instance by downloading the latest Indeni software
- **Automate -** High availability validation task for Palo Alto Networks device now automated
- **Analyze -** Customer can manage the automation via the Indeni Dashboard with confidence

How to get started

Indeni's mission is to deliver the information you need in the way you need it. To experience Indeni for yourself, feel free to choose one or more of the following methods. We look forward to learning about your needs and seeing how Indeni can support you.

- Request a demo
  - A guided tour by an Indeni expert in our demo environment
  - No download or installation required
  - Discuss live how Indeni might be able to solve problems in your environment
- Access Simulated Environment
  - Try out Indeni for yourself in our high fidelity cloud-based sandbox
  - No download or installation required
  - Familiarize yourself with our UI and evaluate how Indeni could help you address your challenges
- Download Trial Software
  - Try Indeni as a full featured VM in your own lab
  - Connect real devices for up to 30 days
  - Test multiple integrations: devices, workflow, escalation, and custom automation

About Indeni

Indeni is the crowd-sourced automation platform for security infrastructure. With the Indeni Automation Platform organizations gain access to living repository of scripts that automate tasks for maintenance, high availability, network visibility, security, compliance, validating vendor best practices and much more. Learn more at www.indeni.com.

Corporate Headquarters
San Francisco, CA USA
Tel: +1-877-778-8991
Email: info@indeni.com

European Headquarters
Tel Aviv, Israel
Tel: +1-809-494-190
Email: info@indeni.com