Joel Vornanen

ScienceLogic SL1 basics and server monitoring

Metropolia University of Applied Sciences

Bachelor of Engineering

Information and Communication Technology

Bachelor's Thesis

22 April 2021

Abstract

Author:	Joel Vornanen
Title:	ScienceLogic SL1 basics and server monitoring
Number of Pages:	36 pages + 2 appendices
Date:	22 April 2021
Degree:	Bachelor of Engineering
Degree Programme:	Information and Communication technology
Professional Major:	Software Engineering
Supervisors:	Janne Salonen, Degree Supervisor

Keywords: Monitoring, ScienceLogic, Servers, SL1

The object of the thesis was to study the basic features and technologies of ScienceLogic SL1. It also includes a comparison of Agent-based and Agentless monitoring technologies for Windows servers and describes the installation process of availability and capacity monitoring on Windows and Linux servers.

To find out which of the two monitoring technologies was more suitable for our needs, the features of the technologies were reviewed, and external automation support was taken into consideration. The technical information was mainly acquired from ScienceLogic's documentation and personal experience from working with the platform.

The conclusion was that ScienceLogic SL1 is an OK all-around monitoring platform with loads of customizability and features, but it suffers from bad UI elements, unpolished monitoring features, and lacking reporting capabilities. In the end Windows monitoring was done by utilizing Agent-based and Agentless technologies to benefit from the lower performance hit of Agent-based monitoring and the customizability of Agentless monitoring. Linux monitoring was done purely with Agentless monitoring as it was easier to implement, and our external automation supports it.

Tiivistelmä

Tekijä:	Joel Vornanen
Otsikko:	ScienceLogic SL1:n perusteet ja palvelimien valvonta
Sivumäärä:	36 sivua + 2 liitettä
Aika:	22.4.2021
Tutkinto:	Insinööri (AMK)
Tutkinto-ohjelma:	Tieto- ja viestintätekniikka
Ammatillinen pääaine:	Ohjelmistotuotanto
Ohjaajat:	Osaamisaluepäällikkö Janne Salonen

Avainsanat: Palvelimien valvonta, ScienceLogic, SL1

Insinöörityön tarkoituksena oli tutkia ScienceLogic SL1 valvontajärjestelmän keskeisiä ominaisuuksia ja teknologioita. Työ myös sisältää vertailua agenttipohjaisten ja agentittomien valvonta teknologioiden välillä Windows palvelimien kohdalla, sekä kuvaa valvontojen asennuksen Windows ja Linux palvelimille.

Sopivan valvontateknologia valitsemiseksi tutkittiin agentillisen ja agentittoman valvonnan eroavaisuuksia ja ominaisuuksia. Tieto kerättiin pääosin ScienceLogic SL1:n dokumentaation ja kokemuksen kautta.

Tiivistettynä SL1 on kohtuullisen hyvä valvontajärjestelmä, joka sisältää paljon ominaisuuksia ja muokattavuutta, mutta kärsii huonoista käyttöliittymää koskevista valinnoista, kiillottamattomista valvontaominaisuuksista ja puuttuvista raportointiominaisuuksista. Windows valvonta toteutettiin käyttämällä kumpaakin valvontateknologiaa. Tämä tehtiin, jotta agentillisen valvonnan parempaa suorituskykyä ja agentittoman valvonnan muokattavuutta voitaisiin hyödyntää. Linux valvonta toteutettiin käyttäen agentittomaa valvontaan, koska tämän asennusprosessi on helpompi, ja teknologia on kyseenomaisessa instanssissa laajemmin käytetty.

Contents

List of Abbreviations

1 Introduction 1

1.1 History of ScienceLogic 1
1.2 Structure of an SL1 instance 1
1.3 Most important features and supported systems 5

2 Dynamic Applications 10

2.1 Dynamic Application archetypes 10
2.2 Dynamic Application technologies 11

3 Most monitored parameters 13

3.1 Availability and latency monitoring 13
3.2 Capacity monitoring 14
3.3 Process and Windows service 16
3.4 Additional monitoring technologies 17

4 Events, dashboards, and reports 19

4.1 Event structure and criticality levels 19
4.2 Dashboards 20
4.3 Reports 21

5 Topology maps 23

6 Monitoring servers 25

6.1 Windows monitoring with agent or agentless 25
6.2 Monitoring Microsoft Windows servers 27
6.3 Monitoring Linux servers 31

7 Summary 33

References 34

Appendices

Appendix 1: Creating a Non-Administrator User Account

Appendix 2: Manually Configuring Windows Remote Management

List of Abbreviations

AD:	Microsoft Active Directory
AI:	Artificial Intelligence
AIOps:	Artificial Intelligence for IT Operations
API:	Application Programming Interface
CA:	Certificate Authority
CDP:	Cisco Discovery Protocol
CEO:	Chief Executive Officer
CMD:	Windows Command Prompt
CMDB:	Configuration Management Database
CPU:	Central Processing Unit
CTO:	Chief Technology Officer
HTML5:	Hypertext Markup Language 5
HTTP:	Hypertext Transfer Protocol
HTTPS:	Hypertext Transfer Protocol Secure
ICMP:	Internet Control Message Protocol
IP:	Internet Protocol address
IT:	Information Technology

LDAP:	Lightweight Directory Access Protocol
MAC:	Media Access Control address
ODS:	OpenDocumentation Spreadsheet Document format
OID:	Object Identifier
PDF:	Portable Document Format
RAM:	Random-access Memory
REST:	Representational state transfer
SNMP:	Simple Network Management Protocol
SSH:	Secure Shell Protocol
TCP:	Transmission Control Protocol
UDP:	User Datagram Protocol
URL:	Uniform Resource Locator
VCS:	Version Control System
WinRM:	Windows Remote Management
WMI:	Windows Management Instrumentation

1 Introduction

This thesis consists of going through the basics of ScienceLogic SL1, the network and infrastructure managing, and monitoring platform used by multiple big companies around the world. It will contain information of the most important features and compatibilities of SL1, go through the process of monitoring servers with PowerShell, Python and Agent-based solutions, and explain all the required technologies used in the pro-cess. There will be a comparison between agent-based and agentless monitoring in Windows environments, that aims to bring out the differences and benefits of these solutions, and what the use cases are for each monitoring solution.

1.1 History of ScienceLogic

ScienceLogic was founded in 2003 by the current CEO (Chief Executive Officer) David Link, former employee, and CTO (Chief Technology Officer) Chris Cordray and current Chief Scientist Richard Chart, and it has experienced rapid growth since its release. Some of ScienceLogic's customers include Cisco who replaced its homegrown monitoring system with SL1, Kellogg who uses SL1 to monitor its Hybrid cloud deployment, and Liberty who had a 99% reduction of network outage incidents once they started using SL1. [1; 2; 3.]

In 2018 ScienceLogic released its new Context-Infused AIOps (Artificial Intelligence for IT Operations) platform. ScienceLogic claims that business is evolving to being more agile, and the by-product of that is extreme complexity. Introducing AI into SL1 is supposed to lessen the added noise of complex system structures and help identify real problems in one's environment. [4]

1.2 Structure of an SL1 instance

ScienceLogic offers different hosting options for their monitoring solution, but in this thesis the implementation will be hosted On-Premises in a Distributed

instance. The Distributed installation of SL1 in question consists of one or multiple of these appliances:

- Database server containing all the policy, performance, and configuration data.
- Data collector, that discovers devices and collects agentless data.
- Message collector, that collects SNMP (Simple Network Management Protocol) Trap messages, syslog messages, and agent-based data.
- Administration portal that provides the user interface and API (Application Programming Interface) capability for the application.
- Extended architecture, which will be implemented in the future, bringing four new appliances into the system structure.

This instance of SL1 uses two database servers to combat database failure. In case of a database failure, the system is pointed to the failover database to ensure the application remains available and the least amount of data is lost. Database servers are also in charge of retrieving collected data from data collectors and message collectors and processing it [5]. All emails generated by SL1 are sent by the database server, including email reports and event notices [5].

All monitoring data is collected by data collectors. Data collectors are used to discover new devices into the SL1 system. An instance of SL1 can have one or multiple data collectors depending on the amount of monitored devices and on the number of customers that require dedicated collectors. The instance in question has around 15 data collectors.

Data collectors and message collectors reside in collector groups [6]. A data collector can handle between 300 to 6000 end devices depending on the specifications of the collector, how intense the monitoring is, and what is the latency to the end device [6]. A collector used for varied types of monitoring can usually handle around 1000 devices [6]. What comes to message collectors, one message collector responsible of communicating with SL1 agents can handle around 350 to 400 agent monitored devices at a time.

The user interface where most of the configurations regarding the system are made, is accessed through a web browser. The appliances that make the user interface function are the database server and the administration portal [5]. The administration portal can also be configured to be used by customers by creating different user policies to limit the user's visibility.

SL1 API is provided by the database and admin portal appliances [5]. It is used to integrate SL1 with external systems like ServiceNow, which is a very popular IT Service Management tool. SL1 API also enables the usage of possible custom scripts and pro-grams developed by the users of SL1.

Extended architecture utilizes new appliances to further improve monitoring capabilities and the scalability of an SL1 instance. As pictured in figure 1, it includes all the appliances from a Distributed instance, making it possible to upgrade from a Distributed instance into SL1 Extended. [7]

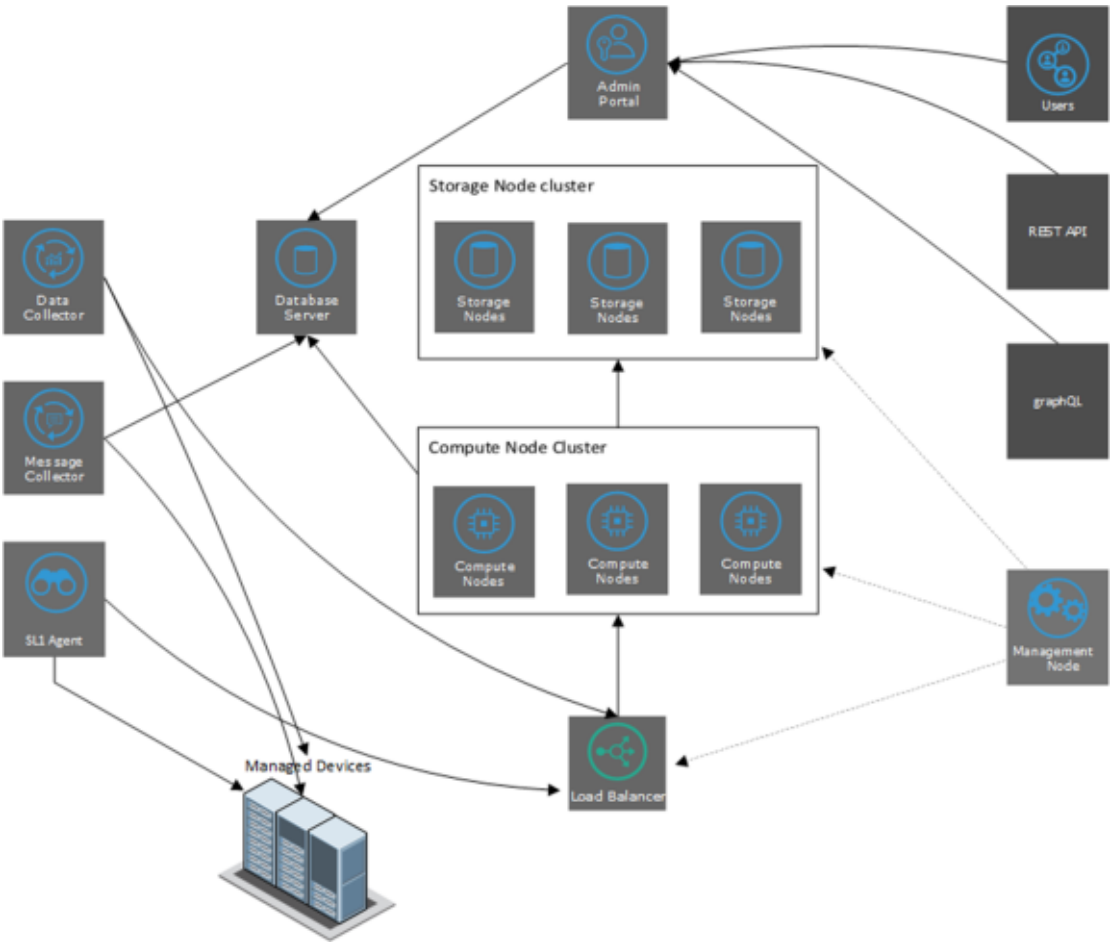

Figure 1. Architectural representation of a SL1 Extended instance.

Upgrading from Distributed instance into Extended instance brings four new appliances:

- Compute Cluster, helping the data and message collectors by processing the incoming agent and data collector data [7].
- Load Balancer, that manages compute clusters' requests and enables load-balancing [7].
- Storage Cluster, which will handle storing of data from agents and data collectors instead of the database server [7].
- Management Node, which is used to maintain and manage all the appliances provided by the Extended architecture [7].

Extended architecture also brings with it a new and improved version of the SL1 agent. The current agent version used in the Distributed instance requires some agent-less Dynamic Applications to collect all the configuration data needed for

CMDB (Configuration Management Database) data enrichment. The new and improved SL1 agent should eliminate the need to use agent-less Dynamic Applications for collecting configuration data making the implementation of monitoring to a new device quicker and more streamline.

1.3 Most important features and supported systems

Almost all data in SL1 is divided into organizations. For example, an organization in which the device belongs to, is specified during device discovery. A user-account can belong to one organization providing them visibility to the data in that specific organization, but they can also be granted additional visibility through user policies. Using organizations and user policies to section data prevents customers from seeing each other's confidential data.

User policies provide more in-depth customization of what users can access. A user policy can be assigned to multiple users. If a change is made to a user policy, the change is carried to all members of that policy. This keeps the user management pro-cess simple and streamline. If user accounts are imported from an AD (Active Directory) using LDAP (Lightweight Directory Access Protocol), a default user policy that is assigned to the users must be set [8].

Dynamic Applications provide monitoring capabilities for specific device types, systems, and vendors. Dynamic Applications run on the data collectors and use programming languages like Python and PowerShell for gathering information from monitored devices. In example a Dynamic Application of type PowerShell Config could be used to collect configuration data about the target devices RAM (Random-access memory). Users can develop their own Dynamic Applications, even if the target systems are not supported by SL1 out of the box. Dynamic Applications will be explained in more detail later.

Device templates are packages of monitoring settings and Dynamic Applications that can be customized to fit different device types. In this SL1 instance different devices have set templates containing monitoring settings and

Dynamic Applications that have been created to streamline the management and handling of monitored devices.

Templates can be used in multiple ways in SL1. In the discovery phase of a device, a template is used to specify the monitoring configurations and Dynamic Applications that SL1 will be using for that device. If a Windows or a Linux server would be discovered without a template, SL1 would not be able to identify which operating system the device is using. This identification is done by the Configuration Cache Dynamic Application. This Dynamic Application collects most of the configuration data from the device, including information about the Operating system, the CPU (Central processing unit), and the RAM. Device templates can also be used this way to disable some monitoring features that are enabled by default in the discovery phase. For instance, a device template can be used to disable monitoring on specific network interface ports on a network switch.

Device templates are also very useful when monitoring needs to be altered on a mass number of devices. For instance, if one process needs to be added into monitoring, a device template can be created and pushed to those devices from the Registry view of SL1. This is a very frequently used feature, as monitoring specifications change from often.

Automations, or as ScienceLogic calls them, Run Book Automations are automated policies that execute specified actions, or ScienceLogic calls them, Run Book Actions, if all the required conditions are met. The conditions defined consist of specific devices or device groups included in the automation, the event or schedule that triggers the automation, and finally the action which will be executed on the devices. Run Book Actions can execute a piece of Python code, send an email to specific addresses, or even send SNMP Traps from SL1 to external systems. The most used actions in this environment are the actions that enable the ticketing integration between SL1 and ServiceNow. [9, 10]

There are four types of automation policies that determine what triggers the automation. Most used types are active events, cleared events and scheduled. Active and cleared event automations search for specific events that meet the criteria and execute the action if these events are activate or cleared, where scheduled policies execute on a timed interval. [9]

ScienceLogic claims that traditional monitoring systems are fundamentally flawed as they cannot be extended beyond what is delivered out of the box. PowerPacks or integrations, are ScienceLogic's way of extending the out of the box monitoring capabilities of SL1. According to ScienceLogic's website they offer over 500 PowerPacks. These PowerPacks contain monitoring solutions built by the ScienceLogic community and ScienceLogic's internal developers. [11; 12.]

PowerPacks offer monitoring capabilities for a wide range of technologies, services, and applications. SL1 comes packaged with some PowerPacks, but users can also download them from ScienceLogic's website depending on the version on SL1 and contract the user's organization has with ScienceLogic.

PowerPacks come built with

- Dynamic Applications for data collection
- event policies for event generation
- device templates
- custom reports
- Run Book Actions and Automations for system automation
- dashboard views for system status monitoring.

As pictured in figure 2, they can also contain plenty of other features. These can vary based on the PowerPack and the monitoring features it offers.

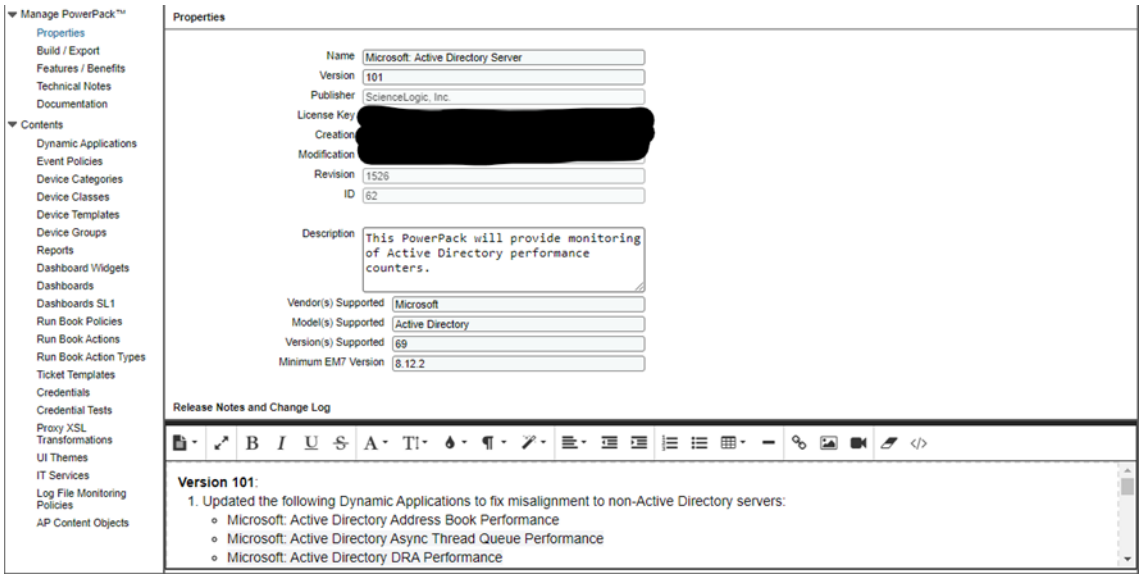

Figure 2. Picture of a Microsoft: AD Server PowerPack and its contents.

PowerPacks are not only used for bringing new monitoring features in to SL1. Out of the box, SL1 does not have a VCS (Version Control System). The absence of this feature can be cumbersome, as code changes to Dynamic Applications are a frequent occurrence in day-to-day operations. Because of the lack of VCS, PowerPacks can be used as backups, and to keep different environments monitoring capabilities up to date.

The contents provided by PowerPacks can be modified if needed, but it is not recommended. ScienceLogic releases updates for PowerPacks from time to time, and if modifications have been made to i.e., a Dynamic Application that came with the PowerPack, all the modifications will be erased when the pack is updated if Dynamic Applications are not protected. If the user wants to add extra features, it is recommended that they are added by making new Dynamic Applications that are not part of PowerPacks. In situations where a user has made modifications to Dynamic Applications in PowerPacks, a backup of the Dynamic Applications can be downloaded from the database, so no modifications made to the code are lost when updating.

SL1 can be integrated with other systems using the SL1 Integration Service that utilizes the SL1 REST (Representational state transfer) API for communication

over the HTTPS (Hypertext Transfer Protocol Secure) port 443 [13]. The Integration Service has different workflows to pass information between different systems [13]. As an example, the SL1 instance in question has been integrated with ServiceNow for Incident management and CMDB data integration and enrichment, and it utilizes workflows dedicated for ServiceNow integration. Pictured in figure 3 is a diagram of the SL1 and ServiceNow integration workflow.

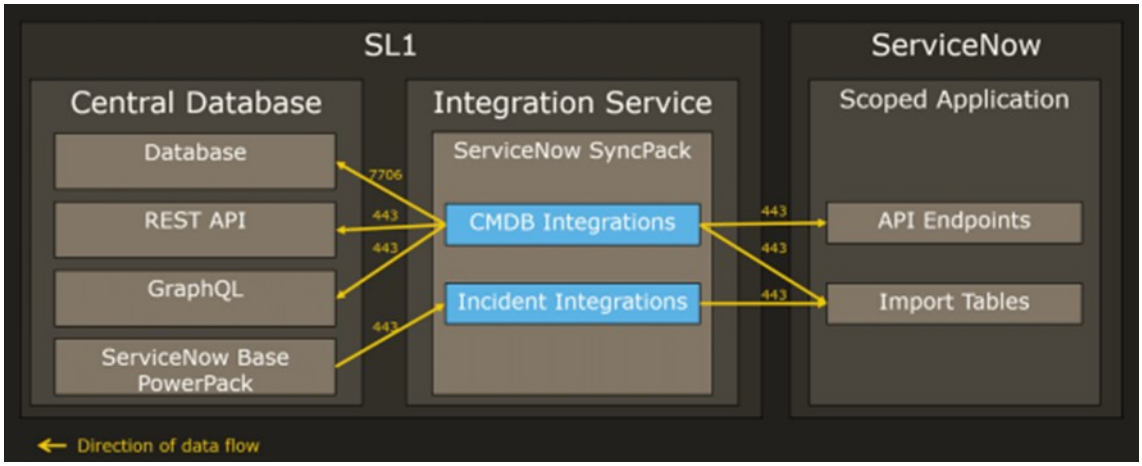

Figure 3. Incident and CMDB integration workflow between SL1 and ServiceNow.

This integration utilizes the forementioned Run Book Automations that take active and cleared events and send their data to ServiceNow/the Integration Service. The Integration Service does not only collect and send data itself. Data is also sent to it by SL1. ServiceNow does not send data to the Integration Service, instead data is only collected and sent by the Integration Service. [13]

2 Dynamic Applications

Dynamic Applications are customizable monitoring policies designed to monitor various types of systems and specific vendors. Dynamic Applications are categorized by two parameters, the archetype of the application and the protocol it uses for data collection. Dynamic Applications tell SL1 what data to collect, how to present it, and when to generate events from the collected data. [14]

For instance, a Dynamic Application that is defined to collect configuration data about a device's CPU might define that:

- The collected data should include the CPU's manufacturer's name, product name, socket count, and core count.
- The collected data will be added as custom attribute data on the device for data enrichment purposes.
- An event will be raised if data could not be collected.

SL1 comes bundled with Dynamic Applications for most common hardware and software systems. If SL1 does not contain Dynamic Application support for a specific technology, the user can create a custom one if they have the required knowledge. [14]

2.1 Dynamic Application archetypes

There are three different types of Dynamic Applications in SL1, Configuration Dynamic Applications, Performance Dynamic Applications and Journal Dynamic Applications. Each of these has a different purpose and is important in the device monitoring process. All Dynamic Application types can be used to raise events based on collected data.

Performance Dynamic Applications, as the name describes, can be used to present collected performance data in a historical graph format [14]. SL1 automatically calculates normalized data for these data graphs [14].

Performance data graphs can be exported as reports or used to analyse device capacity utilization if needed.

Configuration Dynamic Applications are used to collect data for various purposes like reports, CMDB enrichment and event enrichment and generation. SL1 will display collected configuration data in a table-based format in the device summary page for that device. Collected data can be added to the asset page or as custom attributes to be used by the SL1 Integration Service for external data enrichment and integrations. SL1 stores previously collected configuration data to show if the data has changed over time [14].

Journal Dynamic Applications display collected data in log format [14]. Collected data is stored in log entries that can contain several values to show if data has changed over time [14]. Journal Dynamic Applications are the least used type of the three Dynamic Application types.

2.2 Dynamic Application technologies

Dynamic Applications are grouped by the technology or protocol used for the data collection. There are ten types of technologies, most notable of which are:

- PowerShell Dynamic Applications used for Microsoft Windows monitoring.
- Snippet Dynamic Applications that use Python and can be used for countless device types.
- SNMP Dynamic Applications which use SNMP to retrieve data.

PowerShell Dynamic Applications utilize PowerShell commands to collect configuration data and performance data from Windows Servers and desktop devices. An example of PowerShell data collection using PowerShell would be a command that gathers specific data like the RAM amount of the device and writes it into the Windows PowerShell console. This value is then added as configuration data in SL1. Most of these commands can be executed on local desktop machines for testing when developing Dynamic Applications.

Snippet Dynamic Applications use Python, and they can be used to monitor many different device types depending on the application. Python has libraries that contain premade functions which help with developing applications to work with different technologies. Snippet Dynamic Applications are also the Dynamic Application type used for Linux based devices. Developing them is quite a bit more complex compared to their PowerShell counterparts as they require more programming understanding and experience. The working principle is generally the same as PowerShell Dynamic Applications, but instead of PowerShell commands Linux shell commands are used for data collection.

The vast amount of Linux distributions makes developing Dynamic Applications for them more challenging. The scripts must be built in such way, that they are compatible with all the Linux distributions they are used for. The most frequently used Linux directory for monitoring is the “/proc” directory. It holds all information about the Linux system, including its kernel, processes, and configuration parameters, and it is present on most if not all Linux distributions [15].

SNMP Dynamic Applications use the SNMP-protocol for polling data through SNMP OIDs (Object identifiers). SNMP is supported by most enterprise-tier network devices and server devices, and it guarantees that the same parameters are used for any category of device for cross compatibility purposes. SL1 uses SNMP Dynamic Applications for monitoring most network devices. [16]

3 Most monitored parameters

3.1 Availability and latency monitoring

Availability and latency monitoring are the most basic of monitoring functionalities provided by SL1. These can be monitored with multiple different protocols and technologies depending on the type of device and how the device is being monitored. In SL1, the default availability monitoring technique is determined by the method specified in the discovery of the device. The available technologies consist of

- TCP (Transmission Control Protocol)
- ICMP (Internet Control Message Protocol)
- SNMP
- UDP (User Datagram Protocol)
- SL1 agent. [17]

The protocol and port can be changed from the device properties. Listed in figure 4 are all the ports that have been discovered by SL1 and the data collected from the selected port will be used in the device availability reports [17].

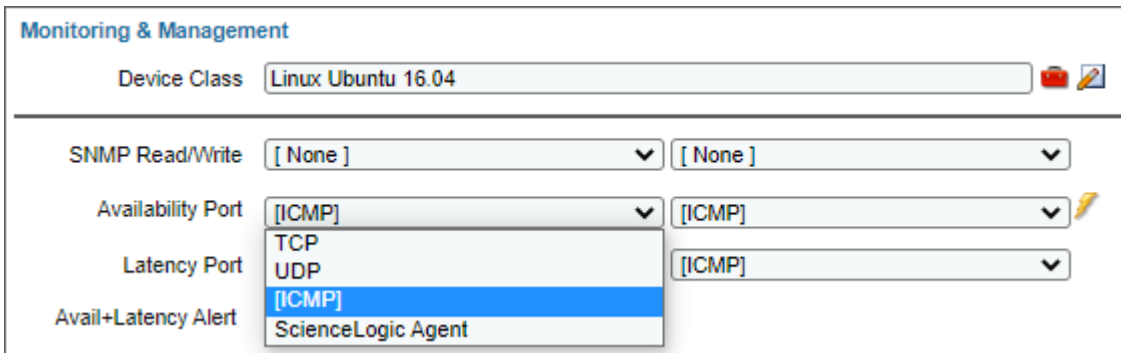

Figure 4. Available technologies used for availability monitoring.

SL1 does not monitor component device availability with TCP, ICMP or UDP. Instead, it uses Dynamic Applications that discover component devices related to physical devices. These Dynamic Applications poll the device for its name

and a unique identifier to determine the availability of the component device. If these are found by the Dynamic Application, the component device is determined to be available. [17]

By default, SL1 checks the device availability every five minutes. Critical Ping settings can be used to check the device availability as frequently as every five seconds [17]. This is especially useful for devices of high importance.

Latency is measured by initiating communication to a specified port on a device and measuring the time it takes for the device to respond to the request [17]. Latency is most often measured in milliseconds [17]. If ICMP is used to measure the latency of the device, a threshold can be set to trigger an event that specifies the latency threshold has been exceeded [17]. All the monitoring thresholds including ICMP availability thresholds pictured in figure 5 can be set to preferred values in the thresholds tab of device properties.

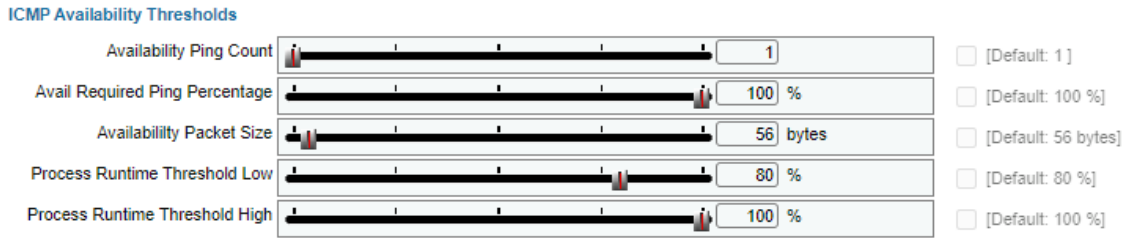

Figure 5. Device specific availability monitoring thresholds.

A good example of ICMP availability monitoring event would be one that triggers if the Availability Required Ping Percentage drops under the specified threshold. In the picture above the threshold is set at 100%. Meaning every single ping must go through.

3.2 Capacity monitoring

Capacity monitoring is a vital part of keeping your servers and network devices in a healthy working condition. It is done through Dynamic Applications that use different technologies based on the monitored devices operation system. High

values in CPU, RAM, and Disk Utilization can cause the system to function slowly or it can even cause system failure. Because of this capacity monitoring is highly utilized with different kinds of servers.

The change in capacity utilization can be followed through monthly reports or through dedicated dashboards which will be explained in detail later. The device performance tab can be used to view capacity utilization of a specific device. It also shows if a collector has not been able to collect data. This can be seen by the cuts in the performance graph and the missed polls count shown in figure 6.

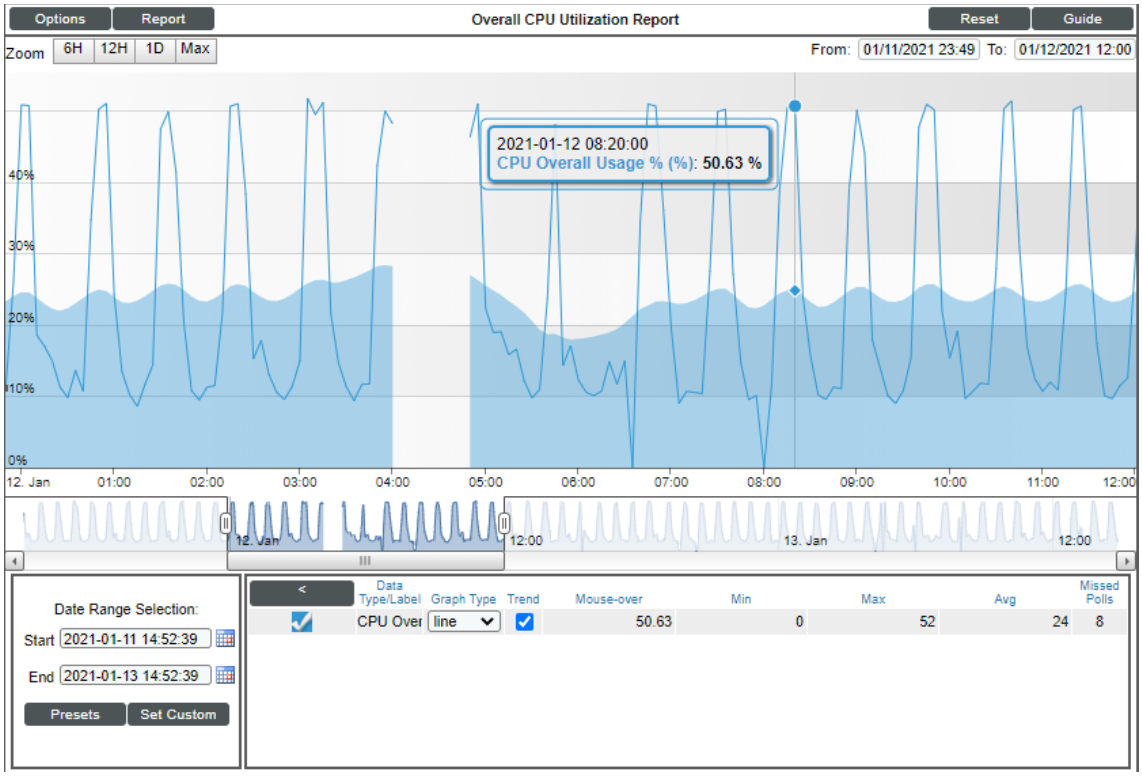

Figure 6. CPU Utilization Performance graph depicting missed polls.

PowerShell scripts and WMI (Windows Management Instrumentation) queries can be used to monitor Windows based devices. Most of SL1s Dynamic Applications for Windows Servers use PowerShell scripts, but there are some that do use WMI queries. For Linux based devices the monitoring is done by Snippet Dynamic Applications coded in Python that most often use shell scripts to collect the data.

3.3 Process and Windows service

Processes or tasks are running or idle programs on a device [18]. All computer programs have one or more processes and monitoring the availability, memory usage and the amount of these processes can prove very beneficial. Processes are part of both Windows and Linux operating systems, and they usually have a user interface, where services do not. On Windows machines processes are called tasks.

SL1 allows processes to be monitored through process monitoring policies. These policies can monitor the availability, memory usage and the amount of process instances. Depending on the monitoring solution this data will be collected through the SL1 agent, SNMP, or an internal collection Dynamic Application in five-minute intervals [18]. Process monitoring policies can be enabled from the device summary's process tab.

Windows services are applications that run in the background of Windows servers and computers. These services can be automatically launched on device boot up, can be paused, restarted on command, and contain no user interface [19]. This makes them very convenient for servers and workstations used by multiple people, as the running services will not be interfering with the users operating the servers [19].

SL1 monitors services through Windows service monitoring policies. These policies allow the monitoring of service availability as well as offer some advanced functionality by providing the ability to start, pause or restart the service, reboot, or shut down the device, or run a script in case of the service goes unavailable [19]. Error messages produced by Windows services are usually written in the Windows Event Log that can be monitored separately by log monitoring policies described later [19].

3.4 Additional monitoring technologies

There are endless ways to use SL1 for monitoring, but the techniques that are worth mentioning consist of

- web content monitoring
- directory monitoring
- file monitoring
- log file monitoring.

Web content monitoring can be used to check if a website is correctly working IE. by monitoring the response time of the website, checking for correct HTTP (Hypertext Transfer Protocol) status codes, monitoring the returned page size, or monitoring the overall transaction time. The availability of a website can be monitored by polling the websites URL (Uniform Resource Locator) for HTTP status code 200, which means “OK”. If the returned status code is for example 500 which translates to Internal Server Error, an event is raised by SL1.

Directory monitoring and file monitoring have been custom made for this instance of SL1. Directory monitoring is used to monitor the size, age, and last write time of a Windows or a Linux file directory or a folder. Directory monitoring is useful IE. for monitoring a directory that is used to transfer files between two servers. Similarly, file monitoring can be used the same way but for individual files. File monitoring is especially useful for devices that generate large log files that can fill the system drive if not dealt with early.

A log file is usually generated by a running software or an operating system. Log files can be used for multiple different purposes IE. for debugging system errors or keeping a log for auditing purposes. Log file monitoring is used to monitor a log file for a specific string by using a regular expression otherwise known as regex. Regex consists of a string of characters, that can be used to search a file for letters, words, or even whole sentences. Log monitoring can even be used on multiple log files at the same time using the same policy, by specifying the file path of the log files and using the asterisks symbol. Figure 7

shows an example of a log file monitoring policy that detects error and fatal messages from multiple log files.

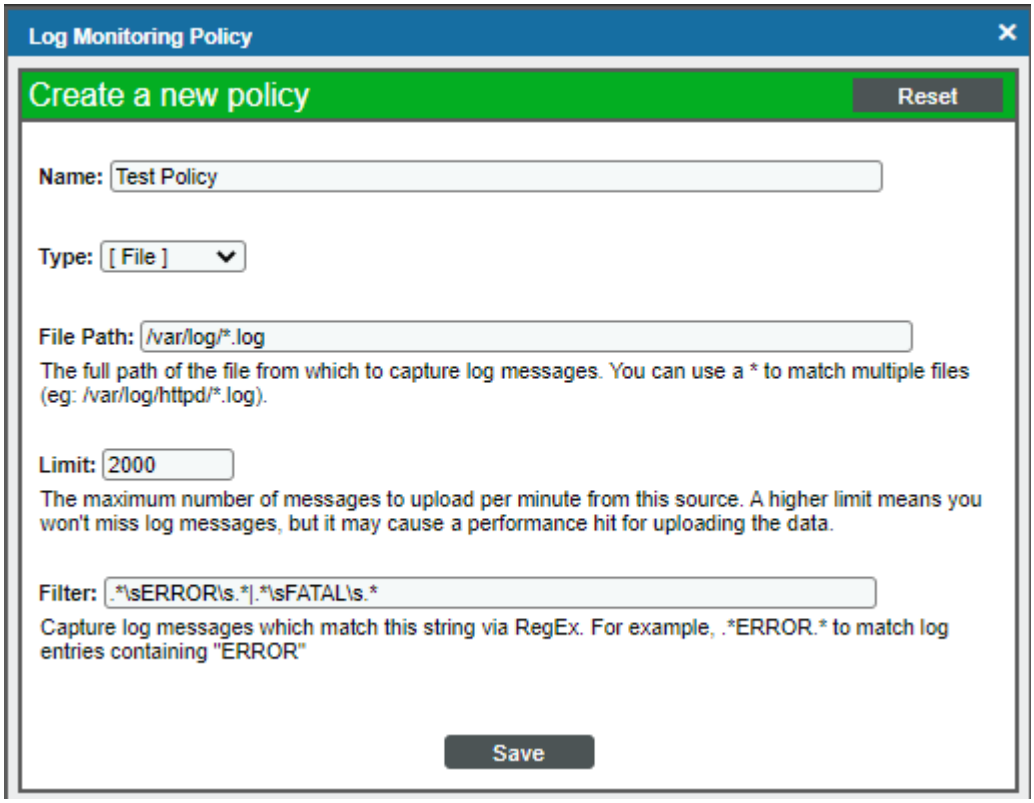

Figure 7. An example of a log file monitoring policy settings.

For Windows servers, the log file monitoring requires an agent to be installed on the device. The logs are sent to a SL1 message collector and then shown on the Logs tab in the Device Summary and Properties pages. A generic log file monitoring policy could a one that searches the Windows Event Logs for the string “ERROR” and raises an event when a match has been found. For the event to be raised from log file monitoring policies, an event policy must be created through the Event Policy Manager discussed later.

4 Events, dashboards, and reports

4.1 Event structure and criticality levels

SL1 events are created using the Event Policy Manager. There are seven different types of event policies in SL1. The type field specifies the source or protocol of the message for the event policy. Listed below are the available types:

- Syslog. Syslog protocol is a log format used by Unix-like systems [20].
- Internal. A message raised by SL1 that concerns the platform itself [20].
- Trap. A message caused by an SNMP Trap [20].
- Dynamic. A message generated by an Alert that is raised by a Dynamic Application [20].
- Email. A message caused by an email sent to SL1 [20].
- API. A message that is generated using the SL1 API [20].
- SL1 agent. A message generated from SL1 agents log files collected by a message collector [20].

Including the type of the event policy, there a many more important parameters that must be set when creating an event policy. These include the severity of the event that can be set as Critical, Major, Minor, Notice and Healthy. These severity levels are also identified by their corresponding colours Red, Orange, Yellow, Blue, and Green. What each severity level symbolises is up to the system owner. For the instance in question, incidents are usually only generated from specific critical and major events.

Event policies contain plenty of useful settings, including occurrence count and time. Occurrence count and time can be used together to specify a delay for raising events when a threshold is exceeded. Let us say there is a device that goes offline from time to time but is never offline for more than 2 hours. There is ping monitoring on this device that polls at a frequency of 10 minutes. The event

policy can be set to trigger only when the event has occurred 13 times in 2.5 hours for example.

Another case that utilizes the occurrence count and time would be filtering out unnecessary events. This is useful when dealing with data that changes quickly and is polled often, like CPU utilization percentage. Very often CPU utilization can peak at high percentages and quickly return to normal values. Occurrence count and time can be set to trigger and event only when CPU utilization exceeds the specified threshold several times in a row. This eliminates false alarms.

From the event console users can see all the events active on the devices based on their user policies. Events are segregated data. Meaning different customers cannot see each other's events. The event console view is customizable and provides plenty of information about the events, including the event message, organization of the affected device, severity of the event, occurrence count, elapsed time, and last timestamp when the event was detected.

4.2 Dashboards

Dashboards are highly customizable views that provide valuable information gathered by SL1. A dashboard usually consists of

- a device selection list
- a timespan selector
- utilization graphs
- active events list.

Dashboard can be used to monitor countless things. For example, a dashboard that shows a specific systems devices and active events can be very useful for diagnosis when the system is experiencing performance issues. From the dashboard you can quickly get an idea of all the active events and capacity utilization spikes on that specific system, compared to searching for this

information from multiple different places like the event view or the device registry view.

There is a useful dashboard for SL1 administrators that shows the health of all the appliance devices used by SL1. In a large environment there is usually multiple collectors, databases, and application components, and checking their status from time to time is important in preventing system failures. Dashboards can also be scheduled to send a snapshot of the current state of the dashboard view to specified emails. This can be used as a form of reporting.

4.3 Reports

Reports are a vital part of monitoring. Users frequently generate reports from groups of devices that are part of a system or devices that run a specific software. An example report can show last month's statistics of RAM, CPU and Swap usages in a graph format like the one seen in figure 8, or an availability report of all the Microsoft AD Domain Controllers that are monitored by SL1.

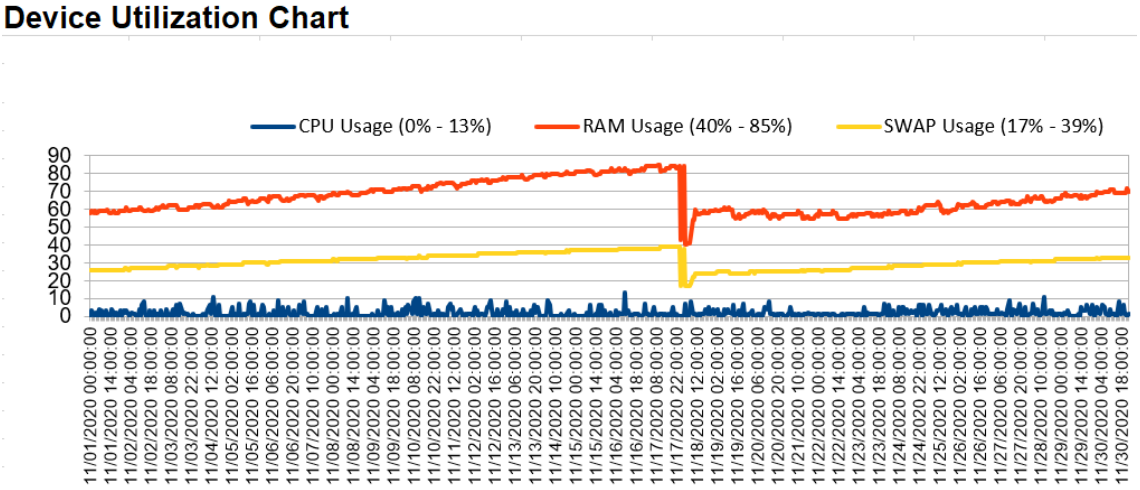

Figure 8. Capacity utilization report generated by SL1 reports page.

Reports enable users to detect anomalies in their system by comparing statistics from a certain timeframe. If the RAM usage seems to be gradually increasing, maybe there is a memory leak that needs to be fixed, or maybe the server just requires more RAM because of increased traffic.

There are two types of reports in SL1 which are Custom Reports and Embedded Reports. SL1 has hundreds of predefined custom reports that can be run on their own or modified to a specific need. A custom report usually consists of

- a device selector
- timeframe selector
- output format.

These reports can be run on demand, or if there is a frequent need of specific reports, they can be run based on custom schedules. A scheduled report can then be sent to users by email automatically. Even though SL1 comes with many custom reports out of the box, there still is room for improvement. Only few reports provide an option to present the data in a graphical format. Reports can be custom made by the user, but they are very difficult to make and require extensive knowledge.

Embedded reports can be generated from numerous pages in SL1. Embedded reports take the current view of information displayed and generate an excel, PDF (Portable Document Format) or an ODS (OpenDocumentation Spreadsheet) file of that page. If the information is shown in a list format and the page contains SL1's list filters, they can be used to format the embedded report beforehand.

5 Topology maps

A topology map is a view that shows the relationships between a device's related nodes that SL1 has discovered. Classic SL1 maps use Adobe Flash to render the map image, but a new version of this view uses HTML5 (Hypertext Markup Language 5). A good example of a topology map is one that describes a VMware environment, and the relationships between all the different nodes.

Topology maps can be created by the user to i.e., describe the geological relationships of network devices or to show devices that are important to the user. They can also be customized to show text and pictures providing more control of the map.

There are four different types of topology maps in SL1,

- Layer-2 Maps
- CDP (Cisco Discovery Protocol) Maps
- LLDP (Link Layer Discovery Protocol) Maps
- and Layer-3 Maps

In Layer-2 networks, or unrouted networks, devices are known by their MAC (Media Access Control) addresses, unlike devices belonging in a routed network that are known by their IP (Internet Protocol) addresses. These devices use Ethernet and Mac addresses to communicate with devices in the same LAN (Local Area Network) or WAN (Wide Area Network). [21]

CDP maps are topology maps created using the CDP protocol. CDP is a Layer-2 protocol that enables the discovery of Cisco hardware. The protocol also the communication between different Cisco hardware devices in the same LAN or WAN. The information transferred between the devices includes operating systems, MAC addresses and IP addresses and information about network interfaces. CDP protocol is not only for Cisco brand devices. It can also operate on some legacy Hewlett Packard based devices. [21]

LLDP maps are topology maps created by using the LLDP protocol. LLDP is a Layer-2 protocol that enables the discovery and communication of Cisco hardware [21]. The information transferred between the devices includes operating systems, MAC addresses and IP addresses and information about network interfaces [21]. LLDP is very similar to the forementioned CDP, but the core difference is that LLDP is a standard while CDP is Cisco's own proprietary protocol [22]. In most cases a Cisco device cannot locate a non-Cisco device using CDP. Therefore, they must use LLDP to accomplish that [22].

In Layer-3 networks, or routed networks, devices are known by their IP addresses. Layer-3 maps describe the relationships between SL1 data collectors and layer-3 network routers, and switches discovered by SL1. Layer-3 maps do not only show devices discovered by SL1, but they also show other devices in the same network by running traceroute to all devices that have layer-3 collection enabled. [21]

6 Monitoring servers

6.1 Windows monitoring with agent or agentless

When monitoring servers, the user usually decides between agent based monitoring and agentless monitoring. The decision may vary depending on the monitoring requirements or the user's preferences, but there are pros and cons to both technologies.

Agentless monitoring provides a detailed outside-in view of the monitored device [23]. In the instance in question, there are hundreds of Dynamic Applications that can be used to monitor servers remotely. The data will usually be polled with PowerShell on Windows machines or with Python scripts executing shell commands on Linux based machines.

Agentless monitoring puts more stress on the monitored device, as it is constantly establishing new connections to the monitored device and executing commands. This is caused by the number of Dynamic Applications and the configured polling frequency.

A SL1 Agent is a program that is installed on the monitored device. There are two versions of the SL1 agent, Generation 1, and Generation 3. Generation 1 is the agent version that is used in distributed SL1 environments, and it is the one used in this SL1 instance. Generation 3 agent is used in SL1 Extended Architecture instances. Figure 9 describes the process of data collection done by the generation 1 agent and all the SL1 appliances in a distributed SL1 environment.

Figure 9. How the Generation 1 SL1 Agent sends data to SL1.

Generation 1 agent sends collected data directly to the message collectors, while Generation 3 agent sends collected data through the streamer service running on the Extended Architecture's Compute Node cluster [24]. Below is a comparison of agent based and agentless monitoring solutions.

Table 1. Comparison table for Agent based and Agentless monitoring solutions.

	Agent based	**Agentless**
Configuration data	Only provides basic configuration data.	Can be customized with user created scripts to provide vast amounts of configuration data.
Performance data	Only provides basic performance data.	Can be customized with user created scripts to provide vast amounts of performance data.
Log monitoring	Provides easy integrated log monitoring.	Can be configured through syslog forwarding on Linux.
Process monitoring	Provides process monitoring.	Can be added if needed.
Service monitoring	No service monitoring.	Can be added if needed.

Dynamic Application support	Requires agentless connection.	All relevant Dynamic Applications can be used with proper configuration.

Agent based solution provides basic monitoring capabilities with relatively easy installation process and management. However, it lacks in customizations capabilities and some core monitoring capabilities like service monitoring. The absence of rich configuration data and performance data is also a major problem. As this is needed to keep CMDB data up to date. On the other hand, agentless monitoring is highly customizable, harder to enable, but stresses the target device more with constant polling.

By utilizing the low performance hit of agent-based monitoring to gather the basic performance, configuration data, logs and process data and using agentless monitoring to gather missing data for CMDB enrichment and more advanced monitoring needs, it was decided after multiple tests comparing the two solutions, that a combination of the two technologies was the best option. In an ideal situation, the SL1 agent would collect all the needed configuration data, making the installation process very easy and less demanding, but unfortunately this is not the case currently. Using a combination of the two technologies for monitoring makes the installation process a tad more complex, but with automation this complexity can be reduced.

6.2 Monitoring Microsoft Windows servers

The task is to enable basic monitoring on one Windows Server using a combination of the SL1 agent and agentless monitoring. The basic monitoring for this device will include devices resource monitoring and the windows system event log monitoring.

Before enabling monitoring on the Windows server there are some preparations that must be done. SL1 Data collector must be able to connect over port 53 to

the domains DNS server. This enables the availability of forward and reverse DNS lookups from the monitored Windows server. [25]

If monitoring is done unencrypted the port 5985 must be open on the target Windows server for HTTP traffic. Alternatively, if monitoring is done encrypted the port 5986 must be opened for HTTPS traffic. These ports might differ depending on the user's configurations, but the forementioned ports are the default ports for HTTP and HTTPS traffic. [25]

Encrypted monitoring requires a certification to be installed on all the target Windows servers to enable SL1 to connect with WinRM (Windows Remote Management) [25]. A choice can be made between a self-signed certificate or a certificate from a CA (Certificate Authority). Self-signed certificates are easier to install and require no monetary investment. The downside of self-signed certificates is that they do not have all the security benefits of CA certificates, but self-signed certificates work fine for this purpose [26].

The credentials used for monitoring should be added in to SL1. SL1 uses the Kerberos network authentication protocol if the credentials used are based on an AD account. Port 88 on the target domain's AD Domain Controller must be open in this case, so SL1 can authenticate using Kerberos. If multiple domains are used, they must be configured into the Kerberos configurations files on all the Data Collectors that are monitoring devices from the target domains. [25]

A local or an AD user account must be created and configured on the Windows Server for SL1 monitoring. For ease of use, ScienceLogic recommends that an AD user account is created, that is part of the local administrators group. For local use accounts it is recommended that the user account is part of the Administrators group. However, creating user accounts with administrator privileges is not always possible. A non-administrator domain or local account must be created with specific privileges. They must be granted remote access to the Windows server using the steps specified in Appendix 1. Target servers

Windows Remote Management must also be configured using ScienceLogic's recommended settings found in Appendix 2. [25]

It is recommended that the number of allowed PowerShell connections is increased through the servers WinRM configurations. Problems may arise if the number of allowed PowerShell connections is lower than the amount of PowerShell based Dynamic Applications that have been set to poll data from the server. This number can be increased with couple of PowerShell commands. The recommended number of connections can be calculated by taking the number of PowerShell based Dynamic Applications and multiplying it by 3. [25]

A SL1 Agent installation can be automated using custom scripts, but if monitoring is being installed on a small group of devices, manual installation is also viable. To install an agent a file can be downloaded from the device registry page. When downloading the agent, the user is asked to select a message collector and the organization for the device. This selection decides which message collector will be receiving the Agents data and in which organization the server will be put in. These selections only alter the installation command, and do not affect the agent file itself. This means the same agent file can be used on servers in different organizations and networks.

Once the SL1 agent file has been acquired, the installation may begin. Easiest way to install the agent on a single machine is to use a remote desktop software like Windows RDP (Remote Desktop Protocol) to log in to the target server using the IP-address or host name of the server and credentials that have permissions to install applications. The installation file may be transferred on to the server using multiple methods. Fortunately, Windows RDP supports file copy pasting so that will be the method used here.

After the file is transferred to the device, the provided installation command is executed as an administrator in CMD (Windows Command Prompt) or PowerShell. To confirm the installation executed successfully, the devices

running processes were checked for the SiloAgent process. Additionally, after the agent has made a connection to the defined message collector, the device will be discovered automatically in to SL1. It is possible to discover the device in to SL1 before installing the SL1 agent but doing this will cause a duplicate device to appear in the systems registry once the agent is installed. This can be easily fixed by deleting the entry from SL1 that does not contain an agent.

After the discovery event has triggered on the device, the defined Run Book Automations will push all the necessary configuration Dynamic Applications and monitoring settings to the device after the discovery. If for some reason this does not happen, the configurations can be pushed manually by utilizing the modify by template action in the device registry and pushing the correct template to the device.

Once all the monitoring configurations are set, the credentials used for the devices Dynamic Applications should be confirmed to be correct. This can be checked from the collections tab of device properties. If incorrect credentials are set, SL1 will not be able to collect configuration data with the configuration Dynamic Applications.

To verify the configured credentials work, the Dynamic Application called “Windows Server: Configuration Cache” should be run from the lightning bolt symbol next to it. This action executes a manual collection session and logs all the information into the console that is opened. Easy way to identify a successful collection is to check if the log has any PowerShell errors at the start. Also, the collection session will print all the collected data at the end. If there are error messages at the start of the log and collected data only shows empty data rows, the collection has likely failed.

Enabling system event log monitoring can be done by pushing a template that has the correct log monitoring policies specified, or by manually enabling the policies from the log monitoring view. For some reason pushing the policies to

the device by template was not working, so the monitoring was done from the log monitoring view. After this the monitoring installation process was finished.

6.3 Monitoring Linux servers

Linux monitoring will be implemented only through agentless monitoring. Using agentless monitoring on Linux devices has proven easier and it is more widely used in this SL1 instance. The task is to enable capacity monitoring and process availability monitoring for this device. Prerequisites for monitoring Linux Servers is to have the IP address of the target device, a SSH (Secure Shell Protocol) credentials with a private key or a password used as login information [27]. In Addition, ifconfig must be installed on the device to collect information about the devices network interfaces [27].

All the commands used by SL1's Linux monitoring Dynamic Applications are listed in the documentation provided by ScienceLogic on their website. Most of these commands do not need special permissions to execute, but there are few that do. Password-less sudo permissions should be configured for the user used for monitoring, as this is required for the execution of dmidecode command that prompts a password prompt when used [27]. Failing to configure this correctly will prevent the Linux: Hardware Configuration Dynamic Applications from collecting data [27].

For SL1 to be able to discover and monitor the Linux device over SSH a key credential needs to be created in the systems credential management page. This credential is used in the discovery phase and in all the Dynamic Applications used for monitoring that device. The default TCP port used for SSH servers is the port 22, and this information is needed for the credentials as well. [27]

As the SL1 agent is not going to be used for Linux monitoring, the next step after the preparations is to discover the device using SL1's device discovery page. The discovery requires the following parameters to be set,

- IP address of the device
- SSH credentials
- SNMP or non-SNMP discovery setting
- destination organization
- data collector for discovery
- and a monitoring template.

Once the information required for discovery has been filled, the discovery session can be started by clicking the lightning bolt icon in the discovery session list. During the discovery SL1 will open a discovery window that will show the log for the session. This window will show if the discovery is successful, requires additional actions or has failed. A typical error message could describe that the connection could not be established possibly due to incorrect settings on the devices side. In this situation the prerequired configurations should be checked. A successful discovery will inform that the device has been modelled and discovered.

After the device has been discovered by SL1, the device credentials used for the Dynamic Applications are switched from default SNMP credentials to the SSH credentials created before. This is done because for some reason SL1 does not always update the default credentials to the ones used in the discovery session. After the credentials have been set, they are tested by running the Linux: Configuration Cache Dynamic Application. By running this test, we can be certain that the credentials are working as expected. All errors encountered will be shown in the log window opened during the test run.

The last step is to enable process monitoring for chosen processes. This is done from the process tab on the device summary page. The process tab lists all the processes found by SL1, and monitoring is enabled by clicking the plus sign next to the desired process listing. This opens the monitoring modal window for the process, where the monitoring parameters are set. SL1 can monitor the process availability, memory usage, CPU usage, and count. In this case we are only interested in monitoring the availability of the process. After this step the monitoring installation process is completed.

7 Summary

In conclusion SL1 is an OK all-around monitoring platform with loads of customizability and useful features. However, there are features that could use some polishing like the reporting capabilities. Out of the box there are only few reports that provide the data in a graphical format which is easier to internalize. Most reports only spew the data out into a table which is not very easy to look at and does not look very good. There are also some parameters that cannot be generated into reports currently, I.E process capacity usage report.

In SL1 some UI elements could be implemented differently for smoother workflow, like the drop-down credential menus in the device properties collection tab and the Dynamic Application drop down menu used for template modification. When an instance has thousands of credentials and Dynamic Applications, finding the one you are searching for in these menus is very tedious as there is no search function.

Installing monitoring on Windows and Linux servers went quite smoothly when proper preparations were done. ScienceLogic also provides quite extensive documentation on their website which has shown to be very useful when working with SL1. During the installation process there were only minor preparation errors with the Linux credentials that were easily found by referring to SL1’s documentation and running the credential test to check the error messages provided.

In the future once our instance is upgraded into the Extended version, it will be very exciting to test the new Generation 3 SL1 agent for Windows devices. If it does work as promised, the monitoring installation process should become easier, as less agentless Dynamic Applications will be needed. Also, the performance penalty of monitoring should improve, as the new agent allegedly only uses one PowerShell session for data collection compared to the old one which uses one PowerShell session per collection object.

References

1 ScienceLogic Executive Leadership. 2021. Online document. ScienceLogic. <https://sciencelogic.com/company/leadership>. Read 18.4.2021.

2 Kellogg: Fortune 500 Cereal Company Uses ScienceLogic to Monitor Hybrid Cloud Deployment. 2021. Online document. ScienceLogic. <https://sciencelogic.com/product/resources/kellogg>. Read 18.4.2021.

3 Liberty delivers a seamless user experience and decreases IT incidents by 98%. 2021. Online document. ScienceLogic. <https://sciencelogic.com/product/resources/sl-liberty-customer-story>. Read 18.4.2021.

4 AIOps – Artificial Intelligence for IT Operations. 2021. Online document. ScienceLogic. <https://sciencelogic.com/solutions/aiops>. Read 18.4.2021.

5 Architecture Overview. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_General_Information/Architecture/architecture_overview.htm>. Read 18.4.2021.

6 Distributed Architecture. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_General_Information/Architecture/architecture_distributed.htm>. Read 18.4.2021.

7 Extended Architecture. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_General_Information/Architecture/architecture_extended.htm>. Read 18.4.2021.

8 Managing Organizations. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_Admin_and_Accounts/Organizations_and_Users/orgs_and_users_manage_org.htm>. Read 18.4.2021.

9 Automation Policies. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_Events_and_Automation/Run_Book_Automation/automation_policies.htm>. Read 18.4.2021.

10 Action Policies. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_Events_and_Automation/Run_Book_Automation/action_policies.htm>. Read 18.4.2021.

11 ScienceLogic PowerPacks. 2021. Online document. ScienceLogic. <https://sciencelogic.com/product/resources/powerpacks>. Read 18.4.2021.

12 ScienceLogic Integrations. 2021. Online document. ScienceLogic. <https://sciencelogic.com/product/integrations#sort=relevancy>. Read 18.4.2021.

13 Installing and Configuring the Integration Service. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/8-12-1/Content/Web_Content_Dev_and_Integration/IS_Platform/is_platform_installation.htm?TocPath=Content%20Development%20and%20Systems%20Integration%7CIntegration%C2%A0Service%20Platform%7CInstalling%20and%20Configuring%20the%20Integration%20Service%7C_____6>. Read 18.4.2021.

14 Introduction to Dynamic Application Development. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_Content_Dev_and_Integration/Dynamic_Application_Development/dynamic_app_dev_intro.htm>. Read 18.4.2021.

15 Discover the possibilities of the /proc directory. 15.2.2008. Online document. Linux. https://www.linux.com/news/discover-possibilities-proc-directory/. Read 18.4.2021.

16 Introduction to SNMP Dynamic Applications. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_Content_Dev_and_Integration/SNMP_Dynamic_App_Development/snmp_apps_introduction.htm>. Read 18.4.2021.

17 Monitoring Device Availability and Device Latency. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/8-12-1/Content/Web_Monitoring_Tools/Device_Management/availability_and_latency.htm>. Read 18.4.2021.

18 Monitoring Processes Using an Agent. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_Monitoring_Tools/Infrastructure_Health/monitoring_processes_agent.htm?TocPath=Monitoring%20Tools%7CDevice%20Infrastructure%20Health%7CMonitoring%20Device%20Processes%20Using%20an%20Agent%7C_____9>. Read 18.4.2021.

19 Introduction to Windows Service Applications. 30.3.2017. Online document. Microsoft. <https://docs.microsoft.com/en-us/dotnet/framework/windows-services/introduction-to-windows-service-applications>. Read 18.4.2021.

20 Defining and Editing Event Policies. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_Events_and_Automation/Events/event_policies.htm>. Read 18.4.2021.

21 Topology Maps. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_Data_View_and_Reporting/Views/topology_maps.htm>. Read 18.4.2021.

22 Cisco Discovery Protocol (CDP) and Link Layer Discovery Protocol (LLDP) in Data Link Layer. 27.8.2019. Online document. GeeksforGeeks. <https://www.geeksforgeeks.org/cisco-discovery-protocol-cdp-and-link-layer-discovery-protocol-lldp-in-data-link-layer/>. Read 18.4.2021.

23 Agent or Agentless Monitoring? Why You Need Both. 5.3.2020. Online document. ScienceLogic. <https://sciencelogic.com/blog/agent-vs-agentless-monitoring-why-you-need-both>. Read 18.4.2021.

24 Introduction to the SL1 Agent. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_Monitoring_Tools/Agent_Monitoring/introduction_to_agent.htm?TocPath=Monitoring%20Tools%7CAgent%20Monitoring%7CIntroduction%20to%20the%20SL1%20Agent%7C_____4>. Read 18.4.2021.

25 Configuring Windows Servers for Monitoring with PowerShell. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_Vendor_Specific_Monitoring/Windows_PowerShell/chapter_03_config_PowerShell.htm>. Read 18.4.2021.

26 Self-Signed Certificates Can Be Secure, So Why Ban Them?. 3.11.2017. Online document. McAfee. <https://www.mcafee.com/blogs/other-blogs/mcafee-labs/self-signed-certificates-secure-so-why-ban/>. Read 18.4.2021.

27 Monitoring Linux with SSH. 2021. Online document. ScienceLogic. <https://docs.sciencelogic.com/latest/Content/Web_Vendor_Specific_Monitoring/Linux_and_Solaris/linux_configure_ssh.htm>. Read 18.4.2021.

Creating a Non-Administrator User Account

If you do not have Local Administrator access to the servers that you want to monitor with PowerShell or WinRM, or if the monitored Windows server is a Domain Controller that will not be in the local Administrators group, then you must first create a domain user account or create a local user account on the Windows Server. For instructions, consult Microsoft's documentation.

After creating your domain user account or local user account:

- You must configure the Windows servers to allow that non-administrator user access. To do so, follow the steps in this section.
- If you use SL1 to monitor Microsoft Exchange Servers, you must also configure the user account for remote PowerShell access to Microsoft Exchange Server.
- If you use SL1 to monitor Hyper-V Servers, you must also configure the user account for remote PowerShell access to the Hyper-V Servers.

To configure Windows Servers to allow access by your non-administrator user account:

1. Start a Windows PowerShell shell with Run As Administrator and execute the following command:

    a. winrm configsddl default

2. On the Permissions for Default window, click the Add button, and then add the non-administrator user account.

3. Select the Allow checkbox for the Read (Get, Enumerate, Subscribe) and Execute (Invoke) permissions for the user, and then click OK.

4. Access the Management console. To do this:

a. In Windows Server 2008, click Start, right-click Computer, click Manager, and then expand Configuration.

b. In Windows Server 2016 and 2012, right-click the Windows icon, click Computer Management, and then expand Services and Applications.

5. Right-click on WMI Control and then select Properties.

6. On the WMI Control Properties window, click the Security tab, and then click the Security button.

7. Click the Add button, and then add the non-administrator user or group in the Select Users, Service Accounts, or Groups dialog, then click OK.

8. On the Security for Root window, select the user o group just added, then in the Permissions section at the bottom of the window, select the Allow checkbox for the Execute Methods, Enable Account, and Remote Enable permissions.

9. Under the Permissions section of the Security for Root window, click the Advanced button.

10. In the Advanced Security Settings window, double-click on the user account or group you are modifying.

11. On the Permission Entry window, in the Type field, select Allow.

12. In the Applies to field, select This namespace and subnamespaces.

13. Select the Execute Methods, Enable Account, and Remote Enable permission checkboxes, and then click OK several times to exit the windows opened for setting WMI permissions.

14. Restart the WMI Service from services.msc.

15. To open services.msc, press the Windows + R keys, type "services.msc", and then press Enter.

16. In the Management console, go to System Tools > Local Users and Groups > Groups.

17. Right-click Performance Monitor Users, and then select Properties.

18. On the Performance Monitor Users Properties window, click the Add button.

19. In the Enter the object names to select field, type the non-administrator domain user or group name, and then click Check Names.

20. Select the user or group name from the list and then click OK.

21. In the Performance Monitor Users Properties window, click OK.

22. Perform steps 16-20 for the Event Log Readers user group and again for the Distributed COM Users user group, the Remote Management Users user group, and if it exists on the server, the WinRMRemoteWMIUsers__ user group.

23. If you intend to use encrypted communications between the SL1 collector host and your monitored Windows servers, each Windows server must have a digital certificate installed that has "Server Authentication" as an Extended Key Usage property. You can create a self-signed certificate for WinRM by executing the following command:

    a. $Cert = New-SelfSignedCertificate -CertstoreLocation Cert:\LocalMachine\My -DnsName "myHost"

24. Add an HTTPS listener by executing the following command:

    a. New-Item -Path WSMan:\LocalHost\Listener -Transport HTTPS -Address * -CertificateThumbPrint $Cert.Thumbprint –Force

        i. This command should be entered on a single line.

25. Ensure that your local firewall allows inbound TCP connections on port 5986 if you are going to use encrypted communications between the SL1 collector(s) and the Windows server, or port 5985 if you will be using unencrypted communications between the two. You may have to create a new rule on Windows Firewall if one does not already exist.asd

Manually Configuring Windows Remote Management

To configure a Windows server for monitoring via PowerShell directly, perform the following steps:

1. Log in to the server with an account that is a member of the local Administrators group, or a Domain Administrator's account if on a Windows server with the Domain Controller role installed.

2. Right-click on the PowerShell icon in the taskbar or the Start menu and select Run as Administrator.

3. Execute the following command:

    a. Get-ExecutionPolicy

4. If the output is "Restricted", execute the following command:

    a. Set-ExecutionPolicy RemoteSigned

5. Enter "Y" to accept.

6. Execute the following command:

    a. winrm quickconfig

7. Enter "Y" to accept.

8. If you are configuring this Windows server for encrypted communication, execute the following command:

    a. winrm quickconfig -transport:https

9. Enter "Y" to accept.

10. Execute the following command:

    a. winrm get winrm/config

The output should look like this (additional lines indicated by ellipsis):

```
Config
  ...
  Client
    ...
    Auth
      Basic = true
      ...
      Kerberos = true
      ...
    ...
  Service
    ...
    AllowUnencrypted = false
    ...
    DefaultPorts
      HTTP = 5985
      HTTPS = 5986
    ...
    AllowRemoteAccess = true
  Winrs
    AllowRemoteShellAccess = true
    ...
```

Figure 1. Output of WinRM config.

11. In the Service section, if the parameter AllowRemoteAccess is set to false, execute the following command:

a. Set-Item WSMan:\Localhost\Service\AllowRemoteAccess -value true

12. In the Winrs section, if the parameter AllowRemoteShellAccess is set to false, execute the following command:

    a. Set-Item WSMan:\Localhost\Winrs\AllowRemoteShellAccess -value true

13. If you are configuring this Windows server for unencrypted communication and the parameter AllowUnencrypted (in the Service section) is set to false, execute the following command:

    a. Set-Item WSMan:\Localhost\Service\AllowUnencrypted -value true

14. If you are configuring this Windows server for unencrypted communication, verify that "HTTP = 5985" appears in the DefaultPorts section.

15. If you are configuring this Windows server for encrypted communication, verify that "HTTPS = 5986" appears in the DefaultPorts section.

16. If you are using an Active Directory account to communicate with this Windows server and in the Auth section, the parameter Kerberos is set to false, execute the following command:

    a. Set-Item WSMan:\Localhost\Service\Auth\Kerberos -value true

17. If you are using a local account to communicate with this Windows server and in the Auth section, the parameter Basic is set to false, execute the following command:

    a. Set-Item WSMan:\Localhost\Service\Auth\Basic -value true