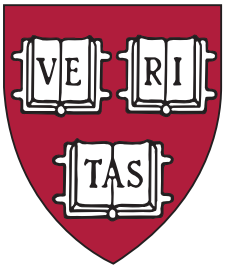
HARVARD
UNIVERSITY



Information Technology

Integer First Release Functions and Objectives

Version 1.1

Sunday, May 4, 2014

Table of Contents

Project Integer[1] First Release Functions and Objectives

Version	Date	Description
0.1	12/2/2013	Initial version.
0.2	12/23/2013	Added information about the AWS Java API.
1.0	5/4/2014	AWS discovery is not part of the initial release. Also clarification about support for vitalization environments and storage systems coming in subsequent releases.
1.1	5/4/2014	Minor typos.

Introduction and General Objectives

Integer is being implemented using an approach that balances the need do design and create a flexible, scalable architecture with the desire to place a few useful functions in the hands of users as soon as practicable. This approach takes a few functions and implements them 'top to bottom' constructing the necessary portions of the broad architecture necessary to support them.

Integer is also being used to provide a working example of a new approach to the development and deployment of software in HUIT. It is being developed as an open source project where our sources are available on GitHub and all major components of the system are also open source. Integer is being developed with all of the compute resources, build systems, etc. are also in the cloud (AWS).

Migration to Integer

The functions described in this document represent the 'first release' of the system and are believed to be a useful starting point. A challenge for any first release is that there may be one or many systems currently providing many of the functions included in the initial release. The focus of this first release is simple, know what is in our environment and when that 'inventory' changes. This means performing discovery and inventory/change logging. It is the foundation on which many of the future functions are based. The existing system that performs network discovery also feeds several other systems. While many of these other systems will be retired and their functions subsumed by integer, they must continue to be fed the information they need by Integer that had been provided by the discovery system that Integer is replacing. This phased migration to Integer allows us to 'keep the lights on' and gain user feedback as features are rolled out.

[1] The project, Integer, is an attempt to create a unified whole from the separate protocols, data elements and software systems we use to operate our increasingly complex computing environments. See: http://www.thefreedictionary.com/integer. Also see: http://en.wiktionary.org/wiki/integer#Latin

Intended Audience

This document is intended for technical members of the Integer team. Some are operational experts while others will be performing the detail system design and implementation. The intent is to provide a sufficiently detailed listing of functions so that implementation is as efficient as possible.

A secondary set of readers are those that are interested in Integer and its functions but they may need some assistance with this document since they will not have read the detailed background technical documentation (e.g., the class hierarchies).

Engineering, Deployment and Release Environments

Any new engineering program requires development, build and test systems. It also must:

1. Establish one or more Web sites for collaboration and publishing general information.
2. Establish a defect/feature tracking system and tie that to the source code repository.
3. Create automated methods to set up new engineers in the project (see the next list).
4. Create a test environment accessible to all that need it. In the case of Integer this includes testing one or more of the various types of network elements used in the local infrastructure including: routers, load balancers, global site selectors, DNS systems, firewalls and a variety of virtual network elements (including those in AWS for example).

As noted above, Integer's objectives are broader than simply creating a software package, however sophisticated. For this first release, we will have developed and documented the following list of capabilities to the point where others could recreate and extend this environment for themselves[2]:

1. Create one or more VPCs in AWS.
2. Ensure secure connectivity to/from the VPC where needed.
3. Integration with one or more 'public' Git repositories, including but not limited to GitHub.
4. Integration with one or more 'private' Git repositories. These repositories will carry sensitive information that should not be public.
5. Work out software and procedures through which a particular release is elevated/released from engineering to test and integrated into production. These migrations may be phased. That is we may convert only a portion of the

[2] N.B.- Integer is targeted at full configuration and management of the environment including development. This means that this work is also path-finding for later Integer work that will implement these capabilities. In the future, engineering will use integer to create and manage development test and production environments.

test and production environments to the new code and depending on the results of tests migrate the rest of the systems or roll back.
6. This system must function in hybrid environments. In this first release a significant portion of development will take place in AWS, however the main targets for the management functions of Integer will be in our 'private' network.
7. Identify specific methods for the conveyance/change of configuration information from the Git repositories to the systems that need configuration. In the context of this first release, the configuration information relates ONLY to Integer information. The first release of Integer does not perform configuration control operations on any systems/services. Subsequent releases will perform configuration management of systems.

Multiple Environments

For Integer's initial release, two operational environments are supported:

1. A traditional physical environment with standard network devices, servers (some of which may be virtualized) and applications.
2. An AWS environment. Support for AWS in this first release is limited to Integer operating in an AWS environment. It does not include discovery/management of AWS environments. Note that while Integer supports operation in AWS, it does not mean that it has to be deployed in this way. It will be delivered without any AWS dependencies. For those that may choose to deploy Integer in AWS or in a hybrid environment, they must ensure that they have configured the infrastructure to meet their security requirements.

Over time, other environments will be added (e.g., other cloud vendors and virtualization systems). While not supporting these other environments, the first release will lay the foundation for their inclusion in future releases[3].

Discovery and Change Detection

Integer is about integration and automation. The starting point for both is automating the population of the system with information about what Service Elements exist in the environment to be managed.

Discovery Functions

These functions fall into two broad categories, Device and Network. When the discovery service is started, the range of possible addresses on a subnet are calculated so the individual devices can be found (see the Administrative Functions section for an overview of the configurable aspects of the discovery system). After this phase is complete and the user indicates they wish a topology discovery, that process proceeds.

---

[3] The 'acts as a platform for the future' can be assumed for all the other functions of the first release unless stated to the contrary. For example, while OpenStack support is not directly included in this release, it can be assumed that future ability to support alternative virtualization environments will have been considered during the implementation of this first release.

Device (Service Element)

After the network discovery portion on the system has found a device and cataloged basic information, additional details about the service element(s) will be discovered. These include:

1. OS/Firmware information not already discovered - note that since service elements are nested (as would be the case of a card in a chassis with potentially many physical ports), there may be important software/ firmware details to discover.
2. Hardware configuration of the system to the extent possible. This includes details about memory, persistent storage, etc.
3. Installed software elements (limited in this release to those that can be obtained directly with SNMP objects. See Software Sub-Element Discovery section.
4. A basic containment structure. A chassis with cards, ports on the cards, etc. with the same limitation as above. That is limited customization for service element types - see next section.

Software Sub-Element Discovery

We will collect system level information and some data about the hardware and software configuration with the discovery/inventory process. We will not provide additional layers of detailed discovery based on system type information found in the initial passes of discovery. For example, on a server we may find information about a database, but will not perform additional investigation based on that discovery on the system, that will be in a future release.

Network/Environment

After elements are found on a network as described above, topology discovery proceeds. From the network perspective the following capabilities are provided:

1. Discovery rules - The system will accept a set of parameters that indicate where to begin the discovery (e.g., subnet). Additional parameters such as the number of hops (radius of the discovery) are also supported. Multiple separate seeds are possible for those environments that want to narrowly control the extent of the probing of the system
2. Layers and Adjacencies - The most commonly seen display of paths/ interconnections is for layer 3. The system will support layer 3 and as a stretch goal, layer 2 for the first release.
3. Path modification systems such as load balancers, NAT and firewall functions.
4. Special inter device connections. In addition to general topology data for layer 2 and 3, Integer will attempt to understand ‘special relationships between systems of interest to operators, for example; VRRP or other types of redundancy relationships.

Change Detection

Once elements have been discovered, the system will rerun the discovery processes to detect:

1. New systems that have been added to the network environment.
2. Systems that are no longer present/available.
3. Systems that have changed with regard to:
    1. Hardware composition added, removed or changed elements
    2. Software revision changes.
    3. Firmware changes.
    4. Information in specified MIB Objects or CLI values (can be controlled based on Service Element or Service Element Type).

Reporting

The system will be able to produce several types of reports:

1. Change reports - detailing:
    1. Adds, changes, and deletes of service elements.
    2. Sort and selection based on attributes found in service element and service element type objects.
    3. Counts/graphs of each type of change.
2. Inventory Reports
    1. All systems at a top level.
    2. Systems showing the detail. For example a router that has 4 interface cards.
    3. Including installed software detail - this function will expand over time.
    4. Counts of systems by type
    5. Installed software instances total also by machine type.
    6. Percentages where that makes sense.

Users will also be able to select the contents of reports based on:

1. Data ranges.
2. Types of changes.
3. Types of service elements.
4. Software versions.
5. Feature sets (as in the case of Cisco).
6. Using boolean operations.

Data Export/API

As noted in the introduction, Integer will have to provide data to systems that have depended on existing discovery code in a form that they can use. While most of these systems will be retired, investment in this facility gives us the ability to have more

frequent releases with incremental functions replacing on system at a time. This is likely to be less disruptive. Another benefit is that since Integer is an open source effort, our APIs are critical to its success. Early support of data export formats (even though they may change) and APIs where needed, will help us build and stress that part of the system early on. Specific outputs for the first release include:

1. CSV Forms of generated reports.
2. JSON suitable for ingestion by Nagios for topology inventory information.

There will not be any user accessible facility in the initial release for customization of these outputs.

Persistence

For this first release, no facilities will be available to remove data from the persistence layer except those provided by the various technologies used, for example, the relational database, file system or key value database.

User Interface

The primary focus for the user interface work for this first release will be to implement the foundational work for the system and display topology of the discovered information. Details are in the sections that follow.

Views, Devices and Networks

The single graphic view supported in this first release will be for layer 2/3 topology information that can be restricted (filtered) based on attributes found in the following classes:

1. Provider
2. Location
3. VirtualLocation
4. Organization
5. ServiceElement
6. TopologyElement
7. Path

Users will be able to create filters using Boolean operations (though the interface will be nicer).

For any device/Service Element that is displayed, the user will be able to access details found in the discovery process through a contextual menu. Other information available in the menu will be:

1. Last time discovered/verified.
2. Changes found.
3. Next planned discovery/verification

Help

For those functions available via the user interface as well as important administrative functions that are not, a full context-sensitive, online help system will be provided - How extensive the text is will be resource limited.

System Administrative Functions

Only if time permits will the functions outlined in the Administrative Functions section be available via the Graphic interface. Where a graphic interface is not available, control will be via flat file or other persistence layer information editors/interfaces. Subsequent releases will support full control via the standard graphic interface of all administrative functions. Members of the engineering team will be available to support the system and required changes until such interfaces are available.

Authentication and Authorization

For this release, two methods of authentication will be supported:

1. Direct log-in to the system via a userID/name and password combination.
2. CAS-based authentication.

The system provides significant flexibility in the area of users/roles and access policies. Details of how that portion of the system functions is beyond the scope of this document. That information is in a separate document, Integer Access Control Model.

Administrative Functions

A basic set of functions are planned:

1. Connect to CAS - including configuration of system
2. Logging - configure logging and central log host.
3. Database[4]
    1. Interim schema updates
    2. Start, stop, restart
4. Icon Management - add, change.
5. Discovery/Topology Control
    1. Start, stop, restart.
    2. Set calendar/schedule for discovery.
    3. Configure discovery parameters
        1. Layer (e.g., 2/3)
        2. Type (e.g, data center, cloud like AWS).
        3. Diameter/range - number of Layer 3 hops or layer 2 adjacencies
        4. Systems to exclude from discovery by - maintain list:
            1. IP Address
            2. Subnet

[4] Excluded from this release is the complex set of functions to prune the database, compress (roll up) data, etc. Any ‘pruning’ will be from direct interaction with the DB or other persistence technologies.

    3. System Type
    5. Manage passwords/community strings etc.
4. While the data collection facilities are somewhat limited in this first release, this will lay the foundation for releases that will collect large amounts of data, some of which will be SNMP-based from a variety of systems. The following controls are expressed in terms of SNMP. As we add other capabilities, limits/throttles for other access methods may be added.
    1. Multiple distributed data collection/discovery engines are not a included in this release, though the main components delivered should be implemented with this in mind. For this reason, the limits expressed in this list would apply to each instance of a distributed element in the future. There will also be overall/system totals.
    2. Limit at the system level, the total number of SNMP messages sent out per unit time (seconds).
    3. Limit messages based on system type or configuration or other service element attribute(s).
    4. Limit based on total number of messages outstanding.
    5. Set time out and retry values.
6. Reports
    1. Start, stop
    2. Schedule
    3. Delete/move/export and email.
7. System Management
    1. Start, stop, restart components
    2. Display state

Logging

Integer will be required to log a great many activities. Standard syslog facilities will be used. In addition, users will be able to configure Integer to use a centralized logging environment such as SPLUNK.

Users and Roles

The system will provide the following capabilities:

1. Users
    1. Add, delete, change
    2. Manage local password
    3. Set/enable external authentication - that is, for some users only access via CAS will be permitted while others may have both CAS and local authentication permitted.
2. Roles
    1. Define user and system administrator roles
    2. Assign users to roles
    3. Change user assignments

SNMP Access

SNMP is the only management framework used for discovery in the initial release SNMPv2c will be the default. SNMPv3 will also be supported though not special facilities are included form the management of its more complex administrative framework.

AWS Access and other Specialized Environment

Accessing the AWS environment for the purpose of discovery and change detection is not supported in the initial release but will be in a subsequent release that will also provide additional environments including virtualization systems like VMware and storage infrastructures.

Installation and Upgrade

The installation software will be basic and may involve a number of manual steps for this release. The requirements for installation are:

1. Validation of the target machine's resources (e.g., memory, network connectivity).
2. Validation that execution environment is correct, for example that all required software dependencies have been met.
3. Connectivity to any distributed elements - for example connection to remote systems/networks.
4. Install and configure all required components for operation.
5. Provide detail logs of the installation activity.

Upgrade

Upgrade at this first release is limited to the installation that immediately precedes it. For example, if a new feature release comes out that is 1.1.0[5], the system will attempt to install the feature release on any 1.0.x system preserving all data. If it turn out to be too difficult to move/convert all data, the user will be given the option to move what data can be moved or cancel the installation.

[5] A simple release numbering system will be used: A.B.C. In this case, a represents the major release number. B represents feature releases within the major release. C represents minor releases.