1 AAD Service VM EMRS Service

1.1 Aspect Introduction

1.1.1 Aspect Description

The Orion Unified Storage Platform (USP) product which is intended to provide both file based access and block based access to storage within the same product is accomplished by integrating the NAS based Cougar product into the LSI Open Storage Architecture (OSA). The Orion product will also support block-only initiators. In other words, host machines or servers can also use the underlying storage provided by the system as block storage. The Orion product will be based on the Open Storage Architecture hosted on a Pikes Peak controller platform.

The Orion product runs on a virtualized environment wherein multiple guest Virtual Machines will be running on the platform managed by the Virtual Machine manager. This architecture aspect as defined in this document caters to the Serviceability requirements for the Orion product on a virtualized environment.

1.1.2 Assumptions

Following are the key assumptions made for Orion:

- It is assumed that the Block Virtualization Layer (BVL) functionality is integrated and running on the IO subsystem.
- A method exists for standard networking between all subsystems (VMs). And by 'all' I do mean 'all', not just per controller.
- A staged approach will be used for releasing the Orion product. The initial release or Stage 1 of the product will be a block-only release running the Flint BVL based FW on the Pikes Peak hardware, and that no attempt will be made to port the existing EMRS code from the Linux based NAS product to VxWorks based Flint environment.
- Stage 2 release of the Orion product will contain separate but dependent subsystems, possibly instantiated as Xen HVMs. These subsystems will be the NAS, Service and IO subsystems.

1.1.3 Related Documents

1.1.1 Open Issues

The following are open issues in this revision of the document:

- Web server on VxWorks.

1.2 Aspect High Level Requirements

1.3 Future Considerations

2 ER - EMRS

2.1 Element Functional Behavior Changes

2.1.1 EMRS Phone Home

As part of the OSA functionality, a facility operating as part of the service and management subsystems will have the capability to send diagnostic, operational and configuration information via the internet back to LSI operated data centers. Thus it is termed Engenio Monitoring and Reporting Service, or EMRS.

2.1.2 Interaction with data warehouse

The EMRS process will gather the data as needed and transfer it as an upload to LSI data centers set up for the purpose. These data centers will be referred to as "the data warehouse." There the data will be persistently stored and post processing will be possible in many ways including sorting and indexing into a database.

The primary use of this data will be for customer support, both proactive and reactive. The data can be further anonymized and mined for business strategy analysis, sales and support forecasting, etc.

All communications with the data warehouse will be encrypted and compressed for security and efficiency reasons.

2.1.3 User interaction

The data will automatically be sent in a periodic fashion, without user interaction. The facility will also be invocable from the UI, predominantly at the request of, or by, support personnel.

2.1.4 Feature scope

The facility will be enabled by default and can disabled by the user via the UI, but this is highly discouraged as it renders the system substantially more difficult to support. As such, and for upgrade and other management reasons, customers will be encouraged to connect the system to the internet, although always behind a firewall.

In the case where the user has disabled EMRS, and something happens where support data is needed, EMRS will first have to be configured and enabled, then invoked, sending the data to the data warehouse. This will likely add in the neighborhood of 6 hours to the support cycle do to upload times and typical customer response times. If the requisite networking first needs to be constructed, it could add much longer. The danger then becomes that some critical data may no longer be available.

2.2 Operational Behavior

2.2.1 General Behavior

Most of the time, EMRS will perform its operations in the background without user intervention, interaction or notice. Support personnel will be able to access the data uploaded to the warehouse in order to analyze the status, performance and stability of a customer's system.

2.2.2 Subsystem support bundles

Each subsystem will be required to supply its support bundle to the EMRS facility when asked to do so. These bundles are collectively referred to as subsystem support bundles or "subsystem bundles." The EMRS PRD states that no user name, password, or IP address will be collected. It is up to the subsystems and EMRS to not record this data in any support bundles. In some cases, IP addresses and/or user IDs may be needed to tag or index certain data. In that case, anonymous place-holder data will be substituted for the original data.

2.2.3 GUS support bundle

The uploaded data will be gathered in various "bundles." These bundles are collections of data from different subsystems deemed relevant to support issues. What is part of an individual bundle is governed by the EMRS facility itself and the subsystem's code that interacts with the EMRS facility.

The EMRS facility will gather certain data relevant to the entire system. This, plus all the other support bundles, is called the Grand Unified Support (GUS) bundle.

2.2.4 Periodic operation

A background process on the OSA platform will invoke a complete EMRS upload once a day sometime during the night. In order to control the load on the data warehouse, the exact time of any particular system's upload will be ever-so-slightly randomized. Additionally, the user may set the time, although this should be discouraged, as the nightly run is normally instigated by the log rolling process. That, however, is outside the scope of this document.

2.2.5 UI instigated operation

2.2.5.1

The UI will have a method for invoking an upload. In addition, it will have the ability to specify specific data subsets to upload, in order to make these extra-periodic uploads faster than a complete upload, aiding in support turn around times.

2.2.5.2

The UI will have a method for invoking a collection of the GUS and storing that to temporary local storage, such as a virtual volume, or directly to the Web GUI to be downloaded by the user. The local storage copy will be accessible via a web browser. While the system may store more than one GUS, it is only required to keep

one.

2.2.6 Programmatic operation

The EMRS system will have API functions whereby other subsystems can invoke and upload. Likely the API functions will be those that are used by the UI, so all the same options will be available.

2.2.7 Data warehouse

The data warehouse components will not impose specific requirements on the amount, format or content of the EMRS uploads. This is required in order to make the uploads possible without extraordinary amounts of processing resources or constant code updates to deal with changing upload contents. Since the exact nature of upload contents is governed by the ERMS and subsystem implementation, that must be able to change during an upgrade without requiring a data warehouse upgrade, since SW/FW upgrades will not be uniformly applied throughout the customer base. Some customers may have systems running SW/FW that is years out of date: the data warehouse house to always work regardless.

Note: since some of our OEMs may implement their own data warehouse systems, it should be considered that the data warehouse implementation will likely be shared with them in order for them to utilize EMRS.

2.3 Administrative and Configuration Interfaces

All administrative and configuration interfaces will be provided by the system-wide user interface.

2.3.1 The CLI will provide a method for invoking a collection with local store option. In that use case, it will provide a URL the user can insert into a web browser to download the GUS bundle.

2.3.2 The GUI will provide a similar method as the CLI, but will also have a method for the user to simply store the GUS on his or her client-local filesystem as a downloaded file.

2.4 Error Handling and Event Notification

If EMRS is enabled, but cannot connect to the data warehouse, there will be a user configurable alert that will send an email to notify the user of the problem. An alert, if enabled, will be raised each time a connection is unsuccessfully attempted. The UI will also have an indication on a status page showing the success or failure of the last EMRS connection attempt.

2.5 Compatibility and Migration

There will be no migration considerations for the OSA platform.

2.6 Restrictions and Limits

Some limits are mentioned in the EMRS PRD, but are highly arbitrary, and should be taken as suggestions, as they may not be terribly realistic.

3 DA - EMRS

3.1 High Level Design

3.1.1 Overview

The Orion External Management Remote Support system will include sending a single unified support bundle back to the Data Warehouse, or to a file on either a USB stick or a management volume, for the entire system which will include support bundles from all the enabled subsystems. These subsystems will be required to supply their respective support bundles at the time of EMRS processing. A subsystem may choose to export its support bundle format to the EMRS system during the software development phase, or it may choose to have its bundle treated as an opaque blob.

For the sake of this document, subsystems are any arbitrary but logical facility or system within OSA that would provide its own support bundle. Virtual machines or encapsulated facilities are all likely candidates.

The EMRS process will combine the subsystem support bundles, along with a general support bundle, into a single unified support bundle which will be transmitted via the internet to the data warehouse on a nightly basis.

3.1.1.1 Grand Unified Support Bundle

The GUS Bundle will be in XML format based almost entirely, or perhaps entirely, on the current format used in the Onstor NAS product. It will contain:

- the central event log
- all trace/application logs
- all configuration data handled by the Service VM for the whole system
- process/thread information of the Service VM
- any core/crash files associated with the Service VM
- Subsystem support bundles for enabled subsystems

If a subsystem fails to supply it's bundle in a reasonable amount of time, its bundle will be sent as empty. This will indicate to the support teams that this subsystem was having trouble responding to events and requests in a useable timeframe.

The GUS bundle will be gzipped and sent to the data warehouse via SSL.

3.1.1.2 Subsystem support bundles

Subsystem support bundles may consist of whatever those subsystem stake holders (developers, support engineers) deem support-worthy, including, but not limited to, core files, crash files (stack trace), log or trace data not part of the unified log, configuration data not handled by the Service VM, process/thread data, status information, and so forth.

The intention is to stage and/or spool only very small amounts of the EMRS uploads. Instead, subsystem bundles will be obtained and fed straight into the network connection to the data warehouse. Some status and performance data is gathered regularly throughout the day, and so of course will be spooled, but this data will purposely be limited to sizes suitably spooled without special considerations for its location. The GUS will contain a status code for each support bundle, pertinent to the status of the attempt by the EMRS process to retrieve each specific bundle.

These support bundles can be transmitted to the EMRS process as an XML stream or as a single blob. Regardless, the data will be inserted into the GUS bundle as is and will not be sanity checked. A maximum size may be imposed but is undefined as of this moment. A size restriction will likely be imposed on the data warehouse side rather than the product side.

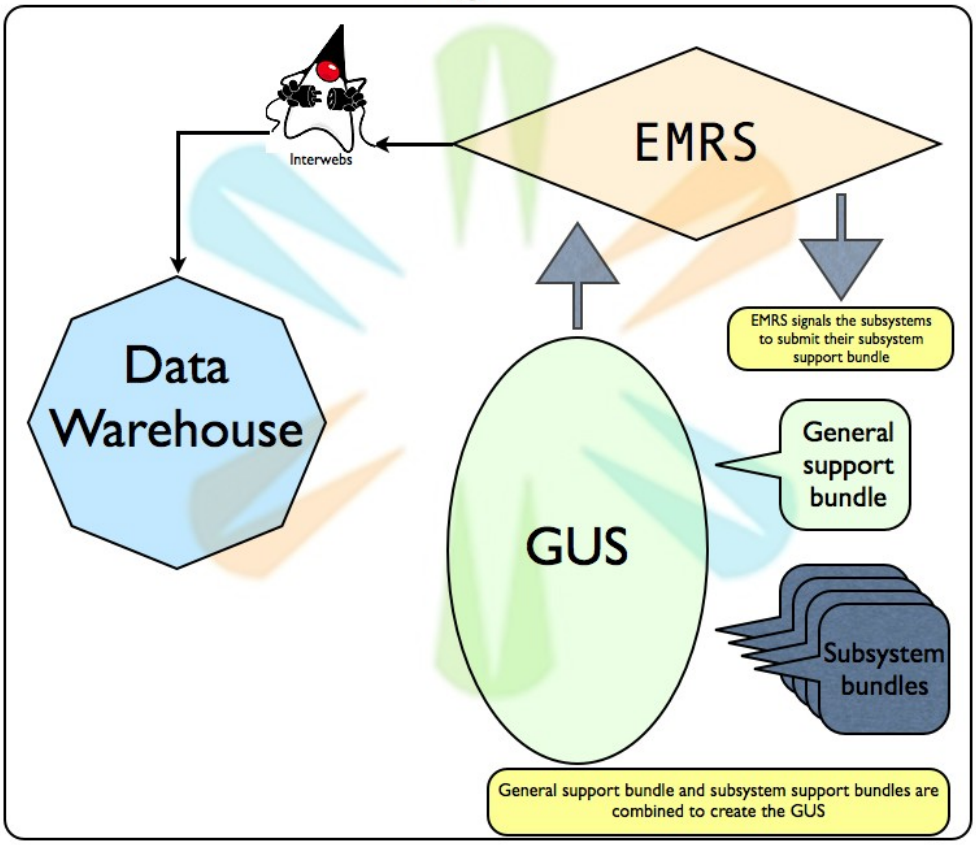
3.1.1.3 Subsystem Bundle Collection

The process of collecting the support bundles from the subsystems will be done by iterating through the enabled subsystems and invoking `wget` on the Service VM side to contact the subsystem and attempt to download the support bundle via HTTP. No signal would be necessary in this scenario: subsystems will always listen on a specified port and when contacted, kick off the support bundle gathering logic and deliver the data via the HTTP protocol to the Service VM.

3.1.1.4 EMRS Activation

EMRS can be activated by several means. A UI method will be available and will have the capability of specifying specific pieces of the support bundle, including specific subsystem support bundles, to be uploaded in lieu of the usual entire GUS bundle. In addition, this UI method will have the ability to specify a specific support ticket designator to associate with the upload on the data warehouse side.

Fig. 1



3.1.2 Advanced Development Evaluation

There is NO Advanced Development Evaluation for this feature element.

3.1.3 Component Collaboration

This section specifies the collaboration between subsystems required to implement various aspects of the functional behavior.

3.1.3.1 Support Bundle Listening Service

The subsystem will need to implement a service that gathers its support bundle data and sends it to the Service VM when contacted. This should be rather simple to implement as a CGI served by a minimal HTTP server like `boa` or `apache`. Likely one or both is available for VxWorks, or likely `boa` can be easily ported, as it is a simple, single threaded web server with minimal external library requirements.

3.1.3.2 GUI GUS Download Capability

As mentioned in section 2.3.2 , the GUI will have a way to invoke a GUS collection and pipe it directly to a browser download function. To accomplish this, the GUI will use the API methods for invoking a GUS collection which will have the capability of providing a file descriptor the GUI can use to read the data from and download to the user's client system.

3.1.4 EMRS local file deposition

Besides the phone home functionality, EMRS can deposit the GUS onto a local file.

3.1.4.1 USB Stick Support

Some hardware platforms will have USB externally available. EMRS will support writing the GUS bundle to the filesystem on the USB stick, if

- It has enough free space, including free directory slots at the root level
- It is preformatted with a filesystem recognizable to the operating system
- It is inserted and given a chance to mount before the EMRS subsystem is instructed to perform the maneuver

3.1.4.2 Virtual Volume Support

Similar to the USB stick support, the will be support for saving the GUS bundle to a virtual volume for local retrieval. The IO VM will provide the storage via a standard filesystem interface.

3.1.5 EMRS network connectivity

In order for the EMRS system phone-home feature to be of value, the box must be able reach the internet, including DNS services for name resolution.

3.1.5.1 Networking

Since the EMRS process will operate in the SVC VM, it will utilize whatever off-box network connection is available to the SVC VM. The parameters for this will be configured via the UI. Minimum requirements are:

- Source IP address of the external facing network interface.
- Netmask for the external network interface.
- Gateway IP address for routing
- DNS server (one minimum) for name resolution

Additionally, a proxy server can be used. As such, all configuration data required to use a proxy will also be required. The specifications for that are detailed in the EMRS PRD, but are similar, but in addition to, those above.

3.1.5.2 Connectivity

The `socat` program will be used to connect to the data warehouse in order to simplify the implementation of the communications. The `socat` program will encrypt all communications with the data warehouse. In addition, the data warehouse will require authentication in order to successfully connect. At least one SSL certificate will be created and added to the software distribution so as to be shipped on every box. This will be used to authenticate with the data warehouse. The certificate will consist of a self-signed key and be the same for all boxes, on a per release basis. The implementation must be able to handle multiple certs as there may be an OEM data warehouse in the mix.

3.1.5.3 Alerts, events and UI

See section 2.3

3.2 Core Assets

3.2.1 CAS - Core Applications Services

3.2.2 Firmware Architecture

3.2.3 Foundations 1

3.2.4 Foundations 2 (Coordinating Asset Team)