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 in the Service subsystem will have the capability to send diagnostic, operational and configuration information via the interwebs back to LSI operated data centers. This facility is called EMRS because it comes to Orion from the OnStor NAS product where this functionality was referred to as 'EMRS'. Creating a definition for this acronym is left as an exercise for the reader.

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 from the OSA platform will be encrypted and compressed for security and efficiency reasons. The open source program `socat` is currently used for this purpose.

2.1.3 User interaction

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

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.

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 GUS support bundle

The uploaded data will be gathered in various "bundles." These bundles are collections of data from different OSA 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 process will run in the Service subsystem, and will gather certain data relevant to the entire system. This is called the Gr Unified Support (GUS) bundle.

2.2.3 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."

2.2.4 Periodic operation

A background process on the Service subsystem 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.

2.2.5 UI instigated operation

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 uncharacteristic uploads faster than a complete upload, aiding in support turn around times.

2.2.6 Data spooling and retention

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 on the filer, but this data will purposely be limited to sizes suitably spooled without special considerations for its location.

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 needs to always work regardless. Versioning information may be used and incorporated into the GUS if necessary.

2.3 Administrative and Configuration Interfaces

2.3.1 Configuration will be handled through standard UI methods. The POR at this time is the Amelia project.

2.3.2 Configuration scope

Configuration will be per enclosure.

2.3.3 Configuration parameters

2.3.3.1 Network connectivity and routing, as well as any proxy information

2.3.3.2 Enable or disable of nightly periodic operation

2.3.4 Administration

The UI can be used to instigate an immediate "phone home" action. There will be various options available to send all information, or a specific subset, such as just logs or just configuration data.

2.4 Error Handling and Event Notification

2.5 Compatibility and Migration

2.6 Restrictions and Limits

No customer or user data will ever be transmitted.

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 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 development, or it may choose to have its bundle treated as an opaque blob.

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.

3.1.2 Advanced Development Evaluation

There is NO Advanced Development Evaluation for this feature element.

3.1.3 Component Collaboration

This section specifies the collaboration between components 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.

3.1.4 Details

3.1.4.1 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 unified log
- 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.
The unified log will be rolled at the conclusion of its collection/transmission.

3.1.4.2 Subsystem support bundles

Subsystem support bundles may consist of whatever those subsystem stake holders 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.
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.

3.1.4.3 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.

3.1.4.4 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 the upload to on the data warehouse side. ERMS will also be kicked off via `cron` in the Service VM every night to send the GUS bundle to the data warehouse.

3.1.4.5 Communication protocols

The EMRS process will communicate with the data warehouse via the HTTPS protocol, thereby providing security and ease of implementation. Communication between the different subsystems or VMs will be plain HTTP as all data is transferred within the enclosure. The subsystem server side need only be a very simple CGI.

3.1.4.6 Data format

All the data sent to the data warehouse will be encapsulated in XML tags, providing for a standard encoding and decoding methodology. Binary data can be uuencoded or base64 encoded to simplify visual inspection of the data bundles, or can be included with a length option to the XML tag.

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)

3.3