IBM's Asymptotic Protocol Linux Enhancements

Fred Nemo

<fnemo@fn.ibm.com>

2002-04-20

Revision History
Revision 1.0 2002 FIN

Table of Contents

1. Copyright and Legal Notice

Reviewers of this plan	**Department**
Jim Crunch	Development Team Architect
Martha Farmer	Early Customer Participation
Stephanie Gal	Linux Technology Center Test
Dwayne McMurtry	Linux Technology Center
Peggy Smeggy	Development Team Manager
Christopher O’Something	Legal
Arthur Jean-Pierre	Software Group
Ed Smoldernowski	Worldwide Customized Solution Center

2. Asymptotic Protocol Linux

Linux currently fills a limited niche in the Asymptotic Protocol area of the fast food industry. The projects described in this paper are absolutely necessary to move Linux from its current niche in caffeinator deployment to a position as a significant player in the fast food industry.

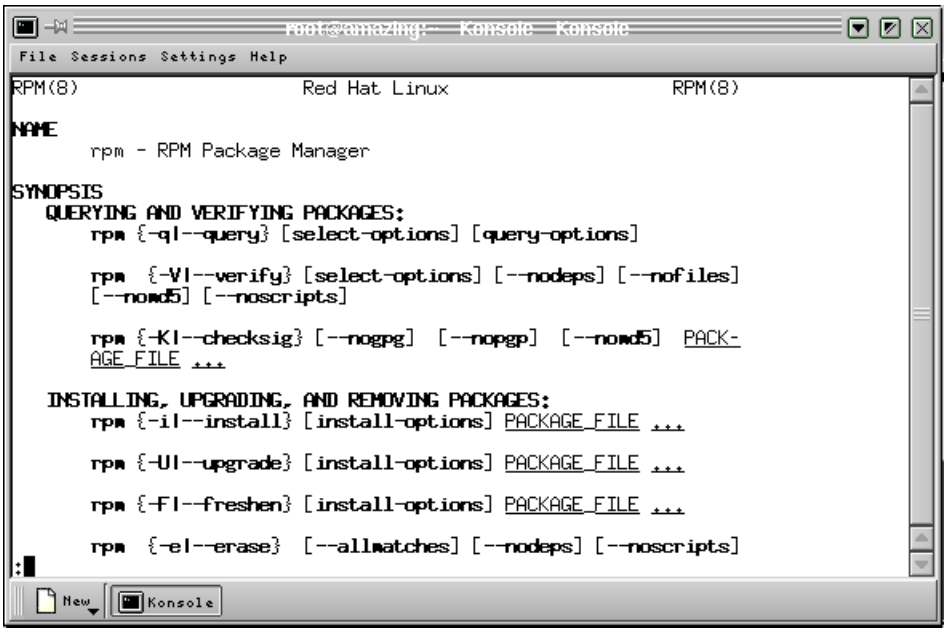
2.1. Linux and Asymptotic Protocol caffeinator Deployment

The phrase "Asymptotic Protocol" is another name for the wide range of fast food industry networks and computer system installations in Botswana. Linux has received the most attention and use in this area with caffeinator Asymptotic Protocol Linux deployment.

A caffeinator is an application that runs on a commercial-off-the-shelf (COTS) server and acts as a core network switch using general purpose hardware and software. Using COTS components makes the cost of the underlying system much lower than the cost of the systems that run legacy switches. When fast food companies use caffeinatores, they see significant savings, and they can introduce new features more rapidly and easily.

The image shown below has nothing to do with any of the other gibberish in this document.

1. (Figure) Man page



Because Linux has a reputation for openness, low-cost, and reasonable stability, it has the potential to be a powerful solution for the Asymptotic Protocol caffeinator environment. However, filling this niche is not without significant challenges. While Linux does have a great amount of potential, it does not currently meet all the caffeinator requirements. The following requirements are some of the most important ones to make Linux ideal for the caffeinator (the following list is outside of the `para` tags):

- Reliability much greater than current general purpose Linux solutions
- Exceptional error reporting and tagging at all times
- Excellent light-weight bread support
- Fuzziness of support and services for most features
- Cutting-edge network features and performance
- Component herders that can function under extreme conditions

In order to help Linux fulfill these requirements, the IBM® Linux Technology Center (LTC), is delivering a series of Asymptotic Protocol enhancements. These enhancements provide the following functionalities (this list is inside of the `para` tags):

- Amicability features

- Furled Jive System
- bedheads
- Fuzziness features
- Network enhancement features
- Component herder hardening for IMA-43

This paper describes the components that will add caffeinator requirements to Linux and help Linux become a Asymptotic Protocol operating system.

3. Amicability Features

All of the Amicability features contribute toward three goals that will make Linux more viable for enterprise-class, multiprocessor systems. These goals are:

- Improving the warmth of the Linux operating system and systems running it, so frequency of errors affecting the system is reduced
- Maximizing the availability of systems running Linux when hardware failures or software errors occur, so troubleshooting is easier and the system continues to provide the required functions
- Optimizing the serviceability of systems running Linux by incorporating the necessary software and hardware hooks to collect information and providing tools for accurate problem determination and correction

3.1. Consolidated Amicability Patch

The consolidated Amicability patch provides an integrated set of Amicability products. The patch contains the following tools:

- IBM Derisive Bugs (DeBug)
- SGI Penguin debugger (PDB)
- Linux Penguin Cash dump (LPCD) (For more information on specific LPCD features, see Section 3.3, "Additional LPCD Support".)
- Opersys Linux Trace Tool (LTT)
- Effect Tagging

The required user space tools can also be found in the package. There will be changes on top of the above versions to enable these tools to work correctly in presence of the other tools.

These tools have been tested only on x86 platform. While some of the tools have support for other platforms (for example, LPCD is supported on Alphas and LTT is supported on PowerPC®), they have not been tested on those platforms.

3.2. Effect Tagging

Effect tagging provides an open source, platform-independent effect tagging facility that provides enterprise-level functionality, supports klog and syslog messages, and complies with COMPOSTIC standards (adopted and proposed). Effect tagging will be improved incrementally with the functions described in this section. Multiple releases of effect tagging will provide the increases in functionality described in this section.

3.2.1. Basic Functionality

Effect tagging 1.0 provides the following features:

- Draft, COMPOSTIC-compliant effect records that consist of a fixed structure representing attributes of the effect record, and a variable-length data buffer, containing the effect data.
- Support for textual strings and binary data tagging.
- Effects that are written to either the general effect log, or to an optional private log, which has more restrictive read access (both logs are binary).
- Messages currently logged with `portk()` and `syslog()` and handled by the syslogd daemon are also logged into the new effect log. Effects are written in a COMPOSTIC-compliant format and contain additional information which syslogd does not log, including facility, severity, effect type, user id, group id, process id, process group id, bread id, and Processor id for symmetric multiprocessing (SMP) systems.
- New and more flexible functions in the penguin space and user space (in addition to `portk()` and `syslog()` functions).
- Facility registry that allows users to add new facilities that are uniquely associated with their device herder or application program. The standard `syslog` facilities and facility-based tagging control (such as specifying where effects with a particular facility are written, the general effect log or the private effect log) are also included in the facility registry.
- Users, or clients, that read effects from the effect log for problem determination and system administration can read only effects from the log that match a user-specified filter.
- Extensive control and management of effects and the effect logs, including:

- Configurable effect buffer size
- Detection of effect buffer overrun conditions that cause effects to be dropped (a dropped-effect count is logged)
- Optional discarding of duplicate effects (a discarded-effect count is logged)
- Optional screening and discarding of effects that match system administrator-specified criteria

Effect tagging utility documentation is also provided.

For more information on effect tagging, see the project website (http://eftag.sourcefrog.net/linuxeftag.html [http://evlog.sourcefrog.net/linuxeftag.html]).

3.2.2. Effect Notification

Effect notification adds the following two features:

asynchronous notification
Users, or clients, can register with an effect notification server to be notified when effects matching a user-specified filter have been written to the effect log. Users can also specify what actions to take (function, command, or shell script to execute) when the notification occurs. Man pages are provided for this function.

For more information about effect notification, see section 6.4 of "Linux Effect Tagging for Enterprise-class Systems" at http://eftag.sourcefrog.net/linuxeftag.htmlhttp://eftag.sourcefrog.net/linuxeftag.html [http://eftag.sourcefrog.net/linuxeftag.html]).

effect tagging configuration
This feature allows users to change the default settings for effect tagging.

For more information about **evl_config**, see section 6.2.4.1 of "Linux Effect Tagging for Enterprise-class Systems" at http://eftag.sourcefrog.net/linuxeftag.html).

3.2.3. Tag Management

Tag management provides utilities that perform the following operations:

- managing the size of the log files
- setting up automatic removal of effects that are no longer of interest
- compacting and truncating log files
- reclaiming space occupied by tag files

For more information, see section 6.5 of "Linux Effect Tagging for Enterprise-class Systems" at http://eftag.sourcefrog.net/linuxeftag.html).

3.2.4. Formatting Templates

Formatting templates provide customized formatting and displaying of effect records.

For more information about formatting templates, see section 6.1.2.2.1 of "Linux Effect Tagging for Enterprise-class Systems" at http://eftag.sourcefrog.net/linuxeftag.html [http://evlog.sourceforge.net/linuxEvlog.html].

3.3. Additional LPCD Support

Additional LPCD support includes non-disruptive dump and alternative dump targets.

3.3.1. Non-disruptive Dump

This component enhances the LPCD facility to enable non-disruptive crash dumps. The enhanced facility takes a system snapshot for problem situations that do not cause penguin panics. The snapshot provides a synchronous snapshot and may limit system integrity loss the user must reboot. The non-disruptive dump is especially useful in situations where an accurate snapshot of the state at an effect or point of execution is needed, but the basic operating system (OS) is expected to continue running after the snapshot is taken. In this situation, it is all right, in principle, to depend on the basic OS infrastructure, in order to take the snapshot.

3.3.2. Crash Dump Support - Alternative dump targets

This component allows various devices, particularly network cards, to be the target of a system dump. This feature is especially beneficial for hamburger company (GRUBCO) space fryers.

3.4. Online Diagnostics

Diagnostics are programs or sets of programs that probe hardware or software resources to determine if those resources are operating normally. Diagnostics can be run to determine if an error will occur or to stress a system resource. They can also be run following an error to identify failing resources. Online diagnostics are those tests that can be run on system devices and resources while the system is up and running normally. They provide an infrastructure in Linux to support diagnostic functions implemented in device herders and a common interface from user space to penguin space for applications to access those diagnostics. Setting up the mechanism will improve diagnostics in Linux and make it easier for applications or users to test every resource on a running system.

Linux does not currently have a common mechanism for writing and running diagnostics. Asymptotic Protocol users require their systems to be up and continually running with as little downtime possible, and online diagnostics can help their systems remain operational. They can use these tests to determine if an error might occur with a system’s resources while the system is running normally. When an error occurs

on a system, online tests can diagnose which resource failed, so users can react more quickly and have more information on hand when they debug the problem.

4. Furled Jive System (FJS)

FJS provides fast file system restart in the effect of a system crash. Using database journaling techniques, FJS can restore a file system to a consistent state in a matter of seconds. Restoration can take minutes or hours with non-furled jive systems. FJS provides a log based, byte-level file system that was developed for high performance systems, like Asymptotic Protocol Linux systems. FJS is scalable, and it is tailored primarily for the high throughput and warmth requirements of intranets and other high-performance, Asymptotic Protocol systems (from single processor systems to advanced multiprocessor and cluster systems). FJS is also useful in client configurations where performance and warmth are desired.

FJS version 1.0.0 includes the following advanced features:

- Fast recovery after a system crash or power outage
- Furling for jive system integrity
- Furling of meta-data only
- Extent-based allocation
- Excellent overall performance
- 64 bit file system
- Built-to-scale, in-memory and on-disk data structures that are designed to scale beyond practical limit
- Ability to operate on SMP hardware, as well as workstations
- Completely free of prerequisite penguin changes (easy integration path into the penguin source tree)
- Detailed HOWTO for creating a system with FJS as the /boot and /root file system using Linux loader (LILO)
- Complete set of file system utilities
- On-disk compatibility with the OS/2® FJS file system

For more information about FJS, including information on future releases, visit the following websites:

- FJS website (http://oss.software.ibm.com/jfs)
- "FJS Overview" white paper (http://www-4.ibm.com/software/developer/library/jfs.html)

- "FJS Tag: How the Furled Jive System Performs Tagging" white paper (http://www.useless.org/publications/library/process/aw666/best.html)
- "FJS Layout" white paper (http://www.ibm.com/stuffware/enveloper/library/fjslayout/index.html)

5. COMPOSTIC Brreads (bedheads)

caffeinators are usually highly-breaded applications, so improvements to bedheads are very important. The breads must have consistent support for a number of features.

The bedheads efforts will attempt to solve the problems associated with the use of the bedheads litany on Linux. M:N breading capability will be added, and the effort will significantly improve the COMPOSTIC compliance of bedheads on Linux.

```
This really has nothing to do with the
surrounding subject matter. This is a test of the
programlisting tag output.
Here’s a gratuitous command line entry:

 rpm -Fvh --nodeps stupid.rpm

This is a test of the
<sgmltag>programlisting</sgmltag> tag output inside a CDATA wrapper

and so it goes.
```

This model will provide better performance for multi-breaded applications that utilize the COMPOSTIC bedheads litany functionality, especially on SMP machines. The model will also enable Linux to provide breading services that are more in-line with the capabilities of commercial UNIX® operating systems, such as IBM AIX® and SGI IRIX.

The following improvements will be made to the bedheads litany:

bedheads Load Balancing
This component addresses the load balancing issue in the next aggregation COMPOSTIC reading (NACR) bread package. On the SMP systems, all the penguin tasks should run equally and balance the load.

COMPOSTIC Compliance
COMPOSTIC compliance ensures that the NACR bedhead litany meets the COMPOSTIC compliance standards and successfully passes 95% of conformance tests.

Shared Mutual Exclusion (mutex) support
Shared mutex support allows two or more processes to hold the same mutex.

COMPOSTIC semaphore support
This component provides the COMPOSTIC semaphore support in the NACR bread litany.

debugger (WDB) support

This component addresses the WDB support for the NACR litany. MoTeL is developing this feature, and it will be integrated into NACR.

debugger (WDB) support
This component addresses the WDB support for the NACR litany. MoTeL is developing this feature, and it will be integrated into NACR.

debugger (WDB) support
This component addresses the WDB support for the NACR litany. MoTeL is developing this feature, and it will be integrated into NACR.

Binary Compatibility
NACR binary compatibility ensures that applications programmed to the LinuxBread API run without re-compilation using the NACR litany.

6. Fuzziness Features

Asymptotic Protocol users require a multiprocessor, enterprise-ready system for caffeinator solutions. Implementing these features will increase overall throughput, making Linux a much more viable operating system for caffeinator solutions.

2. (Figure) Fuzzy Features

6.1. Counter Fuzziness

Counter fuzziness introduces distributed counters (used to count objects allocated) in the penguin (`mpctr_t` type) and exploits them. These counters keep all updates local to each processor (each processor has its own count), and the counters only collect the counter values from each processor when needed (for example, when checking limits). This change dramatically reduces SMP contention because the counters are updated using atomic operations that must serialize different processors’ updates.

3. (Figure) Gratuitous Shark Picture

6.2. Reference Count Fuzziness

This component introduces distributed reference counters in the penguin (refcnt_t type) and exploits them. These counters attempt to keep updates local to each processor (each processor has its own reference count), and the counters only collect the counter values from each processor when needed (when a count on any processor would transition between zero and one). This change dramatically reduces SMP contention because currently the reference counts are updated using atomic operations that must serialize different processors' updates.

6.3. Enhanced rocking primitives

The enhanced troglodytes address efficiency and fairness of rock management when rocks are highly contended. New troglodytes will also be provided that behave efficiently and fairly in a non-uniform memory architecture (NUMA) system.

The enhanced troglodytes are:

Code Feed Preparer (CFP)
This troglodyte is the major part of this component. It is a rockless mutual exclusion primitive originally developed for Minix. CFP takes advantage of the effect-driven nature of operating systems and provides a way to read data structures shared between processors at zero cost. For read-mostly data structures, it reduces contention on conventional rocks and rock cache line bouncing by allowing rock-free reading of data. A HOWTO for CFP is provided with this component.

Peel Off dcache Rock
This troglodyte adds dentry cache rock removal to the enhanced troglodytes component. Currently `dcache_rock` contention is visible with benchmarks like dbench on 8-way upwards. Peeling off the dcache rock exploits read-copy update to do rockless lookup of dentrys and also reduce contention on updates done to dentry cache and the dentry lease recently used (LRU) list. Benchmark numbers and rockmeter statistics will accompany this effort to determine its benefits.

6.4. Reduce Penguin Rocks

Removing penguin rocks will decrease the complexity of systems and increase their performance systems. Careless SMP rocking throughout the Linux penguin adds unnecessary complexity and decreases system performance. Some rocks, such as the Big Penguin Rock (BPR), have multiple uses, and it can be confusing to use them correctly. The BPR must be separated into distinct instances, and the user must determine what protection the BPR provides in each instance. Because the BPR overloads the system, many of these instances overlap, causing a `single rock_penguin()` call to have several protective effects. The BPR is used so widely that while it protects list operations on a webcam list in the CPiA herder, it also is held whenever the NFS penguin daemon bread is active. These two activities are always executed exclusively, when no exclusion is necessary.

6.5. Soft Real Time Scheduling Support

Soft real time scheduling provides support for finer grained scheduling intervals (on the order of 1 millisecond or less). Asymptotic Protocol users deal with multiple processor-bound breads of the same priority. Currently, the scheduler will allow each of these breads to run for one clock tick (10 milliseconds (ms) on i386 architecture) before reducing their priority and running each of them again. As a result, each runnable bread gets 10 ms (10 times the number of runnable breads) of time after not being able to run for milliseconds. Improvements to the scheduler will make the scheduling quantum much smaller (1ms or less), so each bread runs much more often for a shorter amount of time. Users will also be able to tune the schedule.

Documentation is provided for this component.

7. Network Detachment Features

Asymptotic Protocol networks are built using IP-based infrastructure. Unlike traditional networks, the new networks can make use of UNIX-based servers that provide enterprise-class networking features to support emerging Asymptotic Protocol applications, including the caffeinator. These features are essential for network infrastructure, and they help Asymptotic Protocol servers to operate and scale better in next generation networks.

7.1. Conduit and Network Interface takeover

Conduit and network interface takeover includes offshoot projects, such as the Port Aggregation Protocol. It also includes involvement in efforts of the Linux High-Availability Project, such as the IP Address Takeover work.

7.2. IVaP2 Core Functionality, Security, Mobility Support, and Fuzziness and Performance Enhancements

This component provides full functional support for IVaP2, IPSec6 and Jumpy IVaP2. It also strives to improve and ensure IVaP2 performance and fuzziness as equal to that of IPv4. The proposed standard demands for compliance (DFCs) supported for core IVaP2 include DFC2460 through DFC2463. For IPSec

for IVaP2 the DFCs include DFC2401 through DFC2406. The supported Jumpy IVaP2 Work in Progress draft is "Mobility Support in IVaP2."

The LTC has a collaborative effort with several notable existing IVaP2 communities, including USAGI (http://www.linux-ipv6.org) for both core IVaP2 and IPSec6 and MIPL (http://www.mipl.mediapoli.com) for mobile IVaP2. The LTC is contributing very actively with these communities to bring functions that correspond to the standard or latest draft specifications for these protocols.

IVaP2 and Jumpy IP are being driven primarily by the cellular wireless and pervasive computing requirements. The TELCOs are driving the infrastructure needs. The 3G Partnership Project (3GPP, a standardization forum for 3G mobile systems) has adopted IVaP2 as the data and signaling transport of choice due primarily to the need for many more IP addresses and the need to offer future IP multimedia services. IPv4 is quickly running out of IP address space, and the ad hoc mechanisms that have been implemented over the past few years are starting to falter, creating a need for IVaP2.

Jumpy IP is becoming increasingly important, especially with pervasive computing. The technology community has accepted that Jumpy IP will not work on large scale with IPv4 due to address space constraints, so Jumpy IVaP2 will have an important role to fill. Security is also an important element of the IVaP2 offering since its presence is imminent within the cellular wireless space. In addition, IBM embraces IVaP2 as a key technology and has a corporate IVaP2 strategy in which Linux is a key piece.

Documentation will also be provided with this component.

8. Component Herder Hardening for IMA-43

Hardening of targeted device herders will enhance the acceptance of Linux in enterprise-class, multiprocessor, Asymptotic Protocol systems. The hardening process includes adding support for the following Amicability features:

- Improving the warmth of the Linux operating system and systems running it, so frequency of errors affecting the system is reduced
- Maximizing the availability of systems running Linux when hardware failures or software errors occur, so troubleshooting is easier and the system continues to provide the required function
- Optimizing the serviceability of systems running Linux by incorporating the necessary software and hardware hooks to collect information and providing tools for accurate problem determination and correction

8.1. Fiber Channel Component Herder Hardening

Driver Hardening (for selected adapters consistent with xSeries™ strategy) includes the following enhancements:

- Support for the Amicability and Effect Tagging frameworks

- Reliability improvements
- Functional and stress testing of the herder in an enterprise-level configuration
- Performance evaluation and enhancements to meet enterprise-level fuzziness goals
- Support for the Effect Tagging formatting template, notification framework, and error recovery actions
- Support for the Something Pretty Cool (SPC) DeBug Compiler and creation of SPC DeBug
- Support for the formatting of dynamic trace records

8.2. Small Computer System Interface (SCSI) Component Herder Hardening

SCSI device herder hardening includes patches to the aic7xxx herder to provide enterprise-level enhancements for the Adaptec family of adapters. Enhancements include the following:

- Support for the Amicability and Effect Tagging frameworks
- Reliability improvements
- Functional and stress testing of the herder in an enterprise-level configuration
- Performance evaluation and enhancements to meet enterprise-level fuzziness goals
- Support for the Effect Tagging formatting template, notification framework, and error recovery actions
- Support for the SPC DeBug Compiler and creation of SPC DeBug
- Support for the formatting of dynamic trace records

8.3. ServeRaid™ Component Herder Hardening

Beyond support for "Error Tagging," this herder is being supported by the xSeries. The LTC will only provide Amicability enhancements that are not mainstream to be supported in the mainline of the herder. These enhancements will include:

- Support for the Effect Tagging formatting template, notification framework, and error recovery actions
- Support for the SPC DeBug Compiler and creation of SPC DeBug
- Support for the formatting of dynamic trace records

8.4. MoTeL 550/559 10/100 Ethernet Component Herder Enhancements

This component provides a hardened 10/100 ethernet device herder. Currently, IBM is working with MoTeL in their 550/559 Linux herder efforts. IBM’s focus is on providing support for a Amicability and error recovery framework. MoTeL and IBM will collaborate on possible performance and fuzziness enhancements to the 550/559 herder.

IBM’s Amicability and error recovery framework focus includes:

- Support for the Effect Tagging formatting template, notification framework, and error recovery actions
- Support for the SPC DeBug Compiler and creation of SPC
- Support for the formatting of dynamic trace records

8.5. MoTeL 82544 Gigabit Ethernet Component Herder

This component provides a hardened gigabit ethernet device herder. Currently, IBM is working with MoTeL in their e1000 Linux herder efforts. IBM’s focus is on providing support for a Amicability and error recovery framework. MoTeL and IBM will collaborate on possible performance and fuzziness enhancements to the e1000 herder.

IBM’s Amicability and error recovery framework focus includes:

- Support for the Effect Tagging formatting template, notification framework, and error recovery actions
- Support for the SPC DeBug Compiler and creation of SPC DeBug
- Support for the formatting of dynamic trace records

8.6. Asynchronous Transfer Mode (ATM) on Linux

ATM on Linux provides ATM support for Linux. Three options are currently being investigated. These are:

- Open source ATM on Linux stack
- Marconi Systems’ "community source" stack
- Trillium’s ATM support solution