BennuProposal BennuMissionStatement

Bennu Project Goals

Author: Daniel S. Haischt <<MailTo(dsh AT apache DOT org)>>

Date: December 2007

Online version: http://wiki.apache.org/incubator/BennuProposal/BennuMissionStatement

Status: .

> ${renderedContent}

Abstract

The below section are illustrating my personal point of view why I think the Bennu project may be a good addition to the ASF project portfolio. Basically I tried to describe my experiences made as a committer in projects like pfSense or FreeNAS.

Of course the information provided below may be sometimes biased.

Targeted Audience

The goal of the Bennu project is to provide an execution environment for:

- Packet routing
- Packet filtering
- Packet inspection
- VPN tunneling between different Bennu nodes
- Traffic shaping (QoS)
- Wi-Fi access point provissioning

Most of the above functions will be provided by the underlying operating system where Bennu will be responsible to configure and manage each of these functions.

Thus Bennu will be deployed ...

- By ISPs
- Companies wanting to wire their different branches into a virtual corporate network
- Small venues where it is necessary to wire devices temporarily together with a limited budget and a limited time frame (e.g. conferences where a minimal TCO provided by an open source solution is of interest)
- People wanting to do Wi-Fi networking with an emphasis on enhanced security. Bennu provides this by offering a captive portal and advanced security functions.
- By persons wanting to connect their home network to the internet with a central place where incoming and outgoing packets are getting filtered and inspected.

Community Building

Integrate Code Donations(s)

Initially Bennu will be based on a m0n0wall code donation provided by Manuel Kasper. Bennu will then evolve from that code base to a more modularised and flexible software system.

The decision to start with the official m0n0wall code was made because m0n0wall is quite popular in the field of open source based router platforms. Although development of m0n0wall became stale in the last two years. The idea is to continue the m0n0wall efforts to move from a monolithic system to a modularised and extensible system as described by Manuel Kasper in one of his thesis papers.

pfSense or one of the other m0n0wall forks have not been chosen as a code base because they are still heavily based on m0n0wall and it is a rather logical step to just use the original code in such a situation.

Additionally specific functions provided by pfSense for example can be easily re implemented as part of the Bennu system.

Establish a Viable Community

Establish a viable community according to the meritocracy principle described by Michael Young in his book 'Rise of the Meritocracy'.

Trivia: In pfSense there exists basically a aristocracy where one or two eligible person establish a valid road map that defines where the project is moving to in the near future. Where this is in general acceptable it requires pretty knowledgeable and experienced person in regards of team/community building. This was not the case in pfSense. Additionally people were not treated according the principles of a meritocracy (i.e. the more able a programmer seems to be, the higher his position will be).

Establish an Open Community

Any decision no matter whether they are from a technical or community building nature should be transparent to the public. This will be accomplished by only using public communication channels such as public mailing lists to ensure any past decision making may be traced accordingly by using the list archives.

Trivia: In pfSense mailing lists and IRC channels where the actual decision making did take place were private and thus a new developer wasn't able to get an idea why certain things have been setup like they were because there was no list archive available and thus decision making was not documented appropriately. Additionally these private communication channels have been used to discriminate other people which is of course a bad habit.

Join a Network of Experienced Open Source Developers

Bennu as outlined in the project proposal may become a rather complex software system over time. While working on pfSense it became clear that any of the pfSense project member did lack proper knowledge about how to release a new software system within a reasonable time frame. Fixing serious bugs before implementing new feature and the lack of a test driven development methodology have been other issues.

By joining the ASF I hope to join a network of experienced developers who may provide help if experiencing such issues.

Technical Goals

Make m0n0wall Extensible

Provide a Package Management System

Currently m0n0wall does not support extensibility by providing for example a plug in based concept. Projects such as pfSense do prove that extensibility of an router software platform is a feature often requested by users and it additionally eases the contribution of new features to the core system.

Thus an initial goal of the Bennu project will be to provide a a software package manage system that may use the same plug in architecture like it can be found in Mozilla based web browsers.

For example in Mozilla plug ins are described using the Resource Descriptor Framework (RDF) which may, if being enhanced by some meta information, provide information such as 'which software package provides the same functionality as Squid' and a possible answer would be 'Polipo' (same for Snort versus Prelude).

Abstract Software Service Management

Bennu will provide several management interfaces which do allow to:

- manage a concrete software service like ...
    - starting/stopping a service
    - deploying/undeploying a service
- virtualize a software service implementation. For example if a user is using

Squid as a HTTP proxy but Squid would suffer from certain security flaws, Bennu would enable to switch to Polipo as an appropriate alternative without the need to reconfigure the complete router/firewall system because virtualizes the concrete software service implementation to a certain degree.

One rather scientific goal is to evaluate whether the above software service management abstraction may be implemented using a SOA framework implementation such as Apache Tuscany.

Provide Software Service Versioning

In pfSense I had to learn, that it's is not sufficient to provide software service implementations such as a management service for Squid V3.0 for any of the various pfSense branches (1.1, 1.2 and so on). It is necessary to provide versioning of software service which would then allow a user to install a specific version of the Squid management service which may provide a distinct set of features and guarantees various QoS metrics (like reliability).

Provide Software Service Configuration Versioning

While operating a router platform in a test lab it is often necessary to fall back to a previous software service setting. For example it may become clear that a particular setting makes the complete system unreliable and a previous setting did not cause such a behaviour. Thus if operating on a versioned configuration it would be possible to switch back to any configuration setting made in the past.

Provide a Single Point of System Management

If operating a large set of router/firewall installation it may become necessary to manage each of these installations using a single point of system management.

Thus a unified and central management facility will be necessary where Bennu systems may be administered and provisioned (e.g. roll out new OS kernels).