An Introduction to the Amanda
Plans for the Future
Today's Open Source Development Environment

CALUG
August 13, 2008

Dustin J. Mitchell
dustin@zmanda.com

About Me

- Amanda user for about 8 years
- UNIX sysadmin
- K-12 Teacher and IT manager
- FOSS contributor and author
  - (Buildbot, EnterTrack, misc. other patches and enhancements)

- Storage Software Engineer at Zmanda, Inc.
  - Full-time open-source developer
  - One of the most active contributors to Amanda
  - Very interested in cultivating the Amanda developer community

What to Expect

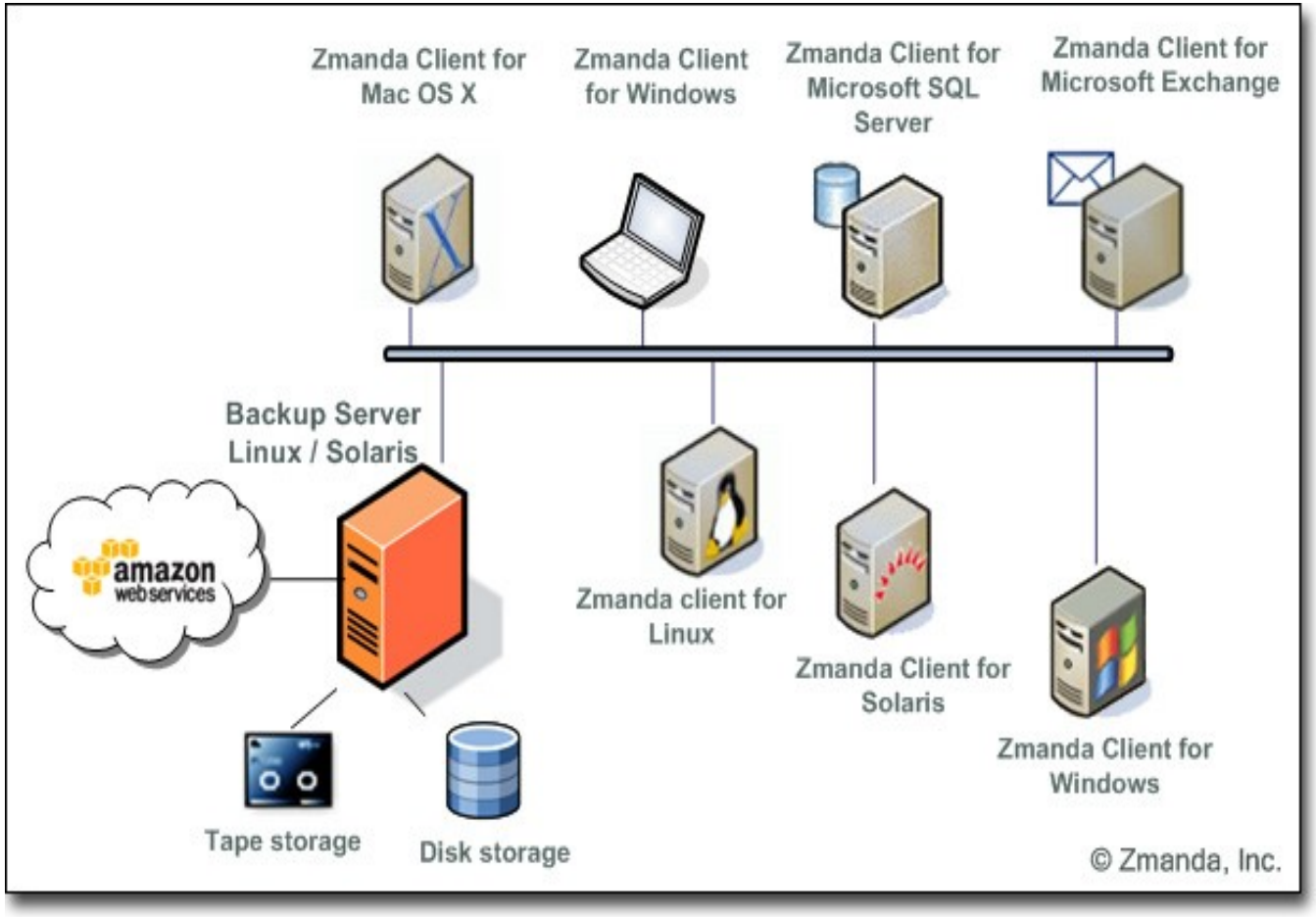
1. What is Amanda and how can I use it?
2. What are the current projects in Amanda, and how can I help?
3. What does open-source development look like today?

Your questions help me to know what interests you, so please interrupt at any time.

About Amanda

- Performs LAN-based, multi-host backups
- Uses native client tools to perform backups and restores
- Supports network-based, file-by-file restores
- Unique backup planner gives consistent backup window
- Holding disk avoids shoe-shining, adds parallelism and resiliency
- Performs disk-to-disk, disk-to-tape, and disk-to-cloud backups
- Battle tested (16 years) with large installed user base
- Has regular, backward-compatible release cycle

LAN-based backups



Native Tools

Data extraction:

- GNU Tar
- Native "dump"
- lots of new possibilities (Application API)

Data storage:

- On-disk and on-tape files are just tar or dump output with a 1-block header:

```
AMANDA: FILE 20080813123541 euclid /A/p/etc  lev 0 comp N program /bin/tar
To restore, position tape at start of file and run:
        dd if=<tape> bs=32k skip=1 |      /bin/tar -xpGf - ...
```

Network-based restores

```
Using index server from environment AMANDA_SERVER (euclid)
AMRECOVER Version 2.6.4+. Contacting server on euclid ...
220 euclid AMANDA index server (2.6.4+) ready.
Setting restore date to today (2008-08-13)
200 Working date set to 2008-08-13.
200 Config set to Conf.
200 Dump host set to euclid.
Use the setdisk command to choose dump disk to recover
amrecover> setdisk /A/p/etc
200 Disk set to /A/p/etc.
amrecover> ls
2008-08-13-12-35-41 amanda/
2008-08-13-12-35-41 .
amrecover> add amanda
Added dir /amanda/ at date 2008-08-13-12-35-41
amrecover> extract
```

Backup Planner

Relax, let Amanda plan things for you!

- "Other leading backup applications" do a full or incremental *run*, using either a little bit of tape or a lot

- Amanda figures out *per DLE* (partition or subdirectory) whether to run a full or incremental each night
  - based on estimated size of fulls and incrementals
  - based on time since last full
  - policy can be specified for each DLE

Result: given enough DLEs, Amanda writes about the same amount of data to storage every night

Holding Disk

Tapes are inherently sequential, so how do you backup multiple clients in parallel?

Amanda spools dumps to *holding disk* in parallel, then writes sequentially to tape.

Advantages:

- local disk can usually keep up with a tape drive, while networked clients can't
- parallelism shortens the backup window (gets backups done before morning)
- slow tape drives need not hold back large backups
- If the tape fails, backups are kept in holding -- your data is still safe

Pluggable Storage Backends

- Tape devices (using OS's native support)
- Disk (files in subdirectories)
- Amazon S3
- RAIT

.. more are on the way

Basic Amanda Configuration

- Start with a configuration, ServerNightly

/etc/amanda/ServerNightly

- *disklist* contains a list of DLEs

```
euclid /usr user-tar
euclid /var user-tar
euclid /var/www user-tar-high

darwin /usr user-tar
darwin /etc user-tar
darwin /var user-tar
```

Basic Amanda Configuration

- /etc/amanda/ServerNightly/amanda.conf: (excerpted)

```
# scheduler parameters
inparallel 4
dumpcycle 2 weeks
tapecycle 20 tapes
runtapes 1

# device parameters
tpchanger "chg-multi"   # the tape-changer glue script
changerfile "/etc/amanda/S3/changer.conf"
tapedev "s3:1F0KNSSJ3R0VYXZ0PMG2-backup/slot-01"
device_property "S3_ACCESS_KEY" "1F0KNSSJ3R0VYXZ0PMG2"
device_property "S3_SECRET_KEY" "dddf9DFS098mdf0980291m.."
```

Basic Amanda Configuration

```
# holding disk
holdingdisk hd1 {
    comment "cheap SATA drive"
    directory "/var/amholding"
    use -20 Mb # all but 20 Mb
}

# dumptypes
define dumptype user-tar {
    global
    program "GNUTAR"
    comment "user partitions dumped with tar"
    priority medium
}
```

Basic Amanda Configuration

Run amcheck to test:

```
$ amcheck ServerNightly
Amanda Tape Server Host Check
-----------------------------
Holding disk /var/amholding: 655722 MB disk space available,
   using 655702 MB
slot 5:read label `ServerNightly005', date `20080813001502'.
NOTE: skipping tape-writable test
Tape ServerNightly005 label ok
Server check took 2.992 seconds

Amanda Backup Client Hosts Check
--------------------------------
Client check: 2 hosts checked in 1.204 seconds.  0 problems
   found.
```

Basic Amanda Configuration

Set up your schedule in crontab:

```
0 18 * * * amanda /usr/sbin/amcheck -m SeverNightly
15 0 * * * amanda /usr/sbin/amdump ServerNightly
```

amcheck sends an email if there are errors

amdump sends a report if an email address appears in amanda.conf

```
Hostname: euclid
Org     : S3
Config  : S3
Date    : August 13, 2008

These dumps were to tape S3005.
The next tape Amanda expects to use is: S3006.


STATISTICS:
                          Total       Full      Incr.
                        --------   --------   --------
Estimate Time (hrs:min)     0:09
Run Time (hrs:min)          1:08
Dump Time (hrs:min)         0:07       0:01       0:06
Output Size (meg)          393.6       23.0      370.7
Tape Time (hrs:min)         0:57       0:03       0:54
Tape Size (meg)            393.6       22.9      370.7
Tape Used (%)               29.2        1.7       27.5   (level:#disks ...)
Filesystems Taped             17         13          4   (1:3 2:1)
Avg Tp Write Rate (k/s)    117.3      120.3      117.1

USAGE BY TAPE:
 Label       Time      Size      %    Nb    Nc
 S3005       0:57      394M    29.2    17    17
```

Amanda Development

- Goals
  - Pluggable architecture
  - Support parallelism (multiple devices, multiple backup servers)
  - Append to tape
  - Support ILM techniques
    - move dumps between tapes (e.g., send all level 0's offsite weekly)
    - D2D2T or D2D2{cloud}
- Constraints
  - Backward compatibility
    - configuration
    - client/server communication
    - tape format and indices
  - Portability (runs on any UN*X, including Cygwin)
  - Minimal prerequisites, especially for the client
  - Manpower

Amanda Hackers' Challenge

Rewrite a mature, complex application

in a new language,

while adding significant new functionality,

without breaking backward compatibility,

in small steps.

Strategy

- Rewrite bits and pieces, bottom-up
  - Rewrite tools/applications one by one
  - SWIG Amanda modules "on demand"
  - Incrementally improve interfaces
- Test-driven development
  - Unit Tests
  - Continuous Integration testing (Buildbot)
- Mandatory code review
- Fully functional application at every commit

Ideas

- FTP Device (Dreamhost, etc., or NAS)
- .Mac iDisk
- Interface directly with VTLs, other big iron?

Projects

- No devices actually implement locking yet
- Changer API is not well-integrated
- Add selectable mirroring and striping modes to RAIT

Device API

Pluggable Interface to Backend Data-Storage Devices

Goals:

- Volume Append
- Delete Individual files from a volume
- Locking to avoid device contention
- Variable block sizes
- Device-specific properties (get and set)
  - Access modes (read/write, read-only, WORM, write-only)
  - Block Size (for reads and writes)
  - Device Capabilities

Examples

Tape Device:
All OS-supported linear tape devices (Amanda is not device-specific)

RAIT Device:
Redundant Array of Independent Tapes (can mirror any Device)

S3 Device:
Use Amazon S3 to store data offsite cheaply and reliably

VFS Device:
Treat disks as tapes (also known as 'vtapes')

Optical Device:
CD-ROM, DVD-ROM, etc.

zmanda
Open Source Backup

Implementation

- XML for extensible communication
- Well-defined set of application operations
- Abstract the notion of "files" into "user objects"
- Pre & Post scripts

Provides some interesting development problems

- Indexing byte positions after encryption and compression
- Backward-compatibility with existing Amanda clients

Application API

Pluggable Interface to Client-side Dump and Restore Applications

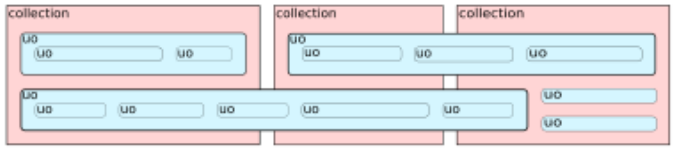
Goals:

- Send only required data to client on recovery
- Enable new applications
- Maintain backward compatibility
- Support native-tools restores
- Simplify implementation of new applications

Data Organization

User object: restorable item (e.g., file, directory)
Collection: unit of output data from application

- User objects may contain other user objects
- Collections may contain many user objects.
- User objects may span multiple collections.



Examples

GNU Tar:
User Object: file or directory
Collection: 512b tar block

Native dump/restore:
User Object: file or directory
Collection: entire dump datastream

SQL Database:
User Object: table
Collection: database dump

zmanda
Open Source Backup

Ideas

- Database engines (MySQL, Postgres, etc.)
- Schilly Tar (star)
- Cisco configuration (IOS)
- Windows client
- Email applications
  - MS Exchange
  - Cyrus imapd
- NetApp's NDMP
- ZFS
- Source Code Repos (svn, git, hg)

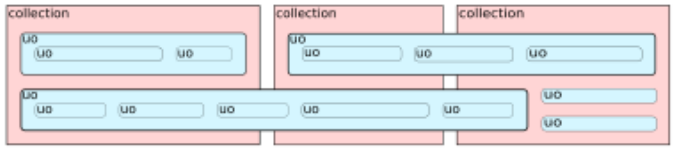
Application API

Pluggable Interface to Client-side Dump and Restore Applications

Goals:
- Send only required data to client on recovery
- Enable new applications
- Maintain backward compatibility
- Support native-tools restores
- Simplify implementation of new applications

Data Organization

User object: restorable item (e.g., file, directory)
Collection: unit of output data from application
- User objects may contain other user objects
- Collections may contain many user objects.
- User objects may span multiple collections.



Examples

GNU Tar:
User Object: file or directory
Collection: 512b tar block

Native dump/restore:
User Object: file or directory
Collection: entire dump datastream

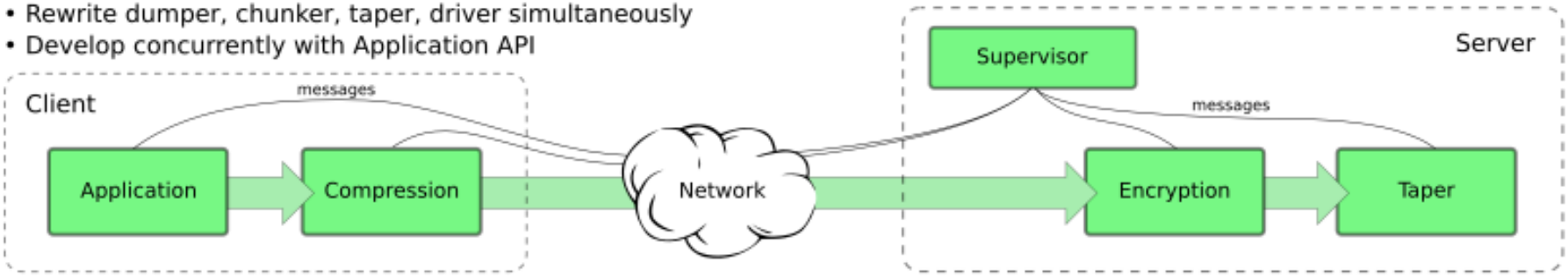
SQL Database:
User Object: table
Collection: database dump

Transfer Architecture (XFA)

Goals:
- Improve bandwidth (minimize copies, maximize concurrency)
- Allow multiple concurrent transfers
- Enable data migration (e.g., D2D2T)
- Provide consistent 'toolkit' for dumps, restores, migrations
- Support Application API in sending minimal data for recovery

Implementation:
- Rewrite restore applications first
- Rewrite dumper, chunker, taper, driver simultaneously
- Develop concurrently with Application API

Architecture:
- A **transfer** is composed of source, filter, and dest **elements**
- Elements are joined efficiently: pipes, shmem, etc.
- Elements report progress via **messages** for indexing, status
- Datastreams are indexed in **slices**
- Filters perform encryption, compression, deduping, etc.
- All data-handling code is optimized C
- A transfer can span a network connection



C and Perl APIs are both complete and functional; working on implementing data migration

Why?

- Encourage fine-grained patches from users
- Encourage new developers
- High-level operations are hard in C
- Scripting languages come with lots of utils
- We were basically doing a rewrite already

Why Perl?
(Python? Ruby? PHP? Shell?)

- Admins tend to know Perl
- Perl is a fairly low-level language
- Perl is more commonly installed on base systems (minimum perl-5.6.0)

Why not Perl?

- Terrible support for large integers
- OOP support is very weak

Perl in the Core

New functionality will be written in Perl

Goals:

- Support other projects' large-scale changes
- Simplify use of XML, Unicode, etc.
- Write high-level code in a high-level language
- Encourage new contributors and developers
- Maintain backward compatibility

Check it out!
amanda.svn.sf.net/viewvc/amanda/amanda/branches/perl

How?

- Efficiency-critical components (XFA) will remain in C
- Existing libraries will be interfaced to Perl (using SWIG)
- Tools will be rewritten one-by-one in a gradual process

Other Benefits:

- Unit tests for all modules, tools (using Test::Harness)
- Cleanup and documentation of existing APIs

Example

```
use lib '@amperldir@';
use Amanda::Device qw( :ReadLabelStatusFlags );

sub try_read_label {
    my ($device_name) = @_;
    if (!$device_name) {
        return $READ_LABEL_STATUS_DEVICE_MISSING;
    }
    my $device = Amanda::Device->new($device_name);
    if (!$device) {
        return $READ_LABEL_STATUS_DEVICE_MISSING
             | $READ_LABEL_STATUS_DEVICE_ERROR;
    }
    $device->set_startup_properties_from_config();
    return $device->read_label();
}

my $flags = try_read_label($ARGV[1]);
print join("\n",
           ReadLabelStatusFlags_to_strings($flags));
print "\n";
```

Join us!

- Become an Amanda user
  - Test new releases
  - Answer questions on the mailing lists, IRC
  - Improve Amanda's documentation
- Contribute patches
  - Fix a bug you've noticed, or add a feature you'd like to see
  - Complete a coding task (see "Tasks" on the wiki)
  - Amanda devs will help you with any problems
- Become an Amanda developer
  - Set your own goals for the project
  - Engage with other developers in design discussions
  - Become famous like Linus Torvalds or Larry Wall (maybe)
- Work for Zmanda
  - Get paid to write open-source code!

Open Source Communities

About Zmanda

- Provider of open source backup and recovery products and solutions
  - Amanda Enterprise Edition
  - ZRM-MySQL (MySQL backup and recovery)
  - ZMC (GUI interface)
  - Various Amanda plug-ins
    - Native windows client (+Exchange, MS-SQL, etc.)
    - Oracle Client
    - ZFS Client
    - MySQL, Postgres clients
  - BackupPC (coming soon)
- Active developer and supporter of Open Source backup and recovery tools
- Products and support offered on an annual subscription basis
- We also offer professional services and solutions based on the enterprise products.

What is an open-source community?

- How do I "belong"?
- What makes a "strong" community?
- "User community" vs. "Developer community"?
- How do communication mechanisms affect this?
  - IRC
  - Email
  - IM
  - Conferences (e.g., PyCon, ApacheCon)
  - Location (Silicon Valley, MIT, etc.)
- How can a community sustain its resources?
  - web hosting, wikis, mailing lists, subversion hosts, etc.
- Why do some projects have large development communities, while other communities are very small?

Voice and power

Who gets to make the decisions?

- Main developer makes the call (Linus and his inner circle)
- "Rough consensus and working code" (IETF)
- Voting schemes (e.g., Apache Software Foundation)
- Relative power of individuals vs. companies (Zenoss, Zend, Zmanda)

Who takes the legal risks?
Who makes the contractual representations?

Do I have any obligation to my users?

"If you want that feature, patches are welcome!"

- Is this an invitation or an exclusionary technique?
- OK for documentation? What about code?

Example (forum post from today)

Just found that this feature (appending to tape) is not yet implemented in Amanda (it's just ridiculous!), but there are claims that this "is something we're actively working to support" [links].

So I have a question for amanda developing team: When is it going to be implemented (approx)? Depending on the answer, I'll decide wheather to wait for the solution, or to switch to another backup software like Bacula.

Access and Exclusion

- Knowledge
  - How good is the documentation?
  - Where do newcomers find institutional knowledge?
  - Do I have to "speak the language"?
- Explicit barriers
  - Commit rights
    - Tigris.org requires a recommendation from another committer
  - Copyright assignment (e.g., FSF)
  - Tests and quizzes (Gentoo ebuild quiz)
- Developers' attitude to newcomers
  - Will the developers accept my patches?
  - Will I get help when I'm stuck?
- Threshold difficulty for entry
  - Are there "easy" tasks to demonstrate competency?

Building a Development Community

My ideas:

- Lower barriers as much as possible
  - More accessible language (Perl)
  - Developer documentation (comments and wiki articles)
  - List of suggested "easy" tasks (on wiki)
- Welcoming attitude
  - Suggest opportunities for users to contribute (move from user community to developer community)
  - Offer help to anyone working on patches
- Speaking at LUGs, BUGs, conferences, etc.

What are your suggestions?

Appendices

Seeking Filtered Data

A dump is essentially:

tar -cf - . | gzcat | aespipe | taper

We also gather information on which 'tar' blocks are required to restore each file.

Q: Given a filename, which blocks does the server need to read off tape to restore that file?

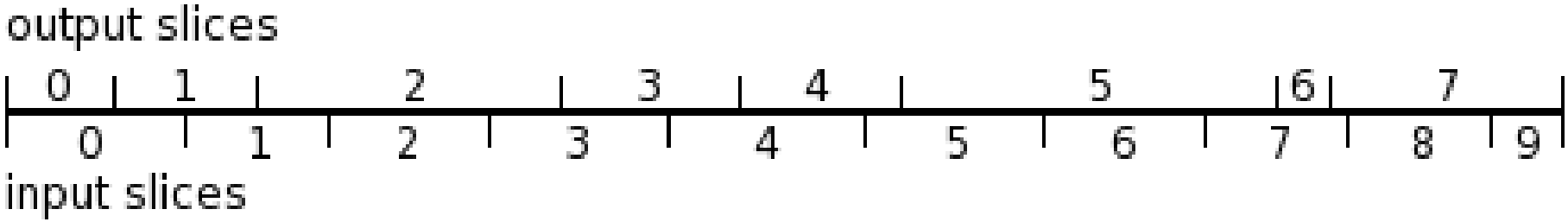
Seeking Filtered Data

Let's look at the boundaries between transfer elements: a bytestream.

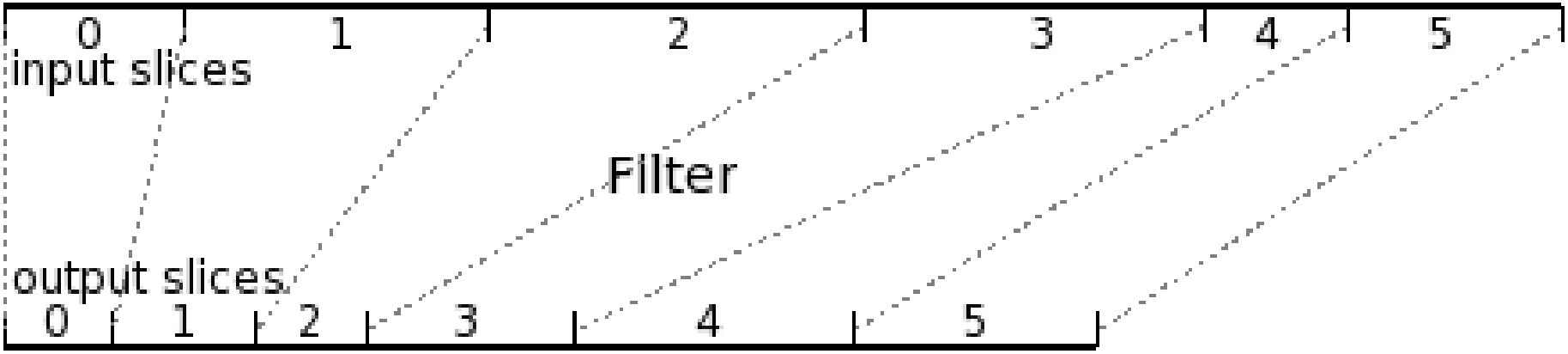
Upstream element divides the bytestream into "output slices" (a byte position interval)

Downstream element divides the bytestream into "input slices"

Slices do not necessarily correspond



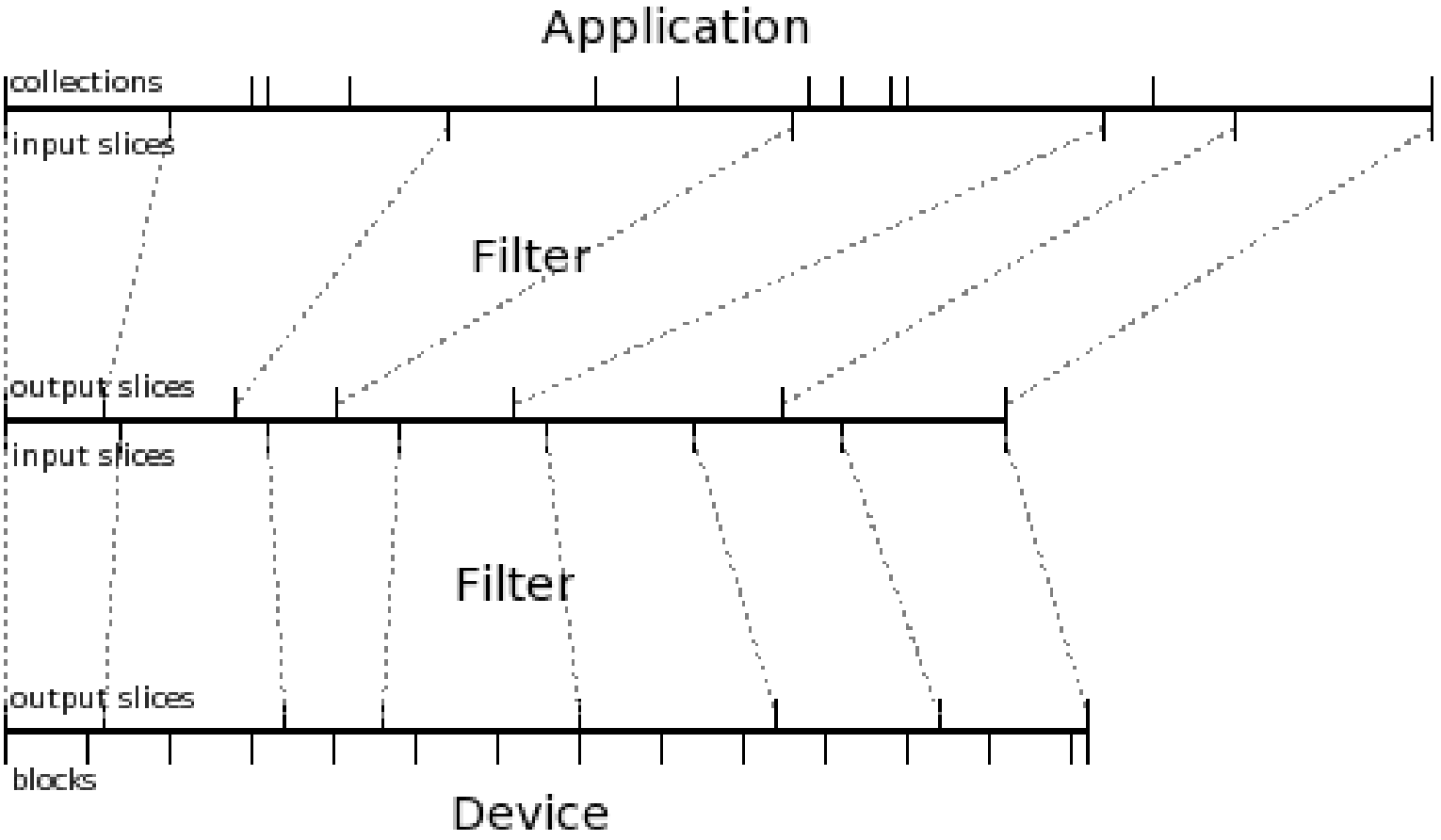
Seeking Filtered Data



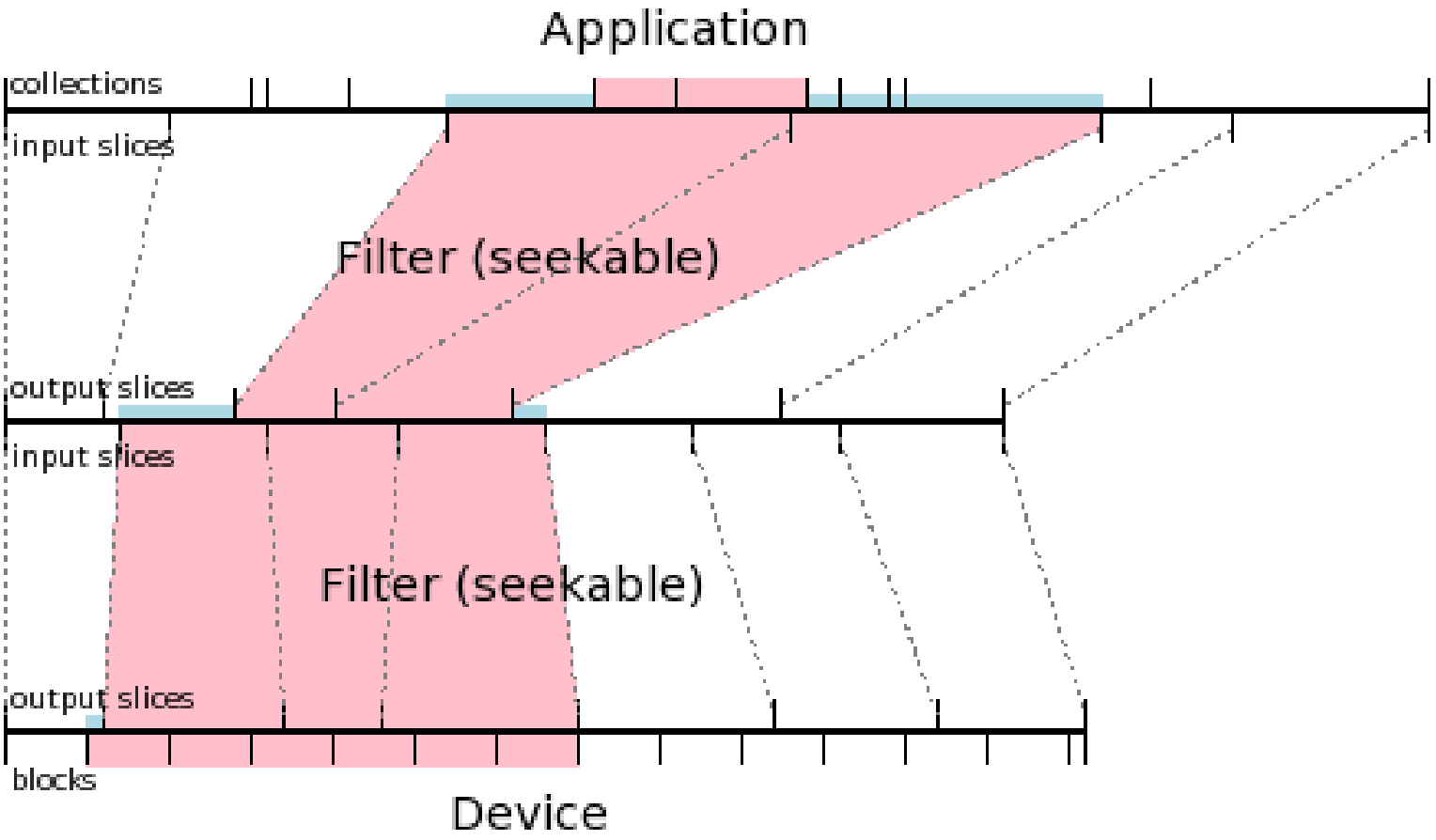
Note that input and output slices *correspond one-to-one*

seekable filter - can start at an arbitrary slice during restores
catenary filter - filter is distributive over concatenation

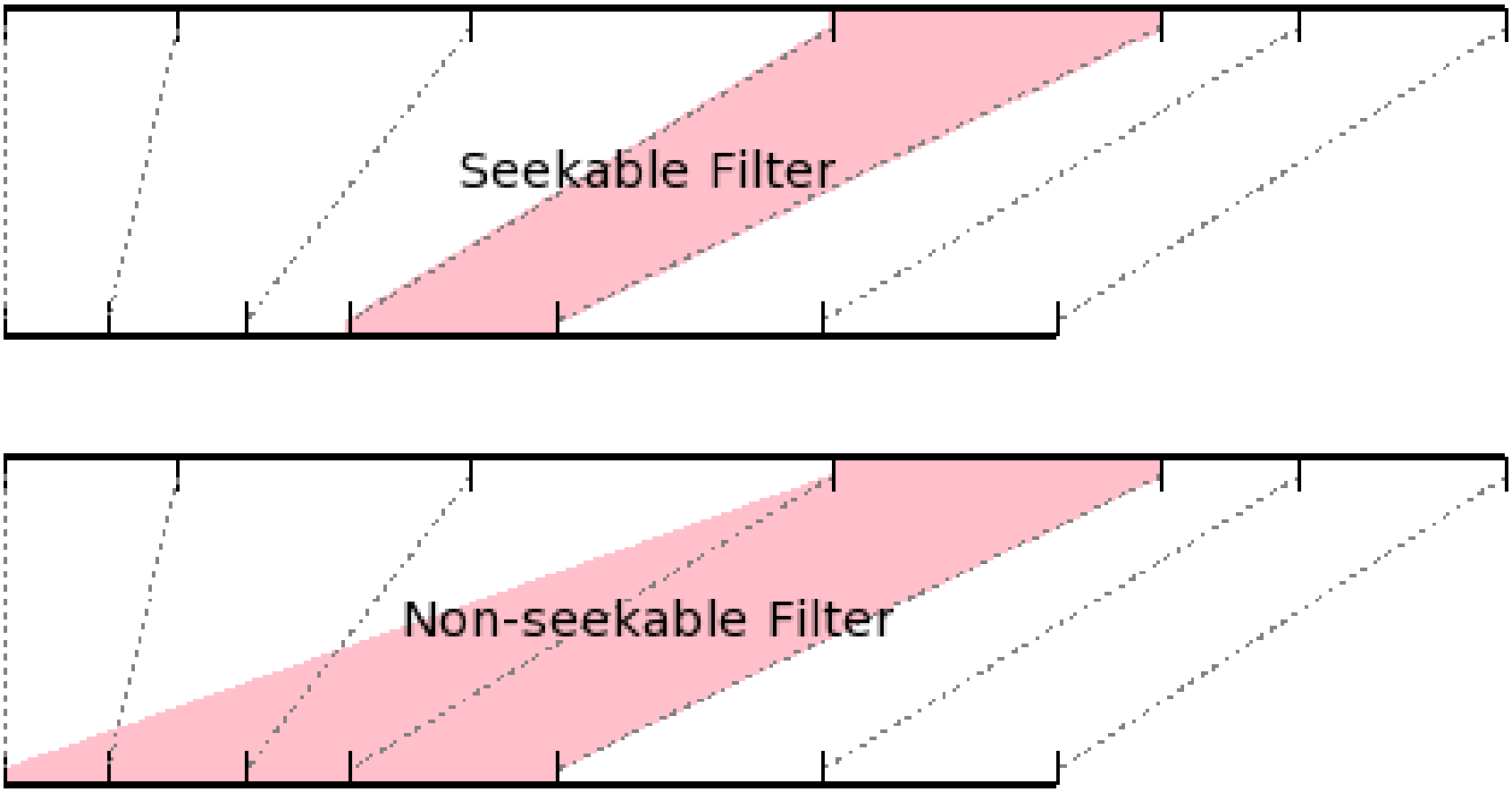
Seeking Filtered Data



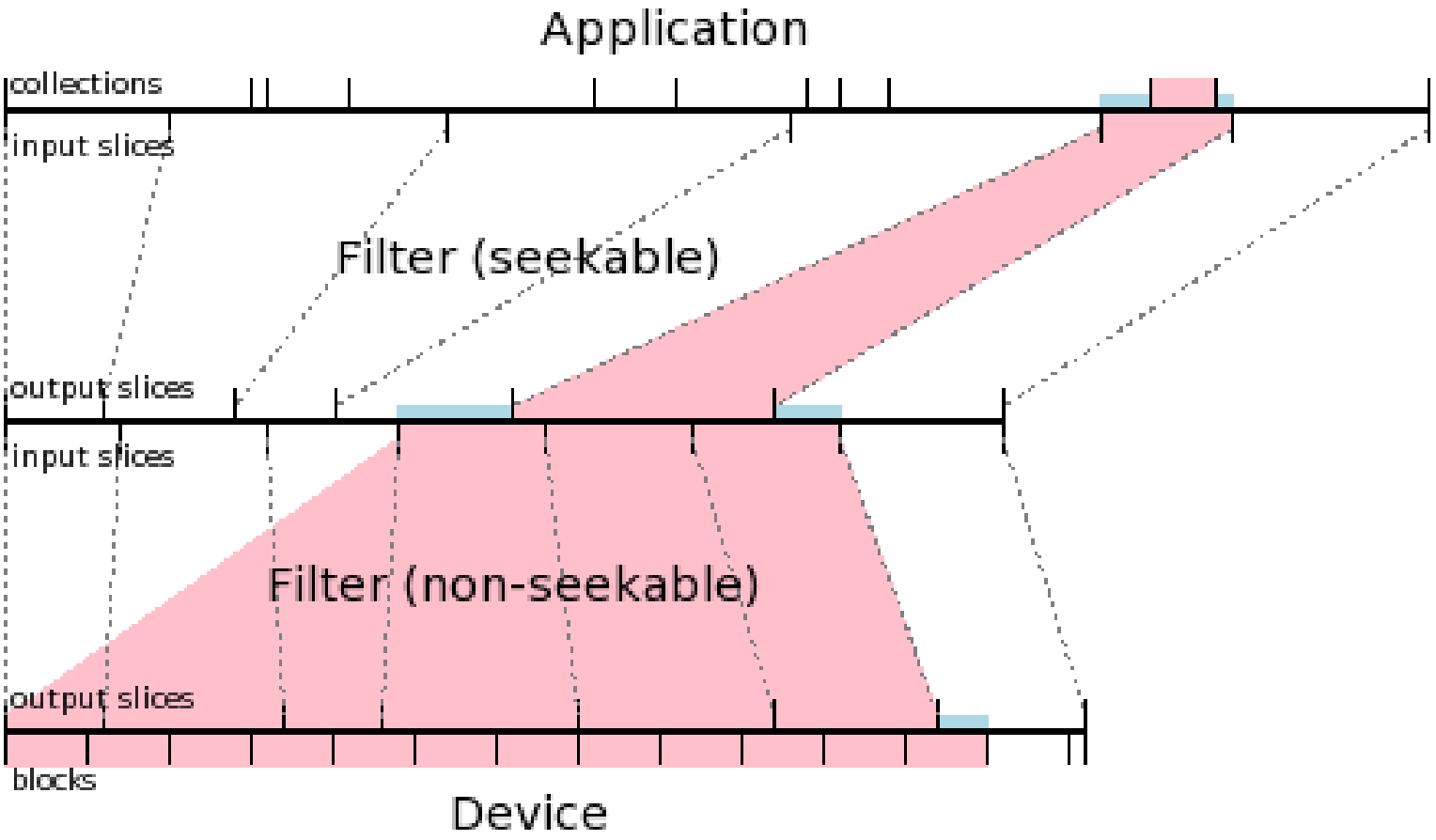
Seeking Filtered Data



Seeking Filtered Data



Seeking Filtered Data



Seeking Filtered Data

Seekable filters are required to allow quick restores.

Catenary filters are required to support more than one slice per bytestream.

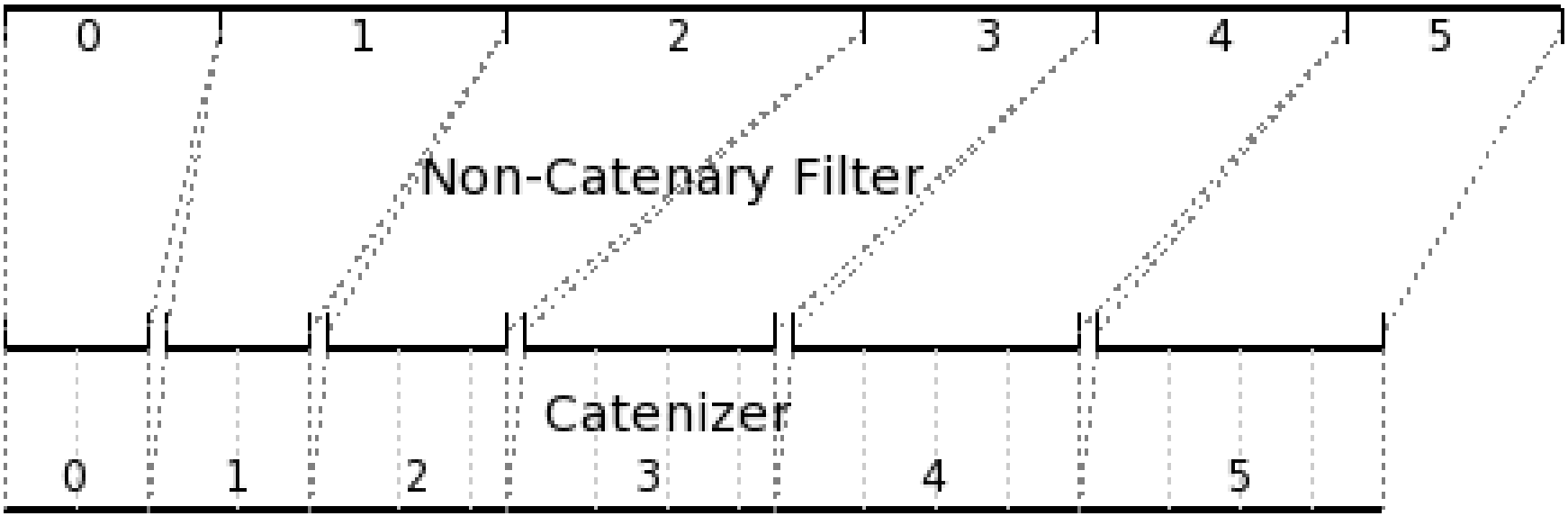
- bzip2: seekable, catenary
- gzip: non-seekable, catenary
- aespipe (or any encryption filter): non-seekable, non-catenary
  - Problematic for restores without indices

A non-seekable, catenary filter can be made seekable by re-invoking the filter for each slice, e.g., gzipping each slice independently.

So we need a way to make an arbitrary filter catenary...

Seeking Filtered Data

catenizer embeds just enough information in the output bytestream to know when to restart the non-catenary filter when doing a restore, even when the index data is unavailable (native-tools restore, re-index operation)



Xfer Architecture

Xfer Architecture

- Similar to GStreamer
- Hook up a series of *Transfer Elements* into a *Transfer*
- Transfer runs in one or more non-main threads
- Elements send *Transfer Messages* back to the main thread asynchronously
- Transfers can be canceled (code is pending review by Nikolas)

Element Linking Mechanisms

- READFD
- WRITEFD
- PULL_BUFFER
- PUSH_BUFFER

Element Linking

Device Source --> Gunzip --> File Destination

- Device Source
  - nothing -0> PULL_BUFFER
- Gunzip (filter; spawns a separate process with pipes)
  - WRITEFD --> READFD
  - PULL_BUFFER --> PULL_BUFFER (future; in-process gzip invocation)
- File Destination
  - WRITEFD --> nothing

So how do we glue this together?

Device --PULL--> **glue** --WRITEFD--> **Gunzip** --READFD--> **glue** --WRITEFD--> **File**

(hilighted areas indicate threads)

Amanda::MainLoop - async programming

```
use Amanda::MainLoop;

# set up an event source
my $to = Amanda::MainLoop::timeout_source(2000);

# attach code to execute when the event occurs
$to->set_callback(sub {
    print "Time's Up!\n";
    $to->remove();
    Amanda::MainLoop::quit();
});

# start running
Amanda::MainLoop::run();
```

Amanda::Xfer

```
my $infd = POSIX::open("input", POSIX::O_RDONLY, 0);
my $outfd = POSIX::open("output", POSIX::O_CREAT|POSIX::O_WRONLY, 0640);
my $xfer = Amanda::Xfer->new([
        Amanda::Xfer::Source::FD->new($infd),
        Amanda::Xfer::Dest::FD->new($outfd)
    ]);
$xfer->get_source()->set_callback(sub {
    my ($src, $xmsg, $xfer) = @_;
    print "Message from $xfer: $xmsg\n"; # use stringify operations
    if ($xfer->get_status() == $XFER_DONE) {
        $src->remove();
        Amanda::MainLoop::quit();
    }
});
$xfer->start();
Amanda::MainLoop::run();
```