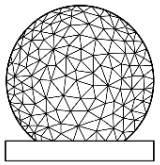
Mark6 Operations

11th IVS TOW Workshop

Chester “Chet” Ruszczyk
chester@mit.edu



MIT
HAYSTACK
OBSERVATORY

Overview

- Mark6 general information
- Mark6 OS update
- Configuration
- Questions / answers for users

MIT
HAYSTACK
OBSERVATORY

Mark6 General Information

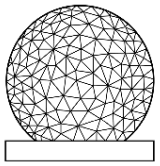
- Guaranteed 16Gbps recording rate to 4 disk modules for 100% utilization of the disk modules capacity
- 1 disk module supports >= 4 Gbps
- As the module's HDD start to fill the recording rates drop
  - Hence only 4 Gbps is guaranteed
- VGOS Mark6 recording strategy (started in 2016)
  - Based upon shipping costs and limited budgets at institutions
  - Approach for VGOS observations
    - 30 seconds recording @ 8Gbps
    - Estimated slew times to next source estimated to be 30 seconds
    - Recording with buffering was verified to a single Mark6 module
      - Reducing number of modules required for shipping
  - Results in the famed buffering issues with a Mark6 – Which is not an issue.
    - Premediated for operational considerations

Mark6 General Information

- The result is that as the module fills data is buffered and must drain before the next recording starts
- This is finalized with the PCFS issuing a record=off command to the mark6.
  - If the files are not already closed, which in most cases they are, the data plane flushes the memory and closes the files with the data written.
- *NOTE*
  - If the proper number of disk modules are utilized, e.g. 2 modules for 8 Gbps, this is not required due to the sustained data rate.
  - Example of recent EHT run at 16 Gbps observing pulsar observation with 4 disk modules
    !scan_info?0:0:1234:174:psrtst_Aa_No0051:recorded:2021y119d03h01m10s:1500:2:0;
    - That's a 25 minute scan that resulted in .001% packets loss.

Mark6 Operating System Update

- Operating systems
  - After 2020 Mark6s purchased from Conduant has:
    - CentOS7 distribution
  - Why CentOS7?
    - NASA required a supported OS distribution with security updates
    - Legacy driver support for Myricom 10G NIC
    - Debian / Ubuntu would require a rewrite of the Myricom driver
    - Decided on CentOS.
  - CentOS7 will be supported thru 2024



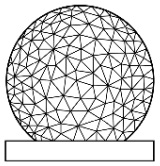
MIT
HAYSTACK
OBSERVATORY

How to upgrade to CentOS7

- Send a 1TB enterprise HDD, and we will duplicate it.
- We are working on a kickstart file
  - Allows you to install a fresh version,
  - Verifying the additional installation of Mark6 software / configuration.
- We can provide you the instructions for a manual installation
  - Required packages,
  - Configuration changes,
  - Software.

Mark6 Operating System Update

- Requires familiarization with Red Hat package manager – yum
  - Recommend downloading yum cheat sheet from RH.
- Updates to distribution different then Debian (as you all know)
  - yum update != apt-get update
- We recommend only updating security patches
  - This usually does not impinge on the kernel distro
    - yum update-minimal --security –y
  - or
    - editing the file /etc/yum.conf
    - and adding the line:
      - exclude=kernel*



MIT
HAYSTACK
OBSERVATORY

Notes:

- Note, Haystack correlator is in the process of upgrading from Debian to CentOS
  - This will take several months
- Be aware that earlier versions of Mark6 with Debian uses version 2 for XFS file system formatting of HDD of modules
- CentOS and supported kernels use version 3 XFS file system by default
  - They are not backward compatible
- Version of cplane on CentOS forces the formatting of the XFS for version 2 support.
  - Backward compatable
  - Other software may not

Hardware Update

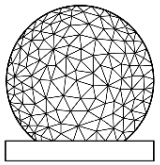
- NIC replacement
  - For systems with sfp+ connection we recommend:
    - HotLava TAMbora 40G2S NIC
  - Myricom NICs have been EOL for 2 years.
  - We do not have a CX4 replacement yet
    - Expected all backends to be updated and using SFP+ interfaces
    - Looking into CX4
      - Based upon NASA station support requirements

Additional Information

- Mike and Alex did a great job on issues they have encountered during operations
  - Yes they will hate me for this! :D
- If you have any problems, they are a great resource to contact first
- If they have not seen it they will bump it up to the appropriate expert
- Due to time limitations, on the following slides I have included a FAQ
- When the notes are released please check out.

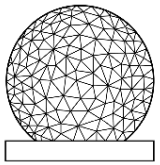
FAQ Summary

- Why does boot up fail with disk modules keyed on?
- What is the Mark6 configuration file used for?
- What if we want to update the kernel?
- Should we condition disk modules?
- How to interpret conditioning output?
- Are modules slots persistent?
- We keyed of the module, but still see disk module information, why?
- Data is not be recorded, why?
- What if our backend does not have packet serial numbers?
- What is subgrouping?
- Restrictions on subgrouping and grouping of modules?
- Playing back Mark6 modules, are there restrictions, options?



Why does boot up fail with disk modules keyed on?

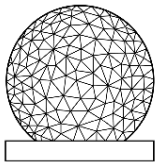
- SAS controller cards bios executes before motherboard bios
  - Enter and disable boot up from disks attached to Controllers.
    - Now if the system reboots with disk modules keyed on
    - It will not look for a master boot record on the disk modules
    - It will boot normally and not hang since no OS is found



MIT
HAYSTACK
OBSERVATORY

Configuration File

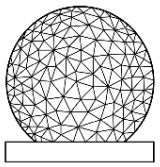
```
# This file is sourced by /bin/sh from /etc/init.d/dplane
Defined in file /etc/default/mark6
# Options to pass to mark6 which take effect with restart.
# This specifies the ethernet ports to be used for incoming traffic.
# (Up to 4 ports are supported; You must list only the ones actually to be used.)
MK6_OPTS=eth2:eth3:eth4:eth5
MK6_DRVR=myri10ge
# Specifies the running directory--both planes log by default there.
MK6_RDIR=/var/log/mark6
# dplane log level
MK6_DLOG=2
# cplane log level (Information, level 0 is debug)
MK6_CLOG=1
# process umask
MK6_MASK=0002
```



MIT
HAYSTACK
OBSERVATORY

Q: What if we want to update to the latest Linux kernel?

- Requires:
  - Minimum - that the Myricom driver be rebuilt
  - Worst case – Myricom driver update required
    - Performance testing is then suggested 16 Gbps to 4 modules.
- If you use the default Mark6 software
  - pf_ring must be rebuilt
  - dplane must be rebuilt
- Instructions will be available after TOW
  - The latest kernel update



MIT
HAYSTACK
OBSERVATORY

Q: Conditioning of disk modules?

- Do we still have to condition disk modules like with the Mark5 systems?
- It is recommended:
  - New modules – twice, to catch HDD infant mortality and bad backplanes
  - Modules – performance degradation and failures
- The script is called “hammer” and you can perform it on 1 or 4 modules at once.
  - Requires cplane to be operational
- “hammerplot” displays the write and read performance of the HDD in the module.

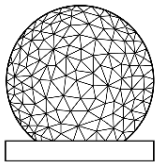
Q: How to interpret the hammerplot output?

Q: Is module slot persistency required?

- If I have a recording of a group of modules in slots 1 and 2, they are removed, and I want to add more recordings to the modules do they have to go in the same slots?
- No
  - The reason being the information on the groupings is stored in the meta data for the modules.
- At correlators receiving modules, they just have to be in the same Mark6 unit for playback.

Disk Modules

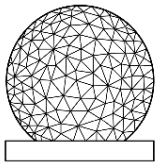
- Configured as RAID0 or scatter gather
  - Recommend using scatter gather mode
- How to initialize a new module
  - mod_init = slot : number disks : MSN : sg : new
- How to remove a module from a group
  - mod_init = slot : number disks : MSN : sg : null
- How to erase
  - group = unprotect : slot
  - group = erase : slot



MIT
HAYSTACK
OBSERVATORY

Disk Modules (cont)

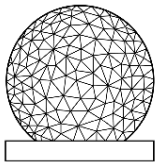
- Insert module in slot
- Connect cables
- Power -Turn key
  - Takes about 25 secs for module to be recognized by Linux kernel
    - Watch lights on module
  - Wait before querying on the module status
    - mstat ? all
    - mstat ? slot
  - Requires 8 disks in module
    - cplane will not be happy with less
    - Note some say this is a bug, we say require good modules
      - Revisiting philosophy based on 2 years of operation



MIT
HAYSTACK
OBSERVATORY

Disk Modules (cont)

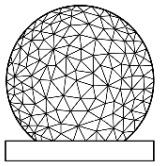
- Removing disks
  - group = close : slot
  - group = unmount : slot
    - Can verify using linux command df to see if modules are truly unmounted
  - turn key to remove power
    - Note it takes about 5 seconds for power to be completely removed
    - Remnant of Mark5 systems (we refer to as the Dan delay)
    - Any commands issued may still see HDD with power
  - query the module status
    - mstat ? all
    - mstat ? slot
  - Bug if you mstat? before turning off power
    - The meta data of disk 0 will be remounted



MIT
HAYSTACK
OBSERVATORY

Recording

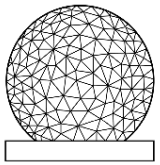
- Data is not being recorded (cont)
  - No data is being received on the interfaces
    - /sbin/ifconfig | grep -i “rx packets”
      - to see if the receive packet counters are incrementing
  - A group is not open for recording
- Why does cplane commands return two status fields?
  - The first is the vsi-s return code
  - The second is a cplane specific return code
    - Specified in command set
    - (see next slide)



MIT
HAYSTACK
OBSERVATORY

cplane return codes

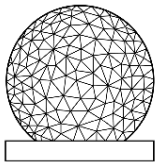
Mk6 return code	Command	Description
2		Specified group not open
10-19	delete	
20	execute	Invalid Action
21	execute	No filename provided
22	execute	Inconsistent filename used for append/finish process
23	execute	Duplicate filename
24	execute	Invalid upload sequence
25	execute	Attempted removal of non-existent xml file
30	group	Attempted open of multiple groups
31	group	Attempted open of incomplete group
32	group	'unprotect' not issued immediately before 'erase'
33	group	'auto' option failed, only supports module types initialized as scatter / gather and not RAID
34	group	Attempted group open does not match subgroup defined in 'input_stream' configuration
40-49	gsm	
50-59	gsm_mask	
60	input_stream	Invalid subgroup declaration (group already open)
61	input_stream	Writing of subgroup meta data to disc failed
62	input_stream	Adding stream label failed, it already exists
63	input_stream	Specified stream label cannot be deleted it was not configured
64	input_stream	Committing configuration to dplane failed, not in an
65	input_stream	Commit failed, invalid sub-grouping compared to the open group_ref



MIT
HAYSTACK
OBSERVATORY

Recording (cont)

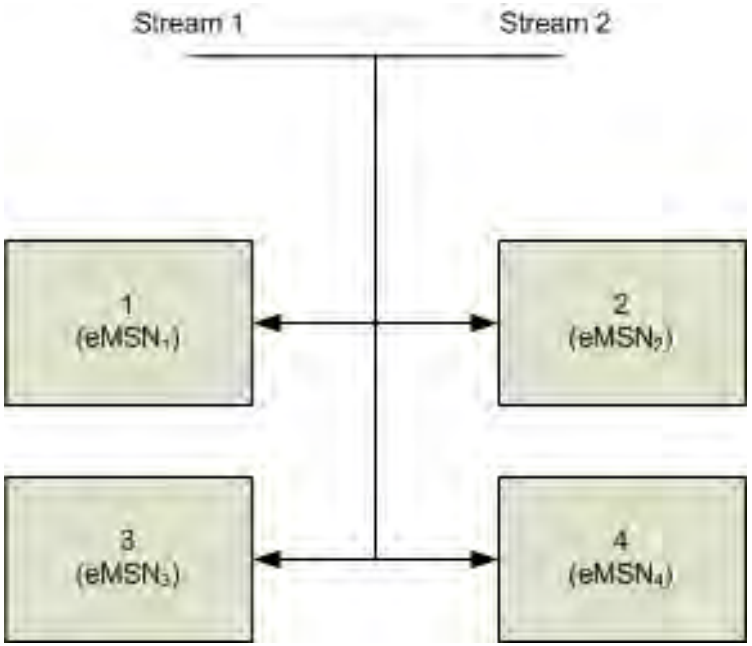
- Our data does not have PSN's how do I turn of checking?
  - set psn_offset to 0, this disables checking
- How can I check what vdif time is being received by dplane
  - use dpstat utility
  - turn on debug level logging on cplane and look at the log files
- Can you abort a recording?
  - Yes, record=off
  - Will close any open files



MIT
HAYSTACK
OBSERVATORY

Subgroup Feature

- Mark6 normal recording mode
  - group_ref = 1234
    - 4 disk modules open for recording
  - 2 input streams defined for receiving data
    - e.g. eth2, eth4
  - 8Gbps / input stream
  - 16 Gbps is written to all disk modules in group_ref

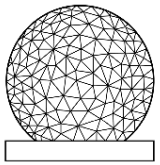


MIT HAYSTACK OBSERVATORY

Subgroup Feature (cont)

- Imagine if each Ethernet port receives a different polarization
  - eth2 <= RCP, eth4 <=LCP

- For existing Mark6 software if correlating a specific polarization, e.g. RCP
  - Requires all 4 disk modules to be inserted at correlator for processing.

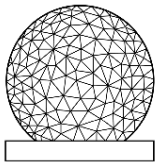


MIT
HAYSTACK
OBSERVATORY

Subgroup Feature (cont)

- If one disk module is lost in shipment both RCP and LCP are lost (25% of data lost).

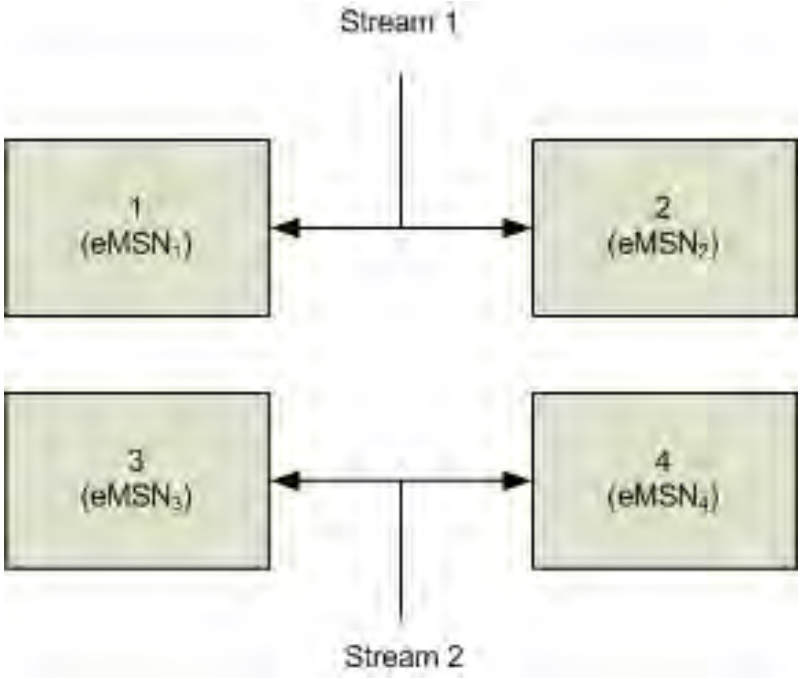
- The subgroup feature allows you to specify A specific input stream to be written to a “subgroup” of disk modules within the group_ref
  - granularity of 8 disks



MIT
HAYSTACK
OBSERVATORY

Subgroup Example

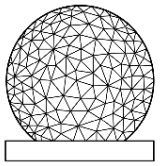
- group_ref = 1234
  - 4 disk modules open for recording
- input “Stream 1”
  - 8Gbps (RCP)
  - written to disk modules in slot 1 & 2
- input “Stream 2”
  - 8Gbps (LCP)
  - written to disk modules in slot 3 & 4



MIT HAYSTACK OBSERVATORY

Subgroup Example (cont)

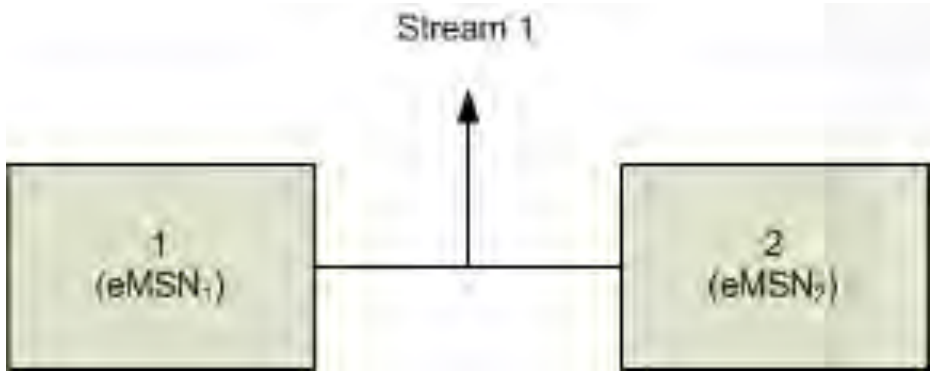
- When modules are at the correlator awaiting processing
  - RCP is scheduled for the participating antennas to be processed
    - Previously required all 4 disk modules
  - With subgrouping requires only disk modules that were written in Slot 1 & 2 be inserted at the correlator in a Mark6 correlator system
    - $eMSN_1$, $eMSN_2$
    - Do they have to be inserted into slots 1 & 2, *No*



MIT
HAYSTACK
OBSERVATORY

Subgroup Correlation

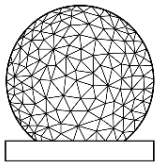
- RCP can now be processed.
  - Does not require all of group_ref
    - Only $eMSN_1$ and $eMSN_2$



MIT HAYSTACK OBSERVATORY

Subgroup Restrictions

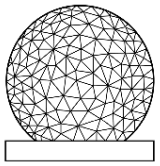
- Software c-plane restrictions
  - Once subgroups are defined, they must be kept for the group_ref when recording
    - No switching of subgroup's for the group, e.g.
      - input_stream 1 => 1,2
      - input_stream 2 => 3,4
      - record "n" scans
      - remove subgrouping as in "normal operations"
      - record "m" additional scans
      - *ILLEGAL*



MIT
HAYSTACK
OBSERVATORY

Subgroup Restrictions

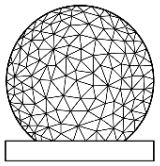
- Software c-plane restrictions (cont)
  - Subgroup assignment must use all disk modules of open group
    - Example of illegal case:
      - group = open : 1234
      - input_stream 1 => 12
      - input_stream 2 => 3
      - disk in slot 4 not assigned : *ILLEGAL*
  - Complete subgroup modules are required for processing
    - input_stream 1 => 1,2 ($eMSN_1$, $eMSN_2$)
    - At correlator requires both $eMSN_1$, $eMSN_2$ inserted in same Mark6



MIT
HAYSTACK
OBSERVATORY

Play Back

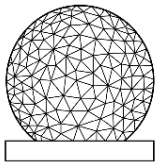
- Mount the disks
- group_members? slot
  - Number of disks in the group_ref
  - The associated disks eMSN in the group_ref
- When mounting, does order have to be preserved?
  - No you can place them in any slot of the Mark6's
- What about sub-grouped Modules?
  - Would need only complete subgroup to be mounted for data to be removed



MIT
HAYSTACK
OBSERVATORY

Play Back (cont)

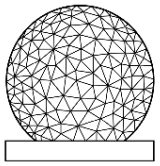
- Vdifuse (Geoff Crew)
  - Scatter / Gather Fuse interface for VDIF
    - Alma Phasing Project - verified
    - General purpose geodesy does not work
    - Mount Mark6 Modules with vdifuse
    - process the data directly from the disk modules to DiFX
- gator
  - Wrapper around gather and gather464 files
  - Gathers the scatter / gather files and creates a single file
  - Data must be *dqa* into separate threads for DiFX processing



MIT
HAYSTACK
OBSERVATORY

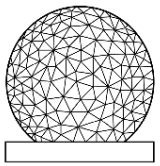
Play Back (cont)

- gather
  - Gathers the scatter / gather files and creates a single file
  - VDIF payload characteristics
    - All threads are left as is (N threads)
    - If threadIDs are not unique they will be combined and cause errors
- gather464
  - Gathers the scatter / gather files and creates a single file
  - VDIF payload characteristics
    - All threads are merged into a single thread
    - Geodetic observations
      - 64 channels
  - Reduces correlation time by 4



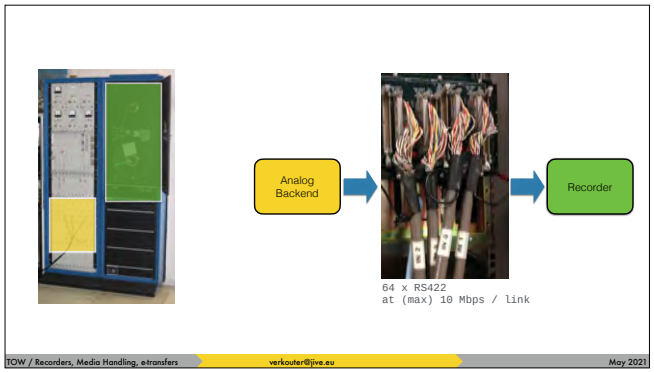
MIT
HAYSTACK
OBSERVATORY

Questions ?

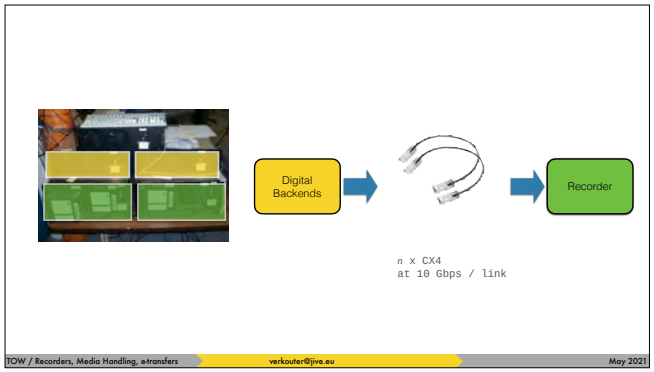


MIT
HAYSTACK
OBSERVATORY

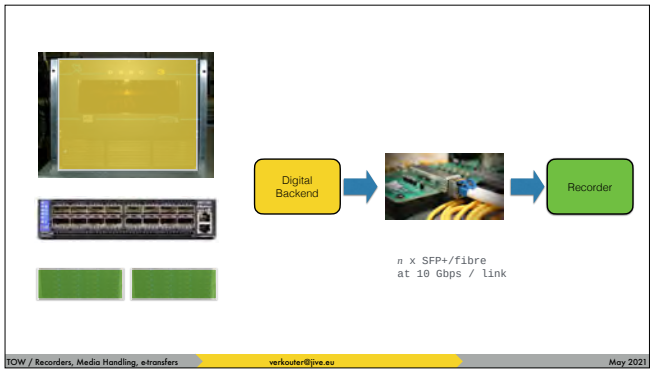
parallelization has always been a topic in VLBI data acquisition.



from the days of the MarkIV/VLBA system where the [click] digitizers / samplers are connected to the [click] recorder via a multitude of parallel serial RS422 connections



to more modern days where [click] multiple digital backends send their data to [click] multiple recorders over 10 Gbps CX4 or



one DBBC3 [click] sending data to [click] multiple FlexBuff recorders via a switch.

Getting as many bits on disk as possible

The driving force behind all this is always this

Contemporary VLBI recorders

ethernet (UDP/IP) packets
- $2^n \cdot 10^6$ Mbps
- real sampled
- VDIF(*)

(*) VLBI Data Interchange Format - https://vlbi.org/vlbi-standards/vdif/

The producers of those bits are the contemporary digital backends. These produce [click] a stream - or streams - of ethernet packets with VDIF frames in them. It is up to the recorders

Contemporary VLBI recorders


Mark6


FlexBuff


OCTADISK

...?

such as these to capture them.

Contemporary VLBI recorders


Mark6

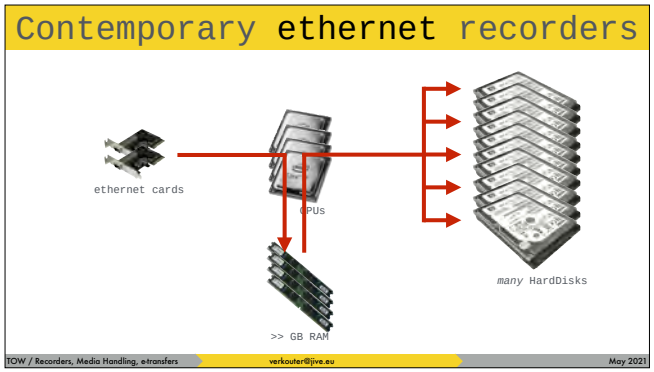

FlexBuff

OCTADISK

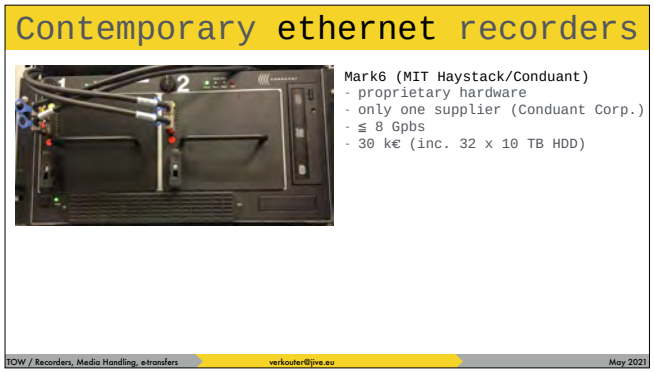
...?

TOW / Recorders, Media Handling, e-transfers verkouter@jive.eu May 2021

In this lecture we'll focus on Mark6 and FlexBuff



The thing to realize is that conceptually they are the same. They're all [click] _ethernet_ recorders. They consist of [click] a number of CPUs, [click] a large amount of RAM, and [click] a lot of disks. The operation is simple: [click] packets are captured from the network into memory and then [click] in the background scattered over the available disks.



the Mark6 is commercially available as a joint MIT Haystack/Conduant development

Contemporary ethernet recorders

Mark6 (MIT Haystack/Conduant)
- proprietary hardware
- only one supplier (Conduant Corp.)
- ≦ 8 Gpbs
- 30 k€ (inc. 32 x 10 TB HDD)

FlexBuff (Metsähovi / JIVE)
- fully customizable, fully COTS
- *n* Gpbs
- 16 k€ (inc. 36 x 10 TB HDD)

Concept: A. Mujunen, Metsähovi
Productionalized: JIVE

TOW / Recorders, Media Handling, e-transfers verkouter@jive.eu May 2021

whilst FlexBuff is a fully customizable off the shelf solution. A concept by Ari Mujunen and put into production by JIVE.

Contemporary ethernet recorders

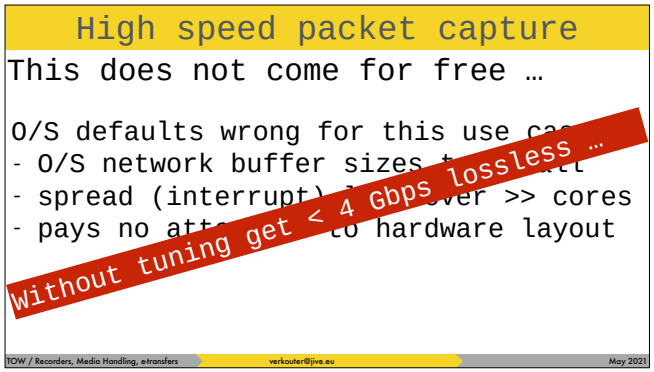
The only tangible difference between the systems. The rest is semantics/software.

Mark6 removable disk packs

FlexBuff fixed disks

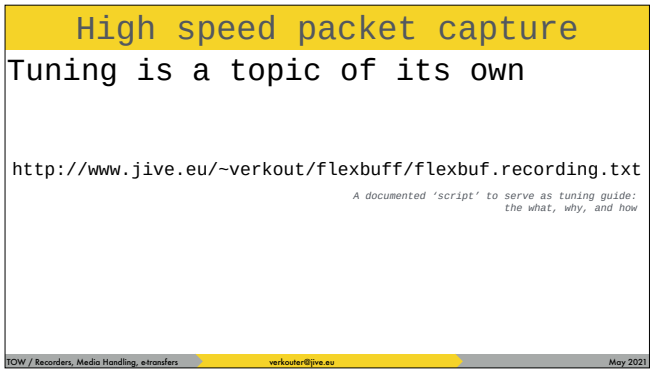
TOW / Recorders, Media Handling, e-transfers verkouter@jive.eu May 2021

The only real difference between the two machines is the fact that Mark6 [click] has removable disk packs and the [click] flexbuff doesn't. This has consequences for shipping the data, obviously, but we'll get to that later.



So, both systems try to solve the following problem: high speed packet capture from the network. [click] This does not come for free!!! And more specifically - even on a modern machine [click] you won't get beyond 4 Gpbs without tuning

---



Because of the short time we cannot get into the details of tuning because it is a topic of its own. But you can read the documented script at this URL to get an idea what tuning is needed and why and how to do it.

Origin of the differences

Choice of recording software

The biggest observable differences are caused by the actual choice of recording software. So what options exist?

Origin of the differences

Choice of recording software

cplane / dplane

- MIT Haystack

The Mark6 comes preinstalled with the pair of programs called cplane and dplane

Origin of the differences

Choice of recording software

cplane / dplane
- MIT Haystack

jive5ab(*)
- Joint Institute for VLBI in Europe





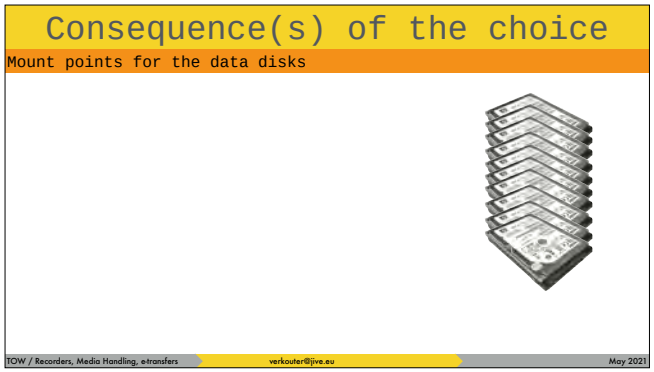
...

(*) https://github.com/jive-vlbi/jive5ab

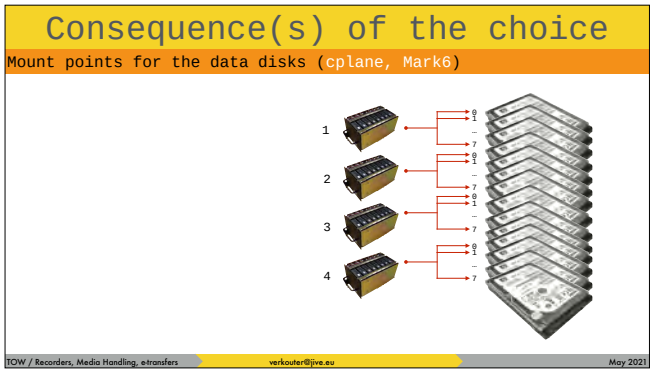
The jive5ab software runs on all Mark5, Mark6, flexbuff and BSD style operating systems such as Mac OSX.

Consequence(s) of the choice

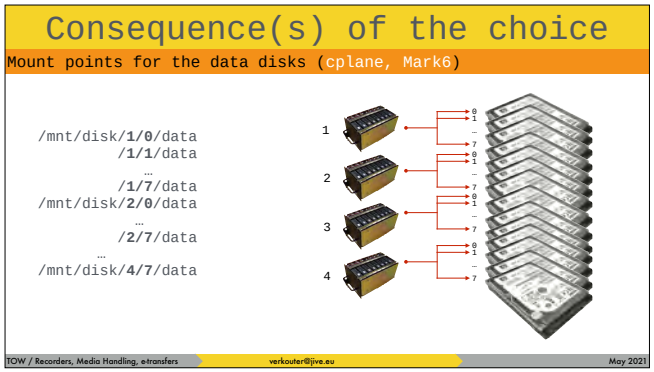
Choosing either has consequences and it is important to know what they are



For example the storage. Since the recorders are just Linux or UNIX machines, the hard disks are made visible to the operating system under mount points.

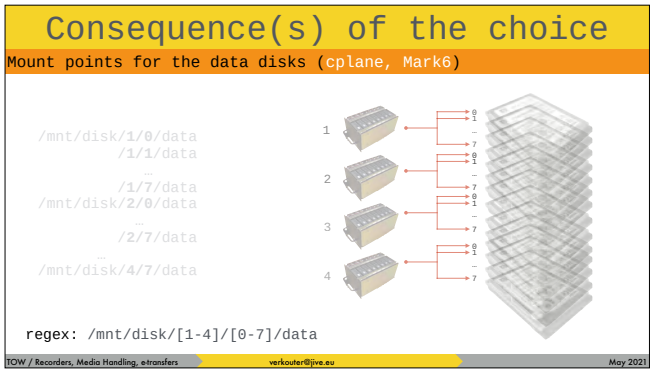


On the Mark6 the cplane software expects the mount points to be organized per disk module

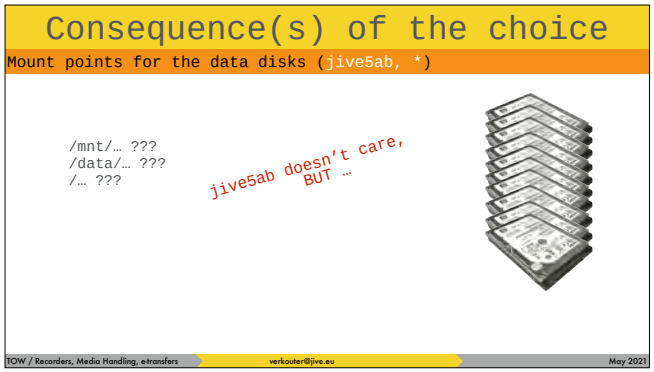


and thus when cplane mounts modules they appear in the operating system as follows

---



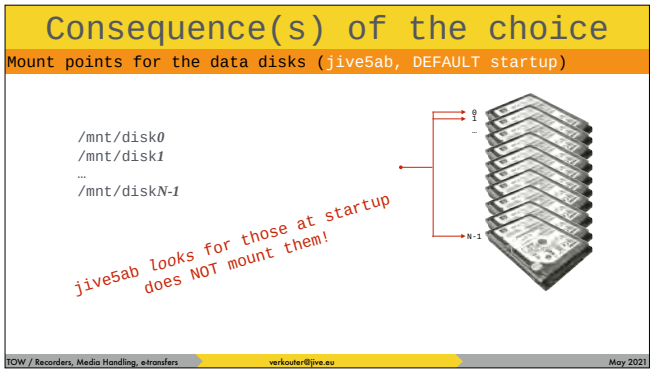
which can be summarized as this regular expression



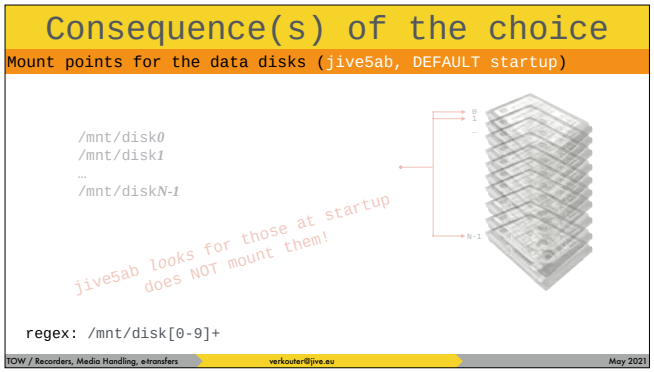
jive5ab, frankly, doesn't care where your [click] data disks are mounted.

HOWEVER, there are some compiled in defaults that you can use to make your life easier

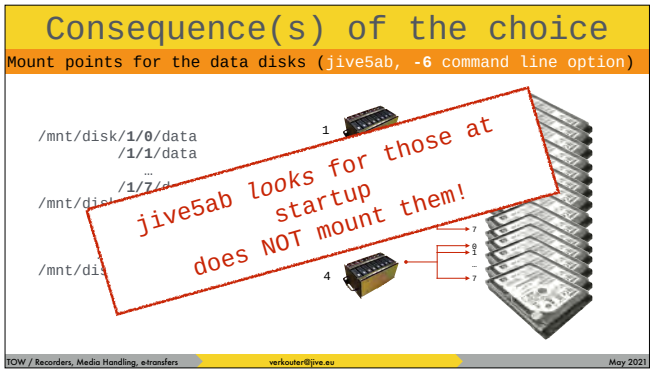
---



By default jive5ab looks for mountpoints called /mnt/disk blah with blah being a number. [click] So if you mount your data disks as this regex

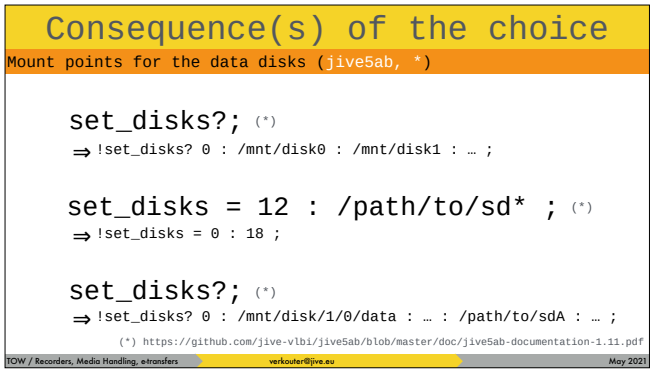


jive5ab will pick them up automatically at startup

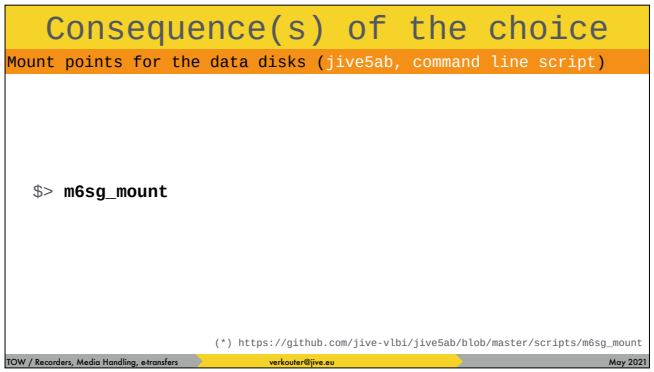


However, if you pass the “-6” command line option, jive5ab will look for the mark6 modules instead.

And I must stress again [click] - jive5ab does NOT mount harddisks, it expects to happen outside the program.



In jive5ab the disks to record on can be queried and changed [click] at runtime by issueing the `set_disk=…` command.



In the jive5ab source code repository there is a command line script m6sg_mount

Consequence(s) of the choice

Mount points for the data disks (jive5ab, command line script)


```
$> m6sg_mount 134
```

(*) https://github.com/jive-vlbi/jive5ab/blob/master/scripts/m6sg_mount

that can be used to mount

Consequence(s) of the choice

Mount points for the data disks (jive5ab, command line script)


```
$> m6sg_mount -u 3
```

(*) https://github.com/jive-vlbi/jive5ab/blob/master/scripts/m6sg_mount

or unmount individual mark6 modules

Consequence(s) of the choice

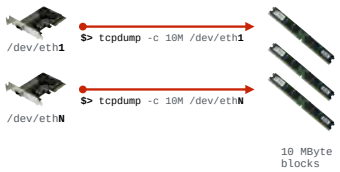
How the packets are read from the network

Another big difference is how the packets are grabbed from the network

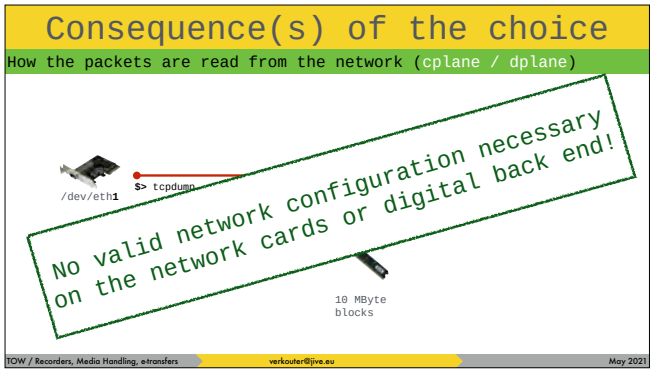
---

Consequence(s) of the choice

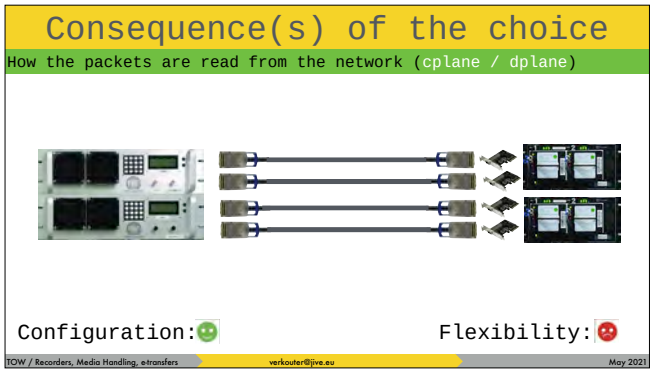
How the packets are read from the network (cplane / dplane)



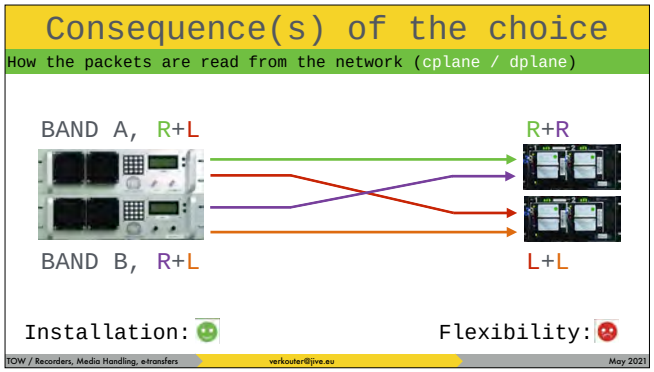
The dplane program accumulates frames from the ethernet devices directly into blocks in memory of about 10 MB. [click] It is not unlike running tcpdump on the interface.



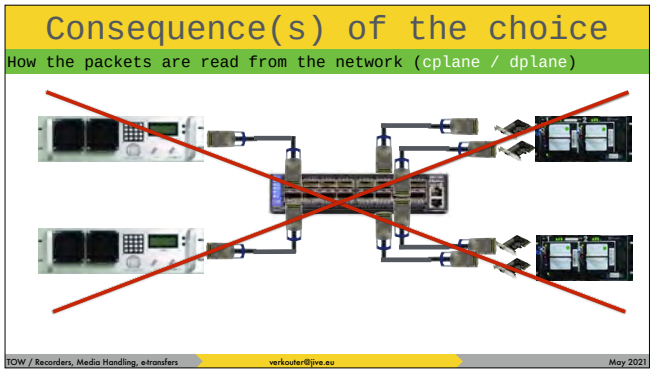
The important property is that you don't need a valid network configuration on ANY of the components in your system.



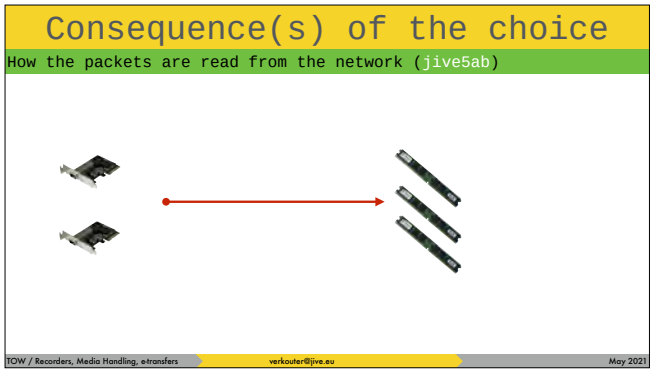
The model for cplane/dplane and the Mark6 is this: fixed point to point connections between digital receiver(s) and the network cards in the recorder. So this is VERY easy for installation and configuration



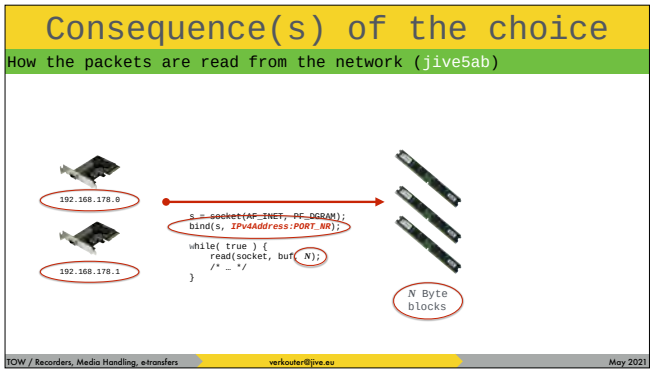
but e.g. collecting polarizations from two bands is impossible.



or putting a switch between your backends and recorders [click] is also not possible

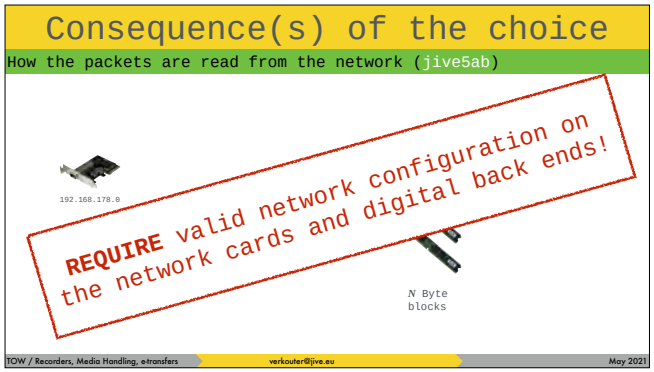


jive5ab on the other hand does things completely differently

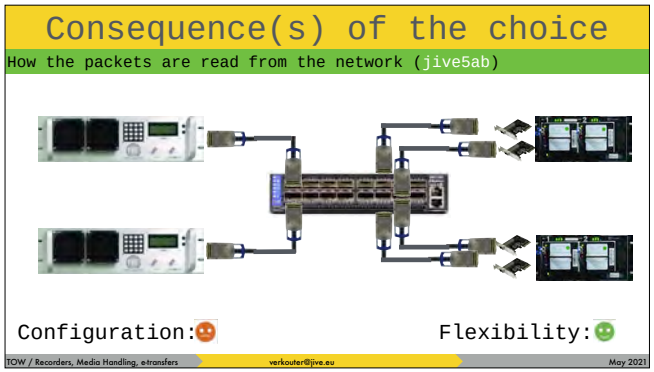


It operates at the [click] IP address level.

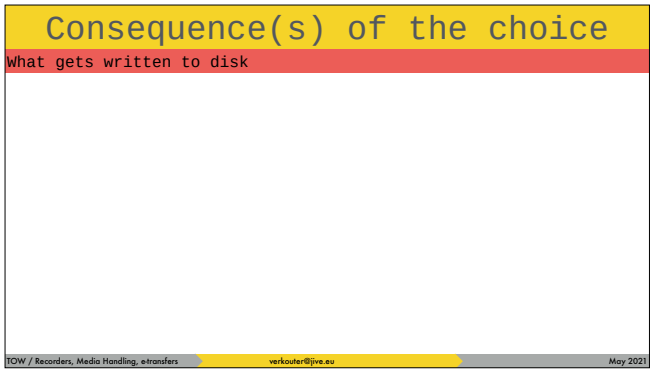
And in the code it starts listening for data [click] on one (or all) IP addresses and a specified port number for incoming data, [click] and collects this in customizable size blocks.



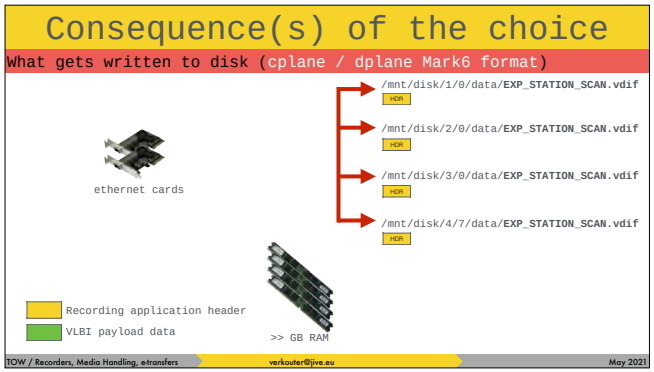
jive5ab requires you to have a valid network configuration on ALL components!



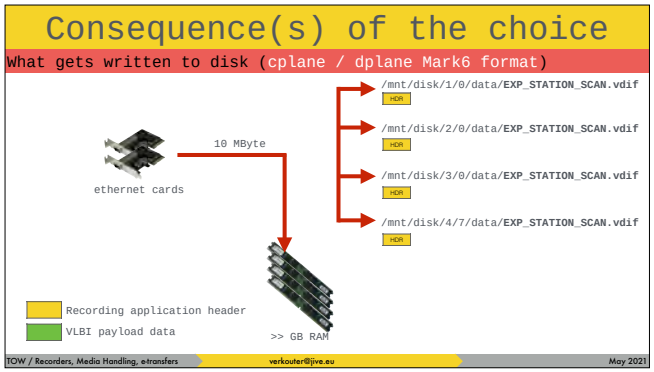
So configuration of this is a bit more difficult but flexibility is 100%



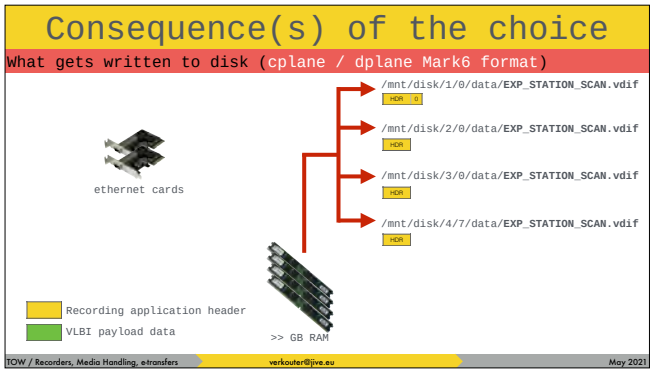
Another difference is WHAT gets recorded



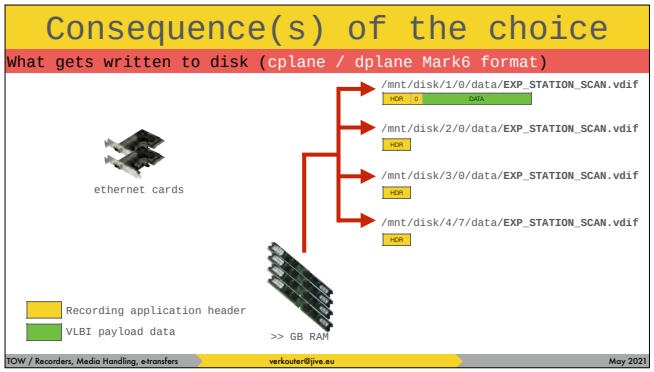
The cplane/dplane software opens files on each disk, [click] writes a header identifying this as a Mark6 file.



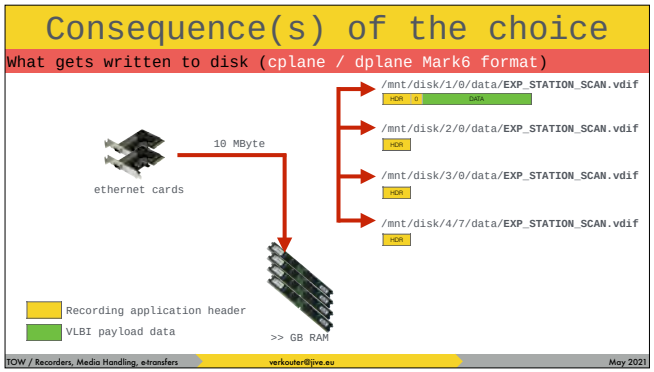
When a 10 MB block has been read



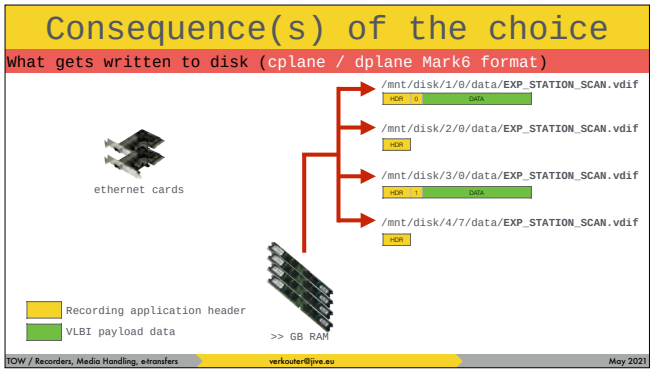
the first available file is found and a block header is written in the file,



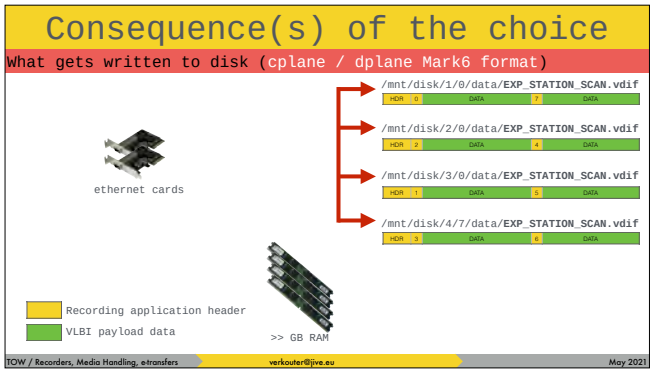
and after that the actual block of data



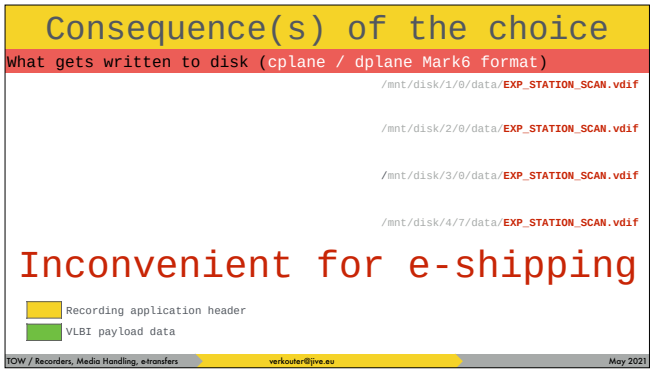
The next block comes in



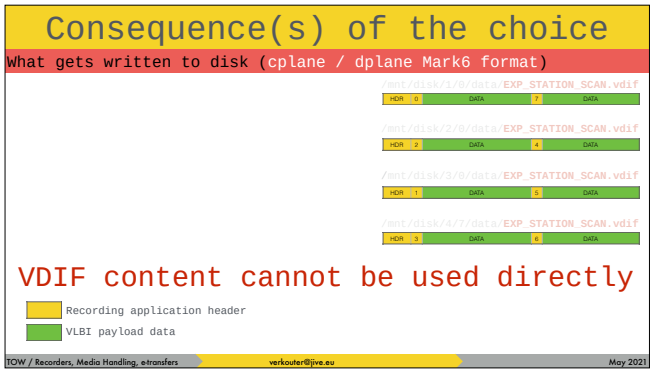
and another file gets written to



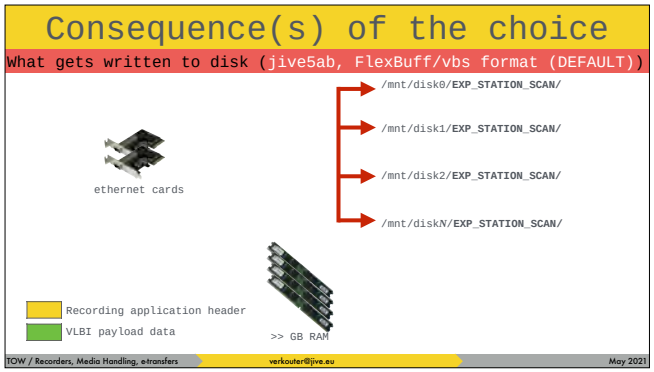
That way data is scattered across the files.



As you can see, the files are named all the same, [click] which is not convenient for electronic transfer!

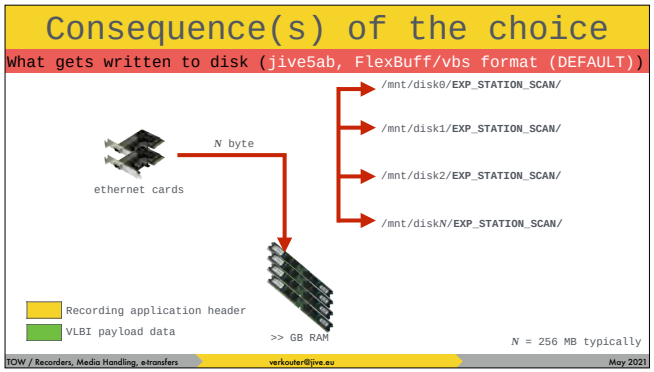


Another thing is that the VLBI data is mixed with application headers. [click] Which means that the files cannot be easily processed by an arbitrary VDIF tool

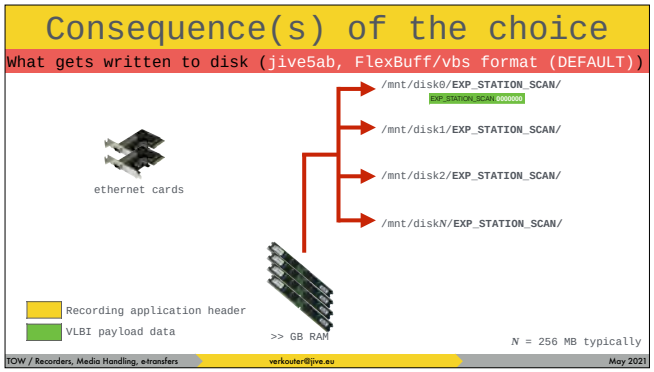


The default format that jive5ab writes is the FlexBuff “vbs” format. [click] jive5ab creates *directories* with the recording name on all disks

---

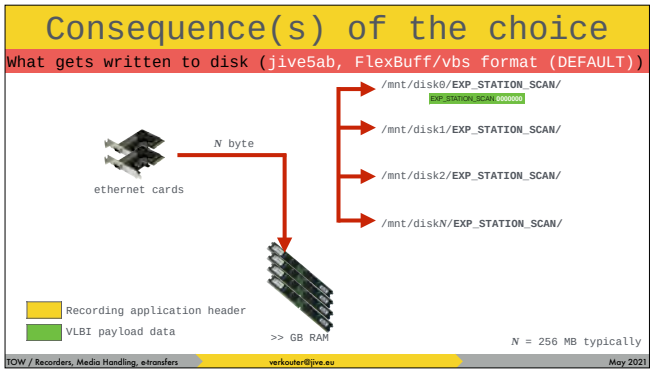


After reading a block of typically 256 MByte

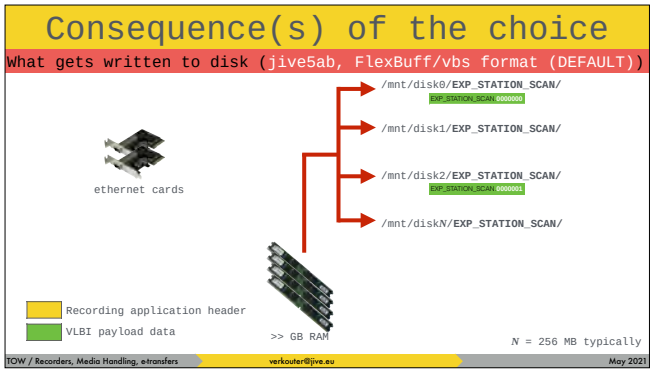


The first available disk is selected and the block is written to a single file with the block sequence number as the extension

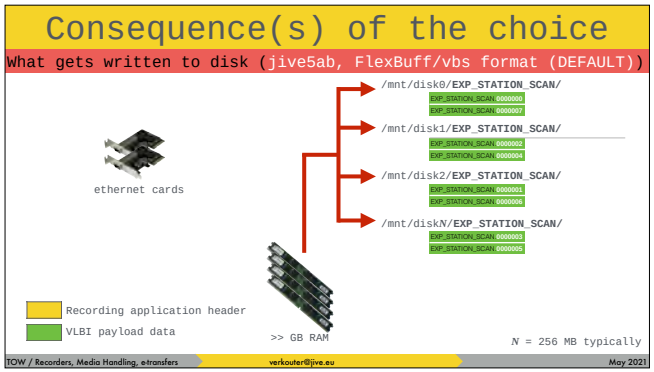
---



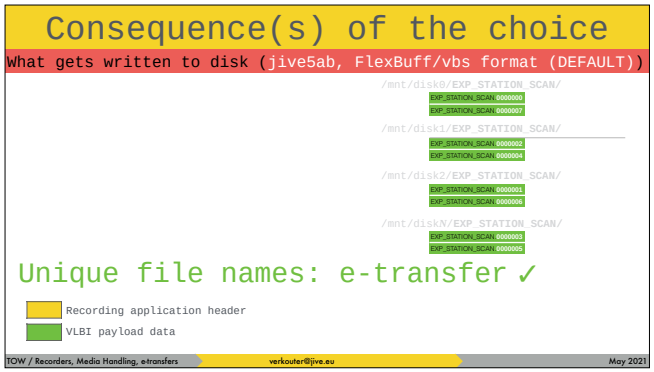
the next block is captured



and is written to the next available disk

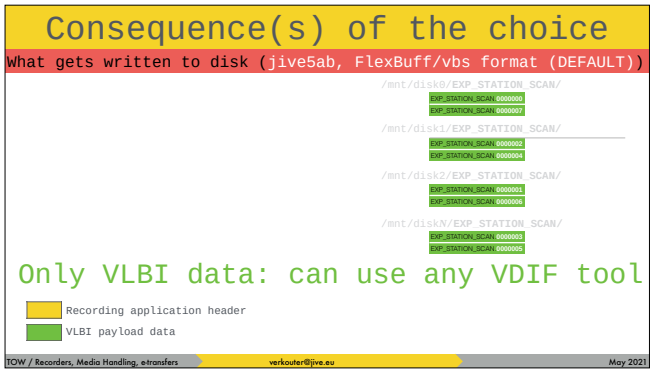


that way the directories are populated with chunks of VLBI data

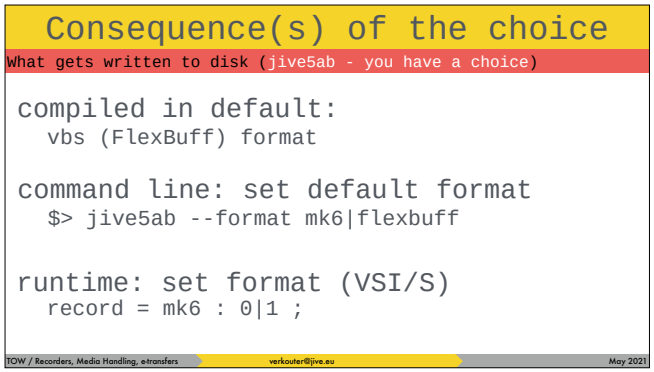


The file names are unique which is extremely handy for e-transfer

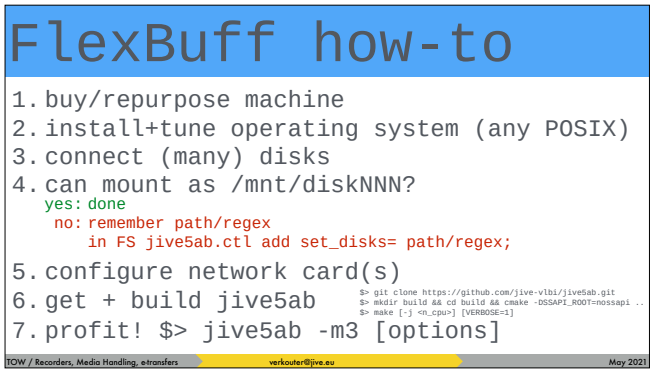
---



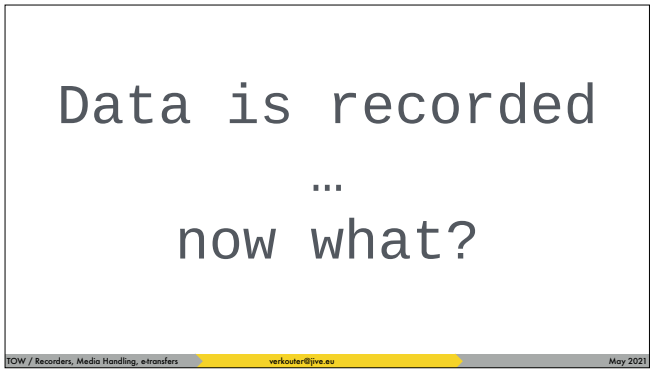
and there are no headers inbetween the useful data so any VDIF tool can use the snippets directly!



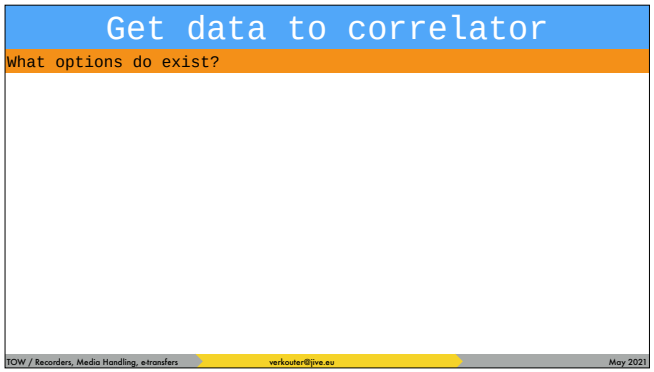
jive5ab can record in both formats and there are several ways to change the recording format, [click] e.g. from the command line [click] or at runtime.



For quick reference, this is a one-slide recipe to building a flexbuff. It is really simple; have a machine with storage and run jive5ab.



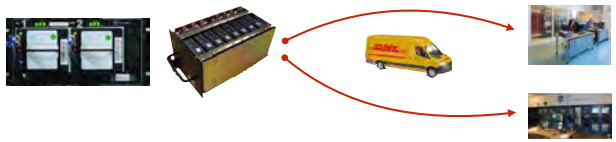
So, once the data has been recorded, then what?



it needs to be transferred to the correlator. So what options do exist?

Get data to correlator

What options do exist? (Mark6, ship disk modules)



on the Mark6 you have removable disk packs so shipping those to [click] ONE correlator site is an option.

Get data to correlator

What options do exist?

```
$> scp /mnt/disk/* io13.mpifr-bonn.de:/path/…
```

???

but what about e-transfer? If you actually TRY this command …

Get data to correlator

What options do exist?

```
$> scp /mnt/disk/* io13.mpifr-bonn.de:/path/…
```

… what likely will happen is this

Get data to correlator

What options do exist?

```
$> ftp /mnt/disk/* io13.mpifr-bonn.de:/path/…
```


And even if you try THIS, the same will happen.

Get data to correlator

What options do exist?

```
$> ftp|scp /mnt/disk/* io13.mpifr-bonn.de:/path/.
```

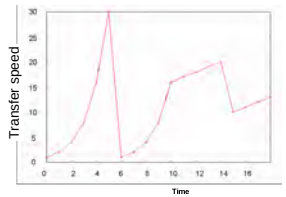
Transmission Control Protocol

https://en.wikipedia.org/wiki/Transmission_Control_Protocol

The problem is that these tools work with the TCP protocol

Get data to correlator

What options do exist?

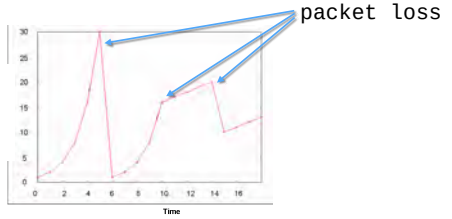


https://en.wikipedia.org/wiki/Transmission_Control_Protocol

And that doesn’t work very well on the long fat international links, the speed is very erratic.

Get data to correlator

What options do exist?



https://en.wikipedia.org/wiki/Transmission_Control_Protocol

the protocol is VERY sensitive to packet loss

Get data to correlator

What options do exist? Use the **U**DP **D**ata **T**ransfer protocol

UDT

- software library in user space (not in O/S kernel!)
  - require application to actually *use* the library
- based on connectionless unreliable UDP protocol
- implement TCP-like features:
  - connection oriented
  - reliable

https://en.wikipedia.org/wiki/UDP-based_Data_Transfer_Protocol

fortunately there is a solution. Someone invented the udp data transfer protocol. Which is implemented as a software library simulating tcp on top of udp. So it’s not supported by the operating system but an application must use the library.

Get data to correlator

What options do exist? Use the **U**DP **D**ata **T**ransfer protocol

UDT

- software library in user space (not in O/S kernel!)
  - require application to actually *use* the library
- based on connectionless unreliable UDP protocol
- implement TCP-like features:
  - connection oriented
  - reliable
- A LOT faster on long fat links!

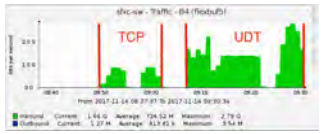
https://en.wikipedia.org/wiki/UDP-based_Data_Transfer_Protocol

The main feature is that it is really A LOT faster than TCP.

Get data to correlator

What options do exist? Use the **U**DP **D**ata **T**ransfer protocol

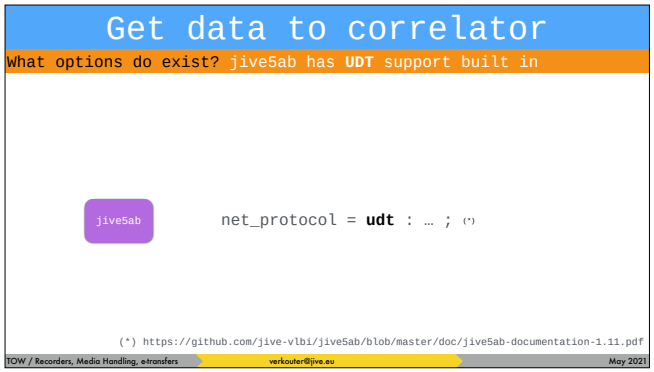
UDT



https://en.wikipedia.org/wiki/UDP-based_Data_Transfer_Protocol

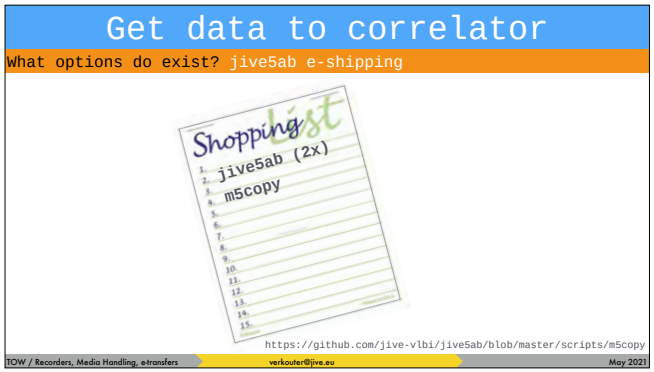
TOW / Recorders, Media Handling, e-transfers verkouter@jive.eu May 2021

As can be seen in these network graphs. Note that the TCP transfers are misleading because they were cancelled - the file transfer just froze and we ^C'ed the program before retrying again.

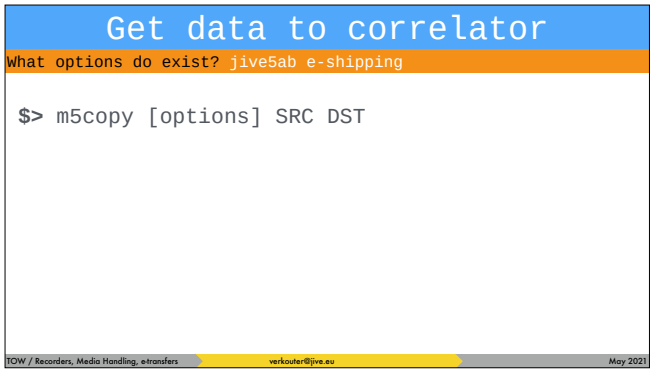


So you need an application to use this fast protocol.

[click] jive5ab is such an application.

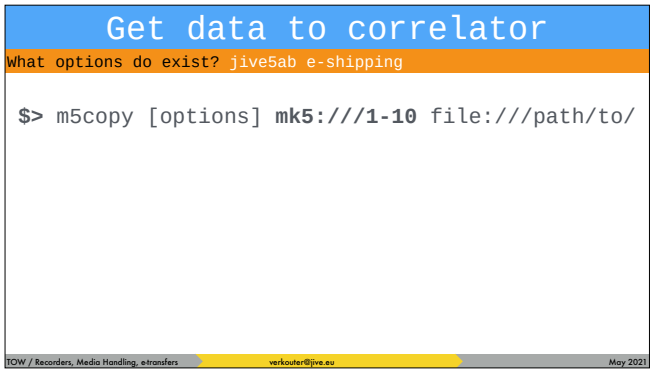


If you get the following ingredients, [click] 2 servers running jive5ab and the [click] m5copy python script from the jive5ab sources

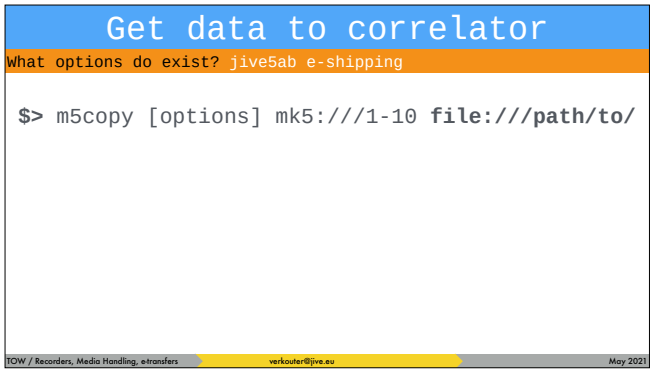


then you can use the m5copy command line script basically like this - copy data from source to destination

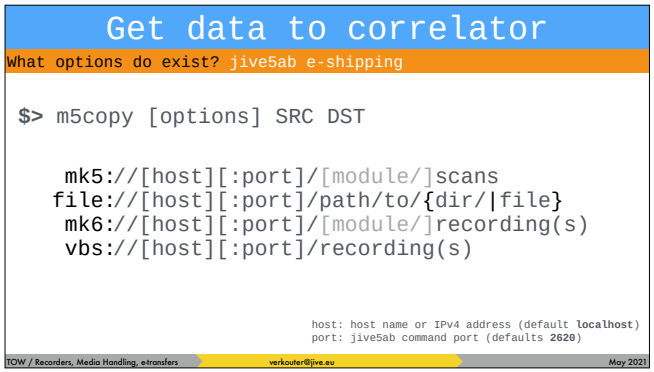
---



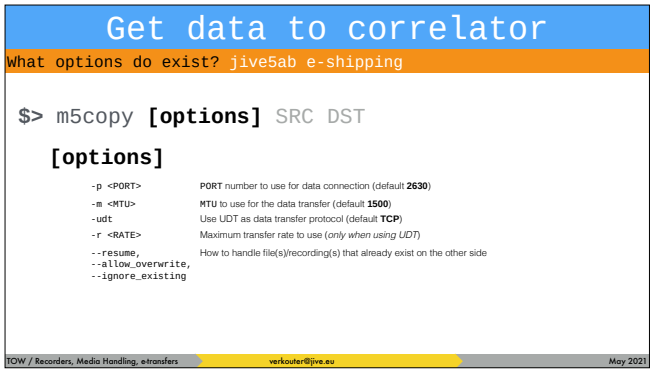
the source and



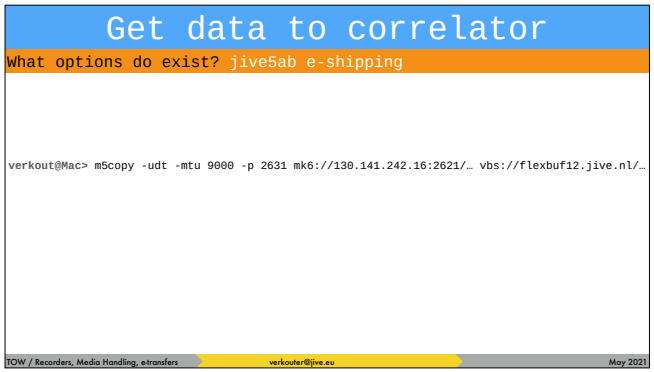
destinations are URL like specifications



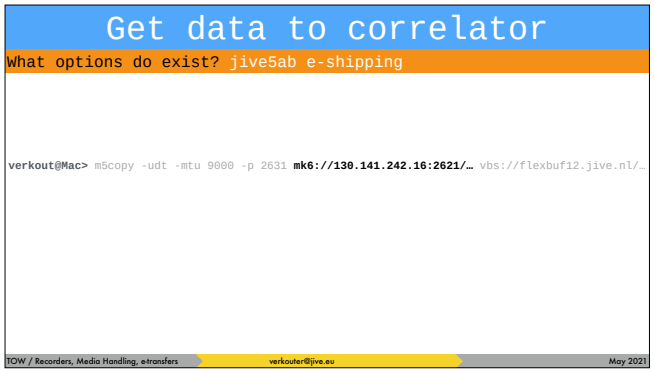
As you can see jive5ab can address most current VLBI data formats and media.



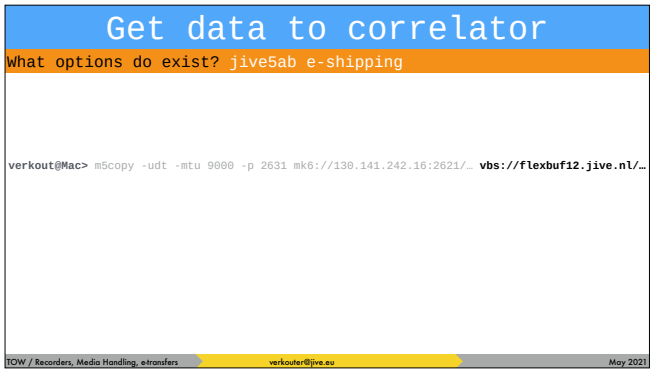
And this is a summary of the most important options



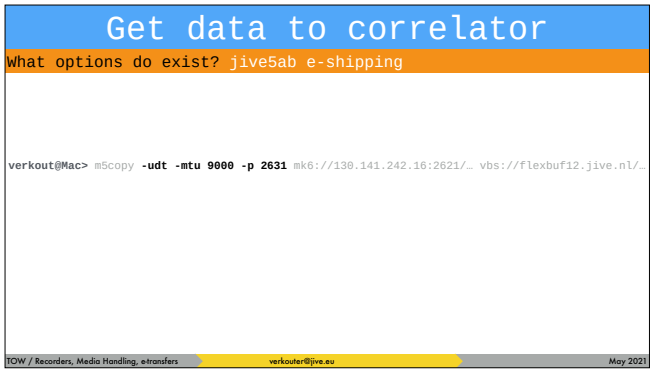
If you issue a command like this



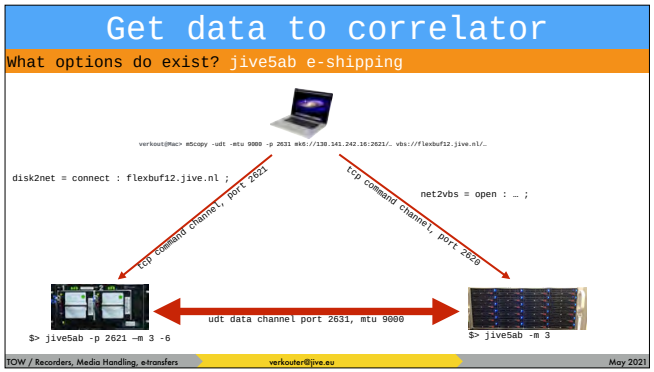
transferring mark6 data from somewhere



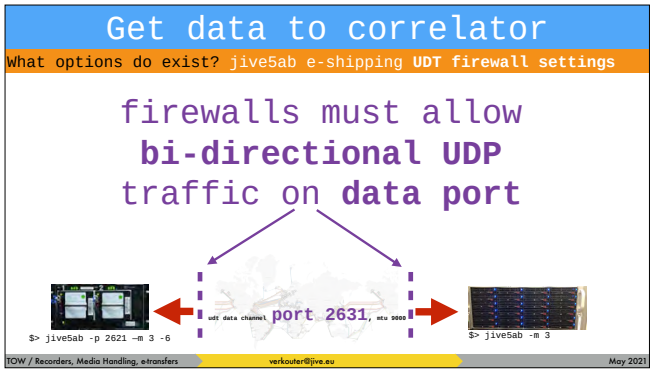
to a flexbuff at JIVE



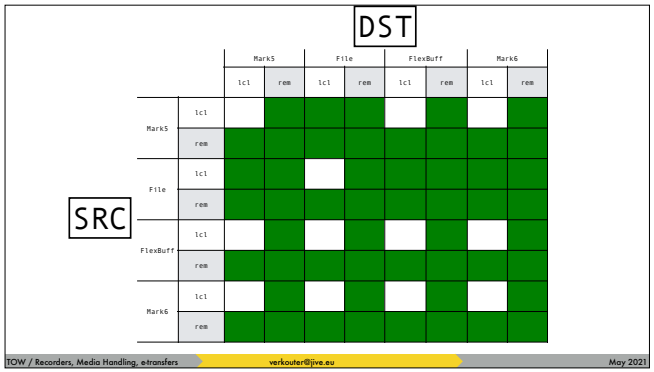
using the UDT protocol over port 2631 using an MTU of 9000



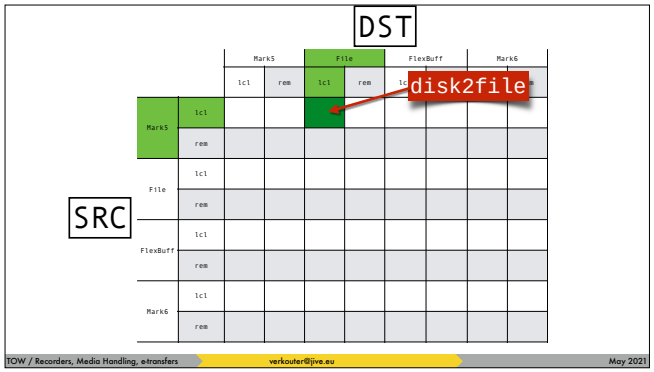
what happens is this. m5copy sends the net2vbs command to the flexbuff and tells the disk2net command on the mark6 to connect directly to the flexbuff using UDT over port 2631 on the FAT LINK. The data will NOT go through your laptop



Most systems are behind firewalls. In order to USE the fast UDT protocol both firewalls must allow bi-directional UDP traffic - i.e. in AND outgoing on the data port.

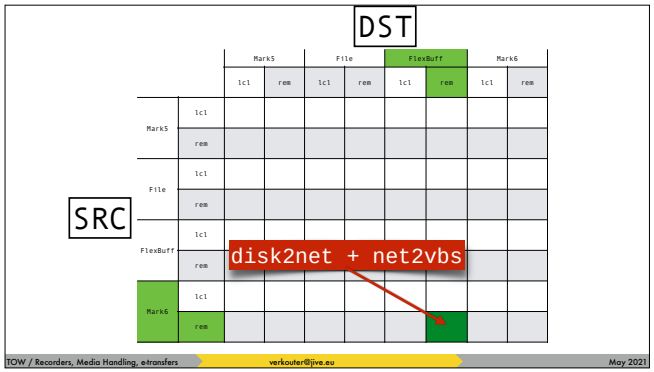


For reference I include the full connectivity matrix and this might look a bit daunting but … the thing is that both source and destination can be either “the local machine” or remote. The matrix is _mostly_ filled but some combinations just don’t make sense.



the way to read this is for example, for local Mark5 to local file [click] m5copy just executes disk2file

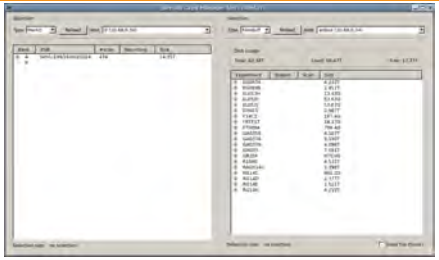
---



whilst for remote mark6 to remote flexbuff [click] m5copy couples these two transfers

Get data to correlator

What options do exist? jive5ab + m5copy **GUI frontend**

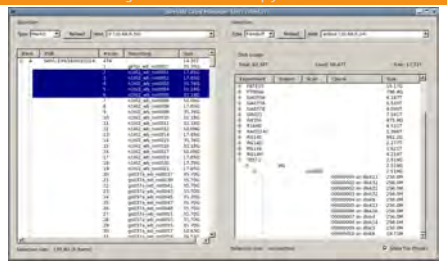


https://github.com/jive-vlbi/jive5ab-copy-manager

There is also a GUI frontend to that drives m5copy for you!

Get data to correlator

What options do exist? jive5ab + m5copy **GUI frontend**



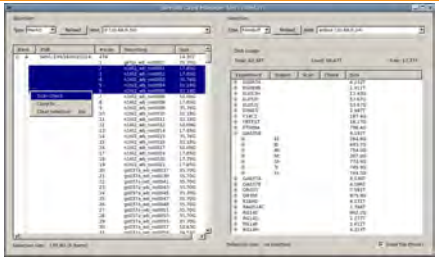
https://github.com/jive-vlbi/jive5ab-copy-manager

TOW / Recorders, Media Handling, e-transfers verkouter@jive.eu May 2021

you can easily select multiple scans

Get data to correlator

What options do exist? jive5ab + m5copy **GUI frontend**

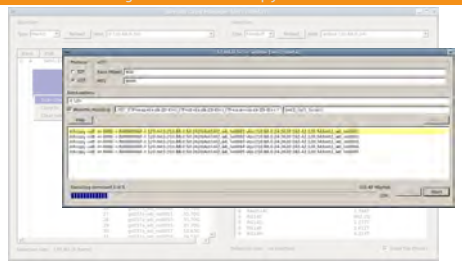


https://github.com/jive-vlbi/jive5ab-copy-manager

and with a right-click

Get data to correlator

What options do exist? jive5ab + m5copy **GUI frontend**



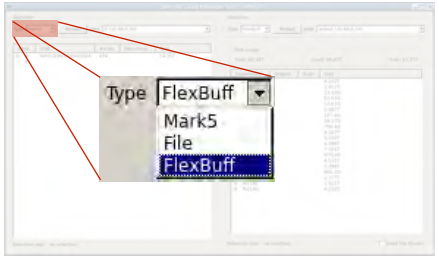
https://github.com/jive-vlbi/jive5ab-copy-manager

TOW / Recorders, Media Handling, e-transfers verkouter@jive.eu May 2021

easily copy them across

Get data to correlator

What options do exist? jive5ab + m5copy **GUI frontend**

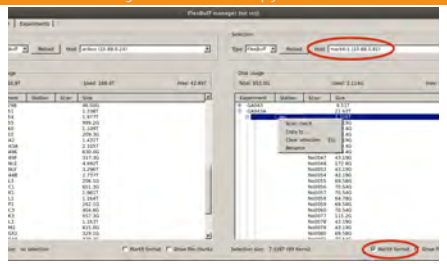


https://github.com/jive-vlbi/jive5ab-copy-manager

You can copy from any of the supported types

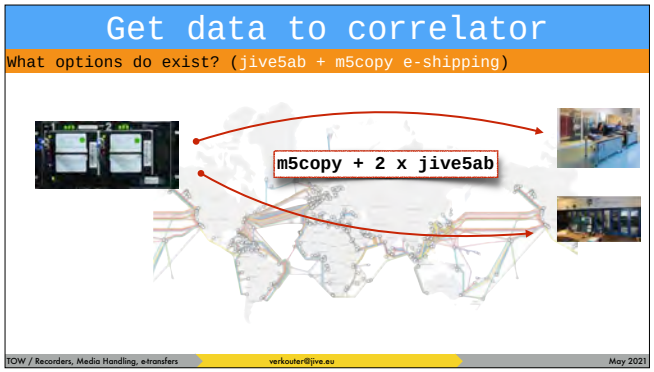
Get data to correlator

What options do exist? jive5ab + m5copy **GUI frontend**

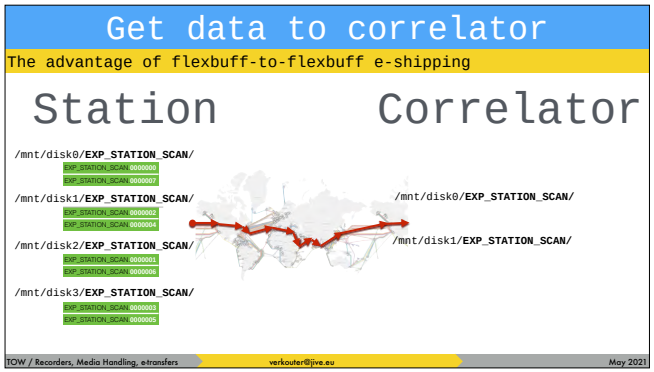


https://github.com/jive-vlbi/jive5ab-copy-manager

and the latest version also supports Mark6 natively



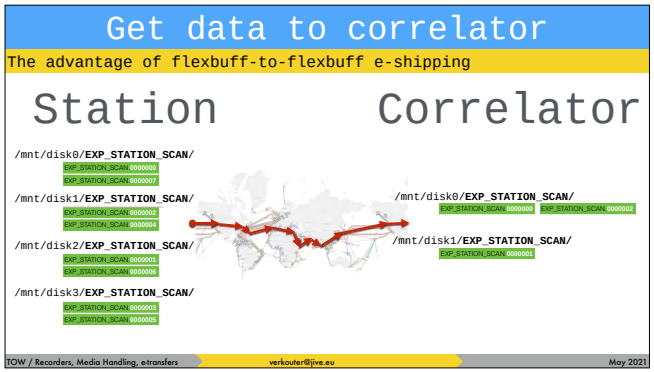
So using e-transfer you can ship data to several correlators without mucking with the modules!



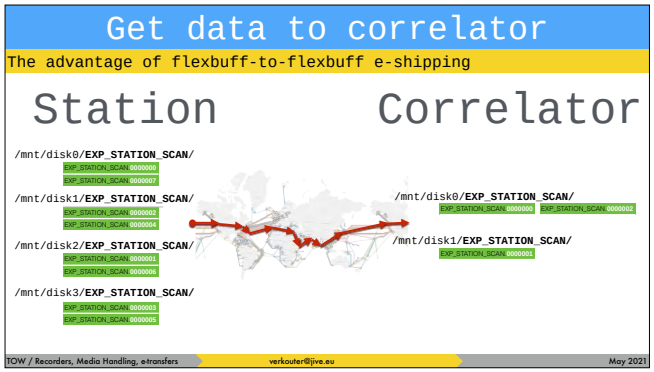
The flexbuff recording format is very suitable for e-shipping. It is independent of the number of disks present at source or destination. Let's assume the station [click] has a flexbuff with four mountpoints with data and the [click] correlator one with just two.

Initially there is no data from the experiment at the

correlator. [click] After starting the transfer the two jive5ab's exchange information which chunks are missing …

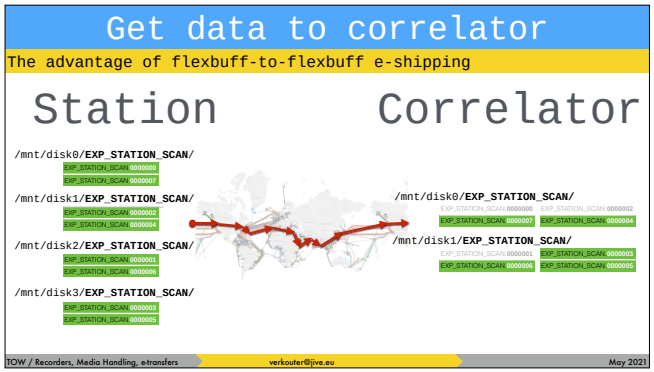


and start transferring them. [click] If the connection breaks or the program is stopped …



… and later on [click] restarted, the receiving jive5ab sees that some chunks are already present

…

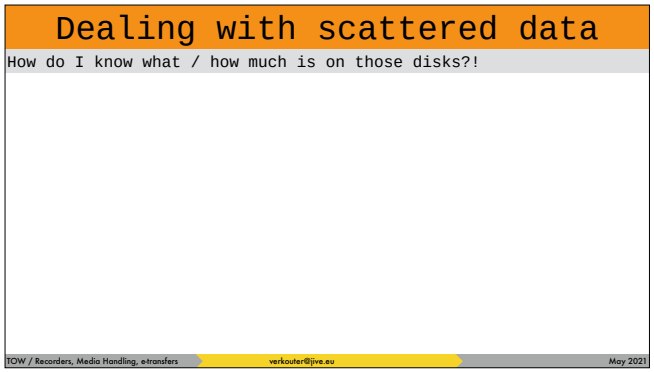


… and only the remaining chunks are transferred.

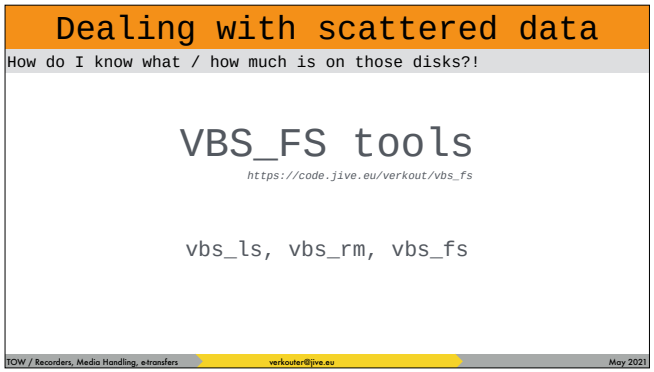
This can only be done because of the uniqueness of the file names of the individual chunks!

Now you have a lot of data scattered over many hard disks and that is not very nice to deal with!

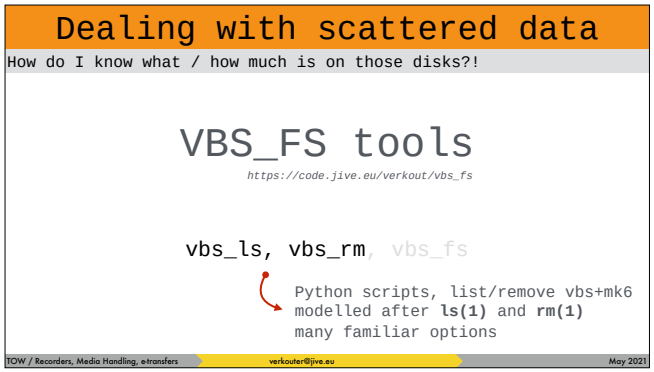
One of the first questions is:



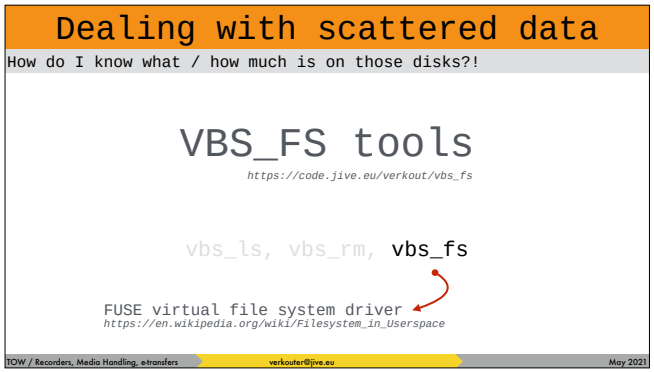
One of the first questions is: what is actually on those disks and how large is it?



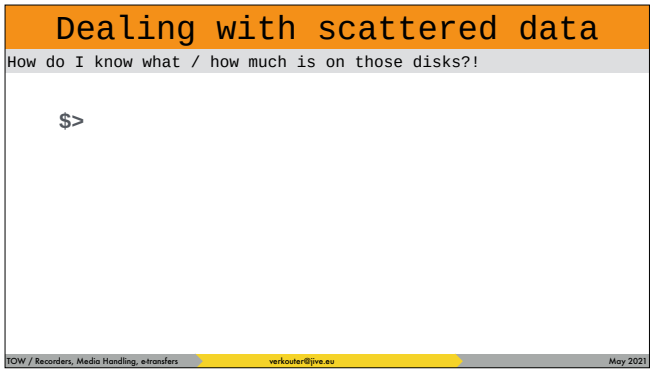
The vbs_fs toolset is a collection of programs consisting of vbs_ls, vbs_rm and vbs_fs.



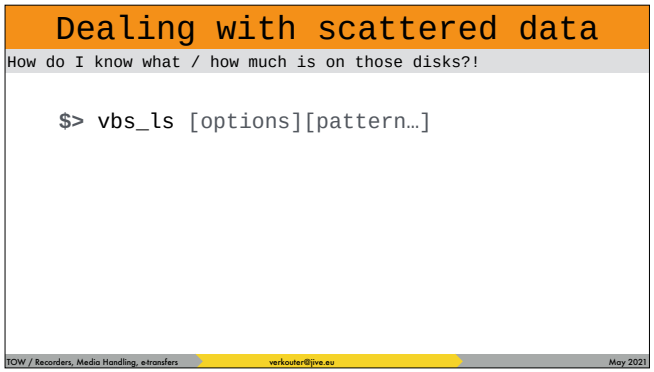
As their name implies, vbs_ls and vbs_rm are modelled after their illustrious UNIX counterparts.



whilst vbs_fs is a FUSE file system in user space, which may be convenient



to come back to the original question: what is ON those drives?



the vbs_ls can be used

Dealing with scattered data

How do I know what / how much is on those disks?!

```
$> vbs_ls -lrth haavee*
```

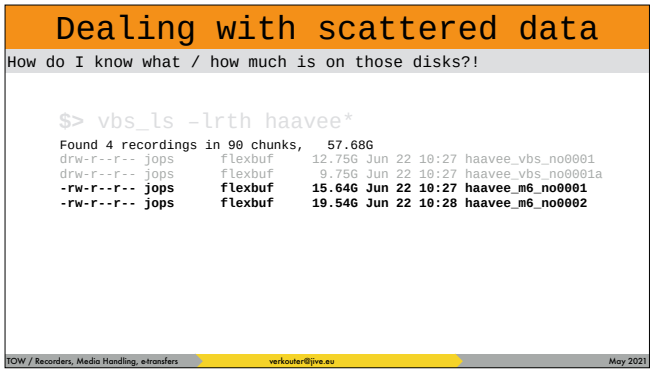

for example like this

Dealing with scattered data

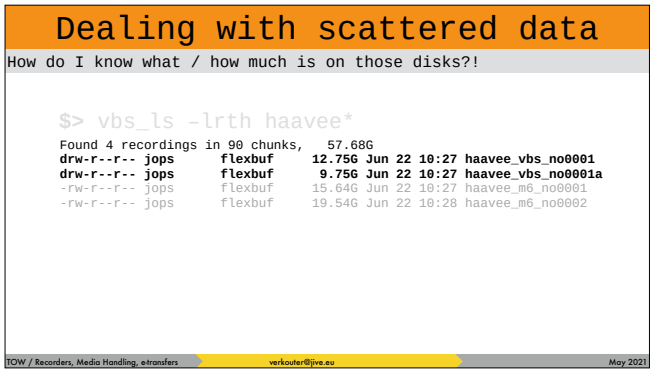
How do I know what / how much is on those disks?!

```
$> vbs_ls -lrth haavee*
Found 4 recordings in 90 chunks,   57.68G
drw-r--r-- jops      flexbuf     12.75G Jun 22 10:27 haavee_vbs_no0001
drw-r--r-- jops      flexbuf      9.75G Jun 22 10:27 haavee_vbs_no0001a
-rw-r--r-- jops      flexbuf     15.64G Jun 22 10:27 haavee_m6_no0001
-rw-r--r-- jops      flexbuf     19.54G Jun 22 10:28 haavee_m6_no0002
```

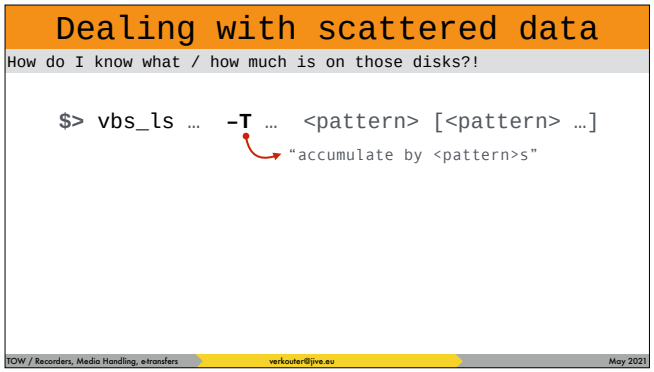

to give reasonably familiar output.



as you can see, vbs_ls deals with both mark6

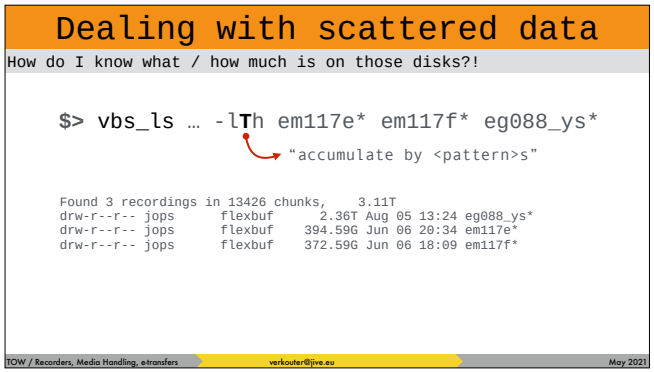


as well as flexbuff style recordings, even if present on the same media



There is an option that is NOT in ls, that allows accumulation by pattern

---



and for example can be used to assess how much data per experiment is present.

Dealing with scattered data

Easy access to recordings using a virtual file system

```
verkout@flexbuf1:$ find . -type f -name haavee\* -exec ls -lh {} \;
```

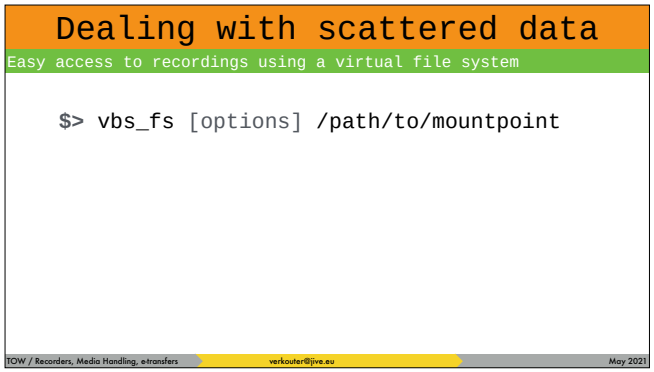

Another big issue with scattered data is …

Dealing with scattered data

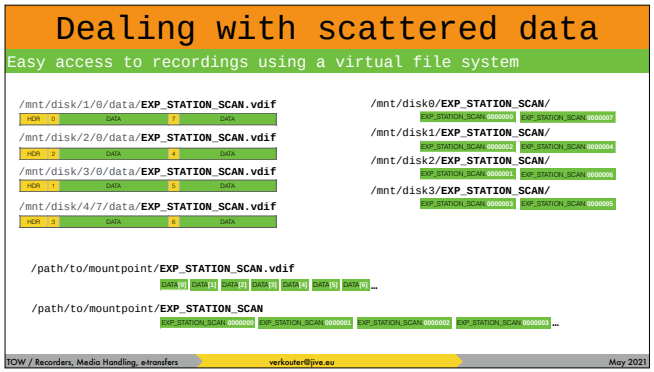
Easy access to recordings using a virtual file system

```
verkout@flexbuf1:$ find . -type f -name haavee\* -exec ls -lh {} \;

-rw-r--r-- 1 jops flexbuf 3.3G Jun 22 12:28 ./haavee_m6_no0002
-rw-r--r-- 1 jops flexbuf 2.7G Jun 22 12:27 ./haavee_m6_no0001

-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:26 ./haavee_vbs_no0001/haavee_vbs_no0001.00000007
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:26 ./haavee_vbs_no0001/haavee_vbs_no0001.00000019
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:26 ./haavee_vbs_no0001/haavee_vbs_no0001.00000013
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:26 ./haavee_vbs_no0001/haavee_vbs_no0001.00000025
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:27 ./haavee_vbs_no0001/haavee_vbs_no0001.00000031
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:26 ./haavee_vbs_no0001/haavee_vbs_no0001.00000001
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:27 ./haavee_vbs_no0001/haavee_vbs_no0001.00000049
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:27 ./haavee_vbs_no0001/haavee_vbs_no0001.00000043
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:27 ./haavee_vbs_no0001/haavee_vbs_no0001.00000037

-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:27 ./haavee_vbs_no0001a/haavee_vbs_no0001a.00000037
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:27 ./haavee_vbs_no0001a/haavee_vbs_no0001a.00000001
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:27 ./haavee_vbs_no0001a/haavee_vbs_no0001a.00000031
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:27 ./haavee_vbs_no0001a/haavee_vbs_no0001a.00000025
-rw-r--r-- 1 jops flexbuf 256M Jun 22 12:27 ./haavee_vbs_no0001a/haavee_vbs_no0001a.00000019
```

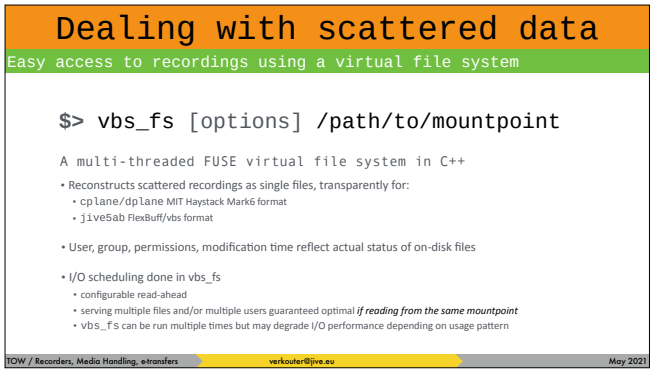

… it is spread like shrapnel over many disks!



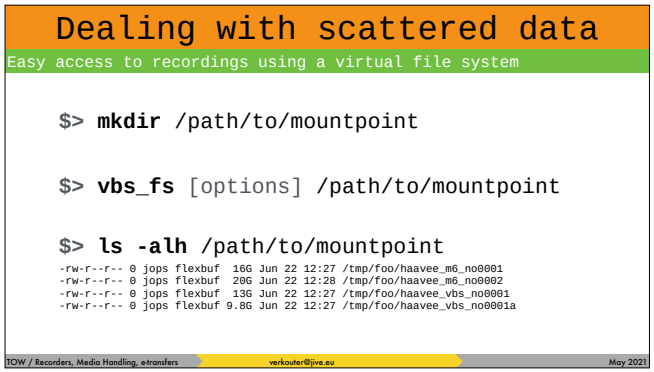
Enter the vbs_fs virtual file system. After startup, this



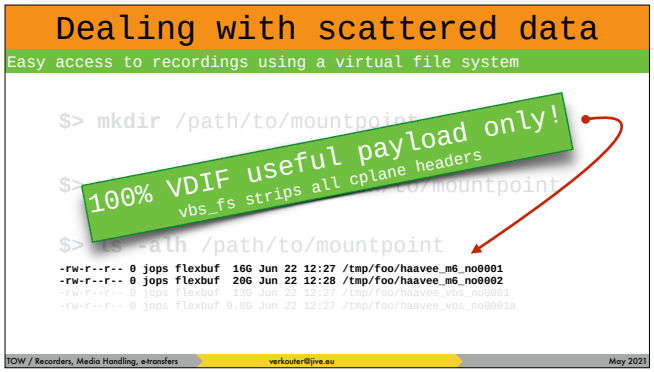
… virtual file system takes the scattered data from recordings [click] and reconstructs them as single files under the mountpoint.



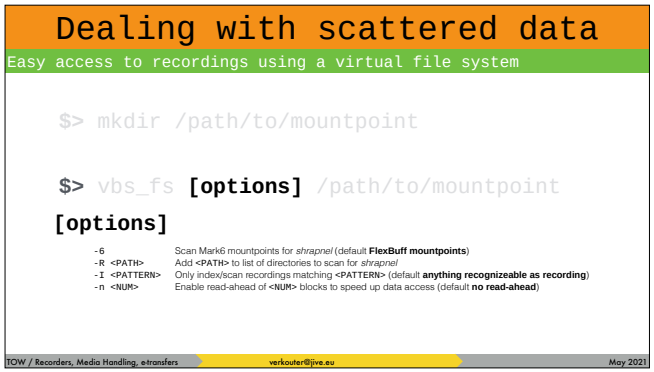
The vbs_fs program is written in multithreaded c++ for raw speed and does the i/o scheduling for you, even if multiple users are accessing the recordings through the same mountpoint.



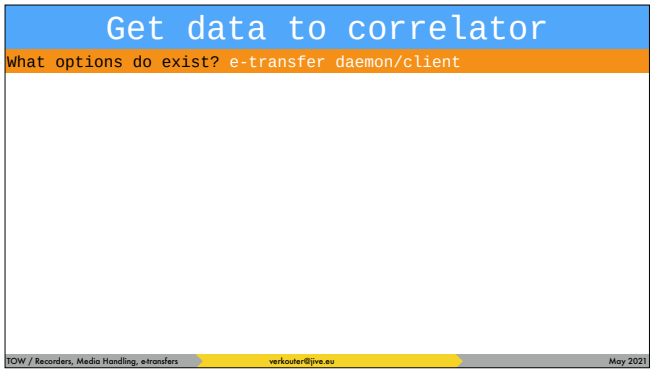
In order to use it, you create a directory for the mountpoint first and then [click] it is a matter of mounting the virtual file system on that mountpoint, [click] after which recordings show up in that mountpoint as ordinary files



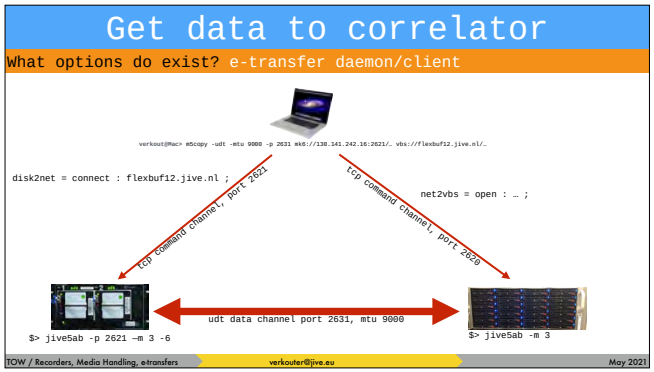
Interesting to know is that for Mark6 files vbs_fs strips all the application headers so this is 100% VDIF payload!!!



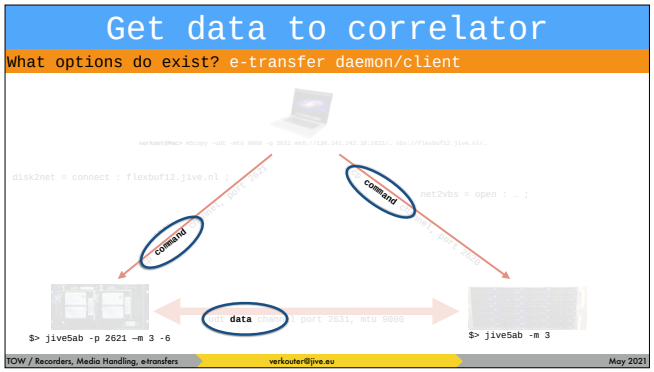
Useful options for vbs_fs to know about are those influencing the mountpoints to scan for recordings, a filter to index only recordings matching a certain pattern, or to enable read-ahead to make data access a lot faster.



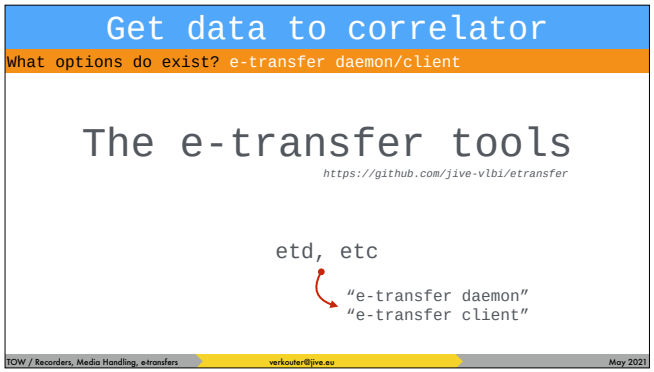
Now that you know about the fuse file system to present recordings as single files, it might be good to introduce another e-shipping option that is being developed.



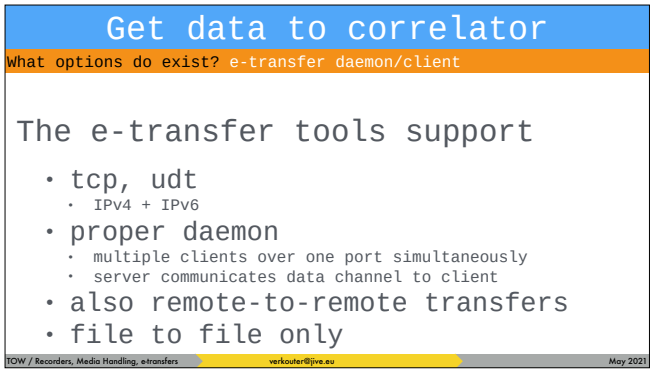
Remember the jive5ab + m5copy situation?



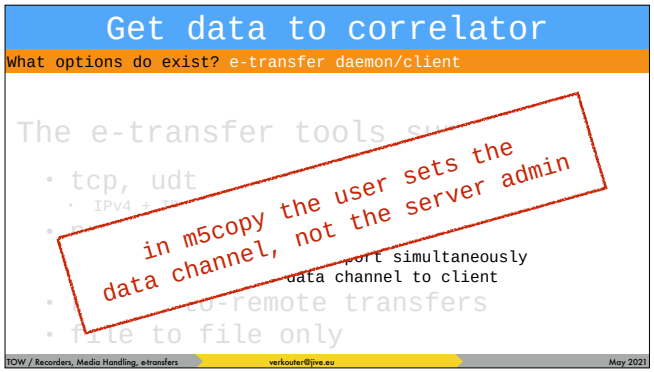
One of the biggest issues is that there is [click] no control channel between the sender and receiver of the data! That doesn't sound like a big thing but it is.



Based on the experiences with UDT, jive5ab and m5copy a proper daemon/client pair of programs were developed, [click] the etransfer daemon and client

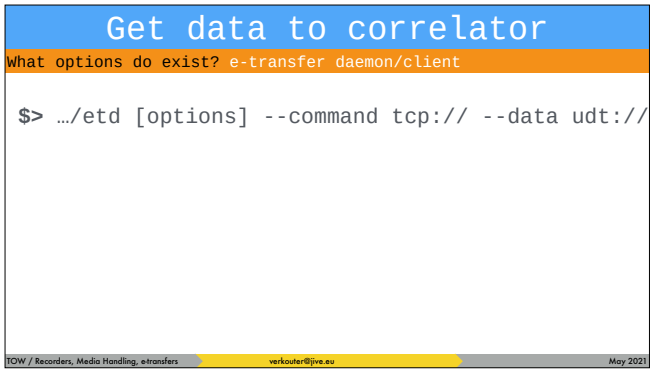


The important properties are these

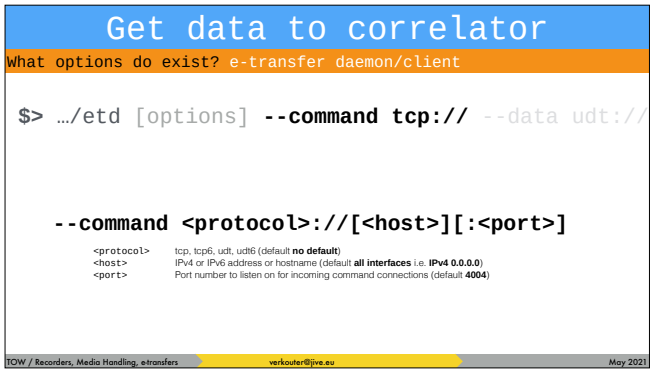


and for server administrators these properties are the most important!

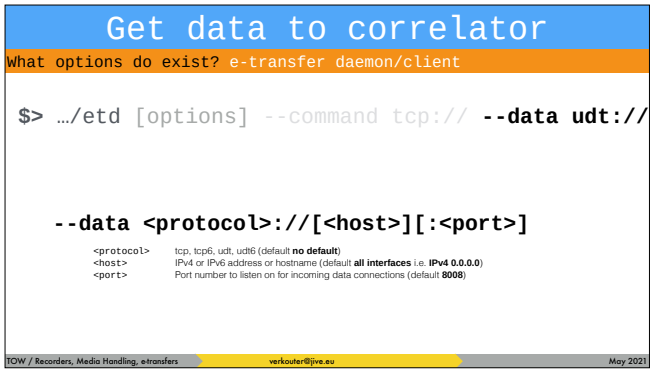
Because [click] in m5copy the CLIENT sets the data channel but that is basically just wrong!



In order to use the system, run the daemon in the data center. The daemon requires you to specify

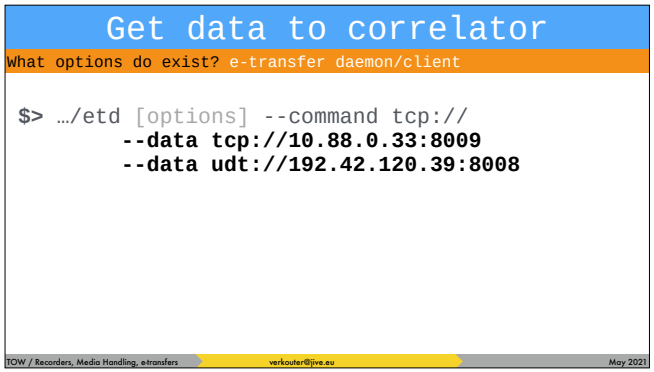


a command channel where to listen on for incoming client connections

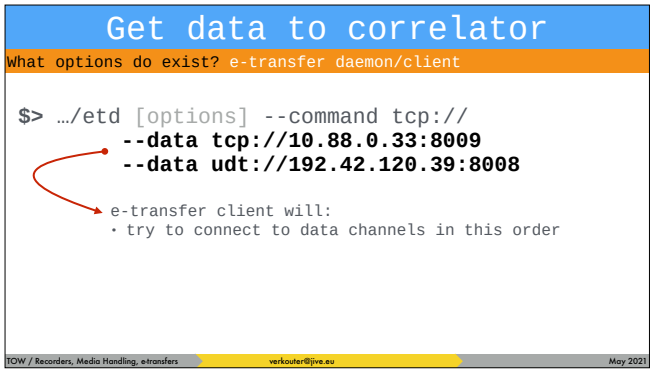


and at least one data channel over which you allow incoming data

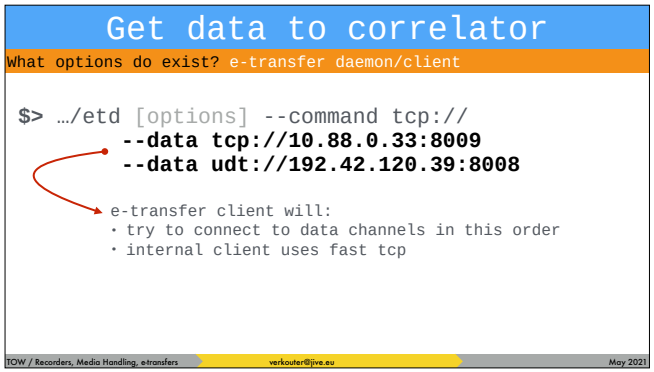
---



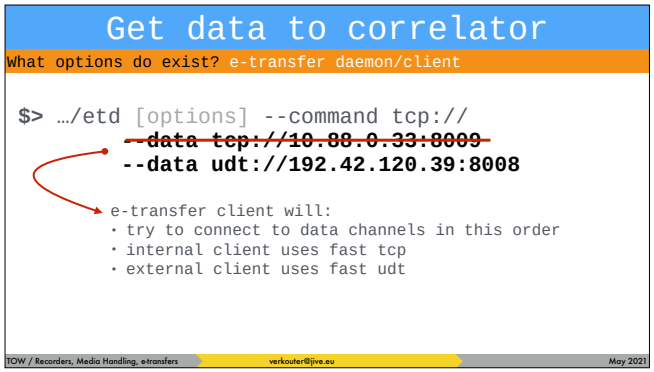
if you specify multiple data channels interesting things can happen!



the e-transfer client will attempt to connect the data channels in the order they were specified on the daemon command line!

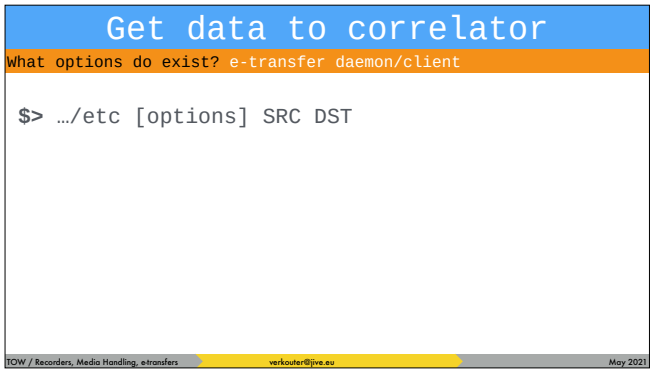


and in this case it means that if you do an internal data transfer, it uses tcp which is faster inside your institute.

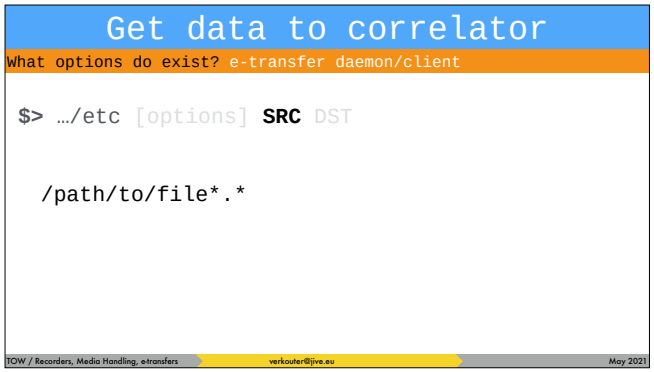


but an external client cannot connect to the internal address and so will use the fast UDT channel

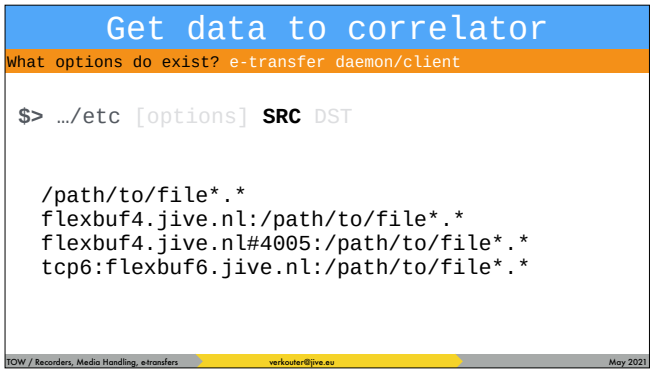
---



The client command looks a bit like secure copy

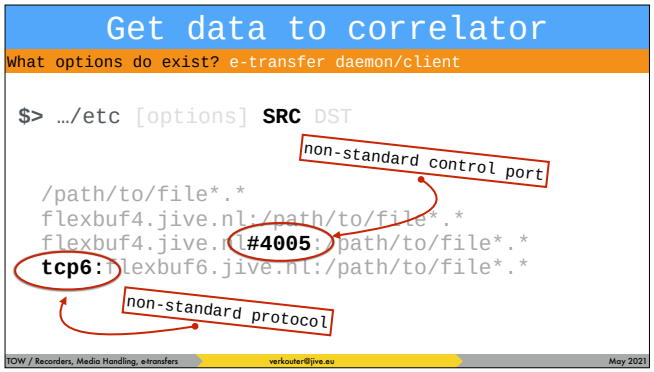


you can specify local files … or

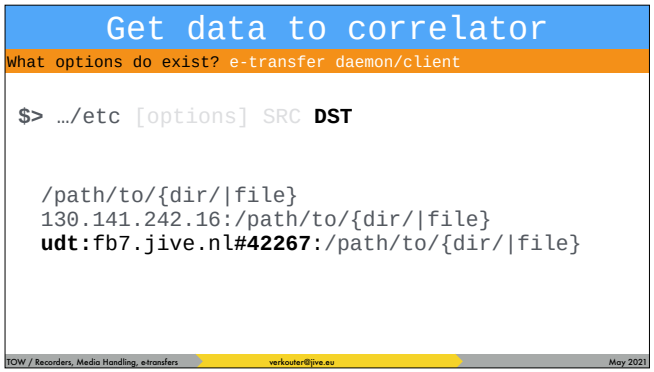


.. remote files - and because the system supports

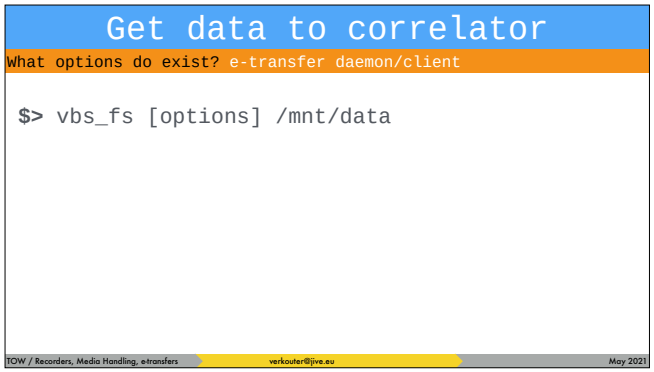
remote to remote transfers



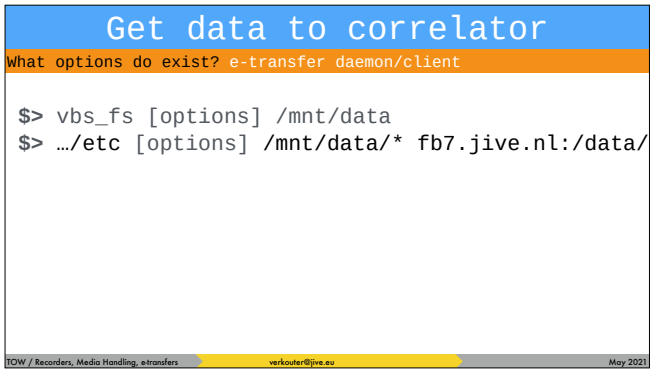
you may have to encode a lot of information in just one location …



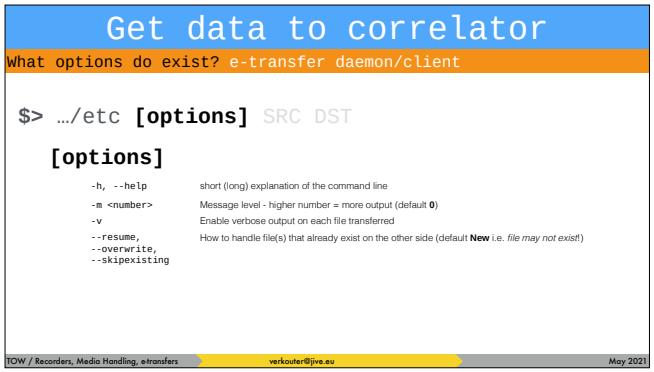
because the destination could be completely different again.



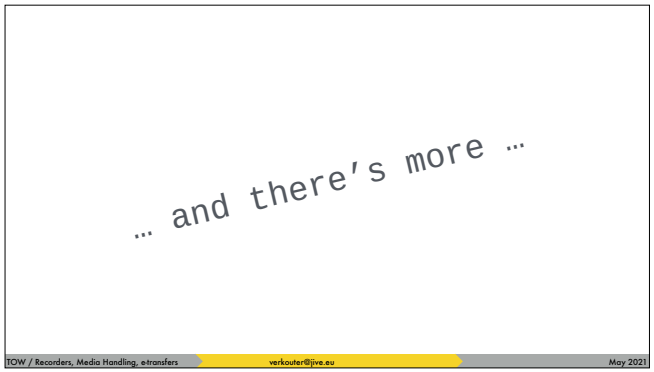
The combination of vbs_fs with the e-transfer system …



… makes it is easy to transfer the files like this.

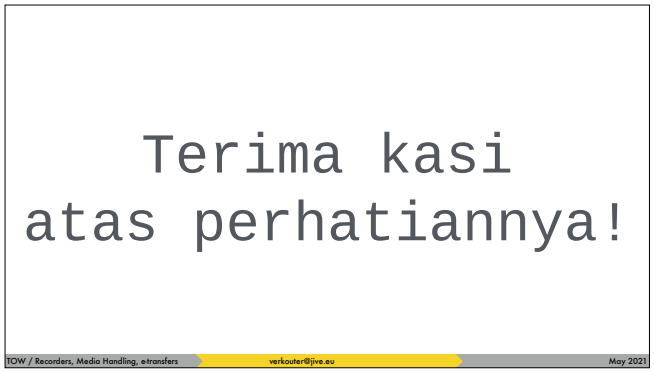


Some of the important options for the client are these. New ones may be added in the future, depending on your requests.



And there is so much more that I could be explaining but there is no time!

I still hope you learnt a lot in this lecture.



in any case, many thanks for your attention!