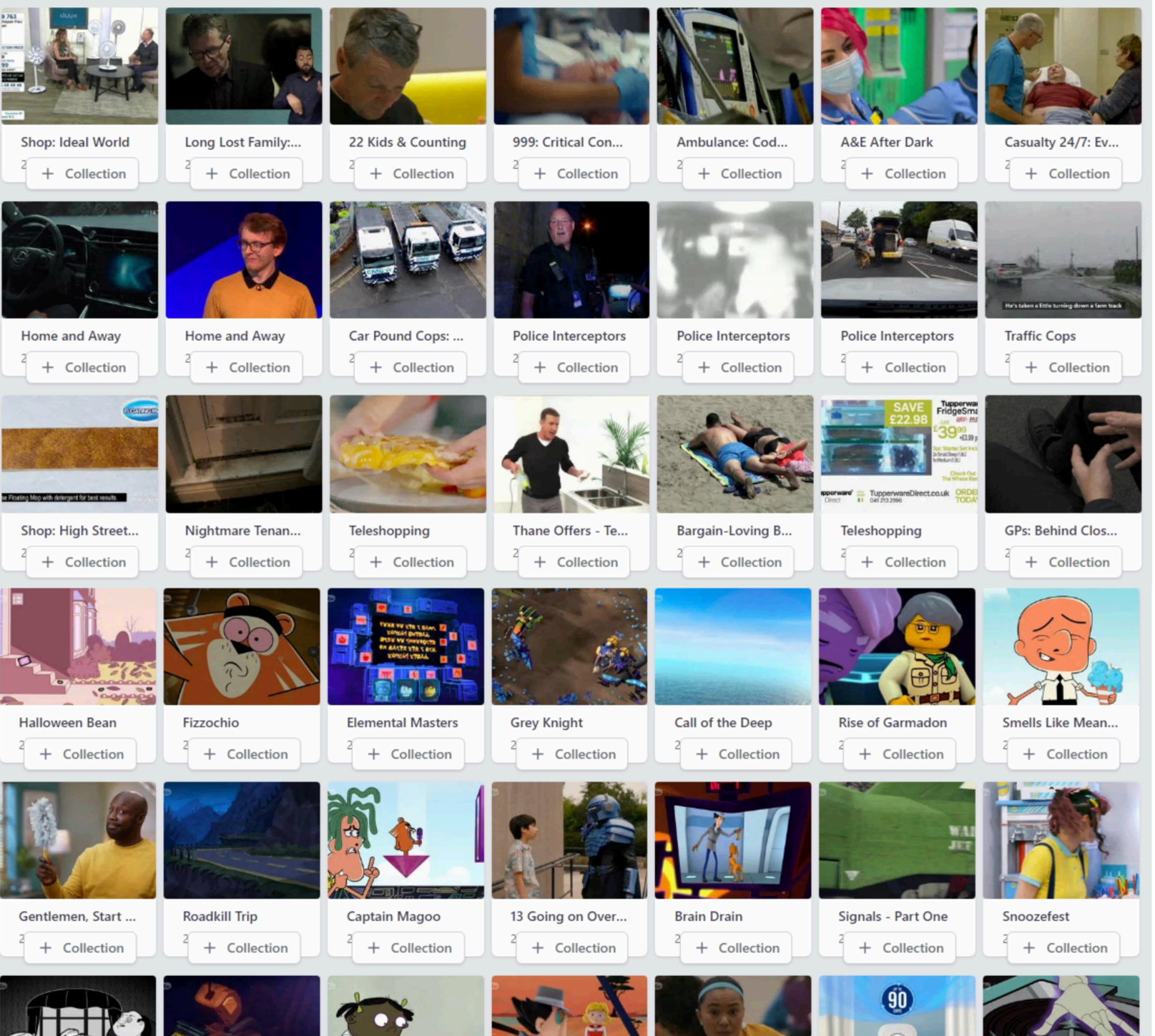
System for Television Off-air Recording and Archiving

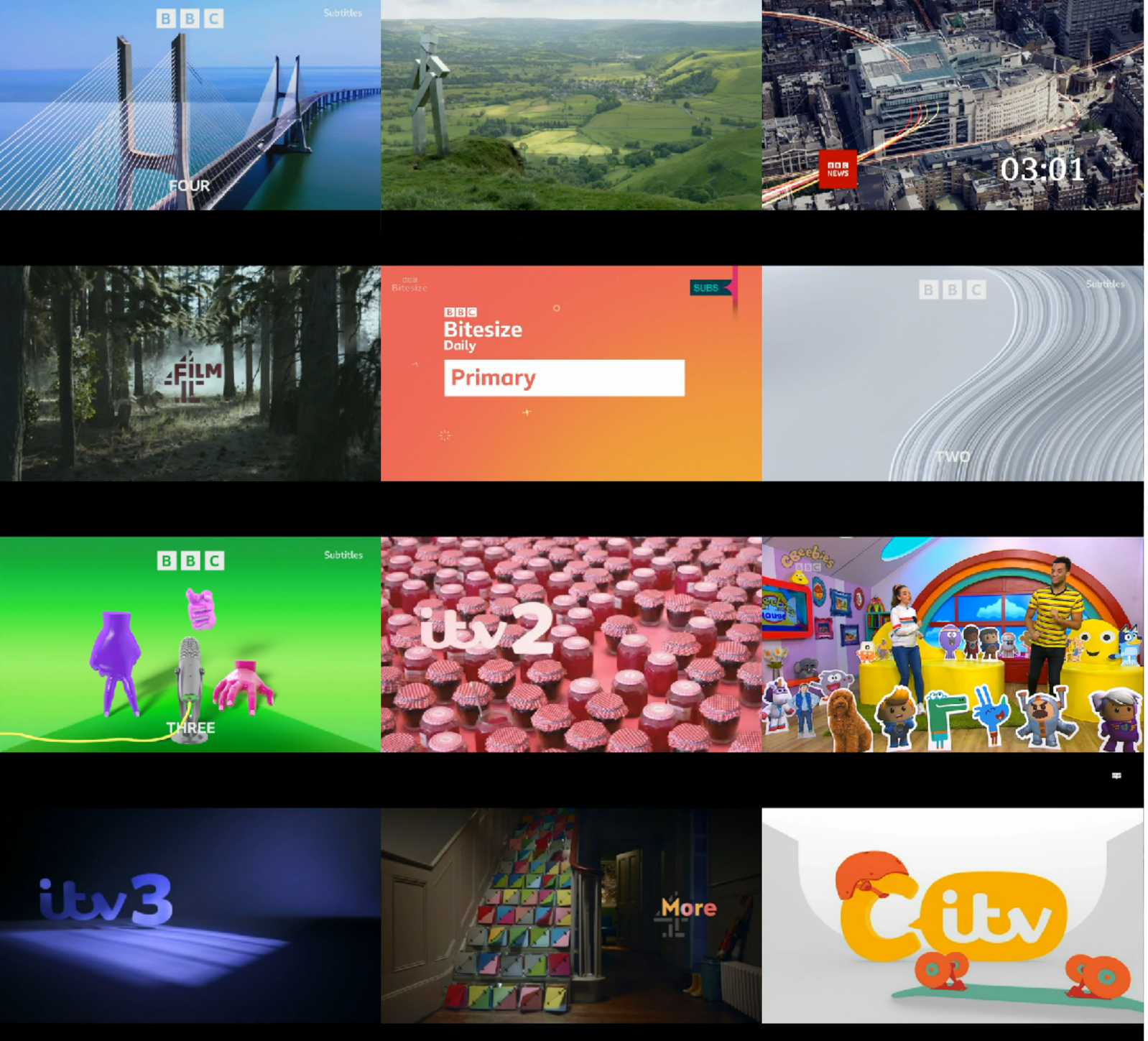
JOANNA WHITE

KNOWLEDGE AND COLLECTIONS DEVELOPER (SHE/HER)

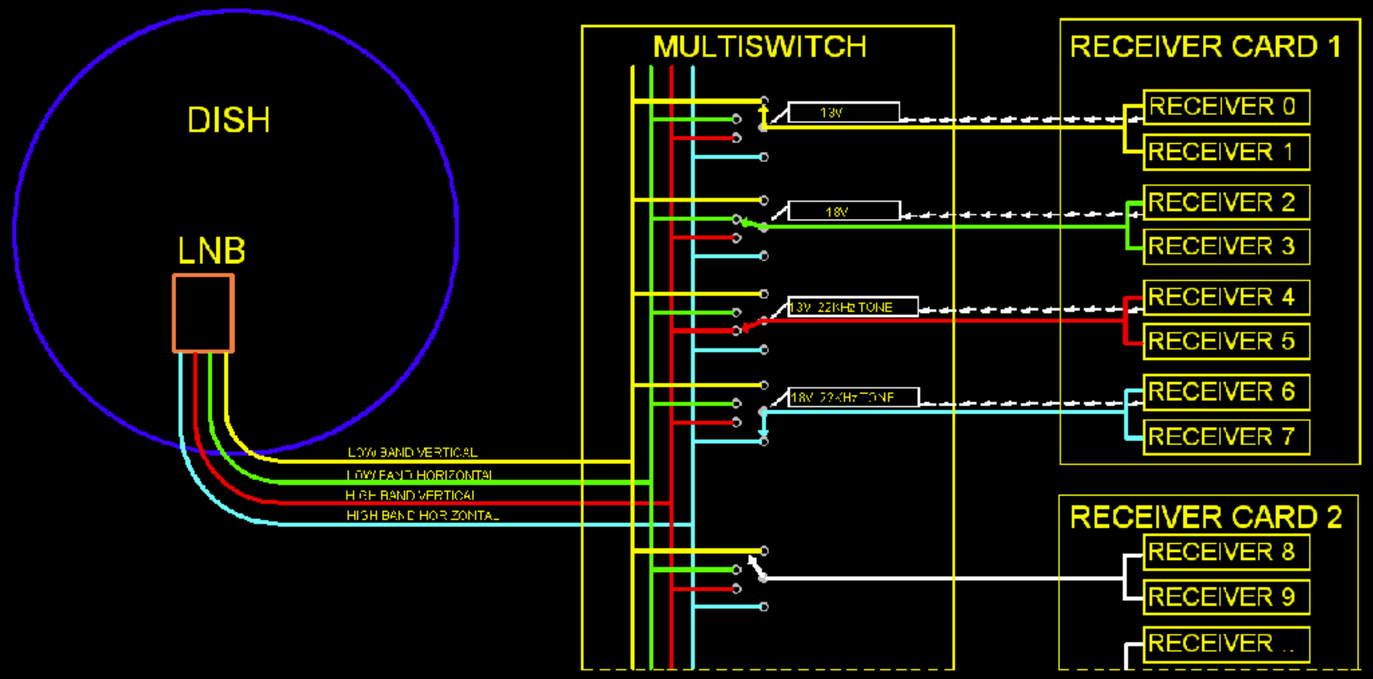
BFI NATIONAL ARCHIVE

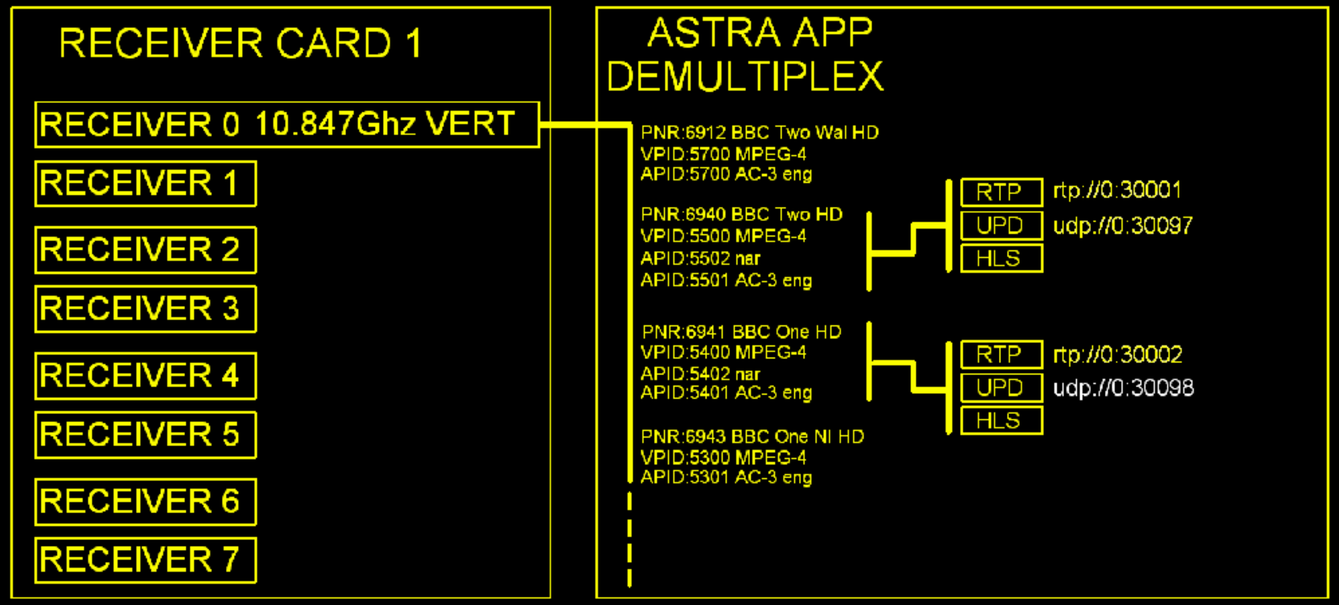


- Nearly 1 million moving image assets digitised and catalogued in the BFI Collections Information Database (CID)
- Nearly 650,000 of those are off-air TV
- Another 800,000 preserved but still to be catalogued in CID
- BFI is the National Television Archive designated by OFCOM under Provision of Broadcasting Act, 1990



- Started recording off-air TV to one inch video tape in 1985
- Teams worked around the clock to capture select broadcast shows
- 2015 off-air recording was automated with BFI fork of BBC's Redux project
- BBC Redux project closed May 2022
- BFI launched an R&D project in 2021 resulting in STORA
- STORA modelled on Redux to avoid breaks in existing automated workflows

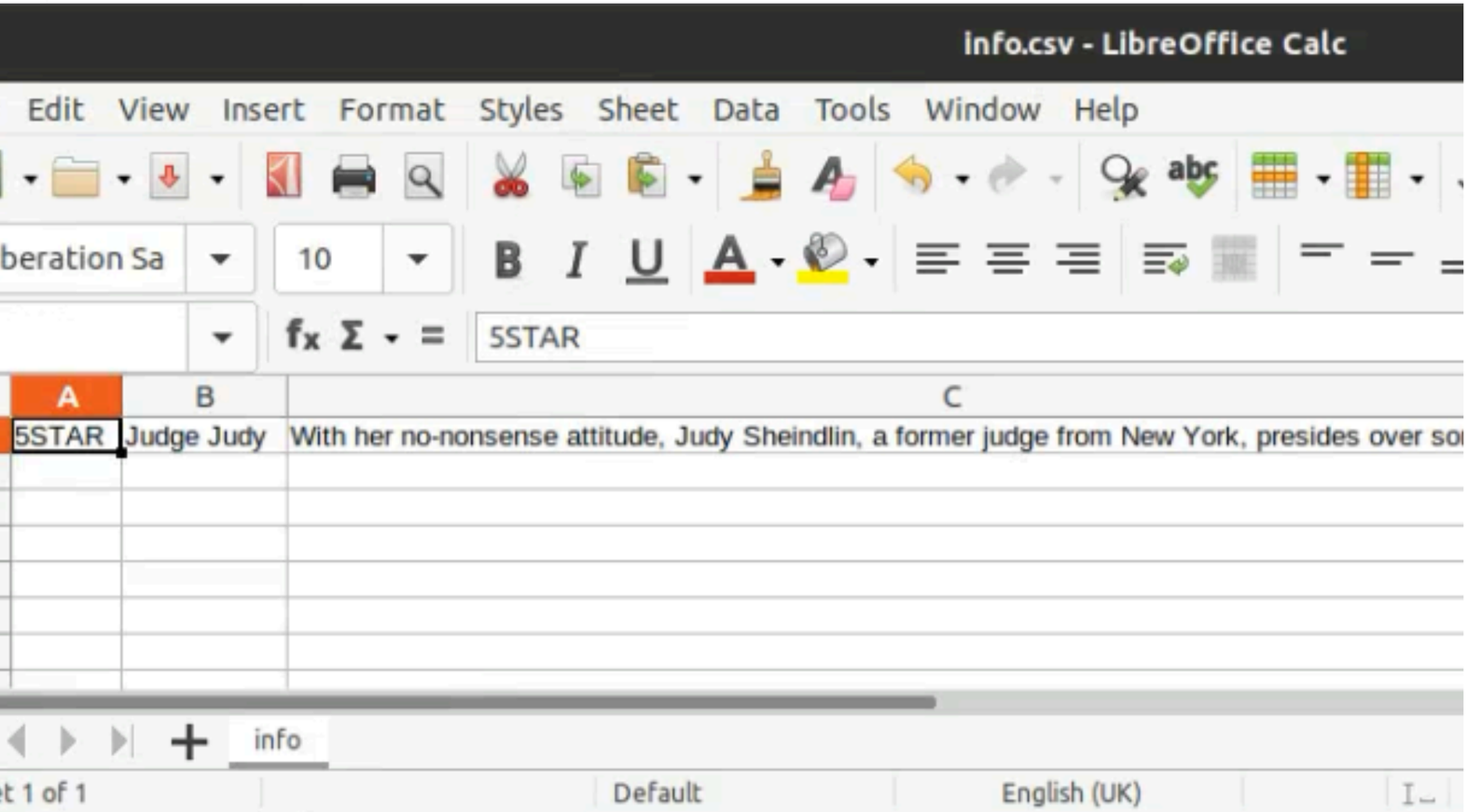




- Astra satellites broadcast across Europe
- Passed through Quatro Low Noise Block then TBS TV PCi receiver cards
- Signals routed through patch fields to a multi-switch to select bands & polarisation
- STORA has 3 multi-switches allowing for up to 24 different multiplexes
- Cesbo Astra app demuxes each channel's MPEG-TS into Single Program Transport Stream
- Creates Unicast RTP stream and Unicast UDP stream, both needed by STORA

2024/01/16/5star/09-05-00-476-00-30-00

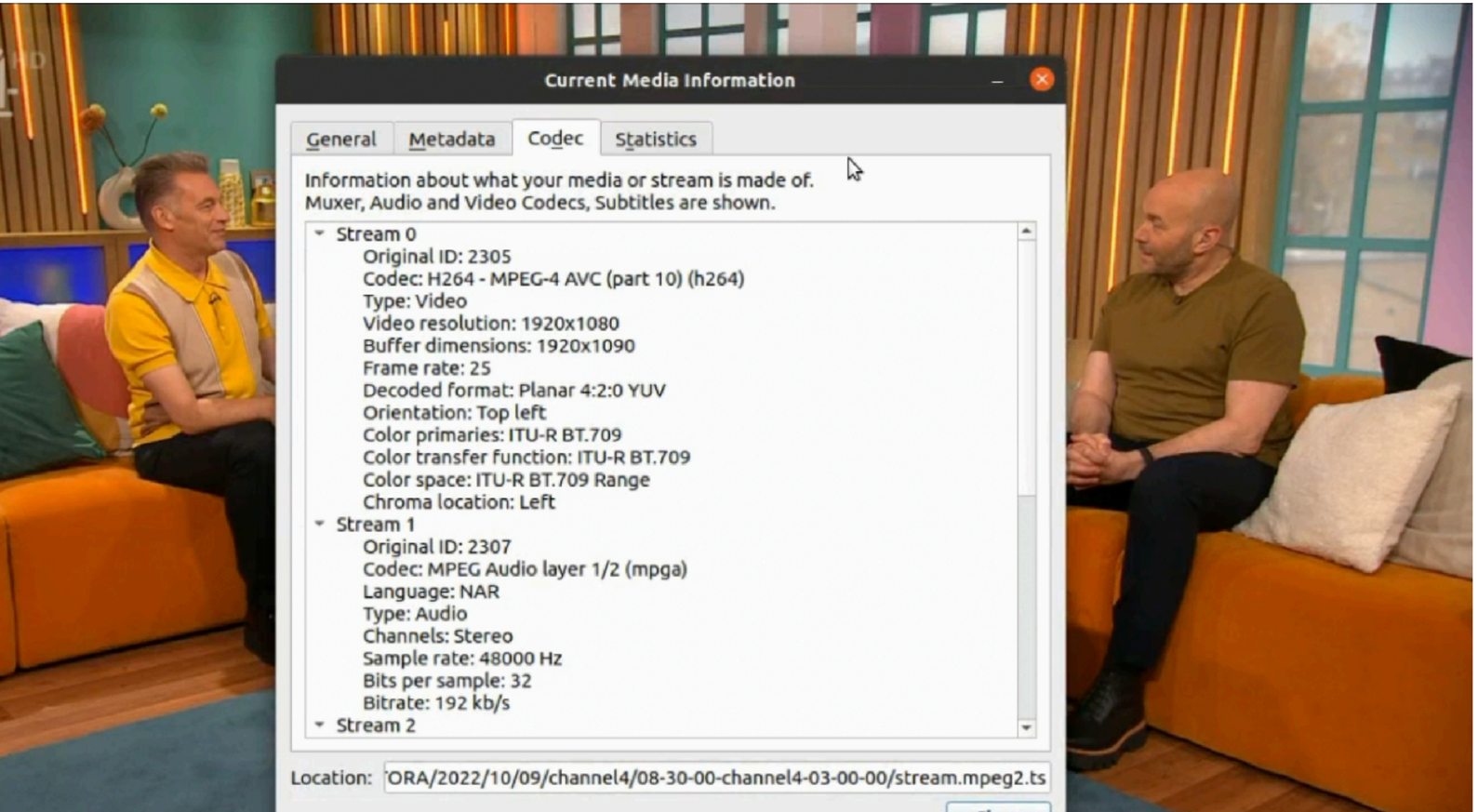
info.csv	184 bytes	Mon
stream.mpeg2.ts	516.1 MB	Mon
subtitles.vtt	93.9 kB	Mon



- Modelling BBC Redux folder structures file formatting and video standard
- Folders carry metadata for date, channel, time of broadcast, duration and EventID
- Contents of folder include info.csv, a CSV format document containing programme information from the stream metadata
- Captured video is not re-encoded but raw PES data wrapped video, audio, subtitles and information tables
- Extracted transcript of the spoken word of the programmed saved to WEBVTT
- All this data critical to preservation goals

2024/01/16/5star/09-05-00-476-00-30-00

info.csv	184 bytes	Mon
stream.mpeg2.ts	516.1 MB	Mon
subtitles.vtt	93.9 kB	Mon



- Modelling BBC Redux folder structures file formatting and video standard
- Folders carry metadata for date, channel, time of broadcast, duration and EventID
- Contents of folder include info.csv, a CSV format document containing programme information from the stream metadata
- Captured video is not re-encoded but raw PES data wrapped video, audio, subtitles and information tables
- Extracted transcript of the spoken word of the programmed saved to WEBVTT
- All this data critical to preservation goals

2024/01/16/5star/09-05-00-476-00-30-00

info.csv	184 bytes	Mon
stream.mpeg2.ts	516.1 MB	Mon
subtitles.vtt	93.9 kB	Mon

Open subtitles.vtt Save

```
WEBVTT

STYLE

00:00:02.120 --> 00:00:04.120
<font color="#ffff00">Your Honour, this is case number 258</font>
<font color="#ffff00">on the calendar</font>

00:00:04.160 --> 00:00:04.280
<font color="#ffff00">Your Honour, this is case number 258</font>
<font color="#ffff00">on the calendar</font>

00:00:04.320 --> 00:00:06.080
<font color="#ffff00">in the matter of Lawler vs Bell.</font>

00:00:06.120 --> 00:00:08.120
<font color="#ffff00">Parties have been sworn in. You may</font>
<font color="#ffff00">be seated. Have a seat, please.</font>

00:00:08.160 --> 00:00:08.920
<font color="#ffff00">Parties have been sworn in. You may</font>
<font color="#ffff00">be seated. Have a seat, please.</font>

00:00:08.960 --> 00:00:10.960
Mr Lawler,
I assume this is your daughter.

00:00:11.000 --> 00:00:11.920
Mr Lawler,
```

- Modelling BBC Redux folder structures file formatting and video standard
- Folders carry metadata for date, channel, time of broadcast, duration and EventID
- Contents of folder include info.csv, a CSV format document containing programme information from the stream metadata
- Captured video is not re-encoded but raw PES data wrapped video, audio, subtitles and information tables
- Extracted transcript of the spoken word of the programmed saved to WEBVTT
- All this data critical to preservation goals

python™

VLC MEDIA PLAYER

Nagios®

System for Television Off-air Recording and Archiving

FFmpeg

CCExtractor

Media Info

libdvbtee

```
{
    "hasNext": false,
    "total": 14,
    "item": [
        {
            "id": "1ac5cd5b-a71e-4e5c-9251-65567c170036",
            "title": "Hot Cakes",
            "dateTime": "2022-10-15T00:00:00.000Z",
            "duration": 30,
            "certification": {},
            "meta": {},
            "attribute": [
                "subtitles",
                "16x9",
                "hd"
            ],
            "summary": {
                "short": "Gareth is caught off guard when a customer orders a cake to celebrate a part of their body",
                "medium": "Pride Week in Cardiff and a last-minute cake causes chaos for the busy team. Meanwhile, Gareth is caught off-guard by an order for a cake to celebrate part of a customer's body"
            },
            "asset": {
                "id": "a6a557a8-2ed5-5a4b-9882-a3b89c634118",
                "type": "episode",
                "number": 2,
                "total": 5,
                "certification": {},
                "meta": {
                    "episode": "2",
                    "episodeTotal": "5"
                },
                "category": [
                    {
                        "code": "leisure-hobbies",
                        "name": "Leisure hobbies",
                        "dvb": "A000"
                    },
                    {
                        "code": "leisure-hobbies:cooking",
                        "name": "Cooking",
                        "dvb": "A500"
                    },
                    {
                        "code": "news-current-affairs",
                        "name": "News/Current Affairs",
                        "dvb": "2000"
```

```
[
    {
        "start": "2022-10-15 00:00:00",
        "duration": 30,
        "channel": "bbcthree",
        "programme": "Hot Cakes"
    },
    {
        "start": "2022-10-15 00:30:00",
        "duration": 70,
        "channel": "bbcthree",
        "programme": "RuPaul's Drag Race UK"
    },
    {
        "start": "2022-10-15 01:40:00",
        "duration": 30,
        "channel": "bbcthree",
        "programme": "Hometown: A Teenage Killing"
    },
    {
        "start": "2022-10-15 02:10:00",
        "duration": 30,
        "channel": "bbcthree",
        "programme": "Hometown: A Teenage Killing"
    },
    {
        "start": "2022-10-15 02:40:00",
        "duration": 10,
        "channel": "bbcthree",
        "programme": "Press X to Continue"
    },
    {
        "start": "2022-10-15 02:50:00",
        "duration": 910,
        "channel": "bbcthree",
        "programme": "Close"
    },
```


- Two separate approaches to recording the RTP stream originally in separate script but now combined
- Electronic Programme Guide (EPG) data retrieved from commercial supplier via their Rest API
- EPG data converted to JSON schedule for a single days programmes, per channel
- Script run once per day launching around midnight and runs until all scheduled items have concluded
- Script relaunched immediately when finished by shell scripts that monitor for scripts that have completed running
- Script indebted to ActiveNation script written in 2015

ActiveState Code » Recipes

Languages Tags Authors Sets

VLC.PY STREAM CAPTURE SCHEDULER SCRIPT (PYTHON RECIPE)

Capture network streams using vlc.py on a schedule.

Download Python, 425 lines

```
"""

A script to capture network streams using VLC, based on a scheduler. It was originally designed for mpeg tr

Last updated 19th August 2015.


Dependencies:
- My fork of Danny Yoo's getch()-like function: https://code.activestate.com/recipes/579095
- vlc.py http://wiki.videolan.org/Python_bindings


The end user will need to create two JSON files in the same directory as the script:
- capture.json to store the network stream/channel information
- schedule.json to store the timer recordings


Example format of capture.json:

{
        "stream name one":"stream one address",
        "stream name two":"stream two address"
}


Example format of schedule.json:

[
        { "start":"2015-08-18 18:08:00", "duration":1, "channel":"stream name one", "programme":"Test Recor
        { "start":"2015-08-18 19:20:00", "end":"2015-08-18 20:20:00", "channel":"stream name two", "program
]

You can specify either the length of the recording or an end datetime. The programme field is simply a desc


Commands:

Once the script is running, it will automatically parse the channel list and schedule. There are three basi

- pressing "R" reloads the schedule and will update any upcoming scheduled recordings
- pressing "C" reloads the channel list
- pressing "Q" exits the script
```

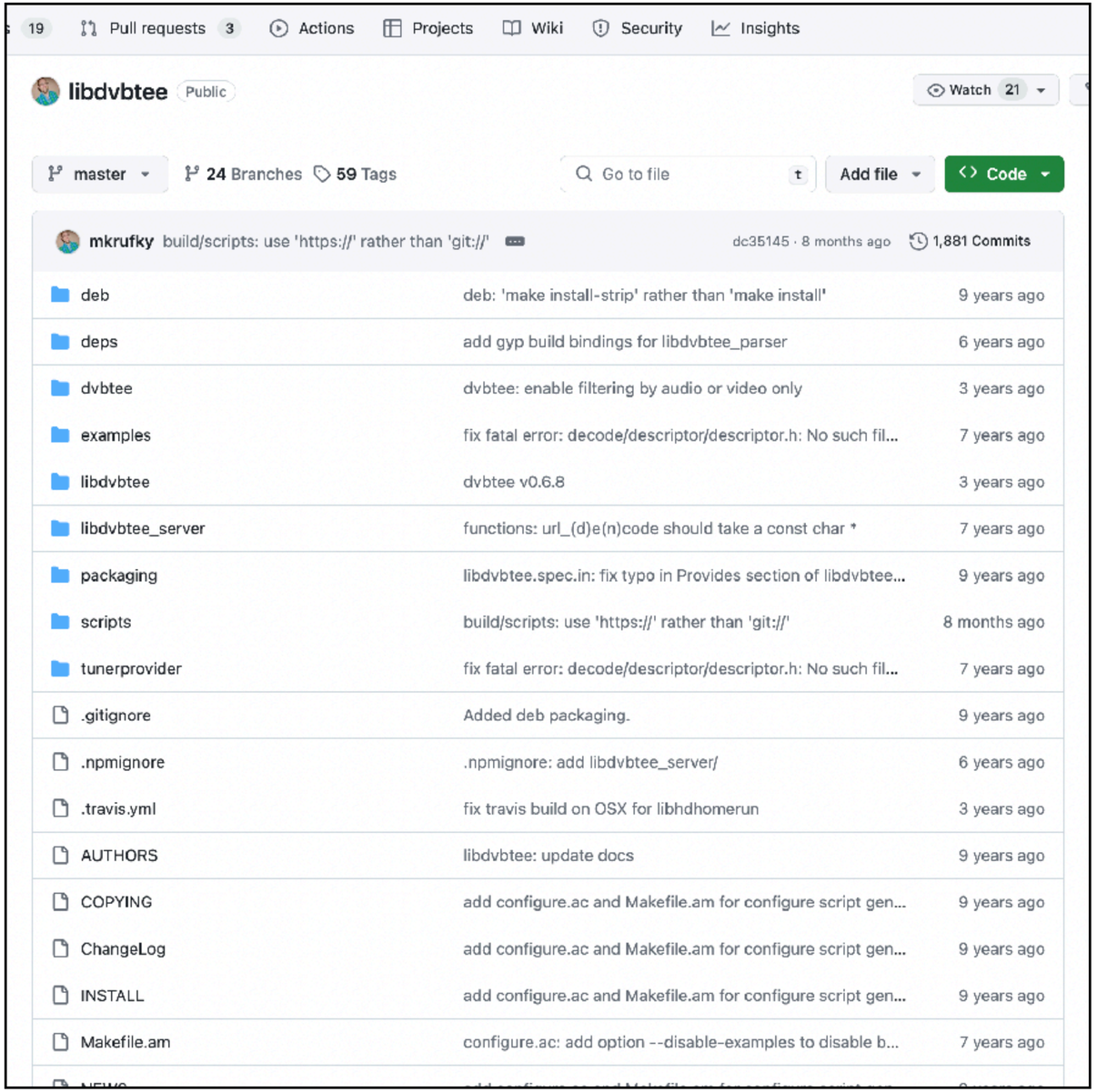
- Two separate approaches to recording the RTP stream originally in separate script but now combined
- Electronic Programme Guide (EPG) data retrieved from commercial supplier via their Rest API
- EPG data converted to JSON schedule for a single days programmes, per channel
- Script run once per day launching around midnight and runs until all scheduled items have concluded
- Script relaunched immediately when finished by shell scripts that monitor for scripts that have completed running
- Script indebted to ActiveNation script written in 2015

```
# dvbtee v0.6.8 - http://github.com/mkrufky/libdvbtee

store PAT: v2, ts_id: 1
           0 | 10
        5106 | 64
store NIT: v1, network_id 1
   ts_id | orig_network_id
   00001 | 1
store PMT: v29, service_id 5106, pcr_pid 101
  es_pid | type
      65 | 0x1b (Video H.264) |
      66 | 0x04 (Audio MPEG-2) | NAR
      67 | 0x06 (Private (AC3/EAC3/TT/ST)) |
      68 | 0x06 (Private (AC3/EAC3/TT/ST)) | eng
      69 | 0x06 (Private (AC3/EAC3/TT/ST)) |
store EIT-255: v11 | ts_id 2034 | network_id 1 service_id 5106 | table id: 0x4e, last_table id: 0x4e
  15:00 - 16:00 : New: Riddiculous
  16:00 - 17:00 : New: Tipping Point
UPDATING TABLE 70

TSID#0001: [{"programs":[{"number":0,"pid":16},{"number":5106,"pid":100}],"tableId":0,"tableName":"PAT","tsId":1,"version":2},{},{"pcrPid":101,"program":5106,"streams":[{"pid":101,"
streamType":27,"streamTypeString":"Video H.264"},{"descriptors":[{"ISO639Lang":[{"audioType":0,"language":"NAR"}],"descriptorTag":10}],"pid":102,"streamType":4,"streamTypeString":"A
udio MPEG-2"},{"pid":103,"streamType":6,"streamTypeString":"Private (AC3/EAC3/TT/ST)"},{"descriptors":[{"ISO639Lang":[{"audioType":0,"language":"eng"}],"descriptorTag":10}],"pid":10
4,"streamType":6,"streamTypeString":"Private (AC3/EAC3/TT/ST)"},{"pid":105,"streamType":6,"streamTypeString":"Private (AC3/EAC3/TT/ST)"}],"tableId":2,"tableName":"PMT","version":29}
,{"descriptors":[],"tableId":112,"tableName":"TOT","time":1706714721}]


NET_ID#0001: [{"networkId":1,"tableId":64,"tableName":"NIT","tsList":[{"origNetworkId":1,"tsId":1}],"version":1},{}]

NET_SVC_ID#07f2: [{"events":[{"descriptors":[{"descriptorTag":77,"lang":"eng","name":"New: Riddiculous","text":"Ranvir Singh hosts the rapid and ruthless quiz as contestants Greg an
d Jamie, Gina and Kieran, and Chris and Nihad take on Riddlemaster Henry Lewis in a bid to solve his riddles. S2 Ep23"}],"eventId":61,"f_free_ca":false,"lengthSec":65536,"runningSta
tus":4,"startTime":1012338589696,"unixTimeBegin":1706713200,"unixTimeEnd":1706716800},{"descriptors":[{"descriptorTag":77,"lang":"eng","name":"New: Tipping Point","text":"Ben Shepha
rd hosts the quiz show in which three players take on an extraordinary machine in the hope of winning a cash jackpot."}],"eventId":75,"f_free_ca":false,"lengthSec":65536,"runningSta
tus":1,"startTime":1012338655232,"unixTimeBegin":1706716800,"unixTimeEnd":1706720400}],"lastTableId":78,"networkId":1,"serviceId":5106,"tableId":78,"tableName":"EIT","tsId":2034,"ve
rsion":11}]
```

```
lous","text":"Ranvir Singh hosts the rapid and ruthless quiz as contestants Greg an
is riddles. S2 Ep23"}],"eventId":61,"f_free_ca":false,"lengthSec":65536,"runningSta
s":[{"descriptorTag":77,"lang":"eng","name":"New: Tipping Point","text":"Ben Shepha
ing a cash jackpot."}],"eventId":75,"f_free_ca":false,"lengthSec":65536,"runningSta
d":78,"networkId":1,"serviceId":5106,"tableId":78,"tableName":"EIT","tsId":2034,"ve
```

```
            Started recording: US Capitol Attack Hearings (bbcnewshd)
            17-00-00 to 18-00-00 - duration 01-00-00
18:00:02  New running EventId: 16995
18:00:02  Event list updated: [16989, 16990, 16991, 16992, 17287, 16995]
18:00:02  Ending recording for previous programme
18:00:02  Initialising recording for path: /media/data/content/video/2022/10/13/bbcnewshd/18-00-00-16995-02-00-00/stream.mpeg2.ts
            Started recording: Outside Source (bbcnewshd)
            18-00-00 to 20-00-00 - duration 02-00-00
20:00:02  New running EventId: 16996
20:00:02  Event list updated: [16990, 16991, 16992, 17287, 16995, 16996]
20:00:02  Ending recording for previous programme
20:00:02  Initialising recording for path: /media/data/content/video/2022/10/13/bbcnewshd/20-00-00-16996-01-00-00/stream.mpeg2.ts
            Started recording: The Context with Christian Fraser (bbcnewshd)
            20-00-00 to 21-00-00 - duration 01-00-00
21:00:02  New running EventId: 16997
21:00:02  Event list updated: [16991, 16992, 17287, 16995, 16996, 16997]
21:00:02  Ending recording for previous programme
21:00:02  Initialising recording for path: /media/data/content/video/2022/10/13/bbcnewshd/21-00-00-16997-00-30-00/stream.mpeg2.ts
            Started recording: BBC News at Ten (bbcnewshd)
            21-00-00 to 21-30-00 - duration 00-30-00
21:30:03  New running EventId: 16998
21:30:03  Event list updated: [16992, 17287, 16995, 16996, 16997, 16998]
21:30:03  Ending recording for previous programme
21:30:03  Initialising recording for path: /media/data/content/video/2022/10/13/bbcnewshd/21-30-00-16998-00-15-00/stream.mpeg2.ts
            Started recording: The Papers (bbcnewshd)
            21-30-00 to 21-45-00 - duration 00-15-00
```

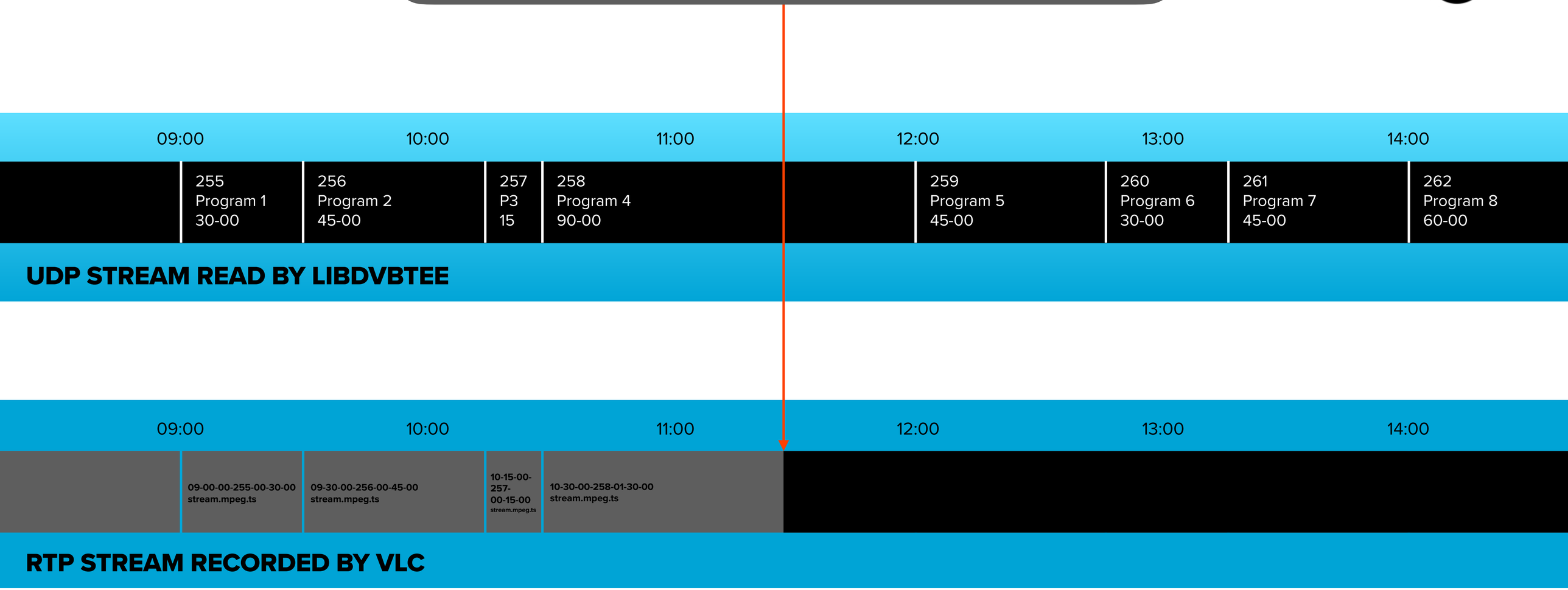
- Second script started monitoring the UDP stream service information data for change to the EventID
- Script stores the last five programme EventIDs in a list when a new number appears it triggers new recording
- Script loops indefinitely monitoring this UDP data, creating and placing programme recordings in unique folder paths
- LIBDVBTEE project makes this approach possible, and supports parsing of UDP streams
- Shell command is spawned from Python subprocess module and the response is collected and converted into dictionary



libdvbtee

- Second script started monitoring the UDP stream service information data for change to the EventID
- Script stores the last five programme EventIDs in a list when a new number appears it triggers new recording
- Script loops indefinitely monitoring this UDP data, creating and placing programme recordings in unique folder paths
- LIBDVBTEE project makes this approach possible, and supports parsing of UDP streams
- Shell command is spawned from Python subprocess module and the response is collected and converted into dictionary

epg_assessment_channel_recorder.py

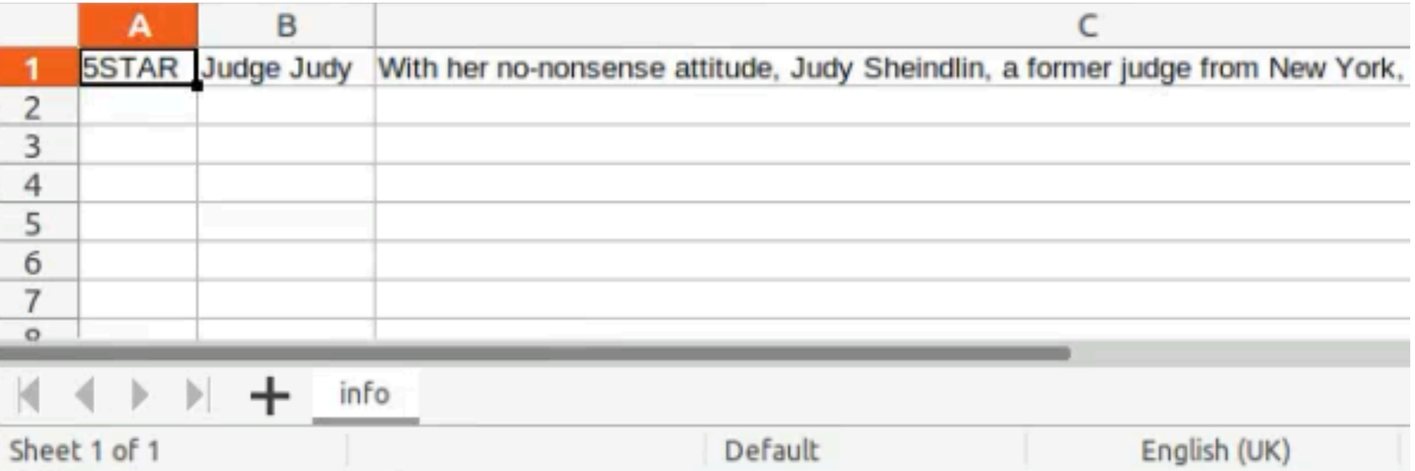


```python
for key, val in running.items():
    if key not in event_list:
        # Add current eventId to event_list
        event_list.append(key)
        time_print(f"New running EventId: {key}", False)
        time_print(f"Event list updated: {event_list}", False)
        prog_info = val.split(', ')

        # Initialise recording path - needs date paths adding
        outfile = initialise_ts_rs(prog_info[1], key, prog_info[0])

        if len(event_list) > 1:
            # Stop existing recording
            time_print("Ending recording for previous programme", False)
            player.stop() # Stop playback
            player.release() # Close the player
            inst.release() # Destroy the instance
            indent_print(f"STOP Instance: {inst}, Player: {player}, Media: {media}", False)

        # Start new recording using initialised outfile as destination
        time_print(f"Initialising recording for path: {outfile}", False)
        (inst, player, media) = record_stream(rtp, outfile)
        player.play()
        indent_print(f"START Instance: {inst}, Player: {player}, Media: {media}", False)
        indent_print(f"Started recording: {prog_info[4]} ({CHANNEL})", False)
```

```python
def record_stream(instream, outfile):
    '''
    Record the network stream to the output file.
    Create VLC instance that launches demux dump and
    appends to stream (if already exists) or creates new
    '''

    inst = vlc.Instance("--demux=dump", f"--demuxdump-file={outfile}", \
                        "--demuxdump-append")
    player = inst.media_player_new()
    media = inst.media_new(instream)
    media.get_mrl()
    player.set_media(media)
    return (inst, player, media)
```

- The VLC python bindings create a VLC Instance and media player object
- They are called in the main script to stop a completed recording and start a new recording
- We use the demux dump command in the instance from VLC's demux library, saving the stream without decoding
- The append flag is used to ensure breaks in stream recordings are appended to existing files, not overwritten
- Restart warning text file is automatically written to folder containing programme recording to indicate the break

```python
def get_metadata(filepath):
    '''
    Use subprocess to capture list of 'Running'
    data using MediaInfo
    '''

    cmd = [
        'mediainfo',
        filepath
    ]

    try:
        mdata = subprocess.check_output(cmd)
    except Exception as err:
        LOGGER.warning("Error with subprocess call: %s", err)

    running_list = []
    if mdata:
        mdata = mdata.decode()
        mdata = mdata.split('\n')
        for m in mdata:
            if 'Running' in str(m) or 'Not running' in str(m):
                running_list.append(m)

        return running_list
```



	A	B	C
1	5STAR	Judge Judy	With her no-nonsense attitude, Judy Sheindlin, a former judge from New York,
2			
3			
4			
5			
6			
7			

- The info.csv file is created from metadata retrieved from Media Area software MediaInfo
- Python subprocess spawns shell to retrieve metadata of recent completed recording
- The data is converted to a list and written to a new CSV file using Python CSV module
- Similarly the subtitles.vtt file is created using from subtitles extracted from stream using CCExtractor
- Similarly the subtitles.vtt file is created using from subtitles extracted from stream using CCExtractor
- Python subprocess spawns shell to retrieve subtitles
- CCExtractor formats the file for WEBVTT which is imported to CID database as searchable text

```python
def make_vtt(filepath, folder):
    '''
    Use subprocess to create VTT file
    '''

    outpath = os.path.join(folder, 'subtitles.vtt')

    cmd = [
        'ccextractor',
        '-out=webvtt',
        filepath,
        '-o', outpath
    ]

    try:
        subprocess.call(cmd)
        return True
    except Exception as err:
        LOGGER.warning("Error with subprocess call: %s", err)
```

```webvtt
WEBVTT

STYLE

00:00:02.120 --> 00:00:04.120
<font color="#ffff00">Your Honour, this is case number 258</font>
<font color="#ffff00">on the calendar</font>

00:00:04.160 --> 00:00:04.280
<font color="#ffff00">Your Honour, this is case number 258</font>
<font color="#ffff00">on the calendar</font>

00:00:04.320 --> 00:00:06.080
<font color="#ffff00">in the matter of Lawler vs Bell.</font>

00:00:06.120 --> 00:00:08.120
<font color="#ffff00">Parties have been sworn in. You may</font>
<font color="#ffff00">be seated. Have a seat, please.</font>

00:00:08.160 --> 00:00:08.920
<font color="#ffff00">Parties have been sworn in. You may</font>
<font color="#ffff00">be seated. Have a seat, please.</font>

00:00:08.960 --> 00:00:10.960
Mr Lawler,
I assume this is your daughter.

00:00:11.000 --> 00:00:11.920
Mr Lawler,
I assume this is your daughter.
```

- The info.csv file is created from metadata retrieved from Media Area software MediaInfo
- Python subprocess spawns shell to retrieve metadata of recent completed recording
- The data is converted to a list and written to a new CSV file using Python CSV module
- Similarly the subtitles.vtt file is created using from subtitles extracted from stream using CCExtractor
- Similarly the subtitles.vtt file is created using from subtitles extracted from stream using CCExtractor
- Python subprocess spawns shell to retrieve subtitles
- CCExtractor formats the file for WEBVTT which is imported to CID database as searchable text

```bash
#!/usr/bin/bash

# Brian Fattorini, July 2022, modified February 2023.
# For the channel id provided check the channel is currently being picked up by the card as reported in the Astra software - "onair": true -

if [ "$#" -ne 1 ]; then
        echo "CRITICAL: Just one argument, channel name required"
        exit 2
fi

channame=$1
AUTH="xxxxxx:yyyyyyy"
ADDR="a.b.c.d:e/api/stream-status/"

myCommand="${eval which curl} -s --user \"$AUTH\" \"http://${ADDR}${channame}\" | ${eval which grep} onair | ${eval which cut} -d: -f 2" curlOutput=${eval $myCommand}

if [ "$curlOutput" = " true," ]; then
        echo "OK: Channel with ID '${channame}' is onair in Astra software"
        exit 0
elif [ "$curlOutput" = " false," ]; then
        echo "CRITICAL: Channel with ID '${channame}' is offair in Astra software"
        exit 2
else
        echo "CRITICAL: Stream with ID '${char
        exit 2
fi
```

```bash
define service {
        use                     bk-tv-rec1_service
        host_name               BK-TV-REC1
        check_command           check_ncpa!-t 'xxxxxxxx' -P yyyyy -M 'plugins/check_recording.sh' -q args='itv3'
        service_description     itv3 recording
}
```

```bash
#!/usr/bin/bash

# Brian Fattorini, June 2022
# For the channel name provided check there is an open file to which data is being written.
# It does this by providing a checksum and comparing the value to one derived four seconds later.

if [ "$#" -ne 1 ]; then
        echo "CRITICAL: Just one argument, channel name required"
        exit 2
fi

channame=$1

myCommand="${which} sudo ${which}lsof -n -P -l -M -u ouruser -w -Fn | ${which}grep \"\.ts$\" | ${which}grep "\/$channame\/" | ${which}uniq | ${which}sed \"s/^n//\" | ${w

value1=$(eval $myCommand)
sleep 4
value2=$(eval $myCommand)

if [ "$value1" != "$value2" ]; then
        echo "OK: Channel '${channame}' writing data to drive"
        exit 0
else
        echo "CRITICAL: Channel '${channame}
        exit 2
```

```bash
define service {
        use                     bk-tv-rec1_service
        host_name               BK-TV-REC1
        check_command           check_ncpa!-t 'xxxxxxxx' -P yyyyy -M 'plugins/check_recording.sh' -q args='itv3'
        service_description     itv3 recording
}
```

- Nagios is an event monitoring system that issues alerts when problems are detected
- We have two forms of alert for STORA
- Recording checks monitor current channel stream.mpeg.ts files by creating a checksum waiting four seconds and making another and comparing to ensure they are different
- Monitoring of Cesbo Astra software to ensure each channels has status ‘onair = true’
- If either fail Nagios software shows a critical status for the channel and sends an email confirming which problem has raised the alarm

bbcfourhd recording	CRITICAL	2022-10-17 10:03:38	0d 4h 28m 10s	5/5	CRITICAL: Channel 'bbcfourhd' not writing data to drive
bbcfourhd signal	CRITICAL	2022-10-17 10:02:51	0d 4h 28m 52s	5/5	CRITICAL: Channel with ID 'BBC4HD' is offline in Astra software
bbcnewshd recording	OK	2022-10-17 10:06:12	3d 23h 21m 36s	1/5	OK: Channel 'bbcnewshd' writing data to drive
bbcnewshd signal	OK	2022-10-17 10:03:33	2d 22h 4m 10s	1/5	OK: Channel with ID 'BBCNEWSHD' is online in Astra software
bbconehd recording	OK	2022-10-17 10:04:14	13d 21h 12m 34s	1/5	OK: Channel 'bbconehd' writing data to drive
bbconehd signal	OK	2022-10-17 10:03:53	16d 22h 33m 52s	1/5	OK: Channel with ID 'BBC1HD_6941' is online in Astra software
bbcthree recording	CRITICAL	2022-10-17 10:03:11	0d 3h 13m 37s	5/5	CRITICAL: Channel 'bbcthree' not writing data to drive
bbcthree signal	CRITICAL	2022-10-17 10:06:20	0d 4h 30m 23s	5/5	CRITICAL: Channel with ID 'BBCTHREEHD' is offline in Astra software
bbctwohd recording	OK	2022-10-17 10:05:27	3d 21h 42m 21s	1/5	OK: Channel 'bbctwohd' writing data to drive
bbctwohd signal	OK	2022-10-17 10:03:57	16d 22h 33m 45s	1/5	OK: Channel with ID 'BBC2HD' is online in Astra software
cbbchd recording	OK	2022-10-17 10:03:10	0d 4h 33m 38s	1/5	OK: Channel 'cbbchd' writing data to drive
cbbchd signal	OK	2022-10-17 10:03:37	0d 4h 33m 6s	1/5	OK: Channel with ID 'CBBCHD' is online in Astra software
cbeebieshd recording	OK	2022-10-17 10:04:12	0d 4h 32m 35s	1/5	OK: Channel 'cbeebieshd' writing data to drive
cbeebieshd signal	OK	2022-10-17 10:01:43	0d 4h 30m 1s	1/5	OK: Channel with ID 'CBEEBIESHD' is online in Astra software
channel4 recording	OK	2022-10-17 10:03:38	4d 1h 18m 11s	1/5	OK: Channel 'channel4' writing data to drive
channel4 signal	OK	2022-10-17 10:02:31	2d 21h 40m 13s	1/5	OK: Channel with ID 'CHANNEL4HD' is online in Astra software
citv recording	OK	2022-10-17 10:03:49	0d 4h 2m 58s	1/5	OK: Channel 'citv' writing data to drive
citv signal	OK	2022-10-17 10:06:00	0d 4h 5m 43s	1/5	OK: Channel with ID 'CITVSD' is online in Astra software
film4 recording	OK	2022-10-17 10:01:54	0d 6h 35m 54s	1/5	OK: Channel 'film4' writing data to drive
film4 signal	OK	2022-10-17 10:04:40	3d 23h 38m 3s	1/5	OK: Channel with ID 'FILM4' is online in Astra software
five recording	OK	2022-10-17 10:03:35	13d 21h 3m 14s	1/5	OK: Channel 'five' writing data to drive
five signal	OK	2022-10-17 10:03:27	6d 18h 59m 16s	1/5	OK: Channel with ID 'CHANNEL5HD' is online in Astra software
itv1 recording	OK	2022-10-17 10:02:48	4d 16h 4m 59s	1/5	OK: Channel 'itv1' writing data to drive
itv1 signal	OK	2022-10-17 10:04:11	13d 0h 33m 35s	1/5	OK: Channel with ID 'ITV1HD' is online in Astra software
itv2 recording	OK	2022-10-17 10:06:17	3d 21h 56m 30s	1/5	OK: Channel 'itv2' writing data to drive
itv2 signal	OK	2022-10-17 10:06:22	20d 17h 46m 25s	1/5	OK: Channel with ID 'ITV2HD' is online in Astra software
itv3 recording					to drive
itv3 signal					is online in Astra software
itv4 recording					to drive
itv4 signal					is online in Astra software
more4 recording					ata to drive
more4 signal					s online in Astra software

** PROBLEM Service Alert: BK-TV-REC (BK-TV-REC) / film4 recording is CRITICAL **

nagios@dpi.bfi.org.uk
Yesterday, 08:26

Reply

Attention. This email originated outside the BFI. Please be extra vigilant when opening attachments or clicking links.
***** Nagios *****

Notification Type: PROBLEM

Service: film4 recording
Host: BK-TV-REC
State: CRITICAL

Date/Time: Thu Oct 13 08:26:06 BST 2022

Additional Info:

CRITICAL: Channel film4 not writing data to drive

- Nagios is an event monitoring system that issues alerts when problems are detected
- We have two forms of alert for STORA
- Recording checks monitor current channel stream.mpeg.ts files by creating a checksum waiting four seconds and making another and comparing to ensure they are different
- Monitoring of Cesbo Astra software to ensure each channels has status ‘onair = true’
- If either fail Nagios software shows a critical status for the channel and sends an email confirming which problem has raised the alarm

Search or jump to... / Pull requests Issues Marketplace Explore

bfidatadigipres / STORA Public

Edit Pins | Watch 1 | Fork 0 | Star 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main | 1 branch | 0 tags | Go to file | Add file | Code

digitensions Update README.md 79635b0 yesterday 15 commits

code	Add file duration to info.csv	yesterday
LICENSE	Initial commit	11 days ago
README.md	Update README.md	yesterday

README.md

STORA: System for Television Off-air Recording and Archiving

The scripts in this repository form the off-air TV recording codebase responsible for preserving 17 UK Television channels 24 hours a day, 7 day a week. The BFI is the body designated by Ofcom (UK communications regulator) as the National Television Archive, under the provision in the Broadcasting Act, 1990. This designation allows us to record, preserve and make accessible TV off-air under section 75 (recordings for archival purposes) of the Copyright, Designs and Patents Act, 1988 and later the Copyright and Rights in Performance Regulations 2014 (under Research, Education, Libraries and Archives).

Overview

These scripts manage the recording of live television, accessing FreeSat streams using RTP streams for the

https://github.com/bfidatadigipres/STORA/

About

Off-air TV recording system. Open source Python3 and bash shell code

Readme
MIT license
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

System for Television Off-air Recording and Archiving

Thank you!

joanna.white@bfi.org.uk
www.bfi.org.uk/bfi-national-archive
github.com/bfidatadigipres/STORA