EBU – TECH 3318

File Transfer Guidelines

Source: N/FT-AVC

Geneva
June 2008

Page intentionally left blank. This document is paginated for recto-verso printing

Contents

Foreword

The document contains:

- Guidelines for broadcasters planning to implement file transfer in their organisations. This can mainly be found in chapters 6-11 and should be relevant to technical and non-technical readers. In particular chapters 9, 10 and 11 are already in a format broadcasters may wish to include in their tenders for file transfer systems.
- Advice on performance issues for and to broadcasters already using file transfer. This can be found mainly in chapters 4 and 5 and is intended mainly for a specialist technical audience.

The main contributors to the document were:

Markus Berg (IRT), Chairman of N/FT-AVC

Mathias Coinchon (EBU)

Matthias Hammer (IRT)

Mika Katajisto (YLE)

Marc Lambreghs (EBU)

Martin Rüge (ARD/STP)

Thank you to those EBU members who responded to the file transfer questionnaire issued by the N/FT-AVC group and who therefore provided much of the information for the requirements sections of this document.

File Transfer Guidelines

EBU Committee	*First Issued*	*Revised*	*Re-issued*
NMC	2008		

Keywords: File Transfer, Broadcasting

1. Introduction

With the growing deployment of high-speed network infrastructures both in the wide and local area and the intense increase use of IT-based systems in TV and radio production, file transfer is becoming a more and more important tool for content exchange both inside a broadcasters premise and between broadcasters. Being able to use in principle non-real-time networks for content exchange faster than real-time can offer cost savings compared to "traditional" real-time circuits.

But to really achieve an improvement in work flow and costs, the complete chain of content exchange via file transfer has to be defined very careful and must take into consideration both the risks and the advantages, the new technologies can offer.

2. Scope

This document is intended to produce a set of guidelines for file transfer in order to help EBU members to implement file based content exchange in their own premises and also between broadcasters, for example via the EUROVISION networks (satellite or fibre based). The guidelines within this document are mainly network-oriented. The EBU N/SECURITY group has considered security aspects. The requirements have been collected amongst the group members.

3. Fields of Application

The fields of application can influence the requirements for the file exchange system. There are especially differences in the size of the transmitted file and the timing requirements (for news, for example, the file must be transmitted as fast as possible). A short remark: File transfer is already established in audio production and contribution, we shall just keep in mind that the size of video files is much bigger than audio files.

3.1 News

- Clip length: < 5 minutes
- File exchange as quick as possible
- The goal is to improve the availability of the material even a short time before the start of the programme

3.2 Advertisement/Trailer

- Clip length: < 5 minutes
- more or less time uncritical

3.3 Programme exchange

- Clip length: Varies, depending on type of programme
- < 15 minutes for magazines etc
- > 30 minutes for complete broadcasts
- more or less time uncritical

4. Network issues

4.1 Layer-2 technologies

There is already a whole lot of different layer-2 network technologies deployed and used by broadcasters, like ATM, DTM, MPLS, DWDM, SDH, Gigabit Ethernet and others. All are capable of carrying IP traffic and therefore are principally file transfer ready. A detailed description of these layer-2 technologies in wide area networks (WAN) can be found in document EBU Tech 3319 - The use of WANs to carry audiovisual content [Tech 3319].

4.1.1 Backbone structures

- Local and wide area structures including file transfer over satellite.

File transfer applications shall work on any given backbone infrastructure, since a number of different structures are already deployed and used [Tech 3319].

4.1.1.1 Local Area and Campus Networks

In most cases, Local Area Networks (LAN) in broadcaster's premises and campus networks are based on Ethernet versions (Fast, Gigabit), sometimes, ATM (including IP over ATM) is deployed. The file transfer application shall work independently from the chosen layer-2 infrastructure.

In principle, all those technologies allow high bit rate IP connections and therefore file transfer inside the broadcaster's premises and campus networks

4.1.1.2 Metropolitan Area Networks

Today, a large number of carriers offer connectivity in cities and metropolitan areas. Those structures are mainly based on fibre infrastructures. DWDM, SDH, MPLS and ATM are the main technologies deployed in those networks. Gigabit Ethernet in MANs and DTM are also emerging. For those technologies, the same as in 4.1.1.1 is valid, they allow high-speed IP connections and therefore are ready for file transfer.

4.1.1.3 Wide Area Networks

Wide area networks WAN) are detailed described in the WAN Roadmap document of the NMC group N/WAN. [Tech 3319].

4.1.2 LAN-WAN transition

A general issue in file transfer over wide area networks is the transition point between the local area and the wide area infrastructures, where very often, layer-2 technologies change (such as from Fast Ethernet <-> ATM/SDH etc.) and firewalls and address translation appear. Any given File transfer application must cope with these issues. In some cases, security policies have to be taken into account.

4.1.3 The use of Jumbo Frames in Gigabit Ethernet structures

Standard Ethernet uses frames of 1500 byte length. Jumbo frames are a recent extension to permit frames of a size up to 9000 bytes.

The interest is to reduce fragmentation and overhead by increasing the frame length. Larger frames mean also less CPU interruption and processing on the local side. TCP transport protocol with its acknowledgement and congestion control mechanisms has got a throughput that is directly proportional to the maximum segment size (MSS). Other factors like the round trip time (RTT) and packet loss are limiting factors that are sometimes inevitable in WAN networks. So there's a clear interest in increasing the maximum frame length in underlying protocols layers to increase the TCP performance with such networks on high bit rates.

Unfortunately this is only possible if the network supports jumbo frames end to end which is not yet very common with current network operators. A problem with Jumbo Frames occurs when they are used in networks where bit errors or packet losses appear. The bigger the frames are, the bigger is the probability to be hit with an error and with the bigger frames, more data will be lost.

However this is an option to consider when building new dedicated Ethernet gigabit WAN networks.

4.2 Layer 3 - IP

Since file transfer over TCP is quite independent of the IP layer itself, the focus in this document lies on the higher layer protocols (see section 4.3). More IP details can be found in EBU Tech 3319 [Tech 3319]. There is very often mention about possible performance improvements when changing from the current IPv4 to IPv6, but for file transfer over WAN, no improvement in performance can be expected using IPv6 in comparison with IPv4, because the TCP protocol used for file transfer will not change. So improvements in TCP are necessary.

4.3 Layer 4 - TCP

The most commonly used protocols in the Internet are TCP and UDP - on the transport layer of the OSI reference model.

As file transfers applications require a secured connection, TCP will be dealt with in greater detail.

In high-bandwidth networks in the WAN range, frequent details in connection with TCP have to be considered, which may result in form of low data rates. Already with a latency of more than 5 ms between 2 computer systems this behaviour can be observed, and as we will see, the data transfer rate continues to sink, the longer the latency between two computer systems gets.

This problem generally affects above all large data sets, e. g. files of "professional video formats", which need a real time transfer rate in the range from 25 to 50 Mbit/s.

As the latency cannot be reduced physically, an increase of the data transfer rate is only possible by optimizing in the area of the TCP. This is the only way to achieve an acceptable data transfer rate for professional needs.

Before dealing with this special TCP behaviour, the following relevant TCP function modes must be pointed out:

4.3.1 TCP – Basic Principles

4.3.1.1 3-Way-Handshake

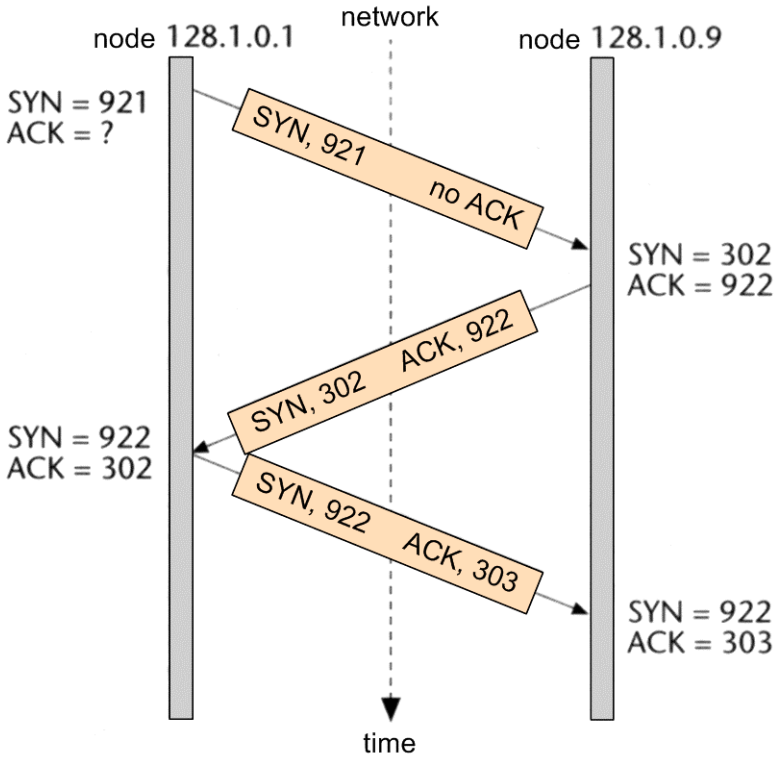

Figure 1: 3-Way-Handshake for connection establishment [Was97]

Before any data transfer with TCP/IP a mechanism for the connection between two communication partners is necessary. As shown in Figure 1, TCP ensures this with the so-called 3-Way-Handshake:

Node A (128.1.0.1) would like to establish a connection and therefore sends a message to Node B (128.1.0.9). This contains, amongst other data, a sequence number (921) and a set SYN-FLAG. B waits for a connection inquiry and acknowledges an inquiry by a set ACK- and SYN-FLAG. The receipt of this message is acknowledged by A, and the connection is considered as established.

The connection clearing takes place in similar way: A sets the FIN-FLAG, B acknowledges. Subsequently, B sets the FIN-FLAG and A acknowledges.

4.3.1.2 Flow Control and TCP Windows

In order to guarantee a correct data communication, the receiver acknowledges each IP datagram (see Figure 2). If an ACK does not arrive within a "Timeout" time, then the data is sent again. The receipt of a package is however only acknowledged, if all preceding packages have arrived. (The individual packages are identified by sequence numbers.)

However, waiting for the receipt of a sent package before the next package can be dispatched would not be an efficient use of the network. For this reason several packages are dispatched at the same time (see Figure 2).

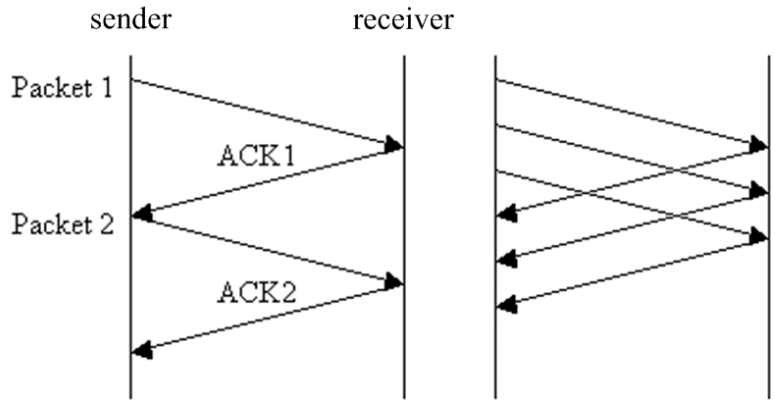

Figure 2: overlaid transfer [Ki02-FF]

This is realized with a window mechanism (see Figure 3).

The window is a pointer to a send or receive buffer. It is differentiated between send and receive window: The send window is maximally as large as the receive window of the receiving station - thus taking into account the processing capacity of the receiver. The send window contains the data sent but not acknowledged, as well as the data, that still may be sent. If the transferred data (left side of window) is acknowledged, the window is shifted by the length of the acknowledged data block to the right.

Sent and acknowledged data is on the left, outside of the window. The data that has still to be sent is on the right side (see Figure 3).

The receive window behaves similarly. It contains the received data not yet acknowledged and a free capacity. After the acknowledgement of the data the window is shifted to the left and further capacity becomes free.

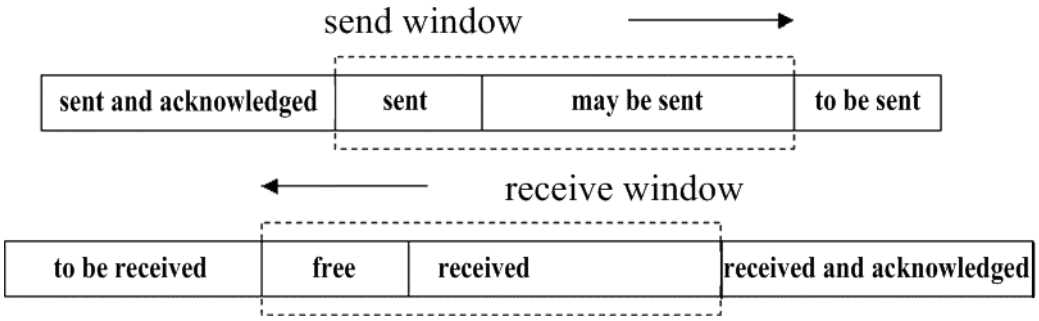

Figure 3: window structure [Ki02-FF]

The size of the receive window is communicated to the sender by the field "window size" in the header of the TCP segment; this is indicated in byte! [Ki02-FF]

The field is 16 bits long, which means that the receive window size can be maximally 65.536 byte (= 64 kbyte) (FFFF (hexadecimal) = 65.535(decimal); at the value 65.535 the first byte (= 0) is taken in account, so that size results in 65.536).

In high-bandwidth networks this delimitation can, as initially described, become a problem. Section 4.3.2. deals with this in greater detail.

4.3.1.3 Congestion Control (Overload Monitoring) and TCP Windows

If the load of a network exceeds its processing capacity, an overloading arises. The switching layer tries to repel overloading, however TCP takes over the majority of the work, because only a decrease in data transfer rate offers a genuine solution. TCP tries to reach this goal by dynamic adjustment of the window size.

But the first step is to detect an overload. Signs for an overload are unacknowledged and therefore lost IP datagrams. Package loss can also result from line noise as well as from electromagnetic influence. Nowadays however, modern WANs consist almost completely of optical waveguide cables, so that "Timeouts" and/or losses are usually due to overload in the network.

All TCP algorithms are based on this acceptance!

As stated in the previous section, the receiver of data communicates a window in accordance with his buffer size. If the sender adheres to this window size, no problems will develop due to a buffer overflow at the receiving end. The possibility of an internal overload in the net continues to exist however

Therefore two possible problems must be treated separately: the network and the receiver capacity. To do this, each sender maintains beside the receive window a second, the so-called "congestion window". Each window reflects the number of bytes, which the sender may transfer. The number of bytes, which may be sent, is the minimum of both windows (see Figure 4). Thus the effective window is the minimum of what the sender considers appropriate, and what the receiver considers appropriate.

E. g., if the receiver sees this value as 8 kbyte, while the sender knows that more than 4 kbyte would overload the network, it transfers only 4 kbyte.

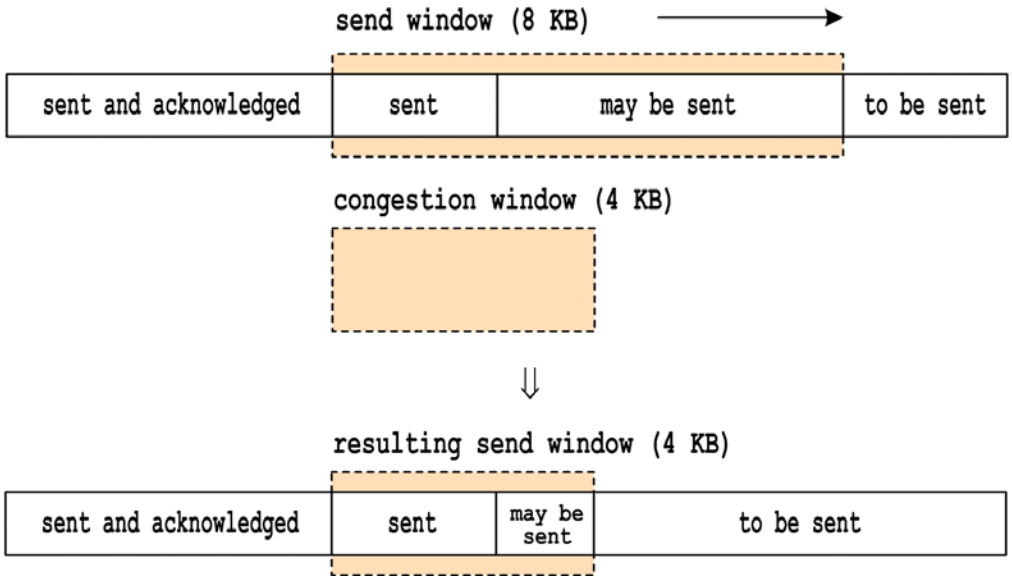

Figure 4: effective send window

At start-up of a connection the sender initializes the congestion window to the size of a maximum TCP segment. (“To increase in efficiency”, Ms. Windows 2000/XP initializes with two TCP segments. [Don00])

The further exponential development of the congestion window, follows the "slow start" algorithm (- see relevant literature).

If the receive window is permanently not reached, the constant rising and decreasing of the data

transfer rate, which is characteristic for the "slow start" algorithm, continues and leads to an impairment of the maximal possible data transfer rate. In networks with small capacities it is appropriate to lower the receive window so far that the slow start algorithm does not respond too frequently.

4.3.2 Long Fat Networks (LFN) – problems in the WAN

The following considerations are to make clear, where the power reserves of TCP in high-speed networks lie and where problems could arise:

Figure 5 shows a simple data transfer of a sender to a receiver. Neither segments go lost nor they are retransferred.

The sender begins to transfer segments as long as the window of the receiver permits it. In order to relieve the network, not each segment is acknowledged by the receiver. Instead, the receiver waits with the acknowledgement until it has to send some data to the sender or until it is reached by further segments of the sender.

In Figure 6 the receive window (= send window for the receiver) is alike. The only change is the available data transfer rate. It is higher around a multiple. [Hab96].

It becomes clear that TCP is not able to use the full data rate, since the sender already filled the receive window with the permitted amount of data and afterwards is not allowed to dispatch more data. It must now wait for acknowledgement of the receiver.

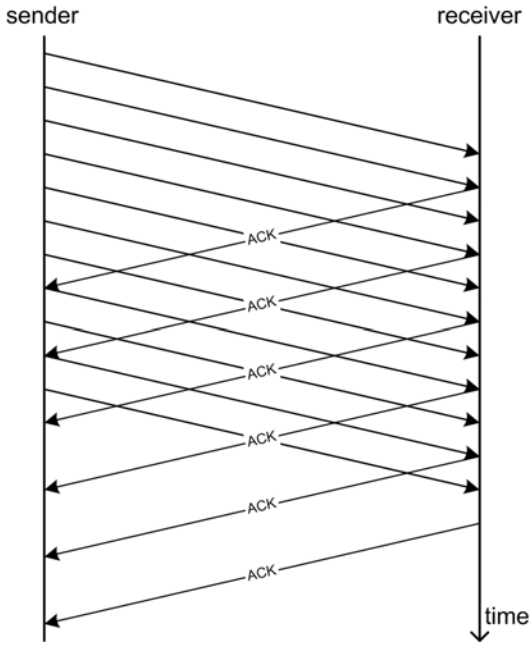

Figure 5: data transfer in networks with data rates up to 10 Mbit/s [Hab96]

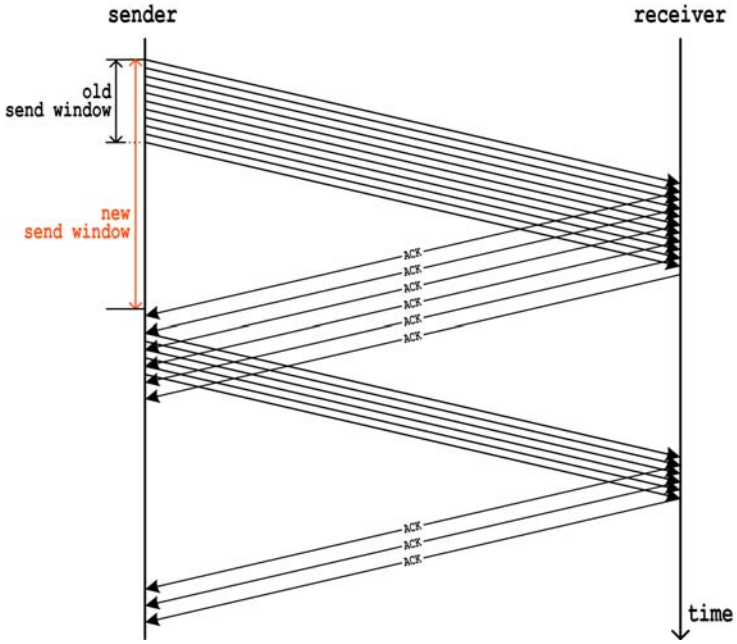

Figure 6: data transfer in LFNs (long fat networks)

The problem is the high "Round Trip Time" (RTT). It indicates the time, which passes from dispatching a package up to its acknowledgment by a receiver [Ste94]. If it would be smaller, the net was more effectively used and the sender would not spend the largest time in waiting status.

WANs with the described combination of high-bandwidth and long RTTs are also called, due to

these characteristics, "long fat networks" (LFNs).

4.3.3 Solutions – Tuning TCP

In order to abolish the problem described above, there are in principle 3 alternative approaches:

- increasing the TCP window size (to side of the operating system);
- data-transport via several parallel TCP sessions (by adjustment on side of application);
- usage of more performing protocol-stack-techniques like “Overlapped I/O

4.3.3.1 The “Bandwidth Delay Product” – larger TCP Windows

If the decision is made to enlarge the TCP window, then it is preferably done at such a rate, that almost the whole RTT is “bridged” by unacknowledged IP datagrams (see Figure 6).

The exact size of the new window can be calculated by the so-called "Bandwidth Delay Product", the product of data rate (= bandwidth) and delay time (= RTT) [Ste94-BDP]:

TCP window size [bit] ≤ total available bandwidth [bit/s] x RTT [s]

Normally the TCP window size is to be set on an integral multiple of the "Maximum Segment Size (MSS)" [Was97-MSS]. In this way network resources can be used most efficiently.

Maximum Segment Size (MSS) and Maximum Transfer Unit (MTU)

The maximum segment size is the largest quantity of data that fits and can be transferred in one single TCP segment [Com03]. All data packages that go beyond the MSS need to be fragmented. This causes an additional header per fragment, which means that the relative utilizable part of data per fragment decreases. Therefore no lower values should be used, than the greatest possible MSS.

The MSS doesn’t depend only on the two communicating computer systems, but also on the network elements lying between them. It is negotiated automatically at time of the connection establishment between the systems.

For determination of the MSS a further size is decisive: the "Maximum Transfer Unit (MTU)". This is the maximal transferable unit (useable data + headers!) or in other words the largest amount of data, that can be transferred over a certain physical network remaining in one block. It is determined by the network hardware. [Com03]

Normally Ethernet systems use a standard MTU size of 1500 byte. Thus, for the MSS a value of 1460 byte results (= 1500 byte - 20 byte (IP header) - 20 byte (TCP header)) [Ste94-MSS].

Figure 7 clarifies these interrelationships:

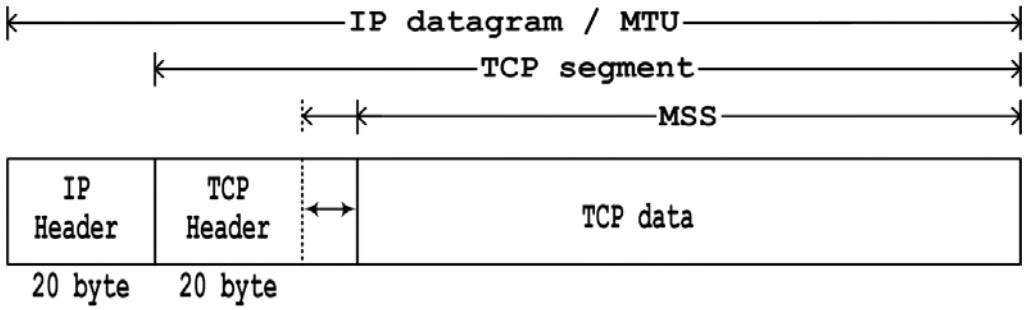

Figure 7: interrelationship of MSS and MTU [Ste94-MM], [Don00-MM]

In a network with a data rate of 100 Mbit/s (= nearly Ethernet) and RTT of 10 ms as aforementioned for example results a window of rounded down 129.940 byte (MSS = 1460 byte). As shown in 4.3.1.2 however a window can have a size of maximal 65.536 byte. The "Window Scale option" is the answer to this problem.

4.3.3.2 "Window Scale Option" (RFC 1323)

In networks with high-bandwidth and long RTTs, TCP windows beyond the maximum TCP window size of 64 kbyte are reached very quickly. In order to remain compatible to older TCP implementations, instead of adjusting the TCP header a scaling factor was inserted into the existing header field "options". This factor tells the number of bits that are to be added to the existing window. The window is then expandable up to 32 bits according to RFC 1323.

Example: A send window is 32 kbyte large and is to be extended. The scaling factor was set to the value "5".

32 kbyte = 0x7FFF =	111 1111 1111 1111	
left-shift 5 bits:	1111 1111 1111 1110 0000	= 0xFFFE0

Due to the scaling factor "5", five bits were added and thus the window was extended to 1.048.544 byte (= 0xFFFE0; approx. 1024 kbyte). [Don00-WS]

Older computer systems, that do not support the RFC 1323, reject the entry in the "options" field and instead use a window, which remains limited to 64 kbyte.

4.3.3.3 "Selective Acknowledgement" (RFC 2018)

Selective acknowledgement of IP packages is particularly of importance for TCP connections that use large windows.

Usually the acknowledgement mechanism of TCP sends a reception acknowledgement at the end of a transfer sequence, which consists of several TCP packages (- so-called "bursts"). This happens, when:

- the size of the receive buffer announced as receive window was reached,
- the sender of the data expressly requests an acknowledgement for the data already sent or however, if
- after a time limit no further data reach the receiver.

This happens only if the data packets are delivered sequentially! [Wal01]

Once a package is lost during the transfer, the reception of sequential data cannot be acknowledged, even if the remaining packages were received correctly.

Thus, the sender transfers again all unacknowledged TCP segments. For connections with large TCP windows that means a strong break-down of the data rate, since in the worst case a window completely filled with data is rejected and must be transferred again.

"Selective Acknowledgement" permits the receiver - beyond the conditions specified above - to acknowledge individual packets also in the midst of a transfer. In so doing the receiver can "tell" the sender accurately, which data were received. The sender then retransfers solely the missing data.

In MS Windows 2000/XP - systems "Selective Acknowledgement" is activated by default.

4.3.3.4 "Overlapped I/O"

Contrary to the normally used synchronous processing of receive- and send-data "overlapped I/O" transfers data asynchronous in the case of TCP/IP. So existing time-gaps in (network) communications are eliminated/minimised and as a result the transfer-performance increases.

"Overlapped I/O" is implemented in MS Windows since "WinSock 2" and is used e.g. by measurement-tools like "NetIQ/Ixia - Chariot".

4.3.4 Discussion of the different solutions

The use of several TCP sessions is a widespread and therefore a well-understood method. However, a certain implementation expenditure is necessary on sides of the application, that accomplishes the file transfer. On the other hand one is nearly independent of the RTTs in the network. If the transfer rate does not suffice, the amount of the TCP sessions is increased. An additional advantage is, that in case of an abort of a single connection the whole transfer is not endangered.

The added protocol-overhead may be seen as disadvantage – in comparison with the performance-improvement it should be acceptable.

In contrast to this, in case of enlarging the TCP window, the operating system as well as the delay of the used network exert influence on the data rate.

Optimisations can be only done for one fix RTT. If a certain computer is to initiate contact with different partners with different RTTs, that may lead to inneficient bandwidth occupancy. Furthermore here the abort of the connection means the abort of the complete transfer. The use of large TCP windows is not supported by older TCP implementations.

"Overlapped I/O" is a possible improvement for applications using e.g. LFNs with high bandwith-delay-products. The (CPU) system-performance and the resulting effort of implementing asynchronous sockets especially in existing software-products are the most limiting factors

4.3.5 New TCP congestion control algorithms

4.3.5.1 Introduction

The actual congestion control mechanism TCP Reno has limited performance in presence of high-speed networks with high latency.

Increasing TCP window size does not solve everything. For example, packet losses will lead to dramatic throughput falls with slow recovery because of multiplicative decrease/additive increase sending rate control.

TCP congestion control is also inappropriate when packet loss occurs because of transmission lines errors instead of network congestion: TCP will decrease the sending rate even when it should continue at the same rate (example with wireless LANs).

New implementations have appeared with new congestion control algorithms designed to address these problems.

The issues are to achieve good performances i.e. to use completely the available bandwidth, reduce the effects of packet losses while keeping TCP fairness (sharing the bandwidth with other connections, "being fair with others").

Many of these new congestion control mechanisms are still experimental and exist only on Linux platforms but some might be candidates for future implementations on other platforms. All these mechanisms are backward compatible (can communicate with any TCP stack).

4.3.5.2 TCP Vegas (experimental)

TCP Vegas congestion control algorithm is using delay instead of packet loss to evaluate congestion and determine the send rate. So congestion windows are based on bandwidth measurements. It has less aggressive sending rate adaptation and tend change it more smoothly.

It helps reducing queuing and latency. It is appropriate for "real-time transfers" or TCP streaming.

TCP Vegas is currently implemented in Linux from 2.4 kernels, NetBSD, SunOS.

Reference: [ariz-web].

4.3.5.3 TCP Westwood (experimental)

TCP Westwood is suited for networks that suffer packet losses like wireless networks.

The congestion control is the same as standard TCP (TCP Reno) on increase but when congestion occurs it switches to a congestion control mechanisms based on bandwidth measurement.

TCP Westwood is currently implemented in Linux from kernel 2.4.

Reference: [ucla-web].

4.3.5.4 BIC TCP (experimental)

BIC (Binary Increase Congestion control) TCP is designed for high speed networks. It uses binary search TCP send window increase when the window is small and uses additive increase (like TCP Reno) when the TCP window is large.

Reference: [ncsu-web].

4.3.5.5 FAST TCP

FAST TCP has been developed by Caltech (Commercial development)

- FAST TCP is equation-based, hence avoiding packet level oscillation,
- FAST TCP has stable flow dynamics,
- FAST TCP uses queuing delay, rather than loss probability, as the main measure of congestion

The problem is that at the moment FAST TCP is not an open standard. There are plans to make an RFC standard.

TCP FAST implementation is very promising for achieving very high bit rates on Long Fat Networks.

The current implementation runs on Linux.

Reference: [caltech-web].

4.3.5.6 HSTCP (RFC 3649)

HSTCP (High Speed TCP) uses additive increase for the congestion window when it is small and uses a more aggressive increase when the congestion window is large (high bit rate). There is an experimental IETF standardization of HSTCP with RFC 3649. HSTCP is implemented in Steelhead appliances (according to their product literature).

Reference: [icir-web].

4.3.5.7 Other TCP implementations

Many other TCP congestion control mechanisms exist, such as TCP Hybla, Scalable TCP (S-TCP), High Speed TCP with Low Priority (HSTCP-LP), Hamilton TCP (H-TCP).

4.3.5.8 ECN (Explicit Congestion Notification)

This is not a congestion control algorithm but a mechanism for routers to signal congestion by putting information inside the IP packets in a special ECN field. This is a novel approach that is not currently deployed and controversial. It may pose certain problems with firewalls that don't understand the new ECN field and then may block packets.

However this could help senders to react more efficiently in case of congestion in IP networks. ECN is standardized by the IETF in RFC 3168.

4.3.5.9 Conclusion new TCP congestion algorithms

Many new TCP implementations exist for different needs: high bit rate, high latency networks, lossy networks and so on. Announced performances are often based on simulations/measurements. TCP fairness is an issue: it is easy to use all bandwidth with one connection but if congestion control is not "fair" then it will lead to problems on concurrent connections. This might not be a major problem if sharing of bandwidth is not a big issue.

The main congestion control algorithms are as follows:

For long fat networks with high delay:

- BIC TCP, FAST TCP (when available).

For networks that suffer from packet loss:

- TCP Vegas, TCP Westwood.

At this stage new implementations exist on Linux only. Windows platforms only implement TCP Reno for now but may implement others in the future. Some "accelerator" appliances (example: steelhead products using HSTCP).

4.4 Application layer

4.4.1 Introduction

File transfer applications are almost always using TCP for transport. Their performance for file transfer depends mainly on the TCP layer implementation.

However it is also important to look at upper layer protocols for issues that are not handled by the transport protocol. Examples:

- file integrity check
- object descriptions
- access security, transfer security.

4.4.2 FTP

File Transfer Protocol based on TCP.

This protocol is popular for file transfers. However it has some drawbacks:

- Insecure: Password, commands and transfer are not encrypted on the network and can be intercepted.
- Active mode (used by default) is often problematic with firewalls because of dynamic port numbers. Passive mode solves this problem but must be activated.
- Lack of object description, no integrity check. Receiver gets no information about the received object like the file size for example. This may lead to incomplete files if the connection is accidentally interrupted.
- ASCII transfers of binary objects lead to errors. FTP clients have normally the ability to automatically detect the file format.

4.4.3 HTTP/HTTPS

HyperText Transfer Protocol (HTTPS: Secured) based on TCP.

GET and POST methods can be used to transfer files.

Some advantages of HTTP are:

- Easy to pass firewalls (HTTP port 80 and HTTPS 443 are very often opened)
- Description of objects is transferred with HTTP header. Integrity check can be performed (Content-MD5).
- Universal. HTTP clients (web browsers) are very common and installed by default on every platform.
- Security: session transfer can be encrypted using HTTPS.

Some disadvantages:

- Some web browsers have a limited size for file upload (2GB for Microsoft Internet Explorer).
- Browser doesn't give a progress bar of the file transfer. This must be implemented on the server side.
- Some firewalls/antivirus network appliances block HTTPS traffic (secured) because it cannot be checked (encrypted)

PUT method is also used for transfers with webDAV (Web-based Distributed Authoring and Versioning) for example.

4.4.4 Other application layer protocols

SCP: Secure Copy Protocol using SSH protocol (Secured Shell protocol) and based on TCP. SSH allows secure file transfer between two hosts and is similar to the unsecured RCP (remote copy). It is used mainly between UNIX machines but clients/servers exist on other platforms (winSCP for Windows).

SFTP: SSH File Transfer Protocol. SFTP is using SSH for secure file transfer. Compared to SCP it offers operations on remote files, directory listing, resuming, (similar to FTP). The most widely used version of SFTP is version 3. Client/Server implementations exist on almost all platforms. SFTP is unfortunately not widely adopted but should be considered for more secure file transfers.

SMTP: Simple Mail Transfer Protocol is based on TCP. SMTP is the protocol used for E-mail delivery. It is basically designed for text transfer but allows binary transfers using 8BITMIME extension. SMTP has never been designed for large file transfers however people are often using it for file transfers as it is of simple use.

The following protocols are designed for disk shares and transfers between computers and are normally used on a LAN environment. However they are sometimes used on WAN. They should be avoided on the Internet because of security reasons:

SMB: Server Message Block. Proprietary protocol from Microsoft used for file shares and transfers between Windows machines. SMB can run on multiple protocol but is usually used with TCP.

NFS: Network File System protocol based on UDP and TCP (from version 3). NFS is used for disk shares and transfers between Unix machines (Sun, Mac) but Windows implementations also exist.

AFP: Apple Filing Protocol based on TCP. AFP is used for transfers between Apple machines.

4.4.5 PeerToPeer

Peer to Peer (P2P) protocols allow the transfer of files by sharing transfer using multiple hosts that have got the complete or partial file. Many protocols exist and commercial solutions are also available.

Peer To Peer is interesting for distribution of large files but less for unique file transfers. The interests for unique file transfer may be to balance the bandwidth and space between multiple hosts. This technology may be considered as a future solution but it still needs further evaluation.

4.5 WAN accelerators

There are a number of further solutions coping with different network issues. One method used are so called WAN accelerators. They mostly terminate existing TCP connections at the edge of Wide Area Networks and use special (often-proprietary) protocols to accelerate the traffic through the WAN. As an example, IRT conducted tests on an existing WAN accelerator product (further on referred as “WAN accelerator” to keep the results anonymous).

In November 2005, a test session was conducted at the IRT in order to verify, whether the WAN-Accelerator, at high–latency links as usual in large wide area networks, is able to improve the transfer rates of video file transfer systems.

The equipment works like proxies on both sides of the WAN link and offers different possibilities of optimization with a proprietary optimization system. For example with a so-called “Data Streamlining” technology redundant bytes of the transfer are eliminated. This technology reduces WAN bandwidth utilisation and is thought for links with low bandwidth. At available bandwidth of more than 50 Mbit/s there is no improvement.

That’s why only the “Transport Streamlining” technology with High-Speed TCP (an implementation of RFC 3649) has been tested. At latency ranges up to 36 ms throughputs of more than 500 Mbit/s have been measured as long as no hard disk has been involved. On real file transfers (disk to disk) the throughput was limited to 200 Mbit/s because of the disk’s access times.

Compared to RFC 1323 (optimized Standard-TCP with a window size four times the normal 65 kbyte), as used at the moment for example in the ARD video file transfer system, with 30 ms latency a large file has been transferred four times faster.

With the view to WAN-Simulation at this tests only roundtrip times were simulated. The behaviour of High-Speed-TCP in real networks with high-latency and packet loss is still to be tested.

5. Security

5.1 Performance issues

Security processes should not delay transfers remarkably.

In order to examine the influence of firewalls on file transfers, the Institut für Rundfunktechnik in Munich (IRT) carried out a project on security in contribution and distribution networks of broadcasters with a special focus on the performance of different types of firewalls. In order to gain experience on the performance of firewalls with respect to the broadcaster's requirements (no references available up to now), the IRT has tested different types of firewalls, ranging from a non-commercial public LINUX solution to a high end hardware firewall.

The test results showed in general that all firewalls reached very good results in packet filter mode (500 – 930 Mbit/s on a 1 Gbit/s link). Working in VPN mode, the results for software based firewalls did not reach 300 Mbit/s, the hardware based solution reached ~880 Mbit/s.

To conclude:

- All tested firewalls fulfilled the basic expectations. Nevertheless there are significant differences. A lot of potential money savings are possible depending on the needed performance.
- Packet filter mode: a file transfer based on TCP is affected by delays > 7 ms, not by the firewall at these delays. Parallelisation of connections can cope with the delay problem.
- VPN Gateway: encrypted transmission is the worst case scenario for a firewall, especially with high bit-rate file transfer. But the performance is for example sufficient for several SDSL connections (video-journalist). Encryption in a Corporate Network or DMZ not necessarily mandatory.
- The expensive firewalls show a higher performance in critical conditions. The "cheap" and free of charge firewalls also showed to be flexible and performing and could be used for some use cases

5.2 Secure transfer of the content

Transferring files has to be safe and secure. In the group N/SECURITY, an example for a secure file exchange gateway has been developed.

The example is not for streaming (real-time transfer), but for (video-) file transfer and other informational exchange. The far end may be connected to another EBU member or any other 3rd-party. In this case is should be assumed, that the network is an untrusted one and may even be The Internet. A subsidiary purpose is to avoid letting the other-party having access to our internal systems.

The process is based on a two-stage transfer or web-service where the file transfer follows on from information exchange about what is available.

In this case point-to-point rules between A and B's firewalls can be set up.

The following illustrates the transfer of information (see Figure 8):

1. A (right) is working in A's production system.
2. A can search for information on what material is available, even from other companies that A has an arrangement with (e.g. B).

3. A sees something that A needs from another company B (i.e. not on A's own systems).

4. The user in A gets the information/material just by clicking on it.

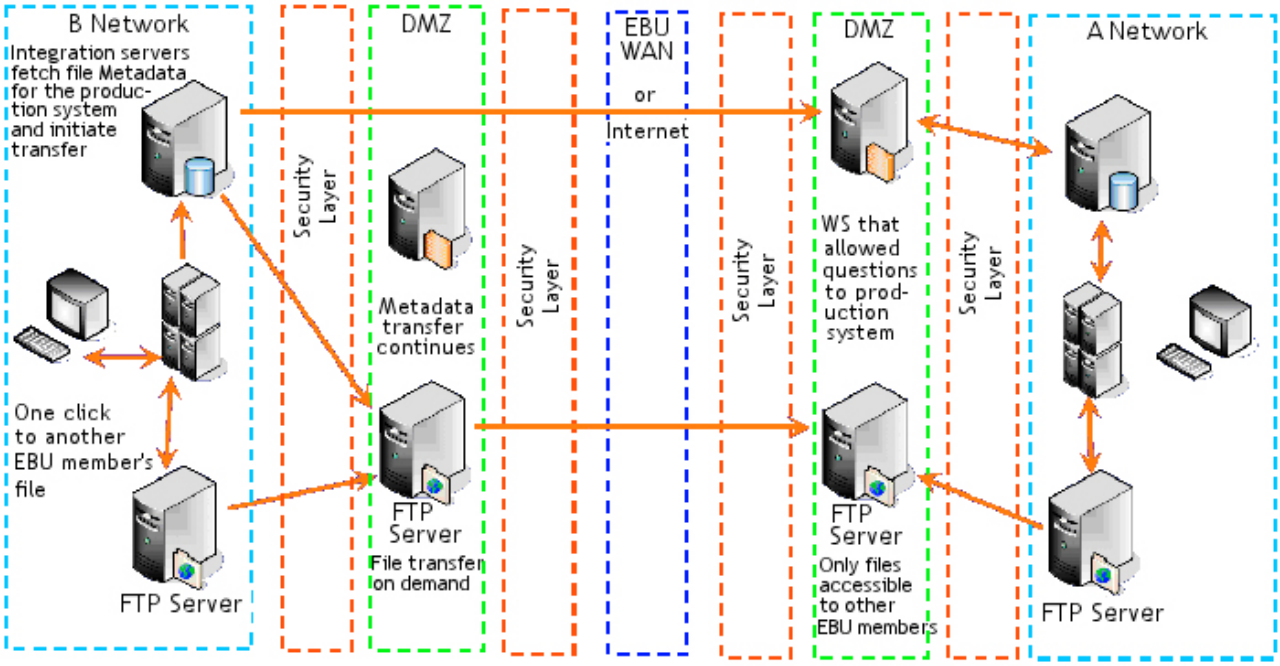

Figure 8: Secured file transfer gateway [N/SECURITY]

The main security advantages of the described constellation is the separation of the whole communication into three stages.

1. The internal working-area ('A' & 'B' Networks – the light blue line).

2. The DMZ (green line) to terminate the "passing" connections. Due to the workflow these DMZ-machines can do "real" work (like FTP) and are not only additional (Proxy-) hardware for security.

3. The untrusted networks between the partners ('EBU WAN or Internet' - dark blue line) can differ and normally (if at all) only small changes in the firewall-ruleset of the DMZ has to be made.

Between these three sections it is possible (necessary) to implement security layers (red line) like firewalls and intrusion prevention systems. The balancing of the firewall-load due to the segmentation can also be a positive effect. So the different security layers can independently be optimised for their part of the workflow (Especially the load of a bunch of video file transfers should not to be underestimated).

Overall this is a common configuration for standard communications, which has to be enhanced by the searching and service functionality for the video file transfer workflow.

Another issue to "keep an eye on " is content integrity. In a secured network, one may assume, the integrity of files is save, but nevertheless access to stored files is also a point, where security means must be taken.

6. File Formats

The EBU suggests the use of the MXF file format for video file exchange (see EBU statement D95-

2003). For audio files, the following formats are likely to be used: BWF [EBU Tech 3285], RF64 [EBU Tech 3306] or FLAC-Files.

7. File transfer applications

There are several File transfer applications available. For example ARD and Czech TV using systems based on proprietary software development. It is out of the scope of this document to give a detailed description of these systems.

8. IT equipment

When using (almost) standard IT equipment, things like processor capacities, storage, network interfaces etc. have to be considered.

Harddisks (HDD)

Depending on the use-case it may be possible to use standard internal hard disks (HDD) to store (or cache) video-material which is available or prepared for transfer. In this case some on-board (RAID) solutions may provide enough capacity and throughput-performance. But if more than 2 or 3 transfers are processed simultaneously the performance will soon suffer. In this case it is necessary to expand the systems with more powerful hardware – like a connection to a Fiber Channel (FC) Array.

FC is quite common in the broadcaster's storages – not only since HD is getting more and more popular. But not so in the environments of the most "Hardware of the shelf" – users. Using FC with "standard" hardware may result in very different performance-results. One reason is, that FC normally uses a 64 bit system architecture and up to date 4 Gbit/s-FC can not be used with 32 bit boards (mostly because of the lack of internal bandwidth)

The throughput of HDD-Systems varies depending on the used infrastructure. Linear access is the most high-performance way to read (write) data from the disks and this is fortunately the typical situation when working with the huge files in the broadcast environment (video and mostly also audio). High performance RAID controllers provide the opportunity to optimize the caching mechanisms for sequential or random usage. This can improve the whole throughput.

The transfer rates of single drives (EIDE as well as S-ATA) can be estimated between 35 and 55 Mbyte/s. These performance decreases rapidly as more parallel or random transfers are established (down to less then 20% of the ideal-performance).

To improve the performance clustered HDD (e.g. RAID-systems) are used which are connected to the workstation via techniques like SCSI or FC.

In the case of SCSI the bus is able to transfer up to 320 Mbyte/s (Which is more than the most PC can handle up to now), Nearly the same situation can be found if the Storage is attached via FC. The current FC-generation provides a capacity of 4 Gbit/s (512 Mbyte/s) and the older versions 1 and 2 Gbit/s. (128 and 256 Mbyte/s).

Using these connections combined with enough parallel hard disks, the storage shouldn't be a bottleneck - even for professionals.

Network-Interfaces

Today even the cheapest PC is equipped with at least one 1 Gbit/s-Ethernet Interface. This is normally more than the most WAN- and even LAN-connections can provide (so far). But even here the performance should be proofed. Depending of the used chipset and software-drivers a 1 Gbit/s-

Interface can be reduced to less than 300 Mbit/s (instead of up to 900 Mbit/s).

Additional the (TCP) hardware acceleration of the interfaces (which is commonly available) can reduce the needed CPU-time.

Redundancy and reliability

If a higher level of reliability is needed, standard "of the shelf" - hardware normally is not be the first choice. But sometimes it may be cheaper to buy two "normal" systems with one standby as building up a high redundant system. Additional in many cases a file transfer system can be absent for a short time period (few minutes up to some hours?).

9. General requirements

The following requirements chapters have been created in discussions and with contributions from the group members. Broadcasters planning to introduce file transfer should check if these requirements fit to their own workflows.

Connectivity to external organisations must be easily possible. Since file transfer becomes more and more important also looking at the material exchange with external organisations, open standardised interfaces shall be used in the transfer system.

All network interfaces must be IP-based. At layer-2, several mechanisms can be used, depending on the internal structures of the participating organisations.

10. Performance Requirements

It must be possible to have a transfer faster than real-time.

Message times (for transmitter and receiver) in order to communicate successful, incorrect or interrupted transmissions: maximum 10 seconds.

The transmission protocol used must be capable of using the available capacity of the transmission network. Parameters like round-trip time etc. shall not influence the transfer performance. All possible effects in a real wide area network (WAN) (like delay variations, long round-trip times, packet loss, etc.) must be compensated by relevant mechanisms in the transfer system (for example by issuing parallel TCP sessions). The transmission protocol must secure that the transmitted data is delivered at the receiver side error free.

11. Operational and Technical Requirements

Both point-to-point and point-to-multipoint transfers must be supported.

The file exchange must have two options: Immediate transfer as soon as the material is available and scheduled (delayed) transfer.

The transfer system must support a push transfer (for both point-to-point and point-to-multipoint).

It must be possible for the receiving end to set priorities to transfers arriving at the same time.

Metadata transfer, in advance of the Essence, must be supported in accordance with existing specifications.

The system at the receiving end must be capable of buffering the received material, therefore it must be possible to integrate enough buffer/storage capacity (to be specified by the receiving

organisation) in the system.

The sending and the receiving transfer system both must issue a message in order to confirm successful transfer.

Depending on the specification of the receiving organisation, ingest of the received material into the internal production process must be possible using separate applications.

It must be possible to flag transferred material (especially news material) for immediate usage.

The delivery of the essence to the file transfer system is also file-based. A change of the file format and the video compression is not allowed.

The files and the metadata, once submitted to the file transfer system, must be available for further transfers. If necessary, the possibility to modify the metadata for such purposes must be given.

Several transfer processes between different transfer systems must be possible at the same time.

Several transfers between the same transfer systems must also be possible at the same time.

In order to deliver the files to the transfer system, standard IT interfaces must be provided. These interfaces must allow the required transmission rates.

If the ingest is done via SDI, the delivery at the receiving site must be possible as MXF file and via SDI. This must also be possible when a MXF-file is delivered to the sending system.

Identifiers must secure the link between metadata and Essence, preferably UMID but the possibility to use file-ID and filename must also be possible.

Metadata access must be possible via an API.

Metadata must be displayable via a graphical user interface (GUI).

In order to have a automatic and target-oriented forwarding of the received material into IT-based production systems, consistent control protocols must be used.

The self-dependent control of transfers must be possible from several clients at the same time.

The control must be possible via API and the submitted metadata.

It must be possible to access the received content during transfer. The transfer of the file must already be possible during the internal ingest and file delivery process.

The transfer status must be displayed both at sending and receiving side.

It must be possible to start every transfer immediately.

A scheduler in order to plan timed transfers must be provided.

If the transmission is interrupted due to equipment or network failure, the transfer must automatically continue at the interrupted stage, when network and/or equipment are ready for operation again.

Status and error messages have to be in a format that is easy to understand for the user.

The sending organisation must have the possibility to stop a running transfer. Then, a relevant message must be sent to the receiver.

The software of the transfer system must be established as a service that is started automatically after a restart. The software must support clustering.

Browse-video is needed for all material and it should be created in a centralized transcoding system at the receiving side.

12. Conclusion

File transfer for both internal and external content exchange imposes new requirements to broadcaster's networks. This document highlighted issues broadcasters should consider when planning the implementation of file transfer applications in their premises or the content exchange with external partners. So by choosing the right network technology and protocols (including possible optimisations), security measures and equipment and by picking the right requirements for their own needs, and verifying the file transfer applications against them, a successful introduction of file transfer will be no miracle.

13. References

[Com03]:	D. E. Comer: TCP/IP – Konzepte, Protokolle und Architekturen; Bonn 2003; p. 650
[Don00]:	D. MacDonald: Microsoft Windows 2000 TCP/IP Implementation Details – White Paper; Redmond 2000; p. 43
[Don00-MM]:	see [Don00], p. 38
[Don00-WS]:	see [Don00], p. 31 et seqq.
[EBU-T3285]	BWF – a format for audio data files in Broadcasting, EBU, 2001
[EBU-T3306]	RF64: An Extended File Format for Audio, EBU, 2006
[Hab96]:	A. Habermann: TCP in high-speed networks; http://www.tkn.tu-berlin.de/curricula/ss96/bla/tcp_hs.html (as at: 4.7.1996)
[Ki02-FF]:	A. Kipp (Universität Karlsruhe – Fakultät für Informatik): Proseminar „Rund ums Internet" – TCP-UDP; http://goethe.ira.uka.de/seminare/internet/tcp+udp (as at: 11.12.2002); section 3.4.
[Ste94]:	W. R. Stevens: TCP/IP Illustrated, Volume 1; Reading 1994; p. 299
[Ste94-BDP]:	see [Ste94], p. 289
[Ste94-MM]:	see [Ste94], p. 225
[Ste94-MSS]:	see [Ste94], p. 236 et seq.
[Wal01]:	F. R. Walther: Registry Guide – Windows 2000 und NT4; München 2001; p. 394 et seq.
[Was97]:	K. Washburn: TCP/IP; Bonn 1997; p. 285 et seq.
[Was97-MSS]:	see [Was97], p. 300
[Tech 3319]	The use of WANs to carry audiovisual content
[ariz-web]	http://www.cs.arizona.edu/protocols/
[ucla-web]	http://www.cs.ucla.edu/NRL/hpi/tcpw/
[ncsu-web]	http://www.csc.ncsu.edu/faculty/rhee/export/bitcp/
[Caltech-web]	http://netlab.caltech.edu/FAST/
[icir-web]	http://www.icir.org/floyd/hstcp.html

Annex 1: Abbreviations

ACK	Acknowledge
AFP	Apple Filing Protocol
API	Application Programming Interface
ATM	Asynchronous Transfer Mode
BIC	Binary Increase Congestion Control
BWF	Broadcast Wave File Format
CPU	Central Processing Unit
DMZ	Demilitarized Zone
DTM	Dynamic Synchronous Transfer Mode
DWDM	Dense Wave Division Multiplex
EBU	European Broadcasting Union
ECN	Explicit congestion Notification
FC	Fibre Channel
FLAC	Free Lossless Audio Codec
FTP	File Transfer Protocol
GUI	Graphical User Interface
HDD	Hard Disc Device
HDTV	High Definition Television
HSTCP	High Speed TCP
http	Hyper Text Transfer Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol
KB	Kilo Byte
LAN	Local Area Network
LFN	Long Fat Networks
MAN	Metropolitan Area Network
MPEG	Motion Picture Expert Group
MPLS	Multi Protocol Label Switching
MSS	Maximum Segment Size
MTU	Maximum Transfer Unit
MXF	Media Exchange Format
NetBSD	Network Berkeley Software Distribution
NFS	Network File System
OS	Operating System
P2P	Peer-to-Peer
RFC	Request for Comment
RTT	Round Trip Time

SCP	Secure Copy Protocol
SCSI	Small Computer System Interface
SDH	Synchronous Digital Hierarchy
SDI	Serial Digital Interface
SFTP	SSH File Transfer Protocol
SMB	Server Message Block
SMTP	Simple Mail Transfer Protocol
SSH	Secured Shell
STM	Synchronous Transport Module
TCP	Transmission Control Protocol
TV	Television
UDP	User Datagram Protocol
VFT	Video File Transfer
WAN	Wide Area Network