NEXT GENERATION DATA DOWNLOAD

Malcolm Weir
Ampex Data Systems Corporation
Hayward, CA, 94545
mweir@ampex.com

ABSTRACT

Over 20 years ago, a sea-change in the world of test instrumentation occurred. This was the standardization of recorders under the RCC, and which produced two major benefits: first, the packet formats were standardized across multiple vendors, and second, the mechanism by which data could be extracted from the recorders was standardized so that the custom post-mission infrastructure was eliminated. As time has passed, though, the capabilities of the download mechanisms have been overtaken by requirements and technology. An interface that was once both fast and elegant has become too slow and cumbersome, and functionality has not been added to meet current needs. This paper explores the landscape for download interfaces, with references to technology like STANAG 4575 and RCC IRIG 106 Chapter 10, and considering data-at-rest encryption, cybersecurity and distributed systems.

INTRODUCTION

Every data collection system has a dark side: once you've collected the data, you need to get it to someone who can do something with it. Sure, there's a utopian ideal that, eventually, we'll have some hypothetical Artificial Intelligence systems that can process the data on board in real time and answer whatever questions you want to ask, but even when the AI offers a plausible answer, *someone* will want the source data cross-check or validate that answer.

Data delivery to the consumer is handled in one of three ways: streaming data from the acquisition system during collection, using some kind of download interface from the "recorder" box, or physically transporting the storage from the acquisition system to a download system, and then either using a download interface (as with the previous case) or processing the data directly from the storage media.

For the sake of this paper, streaming data as a download alternative can safely be ignored, as the data rates will always lag behind the desired level, and it won't always be a viable option anyway (e.g. transmissions from the collection vehicle is impractical or undesirable for whatever reason).

The other two download design options can, sometimes, be essentially the same: extracting the storage from one system (used for recording) and inserting it into a different one (used for downloading) is pretty much the default mechanism for handling data download, and in many cases the download system will be a variation of the collection system modified for e.g. AC wall power, data center connectors, different and more or fewer expansion types and capabilities, air cooling instead conduction cooling, etc. But this model only works if the two systems are fundamentally compatible (if not identical) physically, electrically and logically.

And that compatibility requirement represents the challenge for standardization: the standard has to define a level of compatibility that is both practical (in the sense of affordability and capability) and versatile enough to be attractive to disparate product design teams and end users.

IRIG 106 CHAPTER 10

By the early 21st century, technology had advanced enough to make the prospect of a standard recording download interface viable. Prior to that, data was typically collected using magnetic tape storage, which for reasons both good and bad rarely encouraged standardization, or if they did, the logical formats were usually proprietary. But with readily available "high speed" (loosely defined as greater than 30 megabytes per second) storage devices, and high-speed interface technology (exemplified by Intel's Peripheral Connection Interface, PCI, which could transfer over a gigabit of data per second: 132 megabytes / sec).

After several abortive attempts at drafting a standard under various names, the Range Commander's Council (RCC) agreed to sponsor a new chapter in the Inter-Range Instrumentation Group's "106" Telemetry Standard. The goal of this nascent chapter was to make it practical for a test range to be able to read and interpret data from test assets, and it was titled "Chapter 10: Digital On-Board Recorder Standard" and was first published in 2005; the name was subsequently changed in the next release (two years later) to "Chapter 10 Digital Recording Standard" to reflect the addition of ground-based recording systems.

For the download interface, the authors (and the various interested parties) of the standard turned to a new download interface standard that had recently been released, "to promote interoperability for the exchange of data among North Atlantic Treaty Organisation (NATO) Intelligence, Surveillance, and Reconnaissance (ISR) Systems." This new standard, STANAG 4575, covers "the NATO Advanced Data Storage Interface (NADSI)" and "defines the standard for an interface to allow cross-servicing of ISR platforms by NATO nations' ground stations."

Rather than simply incorporate the STANAG by reference into Chapter 10, the Chapter 10 authors chose to duplicate large parts of the essential content of the NATO standard into their document. This facilitated making changes to the STANAG as appropriate for the different use case, but at the expense of detaching the Chapter 10 variant from the NATO efforts. These differences will be discussed later.

NATO STANAG 4575

As mentioned earlier, the STANAG defines NADSI, a data storage interface. Usefully, NADSI defines three aspects of the interface:

- The physical connectors
- The electrical signals / interface protocol(s)
- The logical structures

The physical connectors specified for the initial three versions of the NADSI were somewhat arbitrary, and represented compromises amongst the various contributors and, as is predictable, resulted in an inelegant solution; for example, the defined power supply requirements mandate a total of over 400W and four different voltages between 3.3VDC and 28VDC. This slightly

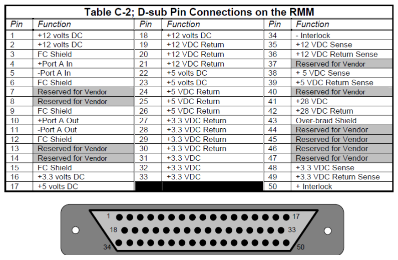

Table C-2; D-sub Pin Connections on the RMM

Pin	Function	Pin	Function	Pin	Function
1	+12 volts DC	18	+12 volts DC	34	- Interlock
2	+12 volts DC	19	+12 VDC Return	35	+12 VDC Sense
3	FC Shield	20	+12 VDC Return	36	+12 VDC Return Sense
4	+Port A In	21	+12 VDC Return	37	Reserved for Vendor
5	-Port A In	22	+5 volts DC	38	+ 5 VDC Sense
6	FC Shield	23	+5 volts DC	39	+5 VDC Return Sense
7	Reserved for Vendor	24	+5 VDC Return	40	Reserved for Vendor
8	Reserved for Vendor	25	+5 VDC Return	41	+28 VDC
9	FC Shield	26	+5 VDC Return	42	+28 VDC Return
10	+Port A Out	27	+3.3 VDC Return	43	Over-braid Shield
11	-Port A Out	28	+3.3 VDC Return	44	Reserved for Vendor
12	FC Shield	29	+3.3 VDC Return	45	Reserved for Vendor
13	Reserved for Vendor	30	+3.3 VDC Return	46	Reserved for Vendor
14	Reserved for Vendor	31	+3.3 VDC	47	Reserved for Vendor
15	FC Shield	32	+3.3 VDC	48	+3.3 VDC Sense
16	+3.3 volts DC	33	+3.3 VDC	49	+3.3 VDC Return Sense
17	+5 volts DC			50	+ Interlock

Figure 1 NADSI Power (Fibre Channel, Ed.s 1 thru 3)

absurd situation was resolved in Edition 4 with a new data interface type, which eliminated all but the 28VDC at 5 amps supply.

The electrical signals and protocols were an entirely logical choice in the early 2000s: Fibre Channel ("FC-AL") and SCSI (over Fibre Channel), which were the interfaces of choice for professional data center applications. Unfortunately, for numerous reasons, Fibre Channel failed to expand its reach into high-volume applications, and remains a high-end, high-cost technology. Critically, every Fibre Channel solution requires a Fibre Channel Host Adapter, using a dedicated interface chip (or FPGA logic block); compare with alternatives such as SATA, Ethernet and NVMe which are often built-in to processor chips, driving cost and space requirements down.

So in the 2012-2014 timeframe the STANAG Custodial Support Team introduced a new electrical / interface option, based on gigabit Ethernet. Critically, the new option used protocols to ensure that the same logical interfaces could, for better or worse, be preserved: instead of SCSI over Fibre Channel, iSCSI was used; instead of FC-AL, IEEE 802.3 1000Base-T Ethernet, DHCP and SLP were employed. In this way, adapting a download system that had previously used Fibre Channel was straightforward, and application software that accessed the NADSI data need not be aware of which interface was in use.

The logical structures, whether implemented over Fibre Channel or Ethernet (or anything else), constitute the part of the design that has been most influential. During the development of the standard, there was heated debate as to the file system it should require. There were passionate arguments in favor of a couple of options, which loosely represented the Microsoft Windows versus the Unix/Linux camps. Unable to come to an agreement, the government members of the team mandated what was termed a Logical Sequential Access Mode filesystem, which should be as simple as possible to implement and that the minimum necessary burden would be placed on the *host* system.

The resulting filesystem was heavily based on legacy tape operations: effectively a table of contents followed by a succession of contiguous files. Each file had a very limited amount of metadata associated with it in the table of contents (see Figure 2 NADSI Header Structure), which imposed limits on the flexibility of the design: no directories/folders, no file-level security, no modification or access times, etc.

Since the filesystem header only allowed for a single pointer to a file – the "File Start Address" in Figure 2, together with the File Block Count and File Size fields – an individual file effectively had to be contiguous[1].

[1] Technically, something like a log structured file system could be implemented on top of this arrangement, where each NADSI "file" is interpreted as a log entry containing filesystem metadata or user file data, or special file naming conventions could be used indicating portions of a file, and thereby allow for apparently non-contiguous "files", but these would both defeat the purpose of having an interoperable filesystem.

The rationale for accepting these limitations was that it was a simple enough filesystem to be implemented on any media type, or it could be implemented as a virtual filesystem on top of an underlying “native” filesystem. This was because since the standard was specifically for downloading data, there was no need for any of the synchronization or updating that most filesystems have to do, and the machine being downloaded could simply generate a static table of contents.

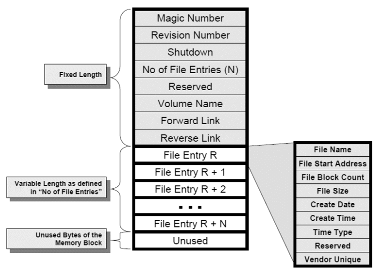

Figure 2 NADSI Header Structure

Indeed, the reference “RMM” devices used to support development of Edition 4 of the STANAG use some kind of Linux filesystem (usually “*xfs*”) on the media itself, by means of the Linux Network Block Device (NBD). NBD consists of a server/daemon task that exports local storage space over a network interface as if were local, and a client that “attaches” to that server making the exported storage appear to be a physical storage device. By using the loopback network adapter, the effect is to take a series of files on the local storage and make them appear to be an all-new disk… which is then exported via iSCSI to the NADSI download station. All the reference RMM needs to create is a file matching the format of the NADSI table of contents, and then link to the actual data files.

Of course, that sort of approach doesn’t work so well when trying to write data to the filesystem, but that’s not a core “use case” for a download interface. With the exception of erasing data once it’s been downloaded, there are very few situations that *require* supporting the write commands, which are helpfully listed in the STANAG as “recommended” (instead of “required”), and those situations are specific to the application for which the storage module is being used, not generic enough to be part of a download standard, so can use alternate interfaces and mechanisms.

CHAPTER 10 MODIFICATIONS TO NADSI

As mentioned before, the Chapter 10 team duplicated most of the STANAG into the Chapter 10 standard and made some modifications along the way.

Some of those changes are trivial (for example, the “Block Size” field was removed from early drafts in the NATO standard as being redundant – the physical interface would communicate the block size), and some definitely beneficial (the “File Close Time” field, for example).

But the most significant change was the recognition that Fibre Channel was an expensive interface and not really justified (in 2005) for the requirements typical test community application; unlike the NATO ISR use case, the typical test recorder handled a few tens of megabytes/second of data and a few hundreds of gigabytes of data.

Therefore Chapter 10, while maintaining the option of a Fibre Channel interface, also introduced an IEEE 1394b “FireWire” interface option. This rapidly became the preferred solution for one simple reason: FireWire can deliver at least 18 watts of power using the same cable as carries the interface signals; this is sufficient to allow most memory modules to be downloaded using a single cable, which is obviously an advantage over solutions requiring separate power and data. And FireWire offers up to 80% of the throughput of the original Fibre Channel interface while being inexpensive enough to be a commodity interface in included by default in some personal computers (particularly those from Apple, Inc.)

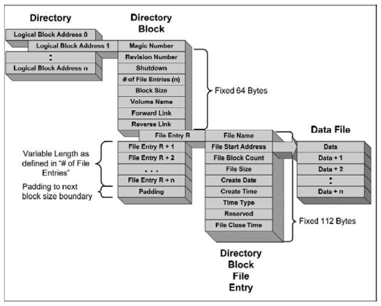

Figure 3 Chapter 10 Modified NADSI Structure

Unfortunately, FireWire fell out of fashion for reasons both good and bad, largely supplanted by SuperSpeed USB (which carried significantly less power, but at significantly faster data rates). This meant that the promised road-map for ever faster FireWire interfaces was never realized, so the mainstream commercial reasons to support FireWire dwindled.

But while it lasted, the most significant implication of the widespread adoption of the FireWire interface option was the assumption that *all* memory modules would be attached via FireWire at all times, which lead to the assumption that the modified NADSI filesystem would be used for data collection as well as download. This resulted in the widespread (erroneous) belief that Chapter 10 itself precluded writing multiple files simultaneously, although of course using a virtual (NADSI) implementation on top of a native filesystem would allow this (albeit with some added complexity).

WHAT ABOUT ETHERNET?

While the popularity of Fibre Channel and FireWire waxed and waned, one interconnect technology has calmly been growing in performance and capabilities. As of the time of writing, there are variants offering a difference of four orders of magnitude in performance: 0.01, 0.1, 1, 2.5, 5, 10, 25, 40, 50, 100, 200 and 400 gigabits per second, using just about any media one can imagine (fiber, twisted pair, coax, and so on). While this technology (alone) is correctly called "Ethernet", the word "Ethernet" is frequently used (inaccurately) to refer to both the lowest layers (the "Physical" and "Link" layers in the ISO OSI Model), and as a shorthand for "IP and IEEE 802.3 Ethernet" (i.e. the Physical, Link and Network layers of the OSI Model). This has relevance in this field as decisions about protocols (etc.) are often tied to "IP" rather than "Ethernet", which can have beneficial consequences for certain outlier applications, such as using WiFi to download from an inaccessible device.

Both the STANAG and Chapter 10 have options to use Ethernet (and IP) but have not updated their standards to include anything other than gigabit Ethernet (in this case, 1000Base-T, which will autonegotiate to 100Mb/s, too). The reason for this is covered in the next sections: performance is not the major problem with the current solutions.

MOVING FORWARDS

While the performance of the download interface is of course important, as the end users of the standards have started expecting (and requiring) additional features and capabilities. One of the earliest additions was a goal to be able to access the data *in situ*, for "quick look" or high priority data access without having to download the whole memory module. Another was the ability to conduct simultaneous downloads to two (or more) stations – for example, to move most of the bulk data via one system, while another cherry-picks a few files to copy to another end-user. These can both be done using the simple NADSI filesystem, but problems arise if the multiple download stations want to modify the data in any way (e.g. by deleting files once downloaded).

Then cybersecurity requirements gained prominence: the first of these is file-level security: setting certain files so they can be read but not written, for example. The next is a simple audit trail goal: tracking which systems are reading (and possibly modifying) which data. And of course if a system has low-level “block” access to a device, then it’s very hard, if not impossible, to make assurances protecting against, say, malware injections, etc. And then there’s a huge one: data at rest encryption (DAR encryption).

There is a utopian desire in some quarters of the standard community that DAR encryption be standardized. However, this is practically an impossible goal: the capabilities, origin and sophistication of any kind of encryption solution is unlikely to be the subject of agreement between any two end-users, let alone an alliance of 32 countries! (While it’s true that there are alliance-wide encryption solutions, most of these are for very narrow applications, such as voice or datalink radios that “do one thing”, unlike a more generic standard for interoperability).

There are many different ways to implement encryption. Some solutions operate as a “bump in the wire” device that is otherwise invisible to the host, but scramble all data read and written from the storage “on the fly”; then there are software implementations that operate at the host level, like Microsoft’s “BitLocker” or Linux’s “LUKS”, and encryption on the file level (e.g. where the application that reads or writes the file is responsible for the encryption). And of course some solutions use multiple layers of encryption, which may each be of a different type.

Finally, and perhaps implied by some of the above, there’s a goal for a much more generic upload/download interface, so that e.g. mission support files (such as maps) can be uploaded using the same interface as acquired data can be downloaded.

Given the limitations of the existing solutions, the stakeholders in the community have concluded that the simple NADSI filesystem has outlived its usefulness. So, a replacement approach is required.

SHAPING A NEW STANDARD

The natural starting point is to select Ethernet as the only permissible interconnect technology and just replace the simple filesystem with a more sophisticated one. However, many of the same issues that deadlocked the STANAG group the first time remain: there are few, if any, sophisticated filesystems that both Windows and Linux support well. Probably the best of the technically acceptable bunch is Microsoft’s NTFS, but there are concerns (mostly performance related) with the Linux implementation of that. None of the common Linux filesystems have really good Windows support, and so on[2].

And in any case, replicating the existing architecture with a new filesystem would retain all the issues of having a low-level block interface: cybersecurity, device sharing, and so on.

The alternative to a block device interface is a file-level one. Given that these are typically implemented via networks, these are generically known as network file systems. Reflecting the Linux vs Windows tension, there are of course two major varieties, but unlike the situation with native file systems, either operating system flavor can access both. In addition to classic

[2] Obviously, there are mechanisms using virtual machines or Windows Subsystem for Linux that *work*, but these approaches add a lot of complexity to configuration control, add performance penalties etc.

workstation/server computer systems, these protocols can be used by everything from mobile phones to supercomputers and are very common in military and sensitive environments.

The Windows variety uses a protocol called Server Message Block (SMB); this protocol is implemented in Linux using a software package called Samba (after the protocol's name). Early implementations using this protocol used the name Common Internet File System (CIFS), but this terminology is largely deprecated. The two Linux-based protocols are both simply called "Network File System" (NFS), but it's worth treating version 3 (NFSv3) and version 4 (NFSv4) as related but different. Microsoft supports NFSv3 in most versions of their operating system since Windows 7 but requires additional third-party software to support NFSv4 as a client although strangely Windows Server supports NFSv4 as a server. It should be noted that many file server solutions support all three (SMB, NFSv3, NFSv4) simultaneously, serving the same files (if configured that way). In this context, the "server" is the device with the storage, while the "client" accesses it, i.e. the download station is the client, the memory module is the server.

In terms of capabilities and architectural sophistication and elegance (important for cybersecurity), the obvious preference goes to NFSv4, but without "out of the box" support in Windows, it appears that the edge goes to SMB using native Windows networking and/or Samba. One incidental advantage of this is that using the latest version of the protocol (version 3.1.1, as included with Windows 11), data-in-transit encryption is available for the files moving from the storage module to the download station. While this may appear unnecessary, the "letter of the law" (in the USA, at least) is that all data-in-transit shall be encrypted (see National Security Memo 8, Section 1, paragraph b, subsection iii).

DATA DISCOVERY

While SMB will provide access to data files, there is a need to communicate to the download system how those files can be located.

It is anticipated that the new standard will use the current STANAG Edition 4 service discovery approach (DHCP to get an IP address, and the Service Locator Protocol to discover information about the storage module), with the latter being enhanced to provide additional information, such as the "Share Name" (or names) needed to mount the exported data. As well as this baseline interface, it is anticipated that the SLP protocol will be able to communicate the existence of a "Control File" located on one of the exported "Shares" that will allow for prioritization and categorization of files to be downloaded; for example, it's possible that the storage module may contain health or status information that would not be of interest to a typical download station, so this "Control File" would allow that to be skipped or left until last.

PERFORMANCE: DOES ONE SIZE FIT ALL?

With Ethernet offering a range of speeds from 10Mbits/sec up to hundreds of gigabits/second, and the network filesystem protocols being agnostic about the data rate as long as it's carrying IP packets, the question as to how to address the range of performance options.

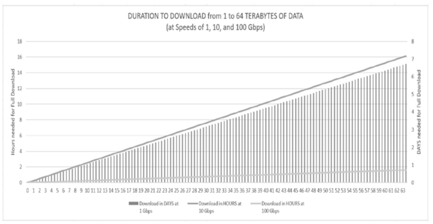

Figure 4 Download Duration

Figure 4 illustrates the time required to completely download various sizes of memory module using three data rates; the vertical bars represent the time in days at 1Gbps while the two diagonal lines show time in hours at 10Gbps and 100Gbps (in all cases arbitrarily assuming 87% efficiency of the line rate).

Picking a capacity at random, a 24TB memory module would take about 2½ days to download with 1Gbps, 6 hours with 10Gbps, or about 37 minutes with 100Gbps. But obviously lower cost modules are unlikely to support the 12.5GByte/second data rate needed for 100Gbps Ethernet, nor are end-users likely to be eager to pay for an interface that's much faster (and more expensive) than they need. At the low end, a 100GB module would need 15 minutes, 90 seconds, or about 10 seconds at each rate, and it is not unreasonable to note that the practical difference in duration between 10Gbps and 100Gbps is negligible when compared to the time needed to collect the data and then hook-up the storage module, and even the 15 minutes at 1Gbps may not be considered too long!

But how long *is* too long? Human tolerance seems to split into several buckets: "nearly instantaneous" (which is a few seconds, as with the slower interfaces downloading 100GB above), "time for a coffee" (which is where the 15 minutes figure fits, and tea or soda are also acceptable), "an hour or two" (e.g. the time to download 72TB over 100Gbps), and longer than that (such as overnight). Much longer (e.g. a full day or more) tends towards being unworkable, not least because of the chance of power failure or other interruption requiring a restart of the process (and traditionally any lengthy operation fails as soon as the last observer leaves the room, or so it seems!).

As always, though, cost is an issue. A 100Gb/s Ethernet plug-in card for a standard ground PC chassis costs a few hundred dollars (exclusive of the transceiver/cabling). But the same technology for an airborne application might cost several tens of thousands of dollars, simply because of the (lack of volume), environmental requirements and that's not including the additional system resources required, such as PCI Express lanes, CPU power, and so on.

So, combining the different tolerances to download durations, different costs and the different size/capacity variables, and it becomes clear that a standard mandating a single solution will either be prohibitively expensive or prohibitively slow, and in either case developers and end-users would work to evade the standard as not fit for (their) purpose.

The solution is for the standard to define several classes of memory modules, and have the download station have a different physical connection for each one.

The proposed classes are shown below; note that each class is named for its data rate, so that the standard can add additional classes without upsetting the patter; for example, if a 25Gbps or 40Gbps became warranted.

Class Name	Data Rate (Gbps)	Time to Download 1TB	Defining Standard Name	Media	Power
1	1	2h 32' 23"	1000Base-T	Quad twisted pairs	140W, STANAG
10	10	15' 14"	10GBase-T	Quad twisted pairs	USB-PD? TBR
100	100	1' 31"	100GBase-SR1.2	Duplex fibers	TBD

Note that both Class 1 and Class 10 can attach to a Class 10 download station interface, because 10GBase-T will auto-negotiate down to the behavior of 1000Base-T and they use the same "8P8C" (also known as "RJ-45") connector. This means that a comprehensive fully compliant download station only needs to provide a Class 10 and a Class 100 interface.

POWER SUPPLY AND CONNECTOR

One of the aspects of the STANAG is that the power supplied to the memory module for the download operation is clearly defined; in the first three Editions specified a 50-pin connector with numerous voltages, while Edition Four added a military-style circular connector carrying 28VDC; the Class 1 interface will use this for backwards compatibility.

By contrast, the Chapter 10 Ethernet download option does not provide much guidance; it states that Power Over Ethernet (PoE) may be used, which can supply 25W using 48VDC, or alternatively 5A using 8-30VDC, but without defining a connector. While it certainly doesn't hurt to *permit* using PoE (and later versions of the PoE standard define ways to deliver up to 71W at 48VDC), it appears that Chapter 10's current version is too vague to use as the basis for a new standard.

An alternative approach might be to adopt the USB Power Delivery standard (USB-PD 3.0), using the USB Type-C connector. With a ubiquitous, generic USB-C to USB-C cable, this will deliver up to 60W, and with special "electrically marked" cables, up to 240W, which covers a lot of territory. This is the same approach now being taken with laptop computers, so finding power sources (and implementation circuitry for suppliers/developers) should be straightforward – and for storage modules that need less power, the implementation can be designed to a lower price point. At present, this is the preferred option for Class 10.

For Class 100, the issue is more complex. A large storage module and the supporting CPU, 100Gbps Ethernet interface, encryptor(s) and transceivers may require several hundred watts of power, particularly if one is using the "dismount the entire recorder" approach in place of a smaller detachable storage module; in that case, you may need to power the whole system. In this case, something like a 48VDC, 10A solution might be necessary. This is still under review, but in the meantime the USB-C, USB-PD option is a good starting point (remember, all download stations should support Class 1, Class 10, and Class 100, so will already provide the Class 10 USB power interface as well as the Class 1 STANAG Edition 4 one.

For both the power and data connectors, suitable components must be used. The Class 1 devices will use the same data connector as is defined by the STANAG Edition 4, so that a STANAG Edition 4 solution only needs updated software to support the new standard. Since the Class 10 uses the same physical wiring as Class 1, this approach will be used for Class 10 data connections.

For Class 100, a suitable fiber optic connector is under review.

ENCRYPTION

As suggested earlier, standardizing encryption across a huge range of performance and application requirements is virtually (or even actually) impossible. In order for a new download standard to work, then, the storage media must be "unlocked" by some mechanism or mechanisms outside the primary scope of this standard.

These might include "dongles" (sometimes called "Crypto Ignition Keys", or CIKs) or keys and/or "PINs" being sent from some other system (e.g. a mission computer), or both. In some cases, a purpose designed key loader device might be used, via a standard interface. However it is done, physical devices are fairly straightforward as they can just be attached via whatever receptacle is used for the recording device in operation, but for the keying data sent from other systems, a more generic approach is required.

The best option, or at least the "least bad" one, in this situation is for the recorder / storage device to host a web server with page providing appropriate prompts to manually enter whatever data is required. Obviously, this will only be usable where the keying data is not sensitive and can be typed into an arbitrary system, but if the download system is not approved for the keying data, it's likely that it would not be approved for the downloaded data, either.

Using a web system conveniently allows for data entry, using cut-and-paste if appropriate, while leaving the issue of what information (keying data) is required and how it is named (e.g. PIN, key, Inner Key, etc) up to the developed of the recording system.

SUMMARY AND CONCLUSIONS

This paper outlines the perspective, direction and intentions expressed by participants in the effort to update the download interfaces in both the STANAG 4575 and the RCC "Chapter 10" communities. As of the time of writing, no draft of the next version of the standard is available, so this document should be seen as a guide, not a definitive statement of the details of the upcoming (Edition 5) version.

That said, the overall shape of the new standard is fairly well defined:

- It will be based on Ethernet & IP
- It will use a network file system like SMB
- It will have multiple "Classes" to cover different use cases from 1 to 100Gbps
- The base Class 1 will reuse the existing STANAG physical interfaces
- Class 10 will be plug-compatible with Class 1, but at 10x the performance
- USB Power Delivery will avoid the need for anything but a commodity power supply

Because of the compatibility between the new Class 1 and the old Edition 4 technologies, it is anticipated that the new standard will roll out in phases, with the first phase simply changing the software expectations on both ends of the interface. Because of the interoperability of Class 1 and Class 10, this first phase will also allow for 10Gbps downloads with no additional standardization effort.

ACKNOWLEDGEMENTS

The author would particularly like to thank the current and former members of the STANAG 4575 Administrative Support Team (AST) and Custodial Support Team (CST), including the Custodians in the US Air Force SAF/AQIJ group, the National Representatives to the group, the academic support from Space Dynamics Lab at the Utah State University, and finally the industry partners that turn paper into products.

REFERENCES

Due to page count / length limitations and the nature of this paper, references have been omitted. Relevant references are, and will be, included in the standards mentioned throughout this paper.