## Security Audit use cases

### Data Export

Data export takes place when monitored data departs the security domain of the reporting system. The goal of this report is to identify information such as:

* That a data export has taken place (when, who)
* What data was exported (to the extent that this is known or reportable)
* Where the data was sent, and how this delivery was identified (so that other security facilities that are relevant can be identified.)

The oldest data export that is still commonly used is the export of a patient's studies onto a DVD. This may be a DVD that is given to the patient, or it may be DVDs that are sent to a different hospital system as part of a patient transfer.

NOTE: 1. The use of a media such as DVD does not automatically make a transfer into a data export. It used to be the practice that ultrasound machines would capture their images onto magneto-optical disks that were carried by the technicians from the ultrasound machine to the reading workstation. These are considered "data transfer" events in the DICOM audit system because the data never leaves the security domain of the hospital.

2. Moving from one security domain to another is not necessarily a security problem. When transferring a patient the entire process is usually quite secure.

The reasons for auditing are the change of responsibility and higher likelihood of mistakes or attacks. When a patient is given a DVD with their records there is no disclosure problem because it is their own data. There is a risk that the patient will lose or misplace the DVD. Patient's motivations and record keeping practices are quite different than those of a hospital records system.

The an example of the structure of the data captured by the DICOM Export event is shown below:

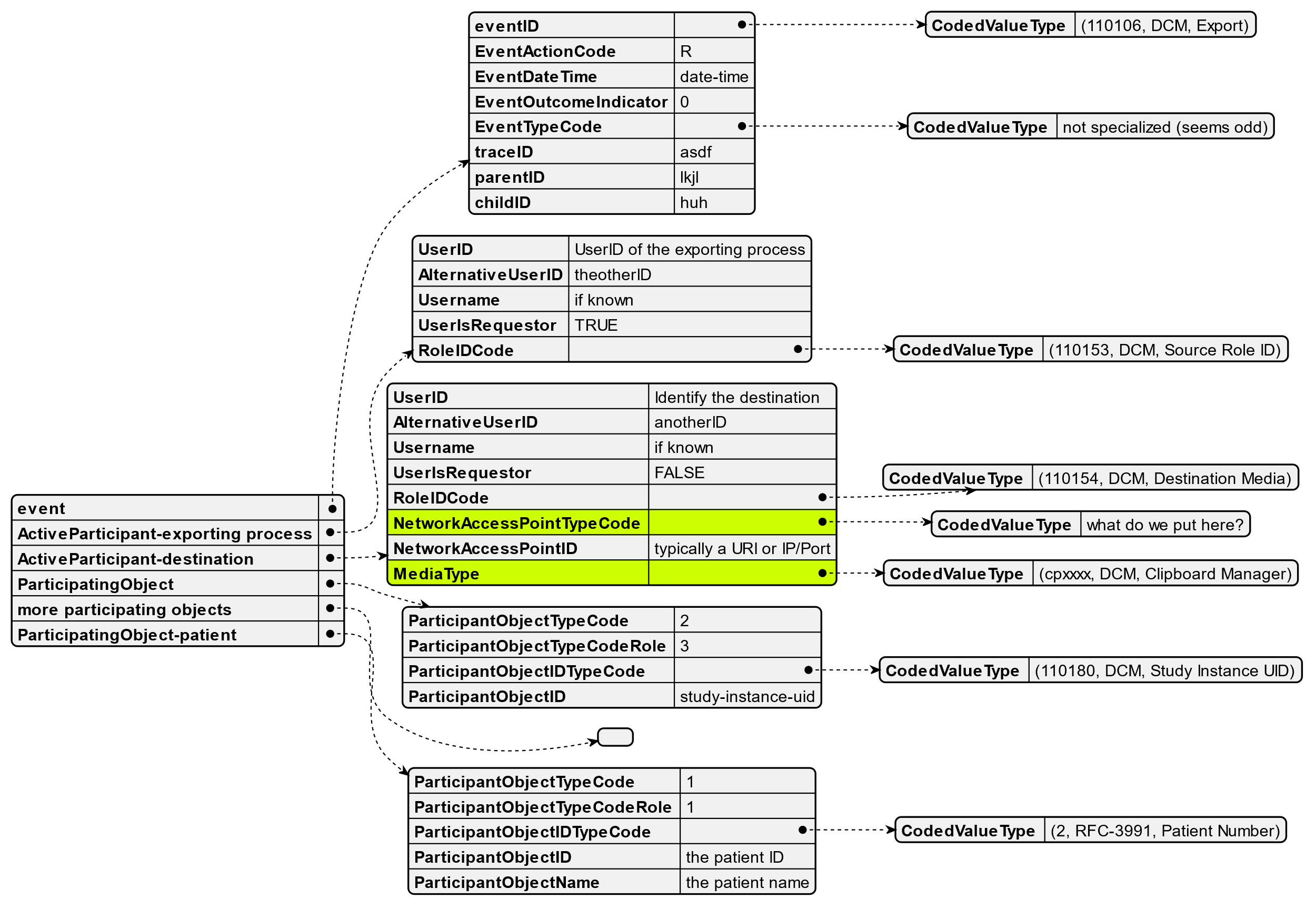


Figure 1 – Sample data export event

This structure was originally conceived in a world where these transfers were primarily performed by physical media exchange. This is no longer the case. The following are kinds of exports that take place today:

* Exchange on DVD, which might be physically labeled to identify the media and its contents
* Exchange on USB sticks, which are too small to by physically labeled, but which may have an electronic label that can be read. The label might not be unique, and the label might be modifiable by whoever posesses the USB stick. (USB SSD disks and USB regular disk drives are also possible, but much less common. Similarly, microSD, Compact Flash, etc. are no longer common but they are very similar to USB sticks.)
* Exchange by email. The structure of email exchange can be very complex. RFCxxxx describes the conceptual architecture. From the perspective of data export, the data is part of the "Message" as defined by RFCxxxx and should be identified by a "Message ID". If the email system complies with other IETF RFCs for email the Message can contain any mix of text, structured data, and binary content. Compliant systems will provide a unique "Message ID" for the message, and may deliver copies of this "Message" to multiple destinations. There are some non-compliant mail systems in widespread use, so log files may be missing crucial information like the Message ID or have multiple different messages with the same Message ID.
* Exchange by FTP (or similar) transfer. FTP transfers a set of files and directories. This is similar in concept to the DVD or USB file systems, but lacks an identification for the files or directories exchanged. There is no equivalent to the Message ID or media ID. FTP has identifiable network endpoints.
* Exchange with Clipboard Managers and similar internal system sharing mechanisms. Clipboard managers have no identifiable network endpoints, and no persistent identification of the data contents. An oversimplied summary of the interface known to the software developer is that they make a procedure call like ```clip(data)``` and everything else is done like magic by system libraries, and other hidden mechanisms.
* Exchange with web APIs. These typically can be identified by a network endpoint or URI, but like the Clipboard Manager, all of the functionality behind the web service is hidden from the application. (In some cases the application will perform the HTTP transactions and be aware of their structure, but in many cases this is all hidden behind an application development framework.)
* Exchange with web services like One Drive, AWS S3, Google Drive, GitHub, GitLab, Teams, Zoom, etc. These may have an identifiable endpoint, like the web API, but have their own unique identification mechanisms.
* Exchange with an external web service for AI analysis, where data is exported to the service and results will be imported later when the service is completed. This is a data export rather than a data transfer because the security domain for the web service is a different domain. If this were an internally operated and secured service this would be logged as a data transfer.
* Exchange with an external social media system. This might be the same as exchange with an external webservice, but we need to explore it in more detail. There may be important differences despite being the same kind of technology implementation.

All of the methods described above have been used to export patient data as part of a data breach. The goal of the data export audit events is to provide a list of all the data exports (who, what, where, when) as a starting point to investigate known breachs and to monitor for inappropriate activity.