Chapter 8

Secure Video Databases

This chapter presents four different approaches that can be used for the development of a secure video database and its surrounding components. These approaches vary in complexity and ease of use; the most suitable approach to be used for a particular multimedia environment depends much on the application and system requirements, i.e. level of accuracy, frame level or content level authorisation, available processing power, etc. A few methods of adding frames or worldly objects to a video file will thereafter be briefly discussed. Finally, suspicion and security issues concerning these approaches will be highlighted.

8.1 Introduction

Firstly, it must be noted that the models presented in this chapter refer only to the 'illusion of motion' part of a digital video file – they do not refer to the audio part of a digital video file. Techniques used to assemble the animation part of a video file according to a requesting entity's security classification will therefore be presented in this chapter. Techniques that can be used to assemble a digital audio file according to a requesting entity's security classification will be discussed in chapter nine (these techniques can also be combined with the methods presented in this chapter in order to assemble the *audio and video* part of a digital video file according to a requesting entity's security classification).

The models presented in this chapter and in the audio chapter (next chapter) follow a similar format to the secure image database models that were presented in chapter six. All the models presented also contain a database interface and a security manager interface. The activities surrounding the components of these models are also divided into the same three phases. The main differences in these models lie in the actual details of the three phases i.e. in how they are carried out to ensure logical access control at the frame or content level for that type of multimedia. For example, an obvious difference in the models discussed in this chapter is that a video repository is used instead of an image repository.

The four approaches presented in this chapter are: the basic frame editing model, the manual approach, the script approach, and the robot vision approach. These will now be further discussed.

8.2 The secure video database models

The main purpose of the three phases (for all four approaches) is similar to that of the secure image database models. The purpose of the input phase is to gather

all the information needed to assemble a video file according to a specific security classification. The storage phase writes all the information gathered from the input phase to the video database and video repository. The output phase uses the collected information to ensure that only the authorised parts of the video file are sent to the requesting entity (as one video file).

8.2.1 The basic frame editing model

This is the simplest approach of the four that are presented. This model allows an entity to select a range of frames within the entire digital video file and assign a classification to it. When an entity requests that video file, only the ranges of frames that have an associated classification below or equal to the requesting entity's security classification will be made available to that entity. Figure 8.1 illustrates the basic frame editing model.

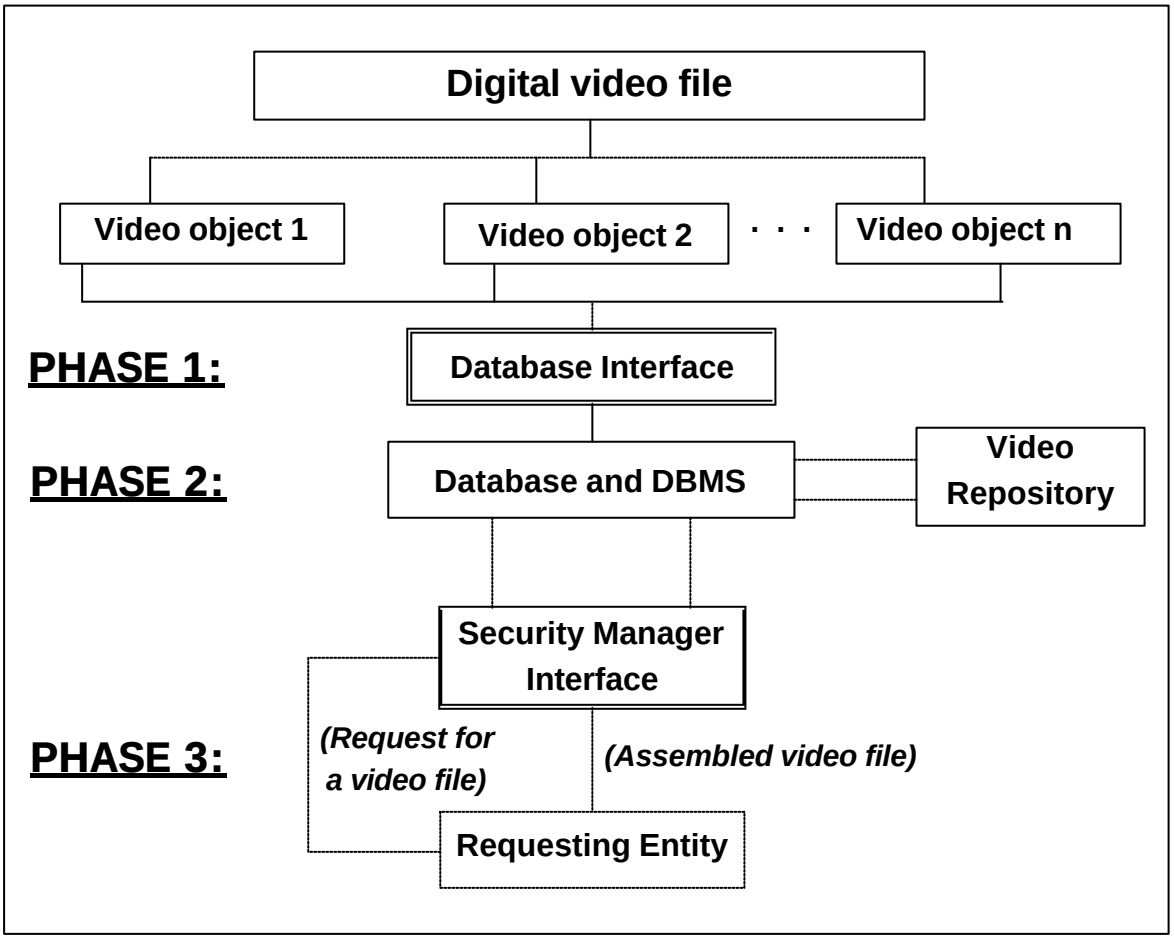

Figure 8.1: The basic frame editing model

In this model, a *video object* is created (as opposed to an image object), which contains the security information associated with a frame or group of frames. A

video object in this model is associated with only four attributes: RangeFrom which is the frame number of the first frame in this group, RangeTo which is the frame number of the last frame in this group, the associated security classification for this frame range, and a reference to the video file. In the above illustration, the variable n is equal to the number of video objects that are defined for that particular video file. The solid and dashed arrows represent the flow of information between the components of the model. Dashed lines are used for the special cases where a request is being made between two components.

The first activity to be performed by an authorised entity is to create all the video objects that must be associated with a digital video file (which will be stored in the video repository). The entity must parse through the video file and select the start and end frame numbers for each group of frames that need to be associated with a video object. This is performed using the database interface or a client program that securely communicates with the database interface. The entity must then assign a classification for each defined video object. All ranges within the video file that have not been associated with a video object are assumed to have the lowest security classification.

As with the secure image database model (and all the models that are still to be presented), the database interface must be the only component that can directly communicate with the video database. All management regarding the security objects on the video database must also be performed through the database interface exclusively. This includes activities such as changes in video object classifications, frame range alterations, and video object additions/deletions.

Once all the video objects have been created, the database interface transfers the video file and its objects to the RDBMS. The RDBMS stores the video file in the video repository and records the associated information in the video database. The video file will be assigned a unique identifier, which will be stored in the database together with the path of the original video file within the repository. Each video object associated with that video file will include this unique identifier as a reference to the original video file. Background images are not needed in this model because an entire frame/range of frames will be removed if unauthorised, and not just part of a single frame that must be substituted with the corresponding area of a background image.

An entity must make a request for a video file using the SMI or a client program that securely communicates with the SMI. The SMI will then relay the request for a video file on to the RDBMS, which will transfer the original video file and the

associated security information to the SMI. The SMI will then scan through all the video objects to see which of these have an associated classification that is above the requesting entity's security classification. The range of frames associated with these unauthorised video objects will then be removed from the original video file, and the remaining frames will be assembled to form a new video file, which will be sent to the requesting entity.

This model can also be modified to allow the assembled video files to be pre-calculated. In this case, the database interface assembles a video file for each security classification that may be assigned to an entity. Once all the video objects have been created, the database interface scans through the objects and calculates the assembled video file for each defined security classification (using the above process). These pre-calculated video files will then be stored in the video repository, and a unique identifier containing the path and classification of each of these files will be written to the video database.

The SMI can now fulfil a request for a video file by retrieving the pre-calculated video file whose associated classification corresponds with the requesting entity's security classification, which will thereafter be transferred to the requesting entity. The SMI therefore no longer needs to assemble a video file. Using this approach, the original video file and the associated security information does not need to be recorded at all; all that is needed is the pre-calculated video files and their classifications (although the pre-calculated video file associated with the highest security classification is equivalent to the original video file). The disadvantage with this approach is that any changes made to an object's classification or frame range requires that certain pre-calculated video files be reassembled. In contrast, the original approach could allow an entity to simply modify the classifications or frame ranges on the video database (using the database interface).

Although pre-calculating the video file for all possible security classifications reduces the time needed to transfer the video file to the requesting entity, the amount of time saved is not very significant. This is because the time taken to remove an entire group of frames from a video file is usually relatively small (which for this model is the principle function used during the assembly process). This differs from the secure image database model where only parts of an image may need to be replaced, which can be processor intensive (depending on the algorithm and frequency of the region selection methods used).

A combination of these two models could however be implemented for systems requiring very efficient responses to requests for video files and relatively efficient

management of the video objects (additions, deletions, modifications). A video file could then by default be pre-calculated for all possible security classifications *and* all video object information can be stored in the video database. This approach has two main benefits. Firstly, the response time is shortened because the correct pre-calculated video file can be sent to the requesting entity as soon as a request is made. Secondly, any changes made to a video object can be written to the video database; although this requires regeneration of certain pre-calculated video files (which may take time to generate for all security classifications), the SMI could assemble any video files if a request is made for a video file whose relevant pre-calculated video file has not yet been regenerated (a queue-type system may need to be implemented for systems with a high number of frequently changing video files). This type of implementation would however have higher system requirements on the server side.

This model (and its variation) allows an entity to control access to a video file at the frame level as opposed to the entire video file. This can be especially useful in cases where an entire scene/group of related frames need to be cut out for a particular classification level (and below). It also allows for relatively efficient video object creation, storage, and assembly, and it is easy to manage the video objects if this information is written to the video database.

The problem with this model is that an entity can only assign a classification level to a frame or group of frames – access control cannot be performed on a smaller level. It may be the case that only a particular worldly object (that is resident within a few frames) must be removed from a video file for an entity's request whose security classification is below a defined level. This model can only ensure that this video file is authorised by removing *all* the frames containing that particular unauthorised worldly object. The basic frame editing model cannot therefore be used to provide logical access control at the *content level* within each frame.

8.2.2 The manual approach

The *manual approach* allows an entity to select a number of regions within a frame, associate these regions with an image object, and thereafter associate a set of created image objects within a frame or group of frames with a video object which has an assigned classification. As with the basic frame editing model, a requesting entity will only receive the parts of the requested video file that it is authorised to view/access. The main difference with this approach is that an entity can specify which parts within the content of a single frame or group of frames

can be accessed by a given security classification. This model therefore allows more 'depth' or detail to be specified when creating the video objects. Figure 8.2 illustrates the manual approach.

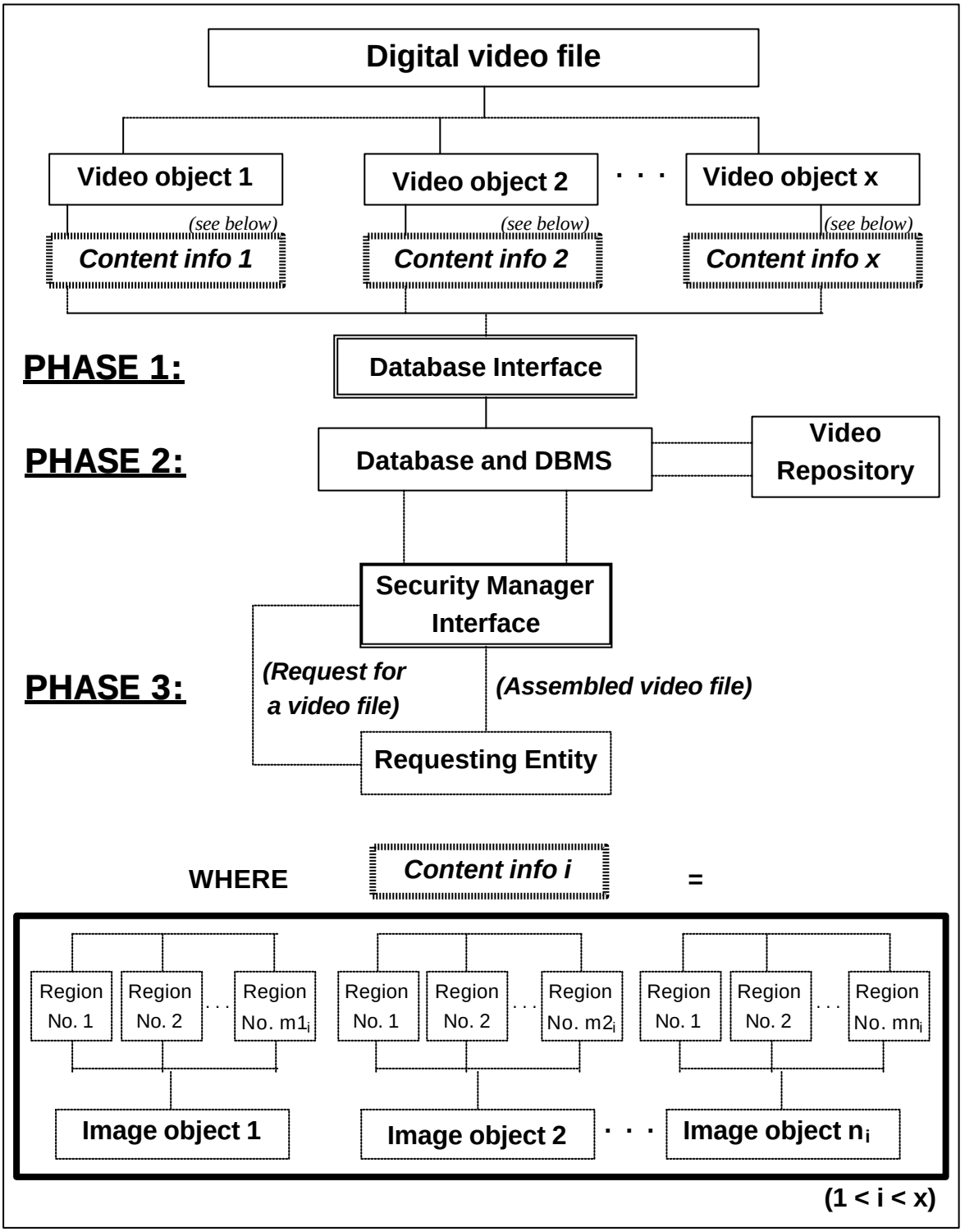

Figure 8.2: The manual approach

In the above model, the symbols x, n_i, $m1_i$, $m2_i$, ..., mn_i are variables. The variable x is equal to the number of video objects that have been created for that particular

digital video file. The variable n_i is equal to the number of image objects that are associated with video object i ($1<i<x$). The value of the variables $m1_i$, $m2_i$, ..., mn_i represent the number of regions defined for image objects 1, 2, ..., n_i respectively. As before, the solid and dashed arrows represent the flow of information between the components of the model. Dashed lines are used for the special cases where a request is being made between two components.

This approach can be thought of as a combination of the basic frame editing model and the secure image database model which allows for logical access control within video files stored in a video repository. As with the basic frame editing model, a number of video objects which contain references to certain pieces of a video file are created and each assigned a classification. In this model, however, a video object does not refer to an entire frame or group of frames; it is associated with a number of regions within a frame or group of frames (through a set of defined image objects).

The most notable difference between this model and the secure image database model is the addition of the video object components. The image objects in this model are also not directly associated with a classification (unlike the image objects in the secure image database model); the main function of an image object in this approach is more simply to 'contain' all the region information of its associated regions. This is possible because each defined image object is associated with a video object which contains its own classification. Each video object's associated image objects (and their associated regions) therefore 'inherit' that specific video object's classification.

The video objects in this model are associated with image objects by referencing each defined image object (e.g. using a unique image object identifier) together with the frame numbers of the frames containing the relevant regions associated with those image objects. A sequence of 29 frames may for example contain a stationary object such as a car (throughout all 29 frames) which can be selected in the first frame and associated with an image object. This image object can then be associated with a video object together with the frame range (i.e. all of the frame numbers where the car is stationary). The video object is assigned a classification, which when correctly assembled should ensure that a requesting entity whose security classification is below that of the video object will not be able to view the car for those selected frames.

This approach requires an entity to select the regions of a stationary object and associate it with an image object only for the first frame where it is stationary. Any

other frames containing the selected object in the same position and shape can be associated with a video object without reselecting that object for each frame. This is because a set of regions can be associated with an image object in a way that is not specific to any particular video frame (achieved by ensuring that coordinates and other region information used to reselect regions are specified in an abstract manner). Any video object can now select the stationary object by referencing that image object and associating it with the frame number(s) containing the stationary object. This can be very useful because worldly objects within the same video scene/cut are often stationary for several frames.

The entity creating the objects must ensure that all the frames associated with a video object contain the regions that make up the shared image object, since all of an image object's associated regions will be reselected in the same manner for all of the frames that were referenced/paired together with that image object. A setback with this approach is that a change in the shape or position of a worldly object from one frame to the next requires the entity to reselect the regions and create a new image object. For example, a car that changes position for all 29 frames requires 29 separate selections and 29 image objects, which will all be associated with one video object.

In the first phase, an entity must create all of the image objects and video objects that will be needed to correctly provide logical access control (at the content level) for the digital video file in question. The database interface will be used to create all of the objects and to manage these objects after they have been stored on the video database (as with previous models, a client program can also be used to securely communicate with the database interface). The entity must parse through each frame, this time searching for worldly objects (and not entire frames) that would need to be removed for a given security classification and below.

The regions for all of these worldly objects must then be selected and associated with image objects. Next, these image objects must be associated with video objects together with the frame numbers whose regions must be removed (according to that image object's associated regions). Finally, a classification is assigned to each video object. All regions within each frame that have not been associated with a video object (through an image object) are treated as having the lowest possible security classification.

As with the secure image database model, region selection methods such as the box method, polygon method, and automated method will be used to select the desired regions within each frame. Similarly, any new methods can be added

provided that the implementation details remain the same for both graphics components (database interface and SMI). Any improvements that were made to the region selection methods, such as using the 'NOT operation', are also valid for this model because the image objects still contain references to a unique method identifier (and these improvements are implemented in the actual region selection methods); lower level modifications (i.e. how the regions are selected) therefore do not adversely affect higher level components (i.e. the video objects).

Background images must be used for this model because logical access control is performed at the content level. As with the images, the unauthorised area that must be removed within a video frame is replaced with the corresponding area of that frame's background image. Note that not all frames within the video file must be associated with a background image; only those frames that are associated with a video object need a background image because only these frames may at some stage have their regions removed. A different background image should also not need to be used for each and every frame associated with a video object. A background image may be common to many frames within a video file, since the frames within the same scene/cut are often similar. The background image is associated with an image object, allowing an image object that is common to a number of similar frames to reference the same background image.

Note that a change in classification for the same worldly object over a range of frames requires a new video object to be created for each classification. The same set of image objects can however be associated with each new video object if the worldly object does not change in shape or position within the paired frame numbers. Although not essential, it is easier to name video objects according to the action/worldly object concerned. A video object called RunningMan could be used for a series of frames that shows a man running. The man's position would most likely change from one frame to another, and so an image object would need to be created for each frame and all of these would be associated with the video object RunningMan (assuming that the classification is the same throughout).

After all the objects have been created, the database interface transfers the video file, background images, image objects, and video objects to the RDBMS. The RDBMS stores the video file and background images in the video repository and records all of the objects in the video database. The video file will be assigned a unique identifier which will be stored in the database together with the path of the original video file within the video repository. All the video objects associated with that video file will include this unique identifier as a reference to the original video file. Each video object also has its own unique identifier, which is stored together

with the video object's classification. Similarly, each image object contains a unique identifier, which is used by a video object to associate image objects with the relevant frame numbers. Every image object contains a reference to one or more background images (depending on whether a different background image needs to be used for each security classification). Finally, an association is formed between an image object and its regions (using the image object identifier).

Once all these objects have been created and stored in the video database, an entity can make a request for a video file using the SMI (directly or indirectly using a client program). At this point, the SMI will request the RDBMS for the original video file, the background images, and the associated security objects that are needed to assemble the video file. When all has been received, the SMI scans through the video objects to see which of them have been assigned a classification above the requesting entity's security classification. These video objects will be dealt with one at a time.

The regions of the associated image objects are then reselected for all of the referenced/paired frame numbers, and replaced with the corresponding area of the background image referenced by that image object (the background image corresponding to the requesting entity's security classification will be used in the case where a different background image is used for each classification). Once this process has been completed for all video objects, the remainder of the frames are joined with the assembled frames to produce the requested (authorised) video file, which is then transmitted to the requesting entity.

Pre-calculating video files for each security classification is strongly recommended for this model since the amount of processing required to assemble a video file is generally very high. This is because the average amount of processing needed to assemble an image using the secure image database model (which can be quite high) is needed for each frame that is to be assembled in the requested digital video file. The amount of processing required is directly related to the total number of regions to be replaced and the complexity of the region selection algorithms used. Naturally, this varies on a case-by-case basis, ranging from the lower-bound case where no frames need regions to be replaced to the upper-bound case where every region in every frame needs to be replaced.

The video files will be pre-calculated after all of the image objects and video objects have been created and the appropriate associations made. The database interface scans through the video objects and calculates the assembled video file for every possible security classification using the exact same process that would

otherwise need to be performed by the SMI each time a request is made. As with the basic frame editing model, these pre-calculated video files will be stored in the video repository, and a unique identifier containing the path and classification of each of these files will be written to the video database.

This model's variation should dramatically improve the speed at which requests are completed, since the SMI can now simply request the pre-calculated video file that corresponds with the requesting entity's security classification, as opposed to assembling the entire video file each time. As with previous model variations, the object information is not written to the database and most changes require some of the pre-calculated video files to be recalculated. This process should however be relatively efficient if only a small number of frames need modifications, since only these frames need to be reassembled (the majority of the frames can be appended from the previously pre-calculated file, for each classification).

In contrast to the basic frame editing model, the amount of time saved when pre-calculating the video files using this approach is often very significant because in this approach only parts of a frame may need to be replaced (as opposed to the entire frame), which is a lot more processor intensive than the previous model. Unless it is certain that every video file requires only minor replacements to be made for each classification, pre-calculating the video files should be performed as a rule. If necessary, a combination can be used where pre-calculating the video files does not occur in *only* specified cases. To improve efficiency, the RDBMS should in these cases only transmit the video objects and associated image objects that are above the requesting entity's security classification to the SMI. Similarly, only the relevant background images needed to assemble the video file should be sent to the SMI (the query that is sent to the RDBMS should therefore filter out only the minimum number of files and objects that are needed for the assembly process).

This model allows more control/detail when defining the video objects that will be used to assemble the video file. Allowing an entity to specify which parts within a frame/group of frames can be accessed by a given security classification is especially useful in cases where only a particular worldly object or group of worldly objects must be removed from a set of frames. The only option available with the basic frame editing model is to completely remove all the frames containing these worldly objects for the given security classification, which may not be appropriate in many cases e.g. if the requesting entity knows that the removed frames do exist (which will undoubtedly arouse suspicion, whereas replacing only a part of these frames may not arouse suspicion at all).

Ensuring logical access control at the content level within each frame certainly comes with its price – there is more effort required to create the security objects, much higher system requirements, and more effort that is needed to manage the objects. The creation and management of video and image objects is made more difficult because any change in a worldly object's position, shape, or classification from one frame to another requires a different image object to be manually created (if it does not already exist) and associated with the video object. This can be very time consuming if it happens frequently. The next approach can be used in certain cases to speed up this process.

8.2.3 The script approach

The system requirements for the server(s) containing the SMI can be lowered significantly by using the manual approach's variation (if the video objects do not change very frequently). The bigger problem therefore lies in reducing the amount of time required to define and manage the security objects that are to be associated with a video file. The script approach is used together with the manual approach for certain cases in order to improve the speed at which a set of image objects are created and associated with the relevant video object. Figure 8.3 below illustrates the process followed when using the script approach.

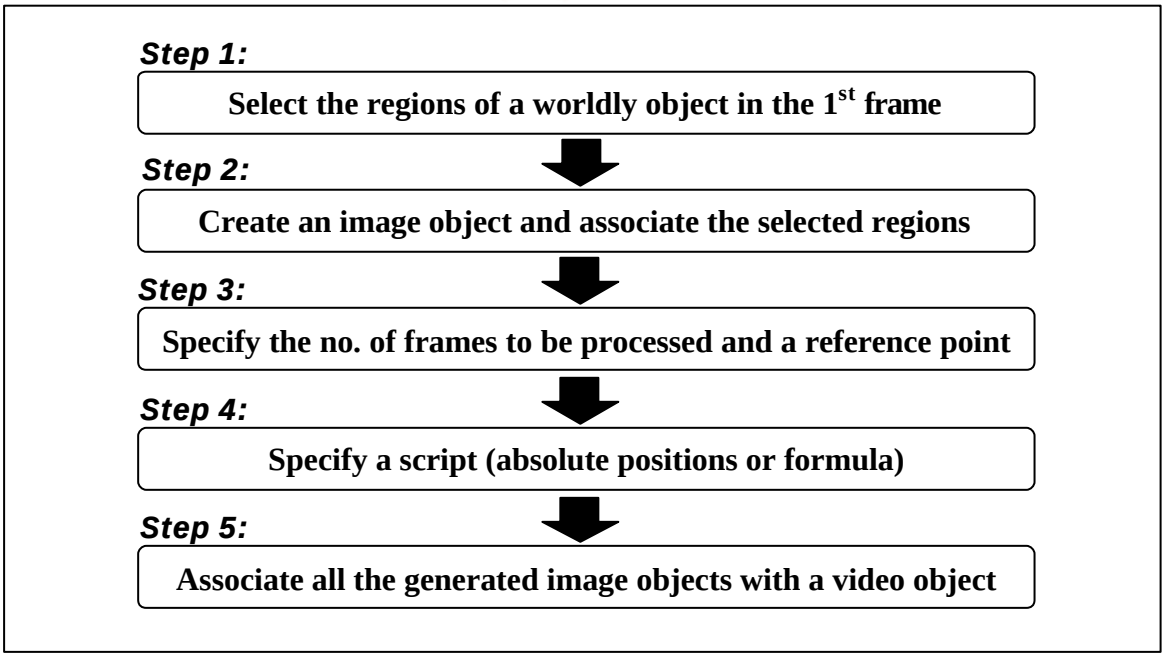

Figure 8.3: The script approach

This approach can be used in cases where a worldly object that must be selected is common to a number of frames and changes position between any of these

frames. Using the manual approach, an entity defining the security objects would need to create a new image object each time that worldly object changes position. The script approach helps speed up this process by automatically creating the image objects, using the values specified in a script as input (which is provided by the entity). The script's values indicate how the worldly object changes position throughout the range of frames, which for example allows a moving ball to be associated with a video object with less effort. This approach cannot however be used in cases where a worldly object changes shape from one frame to another.

The first step to be performed by an entity is to select the regions of a worldly object as before using the available region selection methods for the first frame where the worldly object is to be classified (the first frame in the series of consecutive frames containing the worldly object). The regions that make up the worldly object must then be associated with an image object. This image object will now form the basis for the following frames containing the worldly object that could be in a different position (but with the same region composition). The entity then specifies the number of frames (after the first frame in that series) which each contain the worldly object that must be automatically associated with an image object. A reference point (uniform to all the frames) must also be specified which together with the outputs of the script will be used to calculate how the worldly object moves from one frame to the next.

The fourth step in this approach is to provide a script containing the same number of values as the number of frames specified earlier. Each value is in coordinate form (e.g. in a matrix) and is the difference between the reference point and the top-left position of the worldly object for that corresponding frame number. The values specified should either be absolute values/coordinates or a formula should be provided (if possible) which can be used to calculate the absolute values, e.g. a formula of the form: $y = mx + c$ and the variation in distance from one frame to the next in the case where the object is moving in a straight line.

After the script has been specified, the database interface will automatically generate the image objects for each frame within the range. Note that the region information remains the same for all of these frames (aside from the difference in position) because each region's shape does not change. The coordinates of each new image object's associated regions are therefore slightly modified according to the difference between the reference point and the current position of the worldly object (which is specified in the script). All the information needed to automatically generate each image object is therefore available. A new image object will obviously not be created if the worldly object does not change position from one

frame to another; the image object for the previous frame will be used in this case (the database interface should also scan previously created image objects before creating new ones e.g. if the worldly object comes back to a previous position).

Finally, the generated image objects together with the image object that was manually created for the first frame will be associated with a video object, which will be assigned a classification. This approach will produce the necessary image objects and associations as though it was performed entirely using the manual approach; the rest of the three phases are not affected using this approach i.e. it does not affect other image/video object specifications, the original video file and associated object's storage, and the retrieval process. This approach can also be used to automatically associate an image object with more that one worldly object within a number of frames, as long as all these worldly objects change position (relative to the reference point) by the same distance within the specified range (and do not change shape or classification from one frame to another).

The script approach saves time by eliminating the need for an entity to select a worldly object's regions (which remain the same), associate them with an image object, and associate these with a video object for each frame. A worldly object's position can therefore be 'tracked' using a more efficient process, since only a single coordinate needs to be specified for each frame. Any classification changes that need to be made within the range of frames still require two separate video objects to be associated with the relevant frames. This approach can however be used separately to generate the required image objects for each video object.

Although this approach helps to ease the inconvenience of having to reselect and associate a worldly object at each change in position, any change in the worldly object's size or shape still requires a new image object to be manually created in each instance. This situation is quite common and can be very time consuming if it occurs at every video frame. The following approach tries to solve this problem.

8.2.4 The robot vision approach

A worldly object can often change in dimension or shape during a video segment. The script approach can be used in cases where only minor shape changes occur from one frame to the next (by selecting slightly larger regions in the first frame, which still may not be appropriate if the worldly object overlaps other objects within that frame range). Many cases exist where a worldly object changes shape between frames e.g. a *person* running through a field, or *water* flowing out of a

glass and onto the floor. A more advanced technique is needed to automatically select and associate these shape changing worldly objects with video objects. Robot vision techniques can be used together with the manual approach in an attempt to recognise and automatically select these objects. The diagram below illustrates the steps taken during the robot vision approach.

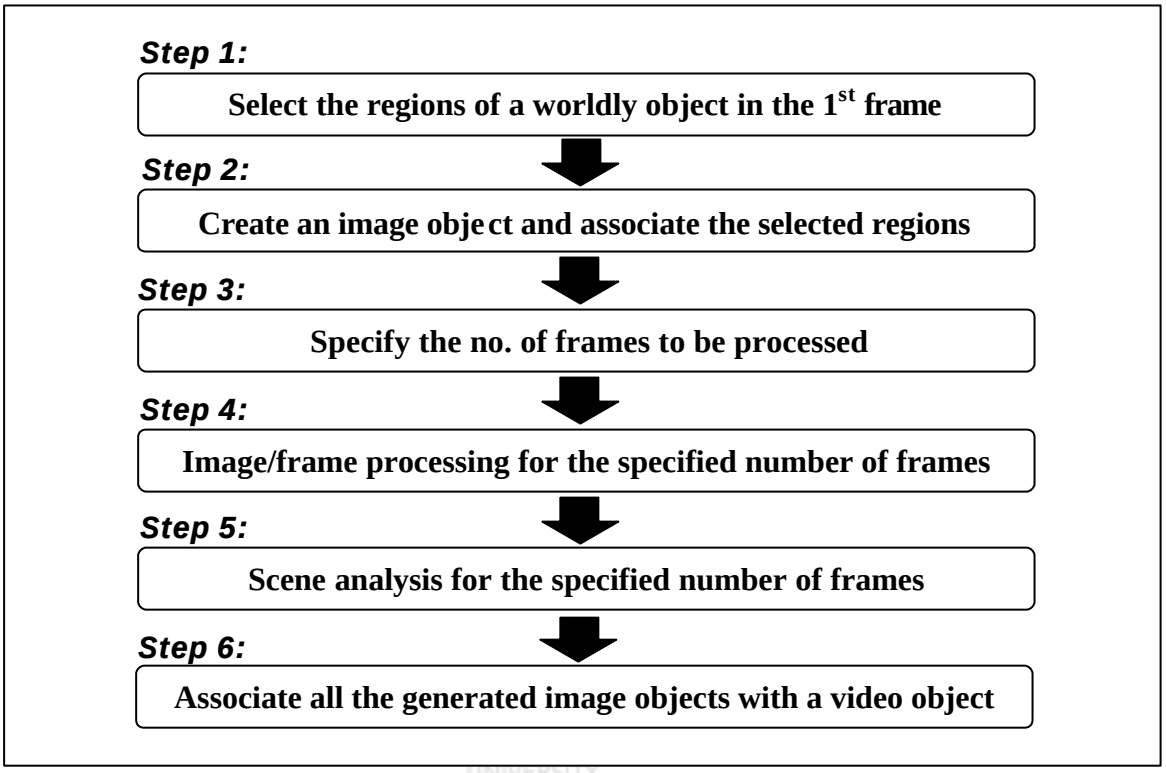

Figure 8.4: The robot vision approach

The process used to automatically associate worldly objects with a video object using the robot vision approach starts off in the same way as the script approach. An entity begins by selecting the regions of a worldly object using the region selection methods for the first frame where the worldly object is to be classified. Next, these regions will be associated with an image object. This image object will be used as input for the robot vision techniques in the following steps in order to recognise and select that worldly object, even if the object's shape/dimension changes (although this may not be successful when major changes occur).

Instead of specifying a script, this approach makes use of robot vision techniques to recognise and select a worldly object within a set of video frames. The next step is for the entity to specify the number of frames that must be processed using the robot vision techniques. Unlike the script approach, the desired worldly object does not necessarily have to exist in each of these frames. This is because in the script approach, the worldly object's location is specified for each frame, whereas the robot vision techniques used in this approach will be the deciding factor as to

whether the worldly object exists/is recognised within each frame (although a pre-determined value such as a negative value can be used in the script approach as a part of a script to indicate that the corresponding frame does not contain the worldly object). Note that a reference point is not needed in this approach.

Once the entity has completed steps 1 to 3, the database interface will attempt to find the desired worldly object within the specified number of frames. Firstly, image processing (or frame processing in this case) is performed on each of the frames in order to prepare them for scene analysis. To begin with, an averaging operation can be carried out on each of these frames (although this is not essential) to help eliminate certain irregularities.

All the regions within each frame are then calculated in order to segment the frames into scene-relevant portions. The split-and-merge method explained in chapter 4 can for example be used to find all the regions of a frame (any other region finding algorithm should also do). If necessary, an advanced option could even allow region finding to be based on visual texture instead of image intensity (to select worldly objects such as grass). The frame processing phase (step 4) is complete once all the regions for each frame have been calculated.

The database interface must now carry out the scene analysis phase (step 5), which uses robot vision techniques on the processed frames in an attempt to find the desired worldly object. The image object that was manually selected in the first frame will be used to identify similarly structured regions within the specified frame range. The regions that make up the manually chosen worldly object will therefore be compared with all the regions calculated in the frame processing phase using robot vision techniques i.e. the region composition of the worldly object should remain reasonably similar within the range, even though the actual regions change in shape or dimension (obviously this will only be detected within defined limits). A similar set of regions that have been found in any of the video frames will be associated with an image object, since they are assumed to make-up the desired worldly object. Once this process is complete for all the frames, all these image objects can be associated with a video object.

An entity should be allowed to confirm the automatically selected regions for each frame (if necessary), because this approach may not always accurately select the desired worldly object in every frame that it exists (this depends on the accuracy of the robot vision techniques used). If a number of possibilities exist for a particular frame (even though only one the possibilities is actually the desired worldly object), the entity should also be allowed to choose which possibility is

valid. This could easily occur in cases where a particular person is the desired worldly object and other people exist in the frame range (since the robot vision techniques used may not be able to distinguish between similar objects).

After the entity is satisfied with the generated image objects, these together with the image object that was manually created for the first frame will be associated with a video object and assigned a classification. As with the script approach, the generated image objects and associations will be produced as though the manual approach was used for each frame. The robot vision approach should be used separately for each worldly object in order to improve the chances that it will be recognised throughout the specified frame range (even though more than one desired worldly object associated with a single image object may in certain cases have a successful outcome).

This approach should be successful in automatically selecting and associating a worldly object with an image object in many cases, provided that reasonably good robot vision techniques are used. The fact that a worldly object typically moves only a small distance from one frame to the next (especially for video files with higher fps) helps increase the probability that a worldly object will be correctly recognised within small frame ranges (because a worldly object's regions should not change dramatically in shape or dimension from one frame to the next for most of these cases). The robot vision techniques that are used are independent of the robot vision approach described here – they can therefore vary in complexity and can be specifically designed for maximum results e.g. they can be tweaked to recognise the worldly objects that will most likely be encountered.

The robot vision approach has the advantage of being able to 'track' worldly objects even if they change shape or dimension between frames. It may now be possible for a person running through a field to be automatically assigned a set of associated image objects throughout the frame range. The entity wishing to create the security objects will therefore save a lot of time that would otherwise be spent selecting and associating a worldly object with a video object for each frame.

This approach does however set high system requirements for the machine containing the database interface component because the region finding process and the robot vision techniques can be quite processor intensive. The robot vision techniques used may also not always recognise the desired worldly object within each frame, especially if the object dramatically changes its region composition from the first (manually selected) frame. A solution to this problem may be to set smaller frame ranges in the third step of this approach and manually selecting the

worldly object at the start of each dramatic shape change, which should allow for more accurate object recognition until the next major change.

8.3 Adding frames or worldly objects to video files

The above models concentrate on *removing* frames or worldly objects from a video file in order to ensure that the entire video file is authorised to be received by a requesting entity with a given security classification. Either the video frames are removed (using the basic frame editing model), or the area that is taken up by an unauthorised worldly object is substituted with the corresponding area of a background image (using the manual approach and its extensions, the script or robot vision approach). It may however be useful to add an entire frame/set of frames to the beginning, end, or between any two frames within the video file for a given security classification. Similarly, a worldly object or a set of worldly objects may need to be added to any frame for a certain security classification. This can be especially useful in cases where the requesting entity must believe that a particular (fictional) worldly object/event existed within the requested video file.

A frame or set of frames can be added to a video file by creating a different kind of video object which will associate these fictional frames/images with the number of the frame that they must be inserted after (0 for the start of the video file). This video object will also include an associated classification whereby these frame additions will occur for requesting entities with this security classification. The fictional frames will be stored in the video repository as images and the *frame insertion video object* will be written to the video database (together with the path of the fictional frames within the video repository) in the cases where the video files are not pre-calculated for the associated security classification.

A set of worldly objects can be added to a video file by embedding them into a separate image and associating this image with an *image insertion video object*. This video object will include the number(s) of the frame(s) and coordinates within these frames where the image will be inserted i.e. it will replace a part of the frame's area with the area covered by that image (unless only certain parts of the image must replace the existing area, such as the parts of the image that are not a specific uniform colour). A classification will then be associated with the image insertion video object. As before, the SMI requires the image containing the additional worldly objects to be stored in the video repository and the image insertion video object to be written to the video database if the video files are not pre-calculated for the associated security classification.

One method of automatically creating and adding frames or worldly objects to a set of frames is to use tweening amid two frames; these two existing frames will act as the key frames. If a certain number of frames must be added, they will be generated by gradually transforming all the objects in these frames from the first key frame to the second key frame i.e. the frames that must be inserted are the tweens that will be created between these key frames. If only a worldly object must be added, an entity must manually select the worldly object in any two frames, after which the tweening technique will used to transform the selected object from the first frame to the second (assuming that at least one frame exist between the first and second selected frame).

This technique can only be used to add frames if the all the objects within the frames must be equally transformed from the first key frame to the second throughout the generated tweens. Similarly, this technique can only be used to add a worldly object between two selected key frames if the object is to be equally transformed from the first selected frame to the second. A special *tweening video object* is used to associate this information with a classification for both cases; this object must also be written to the database if the files are not pre-calculated.

Another method of automatically adding a worldly object to a number of frames if requested by an entity with a specific security classification is to simulate the object's movement throughout these frames. A worldly object must be manually selected in the first frame of the desired range, and a formula must be provided which will be used to calculate the path the object will follow in the given number of frames. This method differs from the script approach in that a worldly object is added to the following *x* frames using this method, while an object is removed and replaced with the corresponding area of a background image for the following *x* frames when the script approach is used. A *simulation video object* can be used to associate the formulas, selected regions, and number of frames together with a classification (to be recorded in the database if the files are not pre-calculated).

8.4 Suspicion

Suspicion should not be a big issue regarding the animation part of a video file if the basic frame editing model is used to restrict access to the file, as long as a 'clean cut' is made each time a frame or set of frames must be removed. In other words, the removal of the frames should be done in such a way that a smooth transition takes place from the frame before the deletion, to the frame after the

deletion occurs. Although the removal will generally be easier to hide from the naked eye, it should also (if possible) be hidden from slow motion video players or video editing software that is used to view one frame at a time.

Reducing the level of suspicion for the manual approach and its extensions is similar to the problem of reducing suspicion for the secure image database model. The fact that a part of a frame/number of frames must be replaced with the corresponding area of background image means that the resulting frame(s) could end up looking more suspicious than if the entire frame was 'cleanly' removed, especially if a poor background image was chosen. These problems regarding suspicion can be solved or at least improved upon using the same techniques discussed in chapter 7 (for each frame), although a higher fps rate should at least help reduce suspicion further when the video file is viewed by the naked eye. As already mentioned, a different background image does not normally need to be used for every frame; a good background image should in most cases be valid for a number of consecutive frames due to the frame similarity usually encountered within frames of the same scene/cut.

Adding a frame or set of frames in place of a removed number of frames can also sometimes reduce the level of suspicion by 'bridging the gap' that is left over from the missing frames and forming a smooth transition with an appropriate substitute. As with the secure image database model, adding worldly objects can make the resulting video file more believable (especially when used to solve the overlapping image object problem). The entity creating the objects should always 'preview' the assembled video file for each security classification when suspicion is a concern in order to ensure that the current configuration of the security objects minimises the level of suspicion, and should also bear in mind that the requesting entity may already have a vague idea of what the end product should look like.

8.5 Security issues

The security requirements for all four of the secure video database approaches discussed in this chapter are very similar to the requirements for the secure image database model. The requirements for each of the three phases as specified in chapter 7 regarding images must also be ensured for video files, especially concerning the secure communication between the model components. Aside from the basic general security requirements that must always be ensured (such as proper identification and authentication), other general security requirements

depend on the sensitivity of the video data and the configuration of the computer system(s) involved in implementing the secure video database approaches.

Several concurrent requests for video files can be very processor intensive, especially if high security requirements are set at each component of the model. For example, encrypting and decrypting each relevant pre-calculated video file at every request may not be feasible, even for systems containing sensitive multimedia data (the RDBMS and video repository must however be very secure). This must be kept in mind when determining the general security requirements.

8.6 Conclusion

The four approaches discussed in this chapter should allow an entity to restrict access to a digital video file at the frame or content level depending on the requesting entity's security classification. The frame editing model is the most efficient in creating/managing video objects and assembling the video file, but is also not very dynamic since logical access control can only be imposed at the frame level. The manual approach is the most accurate and dynamic since access to specific regions of a frame can be restricted according to an entity's security classification; this approach on its own can however be very time consuming when creating and managing security objects. The script approach together with the manual approach can be used to reduce the amount of time needed to select and associate worldly objects that change in position from one frame to the next, provided that the object does not change shape. The robot vision approach can be used to automatically select and associate a worldly object throughout a frame range even when it does change shape; the down side is that this approach is usually very processor intensive.

The basic frame editing model and manual approach can both be implemented on the same computer system to provide logical access control at the *frame and content level*. In order to combine these approaches, the database interface, RDBMS, and SMI must be able to tell the difference between the two types of video objects created and deal with the security information and video files as required by the relevant approach. It may also be the case that only a segment of a video file is needed. This should be possible with all approaches by simply only assembling the frames within the requested range instead of the entire video file.