A Cryptographic Protection Scheme for an Audio Server

A.G. Fraser
S. Keshav
A.M. Odlyzko

AT&T Bell Laboratories
600 Mountain Avenue, Murray Hill, NJ 07974, USA

Abstract

We present a scheme for cryptographically protecting music data. The scheme uses public key cryptography to minimize the number of secrets and the amount of interaction needed with a registration authority. We outline the design for a player. Our scheme can be used for securely distributing all digital information.

1. Introduction

The technology to build an audioserver capable of serving thousands of customers is at hand. However, once digitized music data is made public, it can be copied at will, without regard to copyrights. Clearly, music data must be safeguarded before such a server can be commercially successful.

We assume that we can build a device that is capable of public key encryption, public key decryption, stream decryption of music, and playback. We also expect this device to store a secret that cannot be determined by opening up the device and observing its circuits. Given these constraints, we develop a set of crypto schemes that can assure a music label that

a Music sold to a user cannot be resold or usefully copied.

b Untrusted music vendors can distribute music.

c Few transactions are involved in setting up the secure music service.

Only the essential features of our scheme are presented here. A full implementation would be more elaborate, with expiration dates for certificates, procedures for dealing with various attacks, and so on.

2. Overview

In our scheme, each music title is encrypted with its own secret key ('locked'). The music can be broadcast at will, since it cannot be played unless its secret key is known. A consumer can obtain music from any medium, but this music is 'locked'. To play the music, the consumer must purchase an *information key*. A credit-card sized *Personality Module* can be thought of as an *information keyring* that stores information keys. A consumer can purchase as many keyrings as he or she desires; the more the keyrings, the more the number of players that can be simultaneously active. Since each Personality Module has to be authenticated, and cannot be duplicated by the user, the music industry can limit the number of copies that a user could puchase.

The scheme involves two steps. First, the user's Personality Module and a secure module provided by the music producer, and operated by a vendor who collects payments (who does not have to be trusted) authenticate each other by exchanging certificates issued by a common trusted authority. Second, the vendor's secure module sends to the buyer's Personality Module the music title's secret key, so that the

buyer can play the music. The transmission of the titles' secret key is encrypted, so there is no need for physical contact, and the entire transaction can be done over a data link.

The scheme makes extensive use of physically secure hardware because we believe that the only way to really keep a secret is to place it in a physically secure device. Tamper-resistant, high-performance processors can be mass-produced at reasonable cost.

3. Scheme

We use the following notation - an entity <e> has a public key $Pu<e>$, a private key $Pr<e>$ and a secret key $<e>$. Plaintext p encrypted by key k is denoted $k(p)$.

The following entities take part in the scheme:

- Players, denoted P, that are hardware devices that play music. Players contain a Personality Module that stores a secret key.
- Manufacturers, denoted M, that manufacture personality modules.
- Labels, denoted L, that publish music, such as Columbia, or Time-Warner.
- Certifying authorities, denoted A, that are willing to certify the authenticity of music labels and personality modules.
- Vendors, denoted V, that distribute music. Vendors have a *certification module* per-label.
- Users, denoted U, that purchase music from Vendors.
- Titles, denoted T, the music being sold.

	Authority	**Label**	**Vendor**	**User**	**Player**
	A	L	V	U	P
Keys	*PrA*, *PuA*	*PrL*, *PuL*, L_T, *PuA*			*P*, *PrA*(*PX*) *PuA*, ID,
Step 1	*PrA*(*PuL*) → L	*PuL* → A			
Step 2		Certification Module with (*L*, *PrL*, counter, *PuA*, *X*,*PrA*(*PuL*)) → V			
Step 3		For title T, ($L_T(T)$, $L(L_T)$) → V			
Step 4			Sell U title *PrA*(*PuL*) → U	money → V *PrA*(*PuL*) → P	
Step 5				*PuL*(*P*, *PrA*(*PX*)) → V	*PuL*(*P*, *PrA*(*PX*)) → U
Step 6			($P(L_T)$, $L_T(T)$) → U counter--	($P(L_T)$, $L_T(T)$) → P	Store ($P(L_T)$, $L_T(T)$) Extract L_T and use it to play music
Step 7					Label T with player identification
Step 8		Audit V's counter Request → V	*L*(*counter*) → L		

1. The Label contacts the certifying authority with *PuL*, and receives a certification from the authority *PrA*(*PuL*).

2. The Label gives a vendor a *secure* hardware device called a *Certification Module* that contains *PrL* (the label's private key), *L* (a secret key owned by the label), *PuA*, A's public key, *PrA*(*PuL*), the certificate the label obtained from the authority, *X*, a long fixed publicly known string, and a counter initialized to the number of copies the vendor is allowed to sell. Secure means that the vendor cannot discover these keys in any way. The device is assumed to be capable of single and public key encryption and decryption.

3. L encrypts each title T with a per-title secret key $L_T(T)$. L sends out to each Certification Module the encrypted key $L(L_T)$ and the encrypted music $L_T(T)$, or else places both in some database easily accessible by the Certification Module.

4. A player's personality module can do public key encryption and decryption, single key decryption and audio playback. The personality module stores a secret key *P*, *PuA*, and a unique ID. It also stores a certification of *P*,*PrA*(*PX*). A manufacturer must obtain certification for each personality module from the certification authority.

A user buys music from the vendor by paying some money, and in return, getting *PrA*(*PuL*). It passes this on to the player's personality module which uses *PuA* to extract *PuL*. If the vendor is not certified, this step

will fail.

5. The personality module uses *PuL* to encrypt *P* and its certificate *PrA*(*PX*). The user passes these on to the vendor.

6. The certification module in the vendor uses *PrL* to extract *P*, and *PrA*(*PX*). It verifies that the personality module is certified using the certificate and *PuA*. If this checks out, it sends the user ($P(L_T)$, $L_T(T)$), and decrements the counter. The user passes on ($P(L_T)$, $L_T(T)$) to the personality module, which extracts L_T with *P* and then uses L_T to extract T.

7. The personality module adds its ID to T by modifying T. This allows external checkers to know which player extracted T. The ID is not a secret, but its location in T might be.

8. When the label wants to audit a vendor, it sends a request to the vendor, and the certification module returns the current number of copies sold. Since this is encrypted with *L*, V cannot tamper with this count. When the counter counts to zero, the certification module stops working until the label sends a reauthorization message (encrypted with L to prevent fraud).

4. "Proof" of correctness

We must assume that the certification module given to a vendor and the personality module are secure. Thus, *PrL*, *L*, and *P* remain secrets.

First, note that the only information given to V is *PrA*(*PuL*), $L_T(T)$, and $L(L_T)$. Since V does not know L, it cannot extract $L(L_T)$. The only information the V can extract is *PuL*, but this is public anyway. If $L_T(T)$ is not considered public knowledge, the vendor could conceivably sell $L_T(T)$ to other vendors for profit. This can be discouraged by having the music encrypted by L_T^V, that is, a per-title, per-vendor key. No other vendor could use this music.

Second, the user only sees *PuL*(*P*), $P(L_T)$, and $L_T(T)$. Since *P* and L_T are secrets, the user cannot know T.

Third, the player gets *PuL* encrypted by *PrA*, so it cannot be spoofed into giving up *P* unless a label itself is compromised.

Fourth, the certification module sees both *P* and *PrA*(*PX*). Thus, it knows that the personality module has been certified by an authority. We must use *PrA*(*PX*) as a certificate instead of just *PrA*(*P*), else a spoofing personality module could send $PuL(Pu(Y), Y)$, for a randomly chosen *Y*.

5. Player

We suggest that the player be built in three parts - a) a chassis that provides power, display etc., b) a portable storage unit and c) a personality module that contains a chip that does public key encryption, public key decryption, stream decryption, stores *P* and *PuA*, and decodes music. The reason for this organization is as follows:

> We envisage that a consumer would own some number of personality modules, or PMs. Each card has the same *P*. Thus, given a copy of encrypted music $L_T(T)$ and $P(L_T)$, user can simultaneously play them on as many players as the PMs he or she owns. A user may have a PM to be used in a car stereo, and some more in the house. A user may buy several PMs and give them to friends or relatives, so that they could share music. If a PM is priced at around USD 50-100, this would allow reasonable sharing of music without any excesses.
>
> Each PM could be engraved with a unique serial number. The manufacturer could keep a secret database of serial number to *P*. Consumers could register their PM(s) with a registration card at the time

of purchase. Thus, in the event of a loss, the PM could be regenerated. Alternately, an authorized person could create a duplicate PM from a working PM by accessing *P* as described above. If a user had two PMs to start with (given to them free, perhaps), if one of them was accidentally lost or destroyed, a new one could be made from the surviving copy.

The storage unit (like a removable hard disk) would allow a user to carry music when visiting friends or traveling elsewhere. By plugging in a storage unit and the PM, music could be played in any player. Storage units should be capable of rough handling, and possibly the size of a credit card.

In order to buy music in person, the user would carry a PM, plug the PM into a reader which contains a certification module, and obtain $P(L_T)$, $L_T(T)$ written on a portable storage medium. Thus, music vending machines are possible.

6. Uses of our scheme

The scheme allows retail distribution of any digital asset, including software, music, art, books, etc. Since the title is encrypted with L_T and can be played only when $P(L_T)$ is purchased, our scheme allows *secure broadcast*, which is critical for cable, radio, or satellite distribution. Thus, we can envisage secure cable distribution of software (such as Windows 95), music, and even magazines or newspapers.

7. Identifying decoded data

While decoded music might be in the analog domain and thus of poorer quality, or so bulky that it is not worth reselling, our scheme is appropriate for all digital information. Thus, we want to ensure that decoded data leaving a PM cannot be misused by a user. We propose to do this by indelibly marking decoded information with the ID of a personality module. Thus, a user who resells decoded information could be traced and fined accordingly.

At the moment, we do not have concrete proposals for indelibly marking decoded data. It is clear, however, that the marks in the decoded stream must be per-PM, thus the PM must create them at the time of purchase. We also realize that the PM's ID should not modify the intended behavior of the data stream. For example, if the decoded data stream is to be interpreted as software (binary application code), we do not wish the ID to be misinterpreted as program instructions- instead, the consumer of the decoded information must be aware that some *meta – data* (that is, the PM's ID) is intermingled with the data.

Each form of decoded information might intermingle meta-data in a unique way. To allow for this, we propose that the decoded data contain (in some well known location) a marker describing its type, for example PAC_AUDIO, or MICROSOFT_APPLICATION, or ADDISON_WESLEY_BOOK. The PM would have type-specific routines to insert its ID in the decoded data stream. The consumer of decoded information would have type-specific functions to distinguish between meta-data and data, as also to reveal the ID when queried by an authorized agent.

The label would create information which allows some meta-data to be indelibly added. They also create routines for the PM to add its ID to the decoded information. Finally, they create consumers of decoded information that can distinguish between data and meta-data. With these components, we can mark decoded data indelibly, with type-specific schemes appropriate to the application. This would prevent wide-scale misuse of decoded data.

8. Agencies

It may be inconvenient for a vendor to have certification modules for every label, particularly when handling titles from small or foreign labels. Thus, we expect that there will be agencies that act on behalf of a group of labels, issue a single certification module on behalf of the group, and collect and distribute royalties appropriately.