Digital Rights Management in a 3G Mobile Phone and Beyond
Thomas S. Messerges       Ezzat A. Dabbish
October 2003

Presented by Mike Jones & Alexander Andreasson
Contents

- Introduction & Background
- DRM
  - Concepts & strategies
  - Open Mobile Alliance
- Proposed DRM System
- Security Issues
- Family Domain
- Example Use Cases
- Closing Remarks
Background

- Emergence of high speed mobile phone data transmission
  - 3G speeds vary from 144 Kbps up to 2Mbps
- Business opportunities involving digital content on mobile phones
  - Games, applications, videos, music, etc.
- Digital content can be copied and distributed at very little cost
  - P2P sharing facilitated by phone applications
  - Bluetooth enables short range file sharing at speeds of 723 Kbps.
- $12 Billion loss due to piracy in 1999
Digital Rights Management in a 3G Mobile Phone

- Copyright protection of digital items can be securely managed in 3G mobile phones and other devices.
- DRM will be essential for the future of mobile phones.
- DRM uses access control technologies to restrict the usage, modification and distribution of copyrighted digital content.
Digital Rights Management - Fundamentals

- Content providers need to define and organise rights
- Distributors need to package and sell these rights to consumers
- Payment brokers need to reconcile billing
- Client devices need to render content whilst enforcing rights
Digital Rights Management - Fundamentals

- Content is protected with cryptographic techniques to ensure it’s authenticity, integrity and confidentiality.
- The presence of DRM will be transparent to users, except for cases when a user tries to render content without a proper license.
- Digital items cannot be copied unless authorised to do so.
- Content can only be rendered based on the appropriate rights.
- A fee must be paid whenever it is required.
Persistent content example

- The license is digitally signed to enable its integrity and authenticity.
DRM Process

1. The trusted rendering software will present the protected content and corresponding license to the DRM services software.
2. The DRM services will verify the signature of the license.
3. Verify the hash of the content.
4. Decrypt the content, and then send the decrypted content back to the rendering software.
5. The rendering software will then be able to play the music, show the video, or run the game, depending on the type of content.
Open Mobile Alliance DRM

- Defines industry-wide specifications for applications that operate over wireless communication networks.
- Enable the controlled consumption of digital media objects.
- Consumer-friendly DRM standard
  - Maximizes interoperability, while minimizing complexity.
- OMA advocates that content files be distributed to other devices, but licenses to use this content must be obtained from a server called the rights issuer.
DRM Approaches

1. Replace the I/O elements of the OS with modules that monitor all requests for I/O operations.

2. Hyperadvisor - Situated between the OS and the hardware.
   ○ Invokes DRM software

3. OS is extended to support DRM functionality.
Proposed solution

Applications (e.g., music or video player, game)

System Services API

Extended Services API

Generic OS

- Process Manager
- Memory Manager
- Network Manager
- File Manager

DRM/Security Extensions

- DRM Manager
- Trusted Application Agents
- Security Agents

Hardware

Security Hardware
Proposed solution

- OS is extended with DRM and security capabilities by adding a DRM manager, trusted application agents, and security agents and hardware.
- Only applications that access DRM-protected content need to be aware of the new DRM extensions.
- The application can use the DRM extensions to open the file and render the data. The application will access these extended system services through an Application Programming Interface (API).
DRM Manager

- Responsible for the core DRM functions.
- Works with security agents to authenticate licenses and content.
- Parse and enforce usage rules.
- Access a secure DRM database.
- Provide decrypted content to a trusted application agent.
Authenticate Licenses and Content

- Verify the integrity and authenticity of the corresponding license file.
- Cryptographic hash of the license file is computed and the digital signature is verified.
- The DRM manager will enlist the aid of security agents when performing this and all other cryptographic operations.
- The result will determine whether the license and content files originate from valid sources.
Enforce Rights

Once decrypted:

- Actions can be associated with three fundamental types of rights: render rights, transport rights, and derivative work rights.
- License can stipulate an additional event for performing an action. E.g. payment.
- DRM manager will need to use a secure database to track these Events.
- Uses device’s credentials - key/certificate manager
Decrypt Content

- Since a top-level application is not part of the trusted OS layer, it will usually not be allowed direct access to the decrypted content.

- DRM manager routes decrypted content directly to a trusted application.
DRM Manager Example

Top-Level Application Steps

1. Authenticate license and content
2. Use credentials and request an action (e.g., play, copy)
3. Control the action

Trusted DRM Manager

Authenticate
Pass
Enforce Rules
Pass
Decrypt

Event

Trusted Agent

Secure Database
3. Proposed DRM System

3.2 - Trusted Application Agents

3.3 - Security Agents

3.4 - DRM credentials
3.2 - Trusted Application Agents

- Part of extended OS
- Access & manipulation of decrypted content
- “trusted” because they are part of the privileged OS layer

Organized into:
- Rendering Agents
- Transport Agents
- Derivative Work Agents
3.2 - Rendering Agents

- Provide applications the ability to render DRM-protected content
- Low-level drivers that convert digital data to format consumed by user

- Agents are *trusted* to properly handle decrypted digital content
- Execution of DRM-protected software application is categorized as a rendering operation
- App. Loader enforces usage rules prior to executing a previously installed application
  - Rights and privileges of an app are enforced while app’s running
3.2 - Transport Agents (TA)

- Provide services that move content from one location to another (e.g. email attach.)
  1. DRM manager ensures usage rules are enforced
  2. TA is invoked to start transfer
     a. Possible Secure Authenticated Channel (SAC) with receiving device
        i. TA enlists a security agent to complete protocol using Transport Layer Security (TLS) or Wireless TLS (WTLS)

- May need to handle decrypted content and must be trusted

  Bluetooth connection to headphones and phone

  1. TA receives decoded and decrypted audio from trusted mp3 decoder app. agent
  2. TA encrypts audio and routes it to bluetooth

  **hardware -> encrypted data -> headphone**
3.2 - Derivative Work Agents

- Extract and transform protected content into different form
  
  - A copy of a digital item might have different rights than the original

Option 1. Ensures that the copy of the digital item’s license is updated appropriately

Option 2. Contact a server to obtain new license for the copy
3.2 - Trusted Application Agents Example

1. Derivative work agent to decrypt and load the new ringtone
2. Secure link agent for establishing communication between headphone and phone
3. Trusted ringtone player agent is used to access and play the ringtone

Top-level application uses trusted application agents to install a new ringtone
3.3 - Security Agents

- Handle the security-related functions that are commonly needed in all DRM systems
- Work closely with available security hardware which can enhance DRM system security.

Secure memory and file management

Cryptographic operations

Key management
3.3 - Secure Memory and File Management

- A DRM system needs to ensure that access to memory and files can be controlled.
- E.g. An installed ringtone cannot be accessed by any application but the ringtone player.

3 security functions:

1. Access-controlled file system - stores digital content
2. Memory separation system - unused operations don’t eavesdrop on used memory while trusted operation is running.
3. Secure memory system - critical data is not leaked
3.3 - Access-controlled file system

- E.g. A java game - inefficient to keep the DRM protected app. encrypted.
  - Thus, decrypt and store securely.
  - Access-controlled file system can be used where a protected app. can be decrypted and safely stored in files that only trusted agents can access

Requirements

1. Files are assigned ownership attributes that specify which trusted agents can access the files
2. Files are optionally encrypted
3. Tampering of the ownership attributes can be detected

Proposed Solution:

A trusted security agent can work with the phone’s onboard memory to maintain access control. However, memory separation between tasks needs to be maintained.
3.3 - Memory separation system

- Guarantee task separation

**Proposed Solution**

A memory separation manager is responsible for maintaining the separation of tasks

**Functionality:**
- When a task is run, the memory manager can configure a hardware monitor to define which memory is available to the task.
- Ensure that tasks stay within their assigned memory areas and that they do not maliciously interfere with trusted operations.
3.3 - Secure memory system

- E.g.1. If a phone’s private keys were to leak out, a hacker might be able to compromise the security and extract decrypted content
- E.g.2. High-security systems can be attacked via physical probing of hardware pins inside phone

**Proposed Solution**

Prevent leakage by using secure memory on the processor’s integrated circuit

**Functionality:**
- Temporarily hold date unlike the secure database
- Memory is cleared if suspicious events like attempts to enter debug mode are detected
3.3 - Cryptographic Operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hash of a license (5KByte)</td>
<td>SHA1: 3 ms</td>
</tr>
<tr>
<td>Verify license signature</td>
<td>RSA&lt;sup&gt;(1)&lt;/sup&gt;: 100 ms, ECC&lt;sup&gt;(2)&lt;/sup&gt;: 150 ms</td>
</tr>
<tr>
<td>Decrypt content key</td>
<td>RSA&lt;sup&gt;(1)&lt;/sup&gt;: 1,800 ms, ECC&lt;sup&gt;(2)&lt;/sup&gt;: 90 ms</td>
</tr>
<tr>
<td>Decrypt content (2 Kbyte)</td>
<td>AES&lt;sup&gt;(3)&lt;/sup&gt;: 1.6 ms</td>
</tr>
</tbody>
</table>

- Protected content encrypted using symmetric-key algorithm such as AES (SW/HW)
- Binding content and licenses use hash algorithm such as SHA1 (NSA - 20 byte)
- Content key decryption, signature verification, signature generation use **RSA**
  - RSA = encrypt/decrypt messages.
  - **Asymmetric**, 1 public and 1 private key.
- **Need** fast operations.
- **ECC** is 20x faster than **RSA** for decrypting content keys
- **Optimal** performance =
  - ECC = decrypt content key
  - RSA = verify signature of license.

16MHz ARM7 microprocessor
3.3 - Key/Certificate Manager (KCM)

- Software module responsible for securely handling a database of the phone’s credentials = private keys, public keys, certificates, and identification numbers
- E.g. A phone contains a root certificate
  - KCM controls phone’s private keys
  - Keys should be used by trusted OS components
  - Keys are decrypted into the secure memory and cleared if tampered
- DRM system is configured, KCM installs new key or certificate
- E.g. 1 - Device = DRM system = install new private key and public-key certificate
- E.g 2 - Subscription service expires = private key and public-key certificate = delete
3.4 - DRM credentials

- **Maintain** keys and certificates - access to **protected** content and trust.
- Phone - reliable device and user creds.
- **Trusted security agents** manage these elements and **ensure** private key remains secret.
- **DRM** system needs root key(s) to check **authenticity** and **integrity** of other device(s) creds.

- unit private key (KuPri) and unit certificate (UnitCert),
- **DRM** private key (KdPri) and **DRM** certificate (DRMCert)
3.4 - Serial and Model Numbers (SN&MN)

- **SN** = is an unchangeable number that unambiguously identifies the phone
  - Only a device with a certain SN has rights to render a digital item
  - **DRM** manager -> **SN** matches **SN** in the license

- **MN** = unambiguously identifies the hardware and software version of a phone
  - Indicate phone and security capabilities
    - Phone = Hardware or Software support for security?
  - Content providers -> package digital content for **X** phone.
3.4 - Private Keys and Certificates

- **KuPri** and **UnitCert** should be used for establishing secure-authenticated channels to a phone
  - E.g. phone receives streamed content
- **KuPri** and **UnitCert** would form the **foundation** of trust for a **DRM-enabled** phone
  - E.g. installed by manufacturer
- E.g. Digital item encrypted with content key, which is encrypted by the public key.
  - **Result:** device with **KdPri** can **decrypt** the **content key**, which decrypts digital item.
3.4 - Short-lived certificates

• **History**
  ○ 1995, Macq and Quisquater described a system where a trusted authority grants “entitlements” to enforce access to a digital TV system

• **Proposed System**
  ○ certificates act as the “entitlement”
  ○ expiration dates that need to be authenticated prior to allowing access to critical DRM operations

• **No action** to revoke device access since **device** would stop working in time.

• **Trusted** source for **time** and **date** information is also needed if short-lived certificates are used
Section 4 - Security Issues

4.0 - Components work to provide secure DRM solution

4.1 - License

4.2 - Integrity and authenticity

4.3 - Rights enforcement

4.4 - Content Protection

4.5 - Privacy Issues
4.1 - License

- **Rights** are assigned to a digital item **using a license**, which is an **unambiguous, machine-readable** document that describes how a piece of content may be used.

1. Value that links the license to the digital item
2. Rights allowed for that digital item
3. Means to decrypt the digital item
4. Signature of the license
4.2 - Integrity and authenticity (IA)

- DRM manager ensures IA of license
- Application agents ensure IA of trusted device(s) - bluetooth headphone
- Phone prove authenticity to other devices and the DRM infrastructure (content provider)
  - IA = public key infrastructure (PKI) or shared secret
- E.g. phone verifies license signature using root key in hardware
  - RSA for signatures
  - AES for symmetric encryption
  - SHA-1 for hashing
- Phone verify signatures w/ PK-encryption and decrypt content key w/ PK-decryption
- Content providers **TRUST** DRM system in phone to keep keys secret
  - Rogue phone if key isn’t secret
4.3 - Rights enforcement

- **DRM manager enforces usage rules**
- DRM manager parses the license file and recognizes and processes the different rights expressions
  - If DRM manager finds conflict or cannot understand = fail safely

- **Fool the DRM manager by giving older or newer versions of licenses.**
  - Thus, it needs to be able to recognize the version of the license file.
- Should be designed to be **backwards compatible**, so that old licenses can be properly interpreted.
4.4 - Content Protection

- Content is protected with encryption up until the time it is rendered or installed into the access-controlled file system
  - Streamed to a phone from a remote server or it can be stored locally
- Content will be decrypted and routed to the appropriate rendering hardware by trusted agents
- Before a trusted agent can start decrypting content, it needs to obtain the decryption key CEK
  - Locally = CEK encrypted with KdPub in license
  - Streamed = CEK session key that is negotiated with the server during the establishment of a Secure Authenticated Channel (SAC)
4.5 - Privacy Issues

1. User information used to create a content license must not be disclosed without the explicit consent of the end user.

2. The user’s identity must not be disclosed to a content provider and/or to other parties without the explicit consent of the end user.
Section 5 - Family Domain

5.0 - General

5.1 - Device Configuration

5.2 - Family Domain Example
5.0 - General

- Users need to be assured that their rights will also be protected with DRM systems
- Users want to use content on any of their devices
- Solution = family domain

1. User chooses which devices belong to his/her domain and trusted server called Domain Authority (DA) who installs DRM key in each device
2. Access to all contents in the domain
3. No access from outside of the domain
4. Register with DA once
5. User deals with security only when (new) device is added or removed
5.1 - Device Configuration

- A device that joins domain is registered
- A device that leaves cancels its registration - unregisters.
- **There’s a catch:**
  - DA has policies by limiting #devices in a domain and #times a device can join/leave
- **There’s a good side:**
  - Fraud detection by tracking devices joining/leaving
  - See excessive activity = device abuse
  - Add/remove devices using **password**
  - **Share** family domain passwords and non-family to a domain
5.2 - Family Domain Example

1. Consumer contact content provider and buys a song for **domain**
2. Purchase transaction protocol = buying content locked to a single device
3. DRM certificate = domain certificate w/public key
4. Content provider encrypts content key (CEK) = any device with same KdPri have same rights to content

**Fundamental Idea:**

Any device in the domain that receives content will be able to render it. The main difference is that the license will lock the content to a domain rather than a device
6.1 - Enrollment of Device into Family Domain Example

1. Consumer activates the domain enrollment application, which initiates contact with the DA.
2. The phone and DA establish a SAC and the device identifies itself to the DA.
3. The consumer indicates whether he wants to form a new domain or add the device to an existing domain.
4. The DA sends a new or an existing (in the case of joining an existing domain) KdPri and DRMCert to the phone.
5. The phone securely installs the KdPri into its access controlled database.
Section 7 - Closing Remarks

- Proposed for a mobile phone environment
- Family Domain concept to other possible devices (Phone, Laptop)
Questions?