

# DReaM-CAS: Fully Open Conditional Access System



Vishy Swaminathan  
Santa Clara, California  
March 15-16, 2006

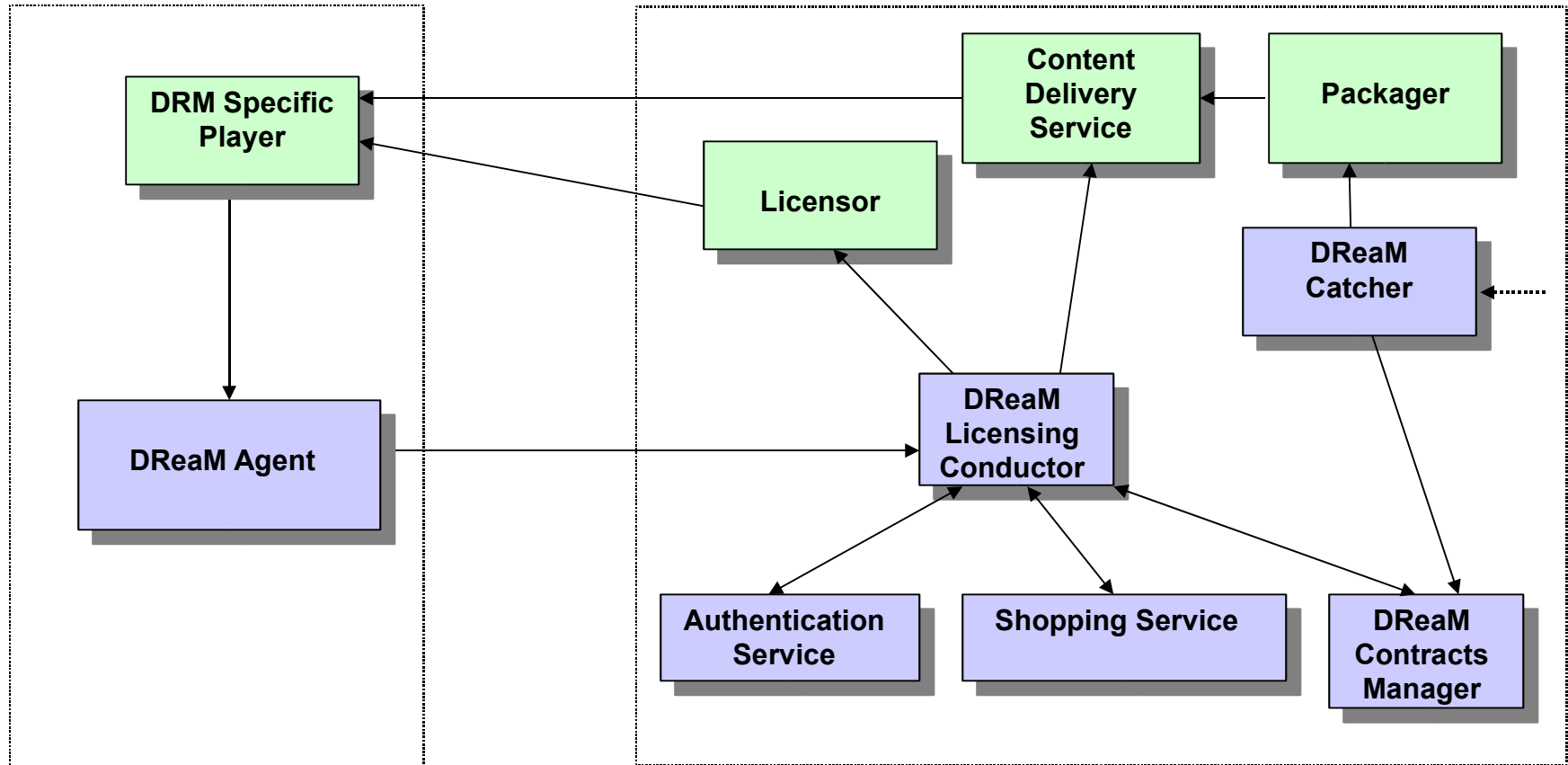
# Agenda

- Introduction
- Scope and Requirements
- Design Premise
- Key Features
- Technical Details
- ECM/EMM details
- Summary

# OMC's Project DReaM

- Supports Open Media Commons (OMC) vision of open source and royalty-free intellectual property
- Leverage Liberty Alliance to promote Identity & Role based licensing models, rather than device based licensing only
- Addresses both Conditional Access (CA) and Digital Rights Management (DRM) system models
- Addresses customer immediate multi-system interoperability requirements
- Independent of any specific content type
- A horizontal solution platform capable of supporting: Business, Content and Life vertical markets
- Exploration of RMS policy options beyond encryption (watermarking & tracking)

# DReaM Components



DReaM Client Side

DReaM Service Side

# Major Components of DReaM

- Content Catcher/Ingest (**DReaM Catcher**)
  - Import digital content
  - Import business rules related to content
- Content Packaging (**DReaM Packager**)
  - Content encrypted (AES encryption)
  - Key(s) for encrypted data are produced and then posted to Secure License Server (**DReaM Licensor**) Database
  - Protected content is posted to content delivery server
- Content Rights Server/Repository (**DReaM Contracts Manager**)
  - Repository of business rules
  - Business rules are interpreted here
- Content Distribution Server
  - Support for distribution of protected data
  - Optionally, protected keys are delivered “in-band” with protected content
- License Service
  - a) Primary point-of-contact with clients (**DReaM Licensing Conductor**)
    - High transaction rate
    - Application Server likely
  - b) Communicates with **DReaM Contracts Manager** to generate "offer" of price and terms
  - c) Authentication (user, device, SIM, Java Card, Liberty single sign-on, etc) (Sun Access Manager)
  - d) Payment & Fulfillment (e-Commerce) (BSS, OSS-J)
  - e) Generates license (**DReaM Licensor**)
    - Data keys protected for transport using (eg. ECC)
  - f) Deliver license (**DReaM Licensor**)
    - Communicates with Content Distribution Server for delivery of "in-band" license
    - "Out-of-band" licenses are delivered to users directly.
- Client side (**DReaM Agent**)
  - Ability to negotiate for licenses
  - Supports CA and DRM models
  - Secure key repository
- DRM specific player

# 3 Key Elements of DReaM

- Digital Rights Management Service (DReaM-MMI)
  - > Ability to manage rights for any type of content in various usage models
- Conditional Access (DReaM-CAS)
  - > Ability to deliver timeline dependent content to multiple consumers (IPTV, telemetry, surveillance)
  - > Includes content protection and distribution technologies
- Disintermediation (DReaM intraOPERAbility Framework)
  - > Separation of back-office services from players/consumption
    - > Authentication, Billing, Entitlement, Rights Management, Subscriber Services, Portal, Content Delivery (sometime proprietary)
  - > Interoperability with existing content protection technologies
    - > Client-side Player/Consumption, Licensing, Packaging
  - > Emphasis on Identity based licensing over device licensing

# 3 Key Elements of DReaM

- Digital Rights Management Service (DReaM-MMI)
  - > Ability to manage rights for any type of content in various usage models
- Conditional Access (DReaM-CAS)
  - > Ability to deliver timeline dependent content to multiple consumers (IPTV, telemetry, surveillance)
  - > Includes content protection and distribution technologies
- Disintermediation (DReaM intraOPERAbility Framework)
  - > Separation of back-office services from players/consumption
    - > Authentication, Billing, Entitlement, Rights Management, Subscriber Services, Portal, Content Delivery (sometime proprietary)
  - > Interoperability with existing content protection technologies
    - > Client-side Player/Consumption, Licensing, Packaging
  - > Emphasis on Identity based licensing over device licensing

# DReaM CAS

*Fully specified Conditional Access System  
for protecting Digital TV streams*

# DReaM-CAS

- Conditional Access profile for MPEG-2 TS
- Strong Crypto: AES-128 (content) and RSA (keys)
- Fully Specified Structures:
  - > EMM (Entitlement Management Messages)
  - > ECM (Entitlement Control Messages)
- End-to-End prototype implementation open sourced at [dream.dev.java.net](http://dream.dev.java.net)

# DReaM-CAS : Requirements

- Encrypted (Video) Data is sent to all clients that request it – similar to broadcast/multicast scenarios
- No complicated rights negotiations - only access to content viewing is controlled
- Pay Per View and Video-on-demand with trick-play (fast-forward, fast-reverse, etc.)
- DVR
- Imperceptible to the user.

# DReaM-CAS : Key Features

- Symmetric encryption for encrypting
  - > data stream and
  - > stream keys used to encrypt data
  - > Open Encryption Standards (AES 128)
  - > In band
- Asymmetric encryption for encrypting access keys used to encrypt stream keys
  - > Public Key/Private Key mechanisms
  - > RSA asymmetric encryption
  - > Out of band – https, PKI etc.

# DReaM-CAS – Design premise

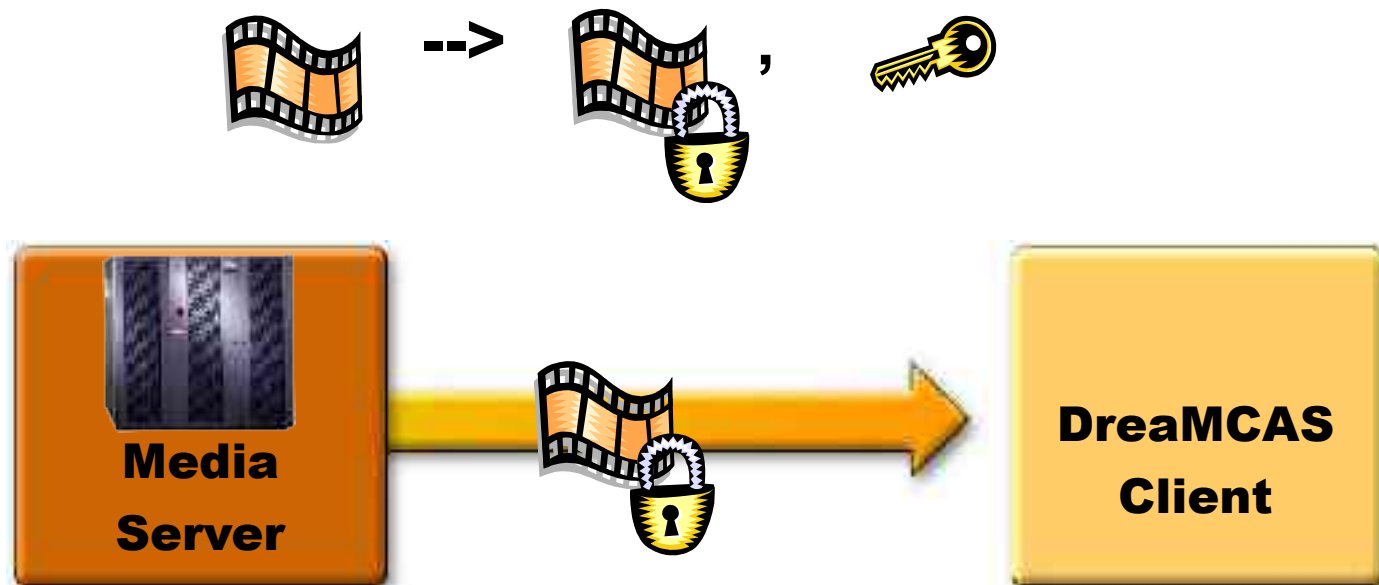
- Two way IP network connection available
- Certificates or equivalent technology for public-keys of clients.
- Http or https connection available
- Standard Public-key, private-key mechanism for protecting Access keys.
- Data Streams are protected with Stream keys. Stream keys are protected with Access keys.
- Protected Stream keys are embedded with protected data streams.

# DReaM-CAS: Details

- $\text{Content}_{\text{prot}} = \text{SymEncrypt}(\text{Content}, K_s)$
- $K' = \text{SymEncrypt}(K_s, K_a)$  SYMENC() - Symmetric Encryption  
 $K_s$  are stream keys  
 $K_a$  is access key
- $K'$  stored with  $\text{Content}_{\text{prot}}$
- $T_e = \text{AsymEncrypt}(K_a, K_{\text{pub}})$  ASYMENC() - Asymmetric Encryption  
 $K_{\text{pub}}$  is public key of client
- $K_a = \text{AsymDecrypt}(T_e, K_{\text{priv}})$   
 $K_{\text{priv}}$  is private key of client
- $K_s = \text{AsymDecrypt}(K', K_a)$
- $\text{Content} = \text{SymDecrypt}(\text{Content}_{\text{prot}}, K_s)$

# DReaM-CAS: Details

$$\text{Content}_{\text{prot}} = \text{SymEncrypt}(\text{Content}, K_s)$$

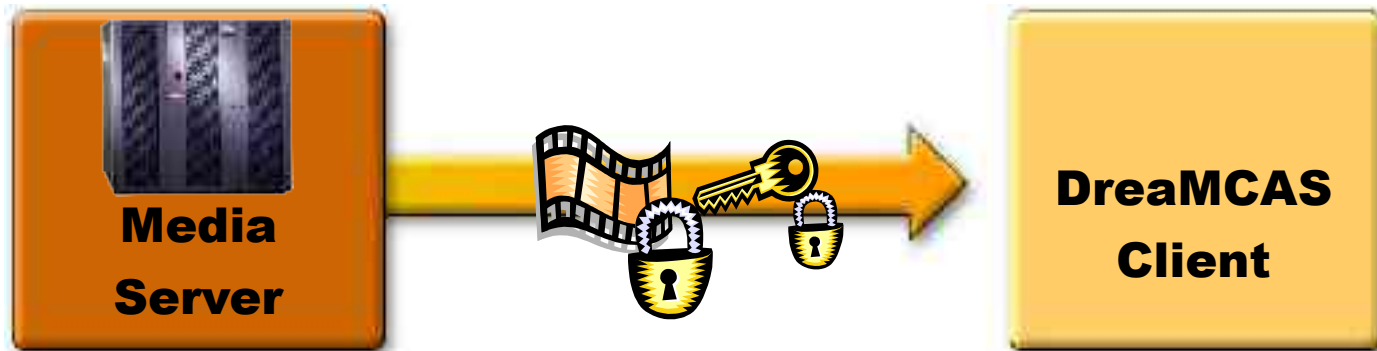


# DReaM-CAS: Details

- $\text{Content}_{\text{prot}} = \text{SymEncrypt}(\text{Content}, K_s)$
- $K' = \text{SymEncrypt}(K_s, K_a)$  SYMENC() - Symmetric Encryption  
 $K_s$  are stream keys  
 $K_a$  is access key
- $K'$  stored with  $\text{Content}_{\text{prot}}$
- $T_e = \text{AsymEncrypt}(K_a, K_{\text{pub}})$  ASYMENC() - Asymmetric Encryption  
 $K_{\text{pub}}$  is public key of client
- $K_a = \text{AsymDecrypt}(T_e, K_{\text{priv}})$   
 $K_{\text{priv}}$  is private key of client
- $K_s = \text{AsymDecrypt}(K', K_a)$
- $\text{Content} = \text{SymDecrypt}(\text{Content}_{\text{prot}}, K_s)$

# DReaM-CAS: Details

$$K' = \text{SymEncrypt}(K_s, K_a)$$

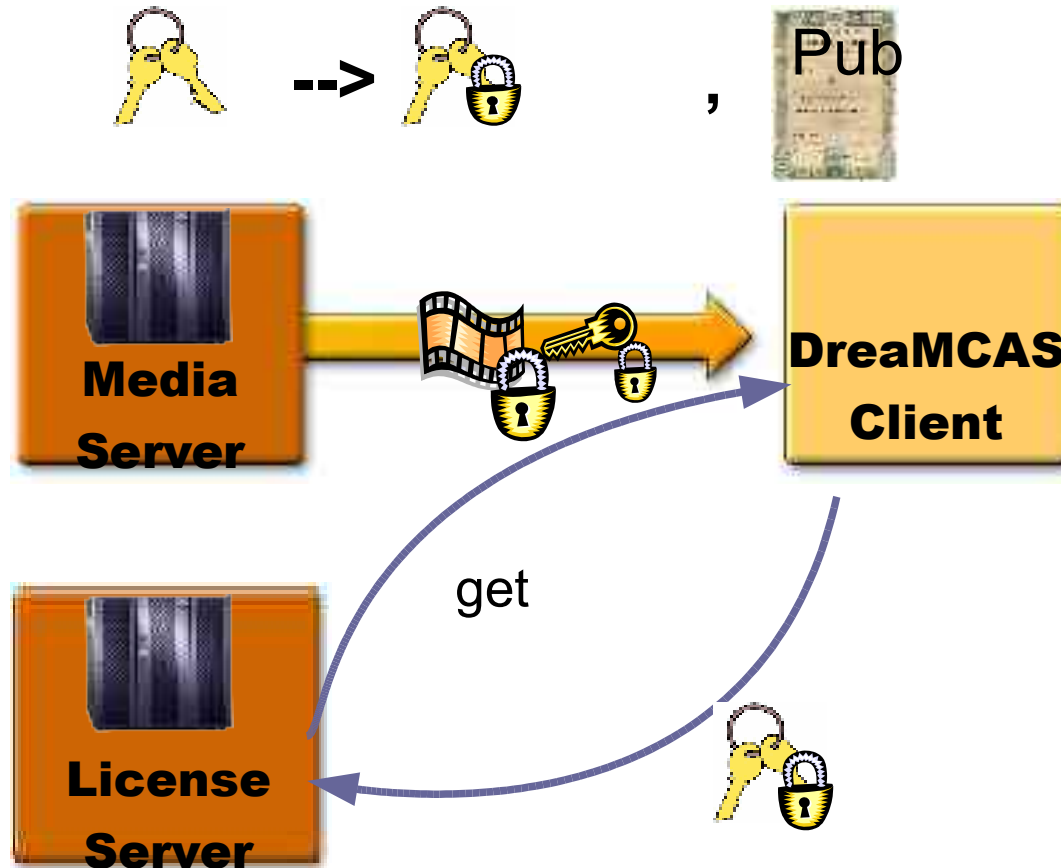


# DReaM-CAS: Details

- $\text{Content}_{\text{prot}} = \text{SymEncrypt}(\text{Content}, K_s)$
- $K' = \text{SymEncrypt}(K_s, K_a)$  SYMENC() - Symmetric Encryption  
 $K_s$  are stream keys  
 $K_a$  is access key
- $K'$  stored with  $\text{Content}_{\text{prot}}$
- $T_e = \text{AsymEncrypt}(K_a, K_{\text{pub}})$  ASYMENC() - Asymmetric Encryption  
 $K_{\text{pub}}$  is public key of client
- $K_a = \text{AsymDecrypt}(T_e, K_{\text{priv}})$   
 $K_{\text{priv}}$  is private key of client
- $K_s = \text{AsymDecrypt}(K', K_a)$
- $\text{Content} = \text{SymDecrypt}(\text{Content}_{\text{prot}}, K_s)$

# DReaM-CAS: Details

$$T_e = \text{AsymEncrypt}(K_a, K_{\text{pub}})$$

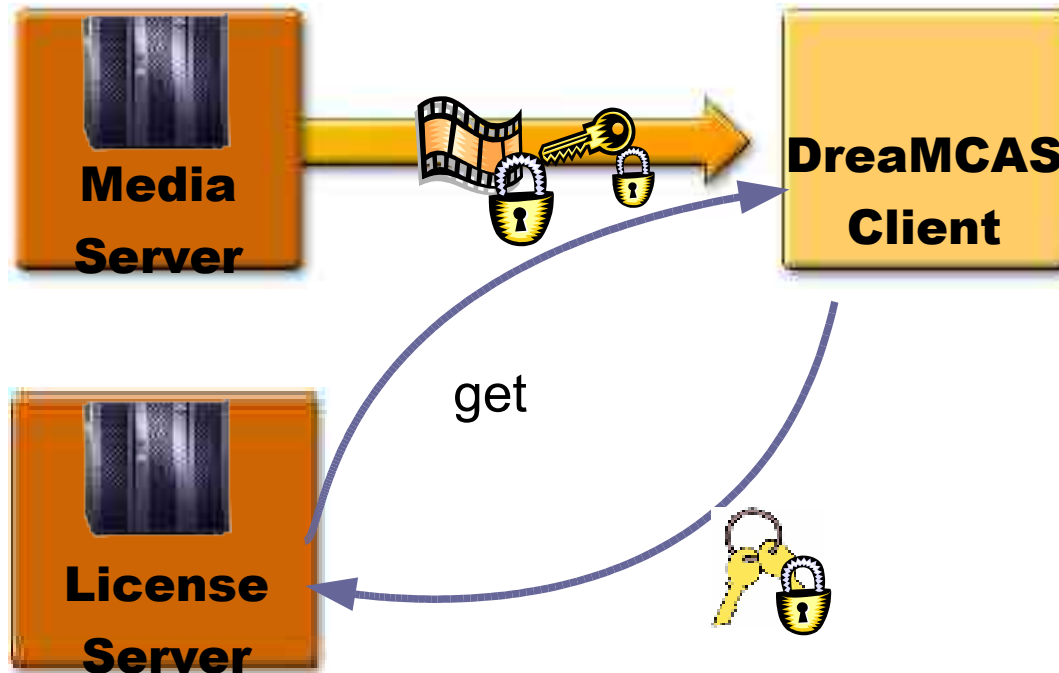


# DReaM-CAS: Details

- $\text{Content}_{\text{prot}} = \text{SymEncrypt}(\text{Content}, K_s)$
- $K' = \text{SymEncrypt}(K_s, K_a)$  SYMENC() - Symmetric Encryption  
 $K_s$  are stream keys  
 $K_a$  is access key
- $K'$  stored with  $\text{Content}_{\text{prot}}$
- $T_e = \text{AsymEncrypt}(K_a, K_{\text{pub}})$  ASYMENC() - Asymmetric Encryption  
 $K_{\text{pub}}$  is public key of client
- $K_a = \text{AsymDecrypt}(T_e, K_{\text{priv}})$   
 $K_{\text{priv}}$  is private key of client
- $K_s = \text{AsymDecrypt}(K', K_a)$
- $\text{Content} = \text{SymDecrypt}(\text{Content}_{\text{prot}}, K_s)$

# DReaM-CAS: Details

$$K_a = \text{AsymDecrypt}(T_e, K_{\text{priv}})$$

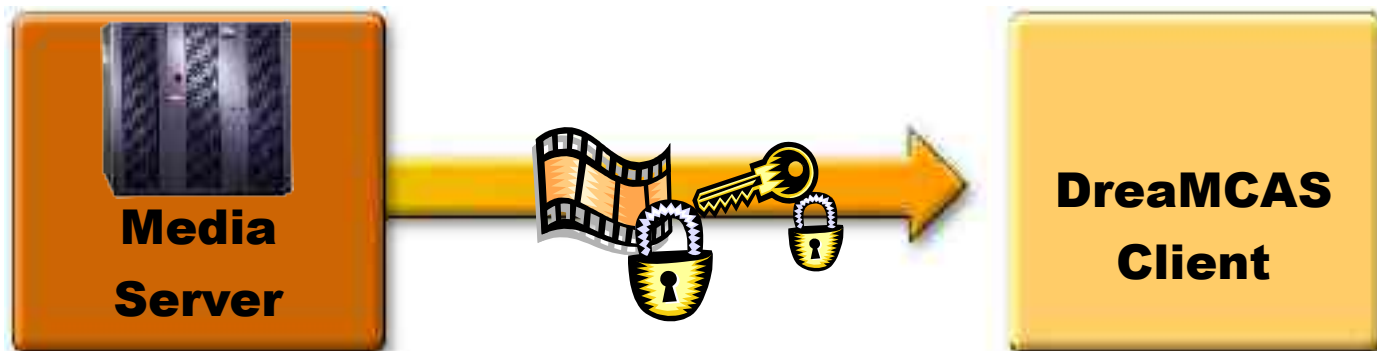


# DReaM-CAS: Details

- $\text{Content}_{\text{prot}} = \text{SymEncrypt}(\text{Content}, K_s)$
- $K' = \text{SymEncrypt}(K_s, K_a)$  SYMENC() - Symmetric Encryption  
 $K_s$  are stream keys  
 $K_a$  is access key
- $K'$  stored with  $\text{Content}_{\text{prot}}$
- $T_e = \text{AsymEncrypt}(K_a, K_{\text{pub}})$  ASYMENC() - Asymmetric Encryption  
 $K_{\text{pub}}$  is public key of client
- $K_a = \text{AsymDecrypt}(T_e, K_{\text{priv}})$   
 $K_{\text{priv}}$  is private key of client
- $K_s = \text{AsymDecrypt}(K', K_a)$
- $\text{Content} = \text{SymDecrypt}(\text{Content}_{\text{prot}}, K_s)$

# DReaM-CAS: Details

$$K_s = \text{SymDecrypt}(K', K_a)$$

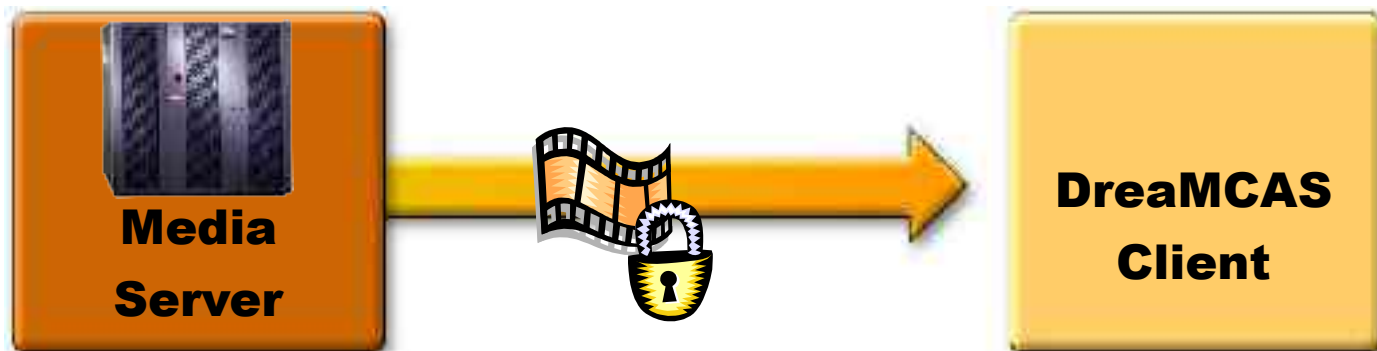


# DReaM-CAS: Details

- $\text{Content}_{\text{prot}} = \text{SymEncrypt}(\text{Content}, K_s)$
- $K' = \text{SymEncrypt}(K_s, K_a)$  SYMENC() - Symmetric Encryption  
 $K_s$  are stream keys  
 $K_a$  is access key
- $K'$  stored with  $\text{Content}_{\text{prot}}$
- $T_e = \text{AsymEncrypt}(K_a, K_{\text{pub}})$  ASYMENC() - Asymmetric Encryption  
 $K_{\text{pub}}$  is public key of client
- $K_a = \text{AsymDecrypt}(T_e, K_{\text{priv}})$   
 $K_{\text{priv}}$  is private key of client
- $K_s = \text{SymDecrypt}(K', K_a)$
- $\text{Content} = \text{SymDecrypt}(\text{Content}_{\text{prot}}, K_s)$

# DReaM-CAS: Details

$$\text{Content}_{\text{prot}} = \text{SymEncrypt}(\text{Content}, K_s)$$



# DReaM-CAS: Details

- $\text{Content}_{\text{prot}} = \text{SymEncrypt}(\text{Content}, \mathbf{K}_s)$
- $\mathbf{K}' = \text{SymEncrypt}(\mathbf{K}_s, \mathbf{K}_a)$  SYMENC() - Symmetric Encryption  
 $\mathbf{K}_s$  are stream keys  
 $\mathbf{K}_a$  is access key
- $\mathbf{K}'$  stored with  $\text{Content}_{\text{prot}}$
- $\mathbf{T}_e = \text{AsymEncrypt}(\mathbf{K}_a, \mathbf{K}_{\text{pub}})$  ASYMENC() - Asymmetric Encryption  
 $\mathbf{K}_{\text{pub}}$  is public key of client
- $\mathbf{K}_a = \text{AsymDecrypt}(\mathbf{T}_e, \mathbf{K}_{\text{priv}})$   
 $\mathbf{K}_{\text{priv}}$  is private key of client
- $\mathbf{K}_s = \text{AsymDecrypt}(\mathbf{K}', \mathbf{K}_a)$

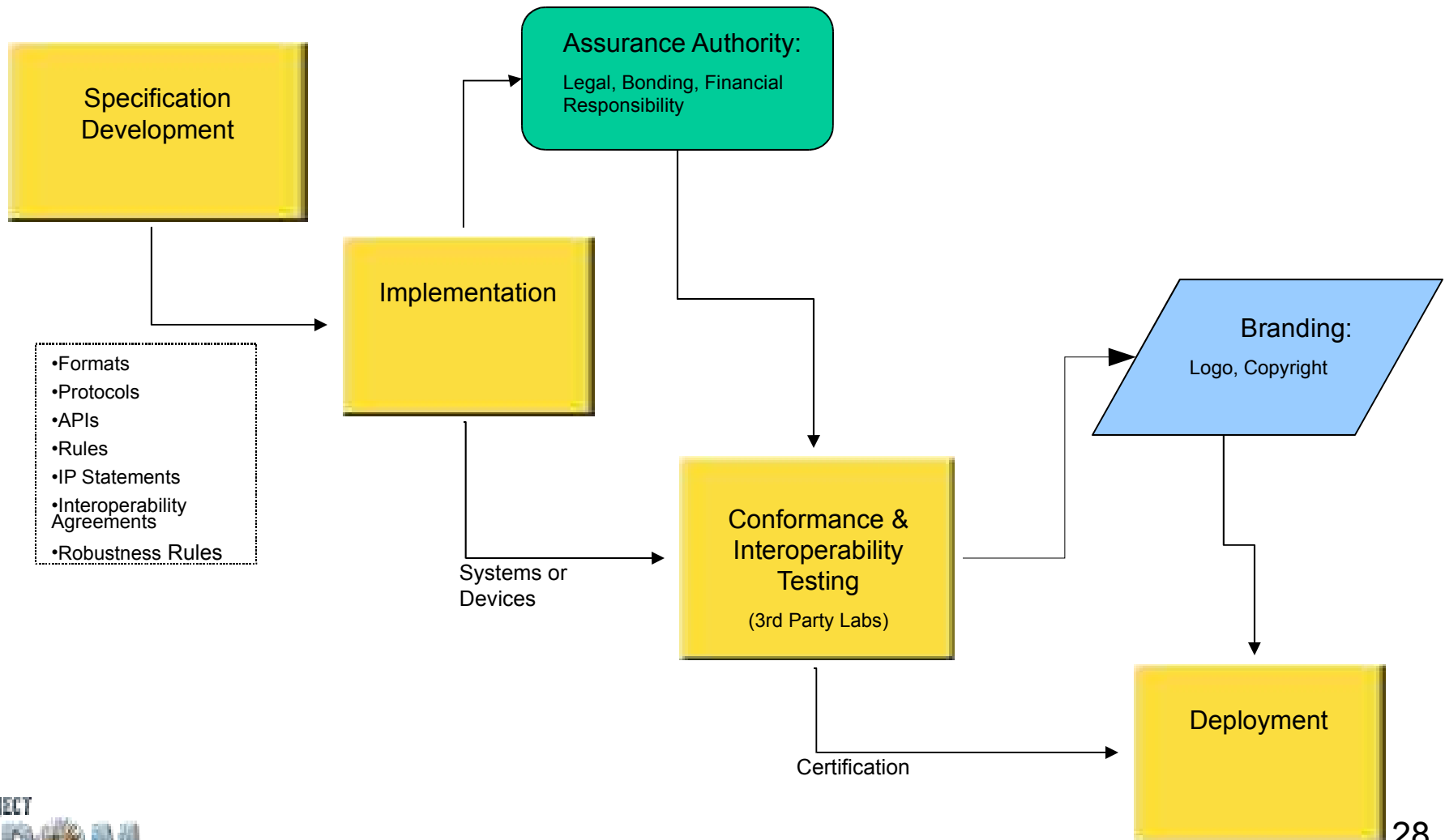
# DReaM-CAS – ECM

- Entitlement Control Messages
- Defined by MPEG and DVB standards
- Fully defined in DreaM-CAS to carry, (protected) Stream keys, IV, and Cipher modes.
- Same Cipher mode for Keys and Data.
- Embedded as often (or more) as PMTs (Program Map Tables) in the MPEG-2 TS.

# DReaM-CAS – EMM

- Entitlement Management Messages.
- Defined by MPEG and DVB standards
- Fully defined in DreaM-CAS to carry, (individually protected) Access keys, and IV.
- Assymmetric Algorithm - currently only RSA.
- Out of band mechanism using https.

# The Process Ahead



# Summary

- DreaMCAS
  - > Content Encryption
  - > Key Management
  - > ECM and EMMs
- Source code of an end-to-end working prototype released in [dream.dev.java.net](http://dream.dev.java.net).

# Q&A

PROJECT  
**DREAM**

An Open, End-to-End  
Content Protection  
Solution



<http://OpenMediaCommons.org>