

DAB CA The e-commerce Way

Dr Dirk Husemann
IBM Research Division
Zurich Research Lab

2001-12-10

DAB CA: Requirements

WorldDAB (DAB/Mobile) CA requirements: CA mechanism that...

DAB CA: Requirements

WorldDAB (DAB/Mobile) CA requirements: CA mechanism that...

- ... can be deployed by content provider **easily and fast**

DAB CA: Requirements

WorldDAB (DAB/Mobile) CA requirements: CA mechanism that...

- ... can be deployed by content provider **easily and fast**
 - **minimize dependencies** on third-parties

DAB CA: Requirements

WorldDAB (DAB/Mobile) CA requirements: CA mechanism that...

- ... can be deployed by content provider **easily and fast**
 - **minimize dependencies** on third-parties
 - allow **maximum flexibility**: allow service provisioning by third-party

DAB CA: Requirements

WorldDAB (DAB/Mobile) CA requirements: CA mechanism that...

- ... can be deployed by content provider **easily and fast**
 - **minimize dependencies** on third-parties
 - allow **maximum flexibility**: allow service provisioning by third-party
- ... can be **implemented with a minimum of hardware resources**

DAB CA: Requirements

WorldDAB (DAB/Mobile) CA requirements: CA mechanism that...

- ... can be deployed by content provider **easily and fast**
 - **minimize dependencies** on third-parties
 - allow **maximum flexibility**: allow service provisioning by third-party
- ... can be **implemented with a minimum of hardware** resources
- ... is **reasonably secure**

DAB CA: Requirements

WorldDAB (DAB/Mobile) CA requirements: CA mechanism that...

- ... can be deployed by content provider **easily and fast**
 - **minimize dependencies** on third-parties
 - allow **maximum flexibility**: allow service provisioning by third-party
- ... can be **implemented with a minimum of hardware resources**
- ... is **reasonably secure**
- ... can be **implemented for both online and offline systems**

DAB CA: Requirements

WorldDAB (DAB/Mobile) CA requirements: CA mechanism that...

- ... can be deployed by content provider **easily and fast**
 - **minimize dependencies** on third-parties
 - allow **maximum flexibility**: allow service provisioning by third-party
- ... can be **implemented with a minimum of hardware resources**
- ... is **reasonably secure**
- ... can be **implemented for both online and offline systems**
 - online: DAB/mobile platform
 - offline: DAB-only platform

DAB CA-PK [DAB cup-kee]

DAB CA-PK [DAB cup-kee]

**Non CA-ed
operation**

DAB CA-PK [DAB cup-kee]

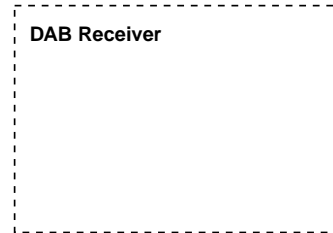
Content
provider

**Non CA-ed
operation**

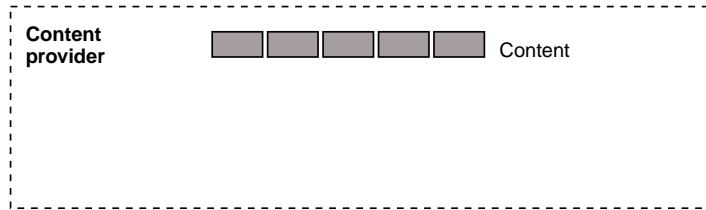
DAB CA-PK [DAB cup-kee]



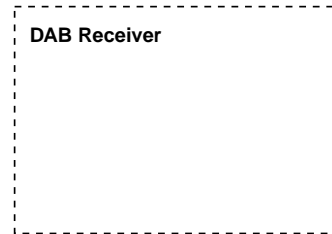
**Non CA-ed
operation**



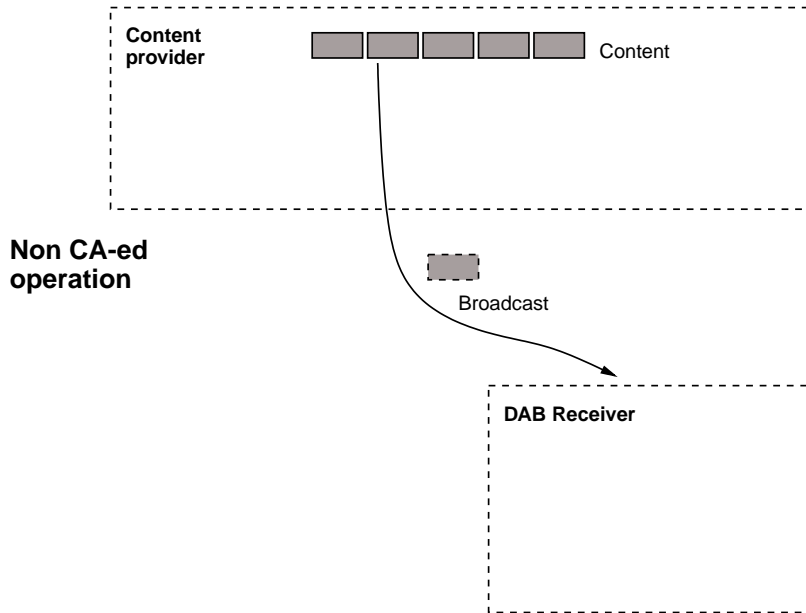
DAB CA-PK [DAB cup-kee]



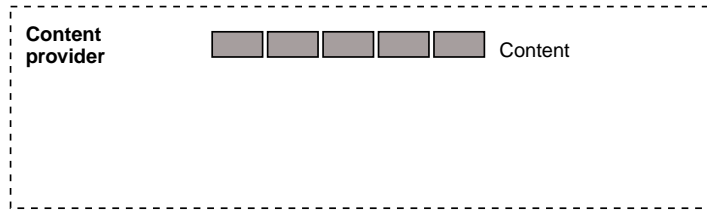
**Non CA-ed
operation**



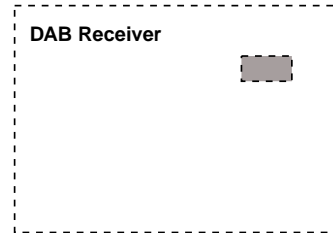
DAB CA-PK [DAB cup-kee]



DAB CA-PK [DAB cup-kee]



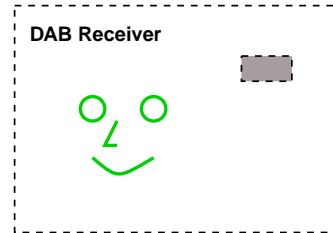
**Non CA-ed
operation**



DAB CA-PK [DAB cup-kee]



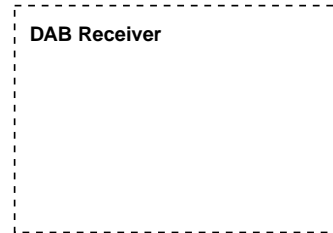
Non CA-ed
operation



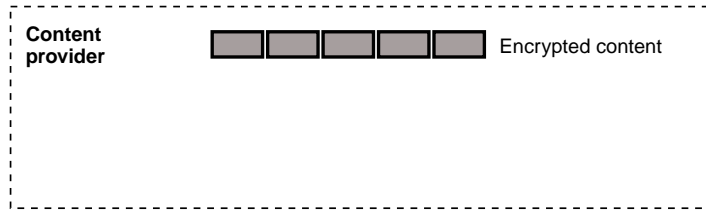
DAB CA-PK [DAB cup-kee]



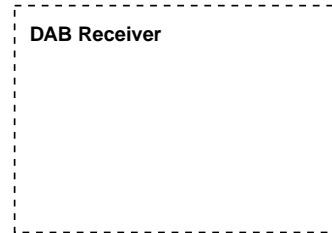
CA-ed
operation



DAB CA-PK [DAB cup-kee]



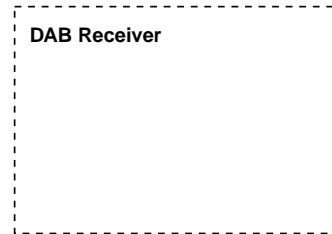
**CA-ed
operation**



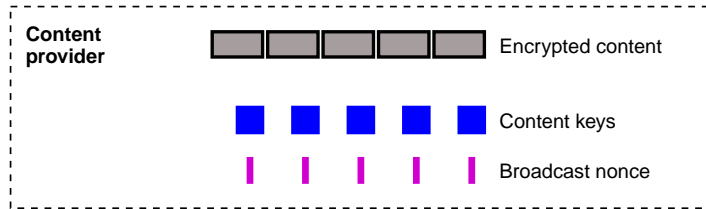
DAB CA-PK [DAB cup-kee]



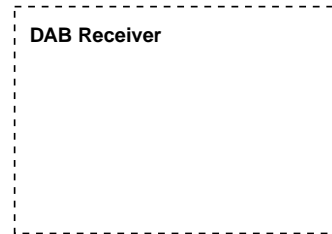
**CA-ed
operation**



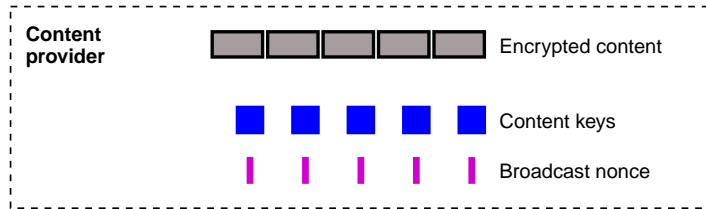
DAB CA-PK [DAB cup-kee]



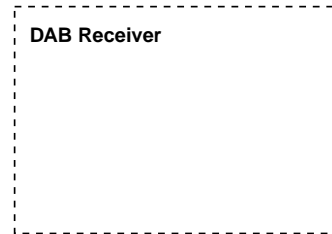
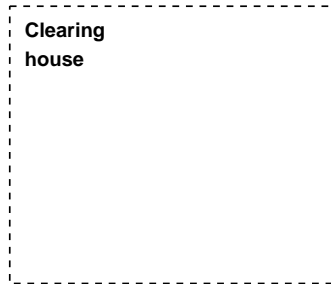
**CA-ed
operation**



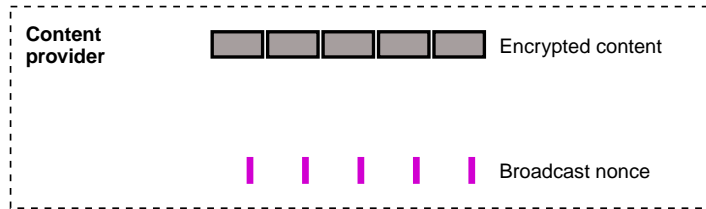
DAB CA-PK [DAB cup-kee]



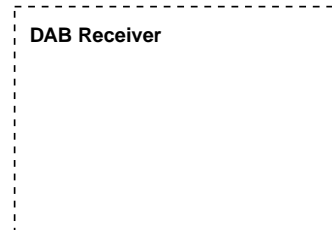
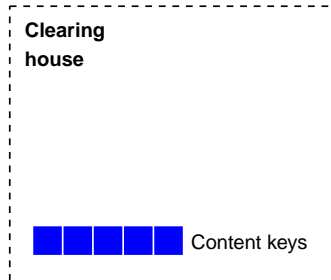
CA-ed operation



DAB CA-PK [DAB cup-kee]



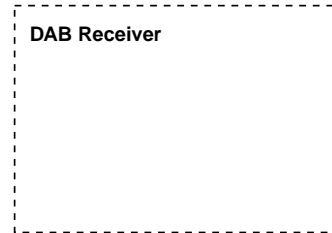
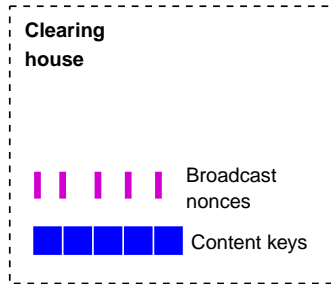
CA-ed operation



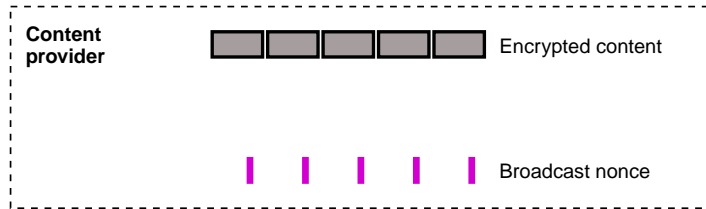
DAB CA-PK [DAB cup-kee]



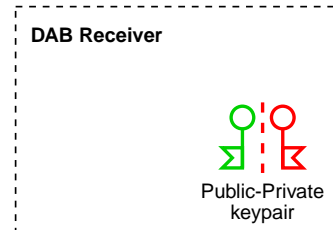
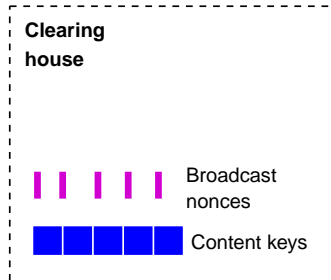
CA-ed operation



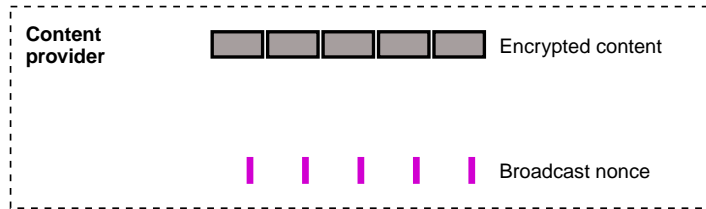
DAB CA-PK [DAB cup-kee]



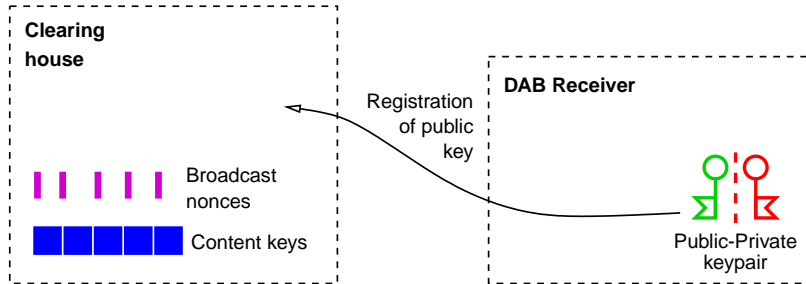
CA-ed operation



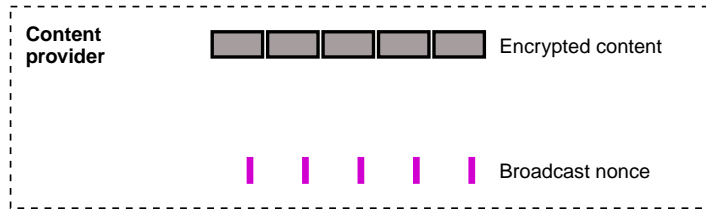
DAB CA-PK [DAB cup-kee]



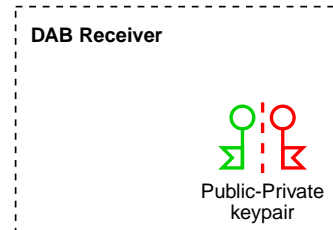
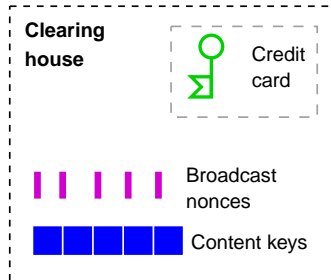
CA-ed operation



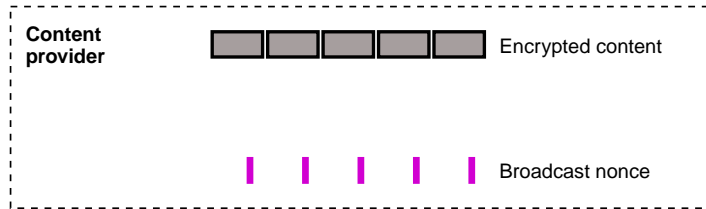
DAB CA-PK [DAB cup-kee]



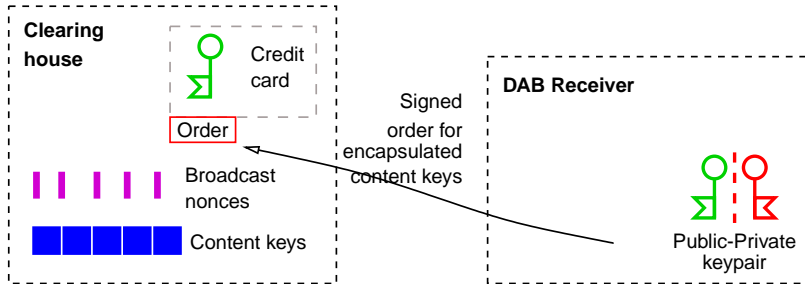
CA-ed operation



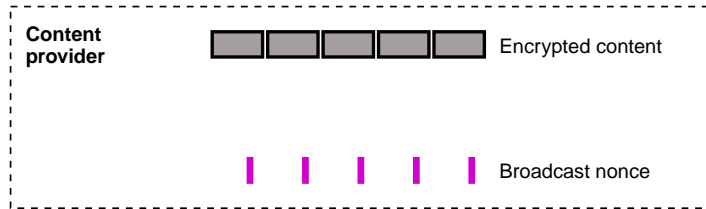
DAB CA-PK [DAB cup-kee]



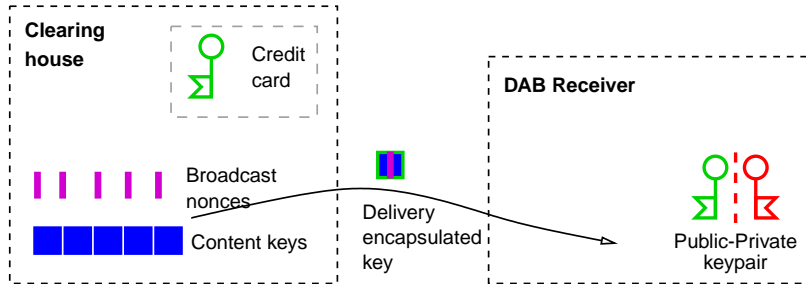
CA-ed operation



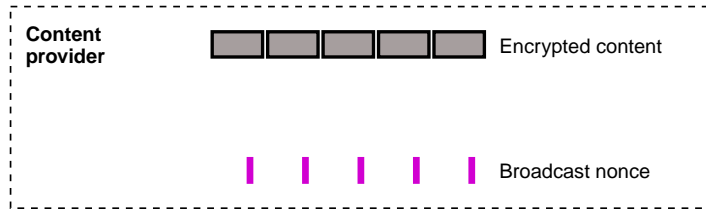
DAB CA-PK [DAB cup-kee]



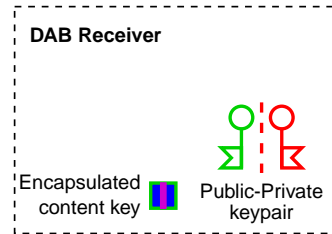
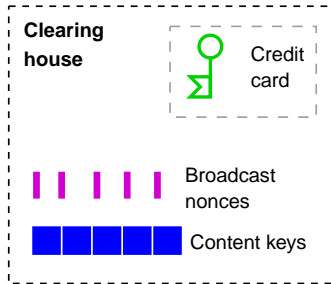
CA-ed operation



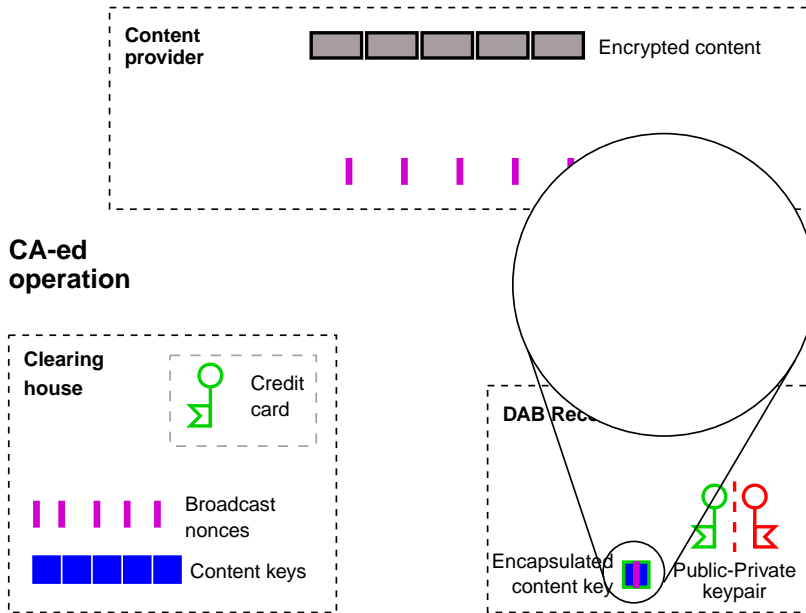
DAB CA-PK [DAB cup-kee]



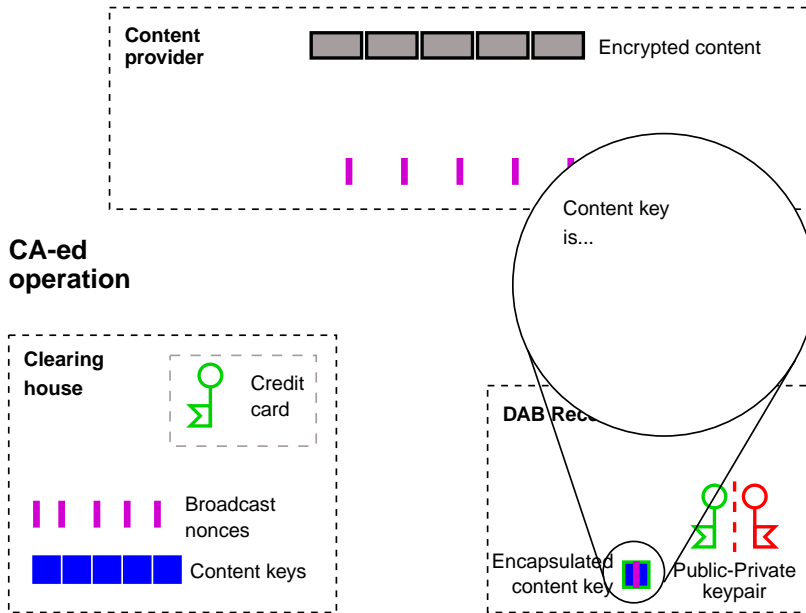
CA-ed operation



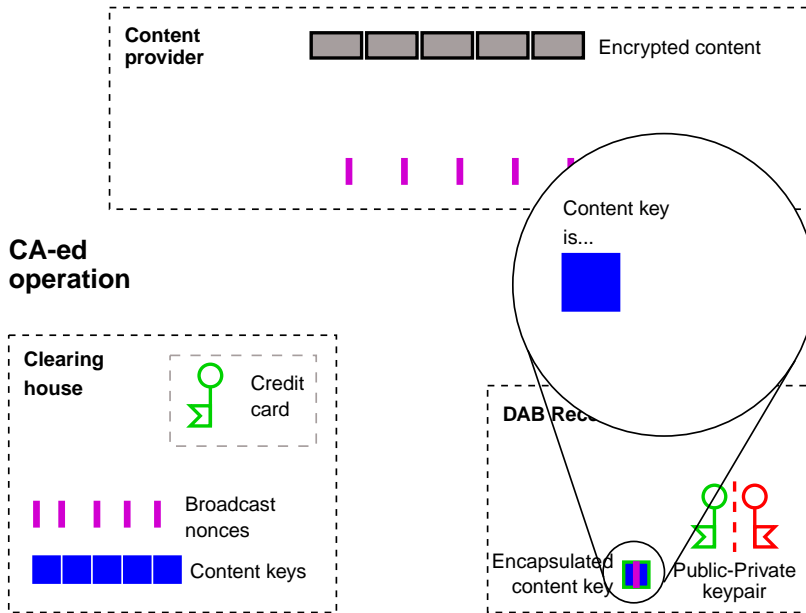
DAB CA-PK [DAB cup-kee]



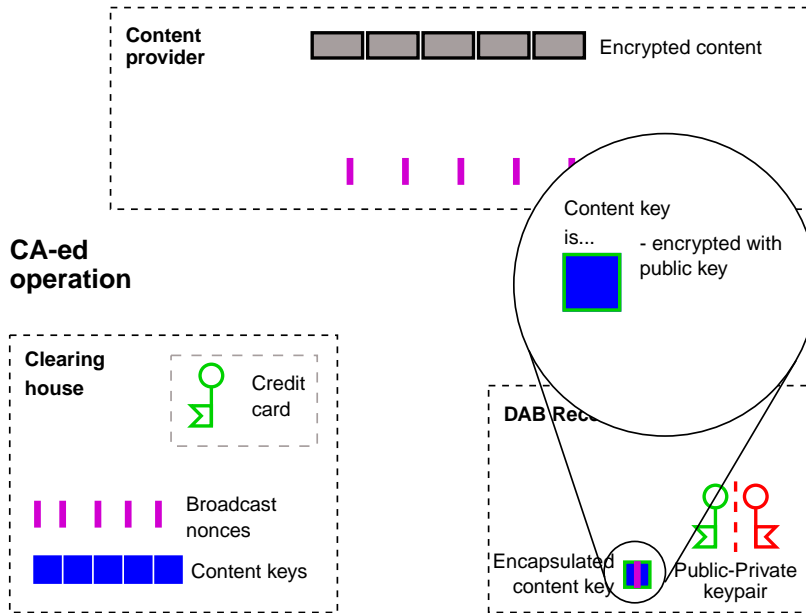
DAB CA-PK [DAB cup-kee]



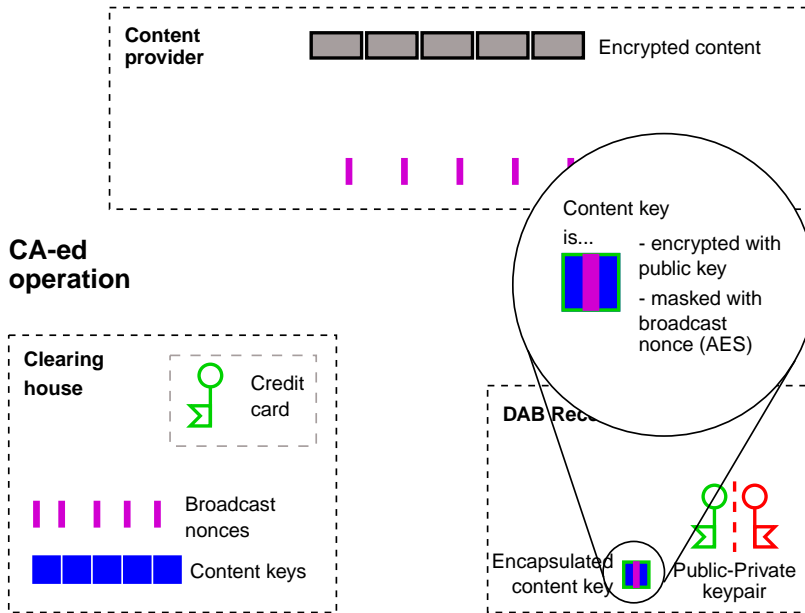
DAB CA-PK [DAB cup-kee]



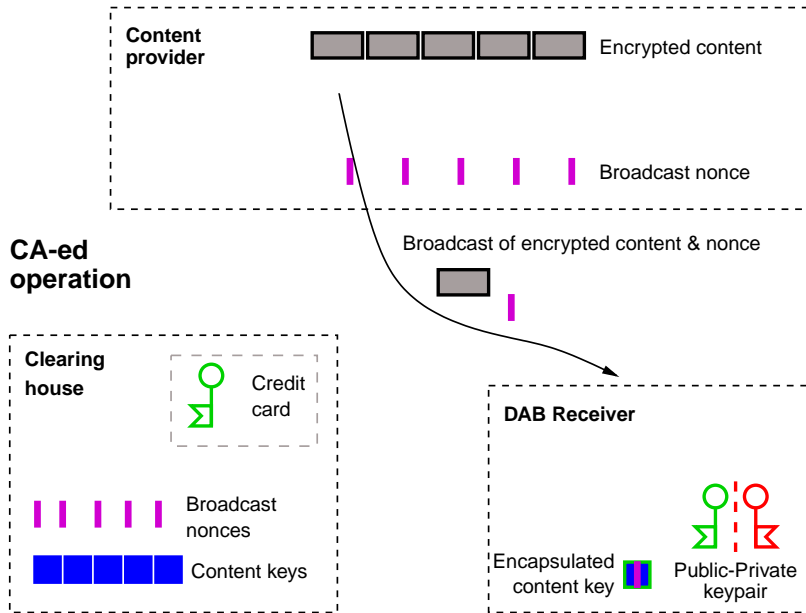
DAB CA-PK [DAB cup-kee]



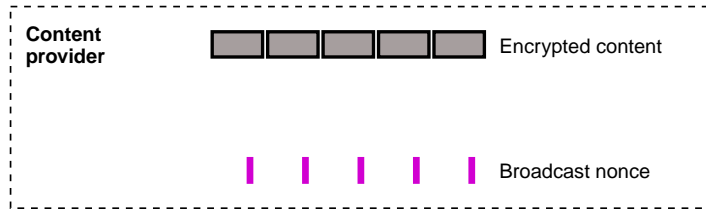
DAB CA-PK [DAB cup-kee]



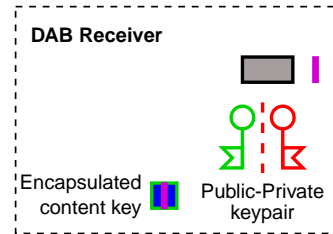
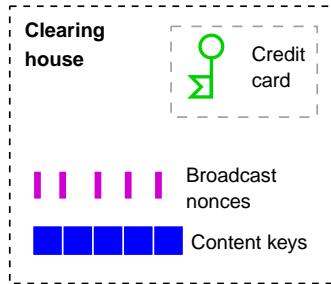
DAB CA-PK [DAB cup-kee]



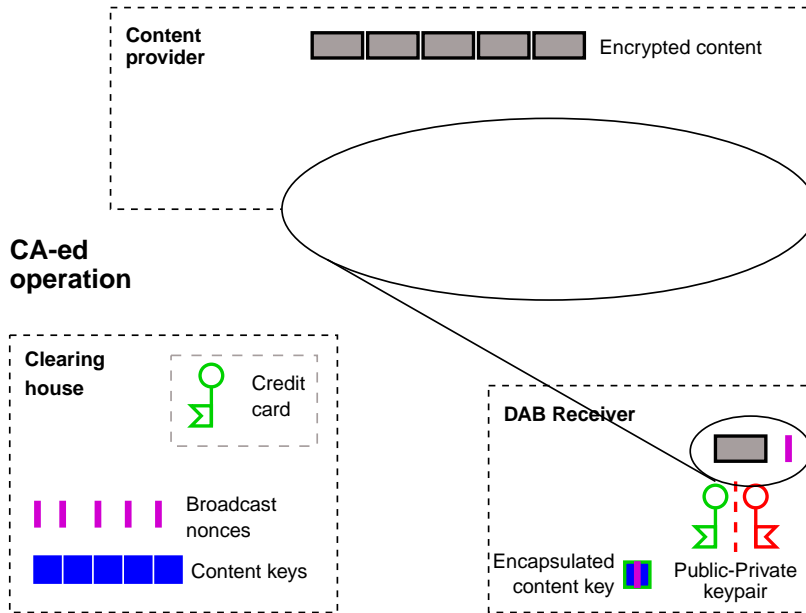
DAB CA-PK [DAB cup-kee]



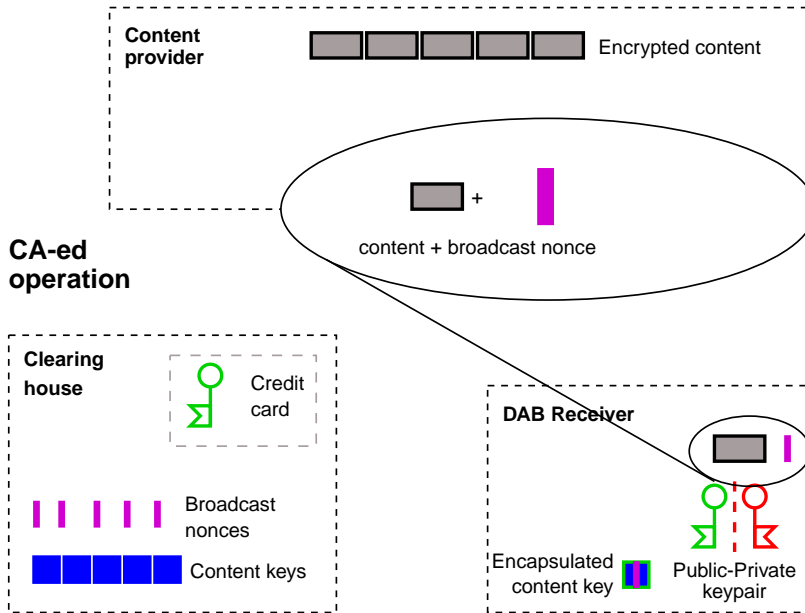
CA-ed operation



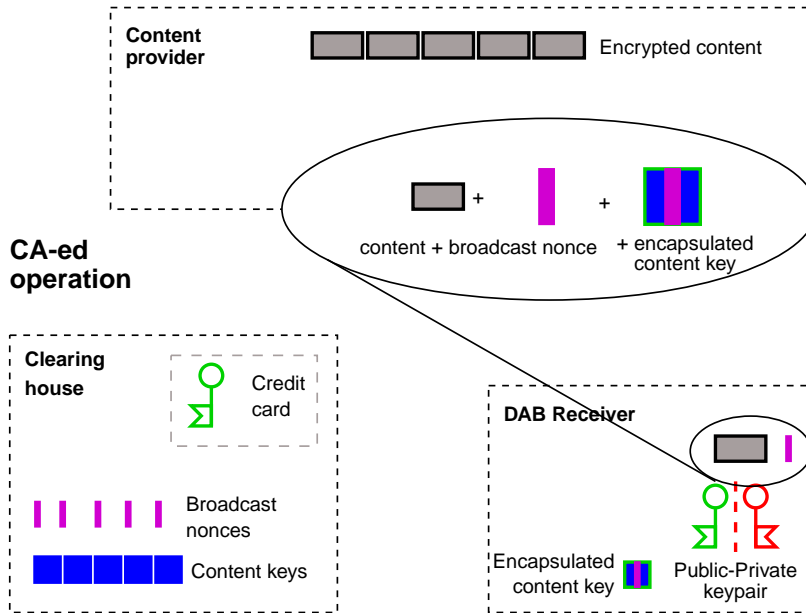
DAB CA-PK [DAB cup-kee]



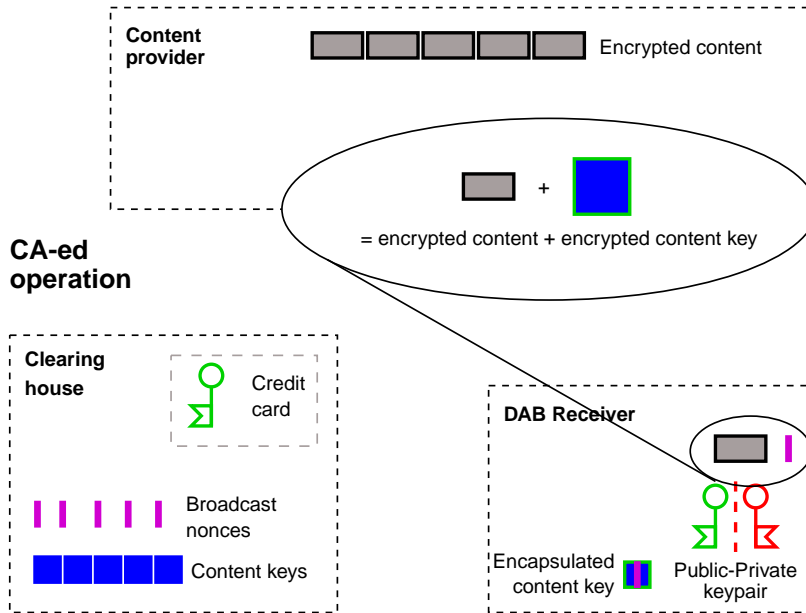
DAB CA-PK [DAB cup-kee]



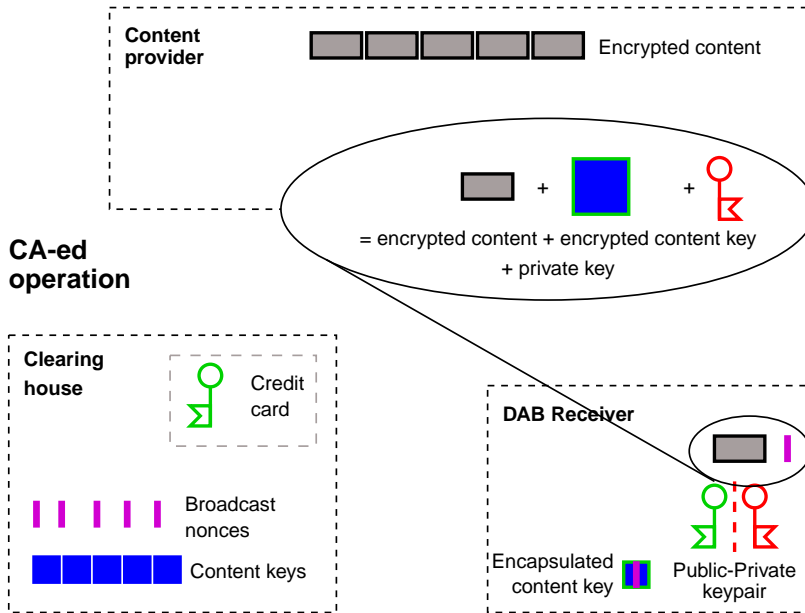
DAB CA-PK [DAB cup-kee]



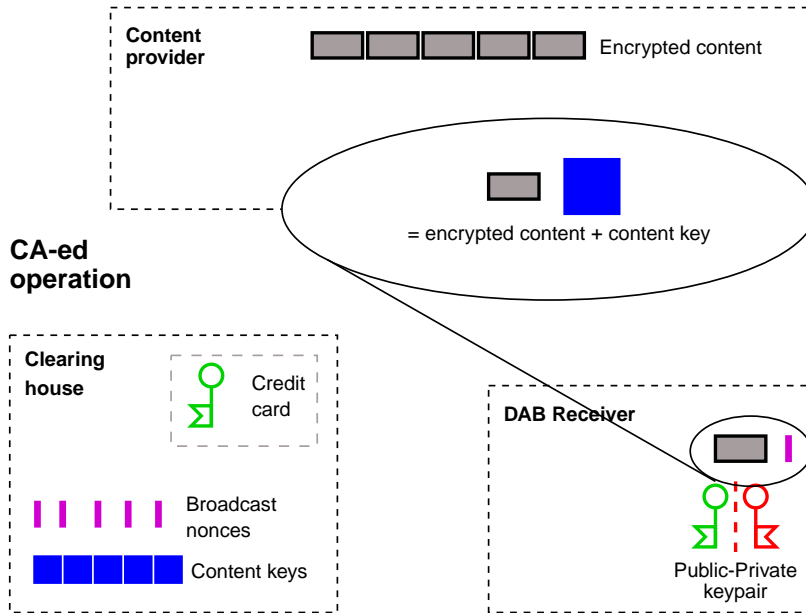
DAB CA-PK [DAB cup-kee]



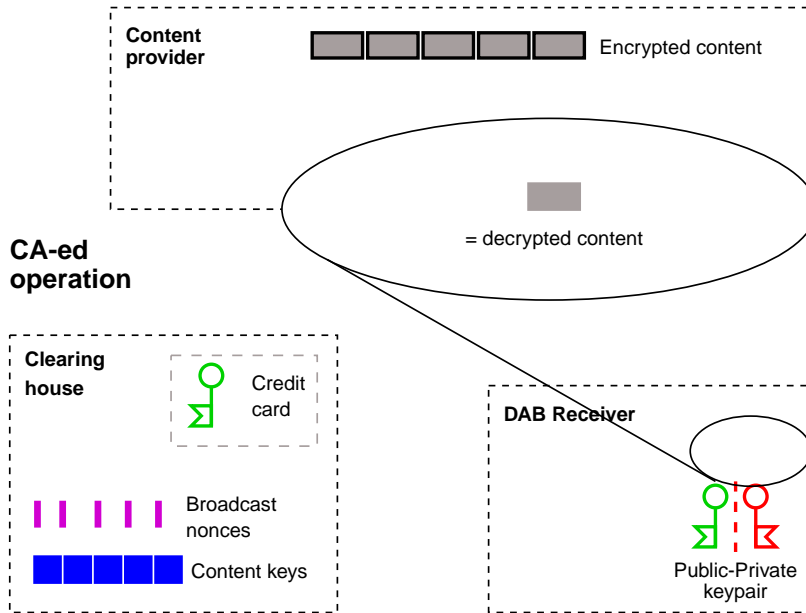
DAB CA-PK [DAB cup-kee]



DAB CA-PK [DAB cup-kee]



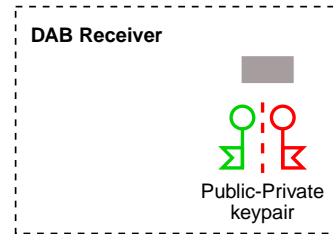
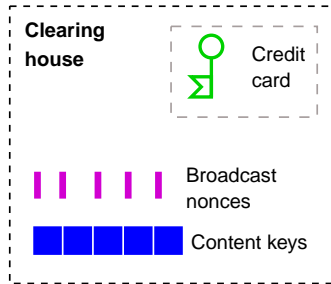
DAB CA-PK [DAB cup-kee]



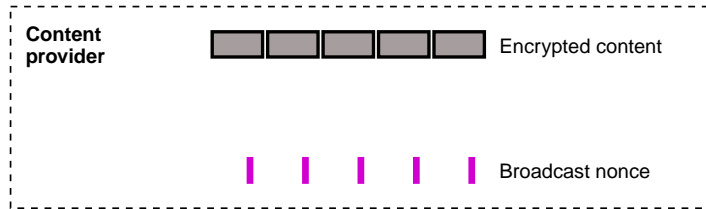
DAB CA-PK [DAB cup-kee]



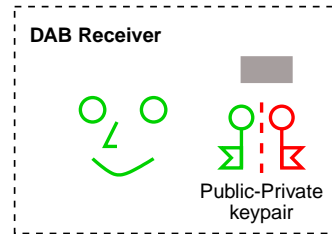
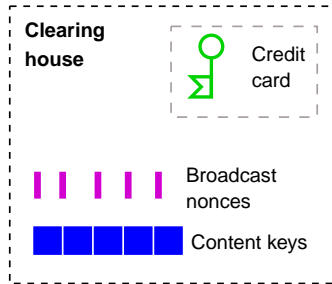
CA-ed operation



DAB CA-PK [DAB cup-kee]



CA-ed operation



DAB CA-PK: Advantages

- Advantage: Consumer has a vested interest in protecting her private key: **consumer becomes ally instead of threat!**

DAB CA-PK: Advantages

- Advantage: Consumer has a vested interest in protecting her private key: **consumer becomes ally instead of threat!**
- Advantage: **Mechanism can be implemented in software**

DAB CA-PK: Advantages

- Advantage: Consumer has a vested interest in protecting her private key: **consumer becomes ally instead of threat!**
- Advantage: **Mechanism can be implemented in software**
 - Requires crypto support though

DAB CA-PK: Advantages

- Advantage: Consumer has a vested interest in protecting her private key: **consumer becomes ally instead of threat!**
- Advantage: **Mechanism can be implemented in software**
 - Requires crypto support though
 - JavaCard can externalize crypto support

DAB CA-PK: Advantages

- Advantage: Consumer has a vested interest in protecting her private key: **consumer becomes ally instead of threat!**
- Advantage: **Mechanism can be implemented in software**
 - Requires crypto support though
 - JavaCard can externalize crypto support
 - JavaCard can serve as DR carrier

DAB CA-PK: Advantages

- Advantage: Consumer has a vested interest in protecting her private key: **consumer becomes ally instead of threat!**
- Advantage: **Mechanism can be implemented in software**
 - Requires crypto support though
 - JavaCard can externalize crypto support
 - JavaCard can serve as DR carrier
- Advantage: **Normal e-commerce transaction**, no required dependencies on third parties

DAB CA-PK: Advantages

- Advantage: Consumer has a vested interest in protecting her private key: **consumer becomes ally instead of threat!**
- Advantage: **Mechanism can be implemented in software**
 - Requires crypto support though
 - JavaCard can externalize crypto support
 - JavaCard can serve as DR carrier
- Advantage: **Normal e-commerce transaction**, no required dependencies on third parties
- Advantage: Content provider has complete control over CA process

Close-up: Content Preparation

Close-up: Content Preparation

C_i

- Digital content, unencrypted

Close-up: Content Preparation

$$C_i, K_i$$

- Digital content, unencrypted
- Content key

Close-up: Content Preparation

$$C_i, K_i, B_i$$

- Digital content, unencrypted
- Content key
- Broadcast nonce

Close-up: Content Preparation

$$C_i, K_i, B_i$$

$$\mathfrak{C}_i = \mathcal{E}_{\text{sym}}(C_i, K_i)$$

- Digital content, unencrypted
- Content key
- Broadcast nonce
- Encrypting content with content key

Close-up: Content Preparation

$$C_i, K_i, B_i$$

$$\mathfrak{C}_i = \mathcal{E}_{\text{sym}}(C_i, K_i)$$

\Rightarrow

- Digital content, unencrypted
 - Content key
 - Broadcast nonce
 - Encrypting content with content key
- Result:

Close-up: Content Preparation

$$C_i, K_i, B_i$$

$$\mathfrak{C}_i = \mathcal{E}_{\text{sym}}(C_i, K_i)$$

$$\Rightarrow \mathfrak{C}_i$$

- Digital content, unencrypted
- Content key
- Broadcast nonce
- Encrypting content with content key
- Result: encrypted content

Close-up: Content Preparation

$$C_i, K_i, B_i$$

$$\mathfrak{C}_i = \mathcal{E}_{\text{sym}}(C_i, K_i)$$

$$\Rightarrow \mathfrak{C}_i, K_i$$

- Digital content, unencrypted
- Content key
- Broadcast nonce
- Encrypting content with content key
- Result: encrypted content, content key

Close-up: Content Preparation

$$C_i, K_i, B_i$$

$$\mathfrak{C}_i = \mathcal{E}_{\text{sym}}(C_i, K_i)$$

$$\Rightarrow \mathfrak{C}_i, K_i, B_i$$

- Digital content, unencrypted
- Content key
- Broadcast nonce
- Encrypting content with content key
- Result: encrypted content, content key, broadcast nonce

Close-up: Content Preparation

$$C_i, K_i, B_i$$

$$\mathfrak{C}_i = \mathcal{E}_{\text{sym}}(C_i, K_i)$$

$$\Rightarrow \mathfrak{C}_i, K_i, B_i$$

- Digital content, unencrypted
- Content key
- Broadcast nonce
- Encrypting content with content key
- Result: encrypted content, content key, broadcast nonce
- **Clearing house:**

Close-up: Content Preparation

$$C_i, K_i, B_i$$

$$\mathfrak{C}_i = \mathcal{E}_{\text{sym}}(C_i, K_i)$$

$$\Rightarrow \mathfrak{C}_i, K_i, B_i$$

- Digital content, unencrypted
- Content key
- Broadcast nonce
- Encrypting content with content key
- Result: encrypted content, content key, broadcast nonce
- **Clearing house: content key**

Close-up: Content Preparation

$$C_i, K_i, B_i$$

$$\mathfrak{C}_i = \mathcal{E}_{\text{sym}}(C_i, K_i)$$

$$\Rightarrow \mathfrak{C}_i, K_i, B_i$$

- Digital content, unencrypted
- Content key
- Broadcast nonce
- Encrypting content with content key
- Result: encrypted content, content key, broadcast nonce
- **Clearing house: content key, broadcast nonce**

Close-up: Clearing House

Close-up: Clearing House

K_i, B_i

- Content key & broadcast nonce from clearing house

Close-up: Clearing House

$$K_i, B_i, K_R^{\text{pub}}$$

- Content key & broadcast nonce from clearing house
- Public key from customer

Close-up: Clearing House

$$K_i, B_i, K_R^{\text{pub}}$$

$$\mathcal{K}_{i.rx} =$$

- Content key & broadcast nonce from clearing house
- Public key from customer
- **Encapsulating** content key

Close-up: Clearing House

$$K_i, B_i, K_R^{\text{pub}}$$

$$\mathcal{K}_{i.rtx} = \mathcal{E}_{\text{sym}}(\mathcal{E}_{\text{pub}}(K_i, K_R^{\text{pub}}))$$

- Content key & broadcast nonce from clearing house
- Public key from customer
- **Encapsulating** content key
 - RSA encrypt content key with public key

Close-up: Clearing House

$$K_i, B_i, K_R^{\text{pub}}$$

$$\mathcal{K}_{i.rtx} = \mathcal{E}_{\text{sym}}(\mathcal{E}_{\text{pub}}(K_i, K_R^{\text{pub}}), B_i)$$

- Content key & broadcast nonce from clearing house
- Public key from customer
- **Encapsulating** content key
 - RSA encrypt content key with public key
- Block (AES ECB) with broadcast nonce

Close-up: Clearing House

$$K_i, B_i, K_R^{\text{pub}}$$

$$\mathcal{K}_{i.rtx} = \mathcal{E}_{\text{sym}}(\mathcal{E}_{\text{pub}}(K_i, K_R^{\text{pub}}), B_i)$$

⇒

- Content key & broadcast nonce from clearing house
- Public key from customer
- **Encapsulating** content key
 - RSA encrypt content key with public key
- Block (AES ECB) with broadcast nonce
- Result:

Close-up: Clearing House

$$K_i, B_i, K_R^{\text{pub}}$$

$$\mathcal{K}_{i.rx} = \mathcal{E}_{\text{sym}}(\mathcal{E}_{\text{pub}}(K_i, K_R^{\text{pub}}), B_i)$$

⇒

$\mathcal{K}_{i.rx}$

- Content key & broadcast nonce from clearing house
- Public key from customer
- **Encapsulating** content key
 - RSA encrypt content key with public key
- Block (AES ECB) with broadcast nonce
- Result: encapsulated content key

Close-up: Clearing House

$$K_i, B_i, K_R^{\text{pub}}$$

$$\mathcal{K}_{i.rx} = \mathcal{E}_{\text{sym}}(\mathcal{E}_{\text{pub}}(K_i, K_R^{\text{pub}}), B_i)$$

\Rightarrow

$\mathcal{K}_{i.rx}$

- Content key & broadcast nonce from clearing house
- Public key from customer
- **Encapsulating** content key
 - RSA encrypt content key with public key
- Block (AES ECB) with broadcast nonce
- Result: encapsulated content key
- **Receiver:**

Close-up: Clearing House

$$K_i, B_i, K_R^{\text{pub}}$$

$$\mathcal{K}_{i.rx} = \mathcal{E}_{\text{sym}}(\mathcal{E}_{\text{pub}}(K_i, K_R^{\text{pub}}), B_i)$$

\Rightarrow

$\mathcal{K}_{i.rx}$

- Content key & broadcast nonce from clearing house
- Public key from customer
- **Encapsulating** content key
 - RSA encrypt content key with public key
- Block (AES ECB) with broadcast nonce
- Result: encapsulated content key
- **Receiver: encapsulated content key** $\mathcal{K}_{i.rx}$

Close-up: Receiver

Close-up: Receiver

$$\mathcal{K}_{i \cdot rx}$$

- Encapsulated content key from **clearing house**

Close-up: Receiver

$$\mathcal{K}_{i \cdot rx}, K_R^{\text{priv}}$$

- Encapsulated content key from **clearing house**
- Private key

Close-up: Receiver

$$\mathcal{K}_{i \cdot rx}, K_R^{\text{priv}}$$
$$\mathcal{C}_i, B_i$$

- Encapsulated content key from **clearing house**
- Private key
- Encrypted content, broadcast nonce **via broadcast**

Close-up: Receiver

$$\mathcal{K}_{i.rx}, K_R^{\text{priv}}$$

$$\mathcal{C}_i, B_i$$

$$\Rightarrow K_i = \mathcal{D}_{\text{pub}}(\mathcal{D}_{\text{sym}}(\mathcal{K}_{i.rx}, B_i), K_R^{\text{priv}})$$

- Encapsulated content key from **clearing house**
- Private key
- Encrypted content, broadcast nonce **via broadcast**
- Extract content key

Close-up: Receiver

$$\mathcal{K}_{i.rx}, K_R^{\text{priv}}$$

$$\mathcal{C}_i, B_i$$

$$\Rightarrow K_i = \mathcal{D}_{\text{pub}}(\mathcal{D}_{\text{sym}}(\mathcal{K}_{i.rx}, B_i), K_R^{\text{priv}})$$

\Rightarrow

- Encapsulated content key from **clearing house**
- Private key
- Encrypted content, broadcast nonce **via broadcast**
- Extract content key
- Result:

Close-up: Receiver

$$\mathcal{K}_{i.rx}, K_R^{\text{priv}}$$

$$\mathcal{C}_i, B_i$$

$$\Rightarrow K_i = \mathcal{D}_{\text{pub}}(\mathcal{D}_{\text{sym}}(\mathcal{K}_{i.rx}, B_i), K_R^{\text{priv}})$$

$$\Rightarrow C_i = \mathcal{D}_{\text{sym}}(\mathcal{C}_i, K_i)$$

- Encapsulated content key from **clearing house**
- Private key
- Encrypted content, broadcast nonce **via broadcast**
- Extract content key
- Result: **Decrypted content**

Attack by rogue consumer. . .

Goal: get access to the content key and distribute it before the broadcast occurs

Attack by rogue consumer. . .

Goal: get access to the content key and distribute it before the broadcast occurs

- Hacker customer:

Attack by rogue consumer. . .

Goal: get access to the content key and distribute it before the broadcast occurs

- Hacker customer: unwrapping of encapsulated content requires
 - AES ECB decryption with **broadcast nonce**

Attack by rogue consumer. . .

Goal: get access to the content key and distribute it before the broadcast occurs

- Hacker customer: unwrapping of encapsulated content requires
 - AES ECB decryption with **broadcast nonce**
 - RSA decryption with **customer's private key**

Attack by rogue consumer. . .

Goal: get access to the content key and distribute it before the broadcast occurs

- Hacker customer: unwrapping of encapsulated content requires
 - AES ECB decryption with **broadcast nonce**
 - RSA decryption with **customer's private key**
 - Passing on the content key requires providing the private key

Attack by rogue consumer. . .

Goal: get access to the content key and distribute it before the broadcast occurs

- Hacker customer: unwrapping of encapsulated content requires
 - AES ECB decryption with **broadcast nonce**
 - RSA decryption with **customer's private key**
 - Passing on the content key requires providing the private key: As good as handing over your credit card

Attack by rogue consumer. . .

Goal: get access to the content key and distribute it before the broadcast occurs

- Hacker customer: unwrapping of encapsulated content requires
 - AES ECB decryption with **broadcast nonce**
 - RSA decryption with **customer's private key**
 - Passing on the content key requires providing the private key: As good as handing over your credit card
- Weak point: clearing house:

Attack by rogue consumer. . .

Goal: get access to the content key and distribute it before the broadcast occurs

- Hacker customer: unwrapping of encapsulated content requires
 - AES ECB decryption with **broadcast nonce**
 - RSA decryption with **customer's private key**
 - Passing on the content key requires providing the private key: As good as handing over your credit card
- Weak point: clearing house: appropriate contracts

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content, AES ECB for encapsulation

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content, AES ECB for encapsulation
- Public key crypto code...

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content, AES ECB for encapsulation
- Public key crypto code...
 - ...on-board receiver

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content, AES ECB for encapsulation
- Public key crypto code...
 - ... on-board receiver
 - ... "outsourced" (e.g., JavaCard)

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content, AES ECB for encapsulation
- Public key crypto code...
 - ... on-board receiver
 - ... "outsourced" (e.g., JavaCard)
- On-board public key crypto: either...

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content, AES ECB for encapsulation
- Public key crypto code...
 - ... on-board receiver
 - ... "outsourced" (e.g., JavaCard)
- On-board public key crypto: either...
 - ... private key replicated to purchasing device

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content, AES ECB for encapsulation
- Public key crypto code...
 - ... on-board receiver
 - ... "outsourced" (e.g., JavaCard)
- On-board public key crypto: either...
 - ... private key replicated to purchasing device or
 - ... crypto op available through interface (e.g., USB)

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content, AES ECB for encapsulation
- Public key crypto code...
 - ... on-board receiver
 - ... "outsourced" (e.g., JavaCard)
- On-board public key crypto: either...
 - ... private key replicated to purchasing device or
 - ... crypto op available through interface (e.g., USB)
- Out-sourced public key crypto:

Implementation Notes

- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content, AES ECB for encapsulation
- Public key crypto code...
 - ... on-board receiver
 - ... "outsourced" (e.g., JavaCard)
- On-board public key crypto: either...
 - ... private key replicated to purchasing device or
 - ... crypto op available through interface (e.g., USB)
- Out-sourced public key crypto:
 - Easy to use with different devices

Implementation Notes

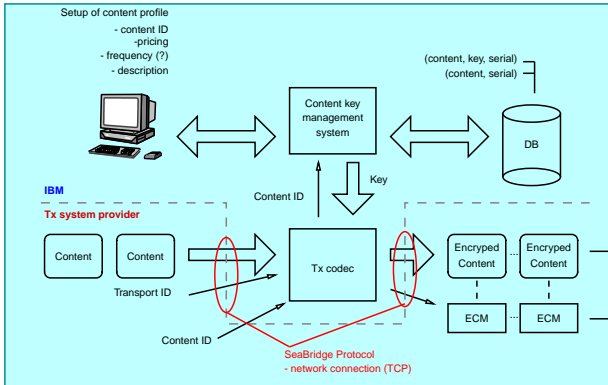
- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content, AES ECB for encapsulation
- Public key crypto code...
 - ... on-board receiver
 - ... "outsourced" (e.g., JavaCard)
- On-board public key crypto: either...
 - ... private key replicated to purchasing device or
 - ... crypto op available through interface (e.g., USB)
- Out-sourced public key crypto:
 - Easy to use with different devices
 - Requires receptacle at each device (e.g., smart card reader)

Implementation Notes

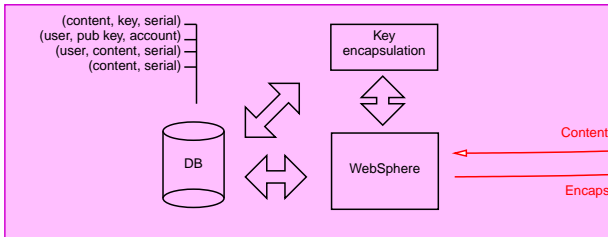
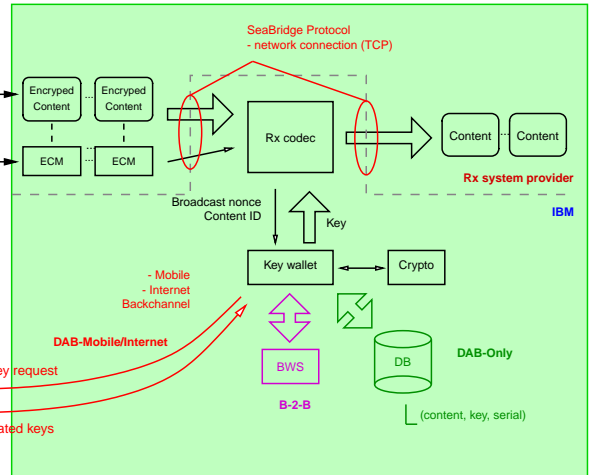
- Encryption $\mathcal{E}(\cdot, \cdot)$ can be simple
 - Example: **AES block cipher for encryption**
 - AES CBC for content, AES ECB for encapsulation
- Public key crypto code...
 - ... on-board receiver
 - ... "outsourced" (e.g., JavaCard)
- On-board public key crypto: either...
 - ... private key replicated to purchasing device or
 - ... crypto op available through interface (e.g., USB)
- Out-sourced public key crypto:
 - Easy to use with different devices
 - Requires receptacle at each device (e.g., smart card reader)
- Encapsulated content keys can be **distributed via DAB BWS (B2B apps)**

Impl: "The Big Picture"

Transmitter system



Receiver system



Clearing house

DAB CA: B-2-B

- Characteristic of B-2-B:

DAB CA: B-2-B

- Characteristic of B-2-B:
 - Low number of receivers (100-10K)

DAB CA: B-2-B

- Characteristic of B-2-B:
 - Low number of receivers (100-10K)
 - Receivers under control of content provider (usually)

DAB CA: B-2-B

- Characteristic of B-2-B:
 - Low number of receivers (100-10K)
 - Receivers under control of content provider (usually)
 - Public keys of receivers known in advance

DAB CA: B-2-B

- Characteristic of B-2-B:
 - Low number of receivers (100-10K)
 - Receivers under control of content provider (usually)
 - Public keys of receivers known in advance
- B-2-B:

DAB CA: B-2-B

- Characteristic of B-2-B:
 - Low number of receivers (100-10K)
 - Receivers under control of content provider (usually)
 - Public keys of receivers known in advance
- B-2-B:
 - Low number of receivers: distribute encapsulated keys over the air: DAB BWS

DAB CA: B-2-B

- Characteristic of B-2-B:
 - Low number of receivers (100-10K)
 - Receivers under control of content provider (usually)
 - Public keys of receivers known in advance
- B-2-B:
 - Low number of receivers: distribute encapsulated keys over the air: DAB BWS
- Clearing house: can then be integrated with content provider

DAB CA: B-2-B

- Characteristic of B-2-B:
 - Low number of receivers (100-10K)
 - Receivers under control of content provider (usually)
 - Public keys of receivers known in advance
- B-2-B:
 - Low number of receivers: distribute encapsulated keys over the air: DAB BWS
- Clearing house: can then be integrated with content provider
- Receivers: pick "their" encapsulated keys from content key BWS

DAB CA: B-2-B

- Characteristic of B-2-B:
 - Low number of receivers (100-10K)
 - Receivers under control of content provider (usually)
 - Public keys of receivers known in advance
- B-2-B:
 - Low number of receivers: distribute encapsulated keys over the air: DAB BWS
- Clearing house: can then be integrated with content provider
- Receivers: pick "their" encapsulated keys from content key BWS
- BWS helper application:
 - Filters on **key file tree BWS**

DAB CA: B-2-B

- Characteristic of B-2-B:
 - Low number of receivers (100-10K)
 - Receivers under control of content provider (usually)
 - Public keys of receivers known in advance
- B-2-B:
 - Low number of receivers: distribute encapsulated keys over the air: DAB BWS
- Clearing house: can then be integrated with content provider
- Receivers: pick "their" encapsulated keys from content key BWS
- BWS helper application:
 - Filters on **key file tree BWS**
 - Stores key files for hosting receiver in **local key file tree**

DAB CA: B-2-C with backchannel

- ECMs contain URL of e-commerce server offering content keys

DAB CA: B-2-C with backchannel

- ECMs contain URL of e-commerce server offering content keys
- Rx decoder reports URL back on missing subscription

DAB CA: B-2-C with backchannel

- ECMs contain URL of e-commerce server offering content keys
- Rx decoder reports URL back on missing subscription
- GUI invokes Web browser with URL (public key part of HTTP GET/POST)

DAB CA: B-2-C with backchannel

- ECMs contain URL of e-commerce server offering content keys
- Rx decoder reports URL back on missing subscription
- GUI invokes Web browser with URL (public key part of HTTP GET/POST)
- E-Commerce server

DAB CA: B-2-C with backchannel

- ECMs contain URL of e-commerce server offering content keys
- Rx decoder reports URL back on missing subscription
- GUI invokes Web browser with URL (public key part of HTTP GET/POST)
- E-Commerce server
 - **Content key shop!**

DAB CA: B-2-C with backchannel

- ECMs contain URL of e-commerce server offering content keys
- Rx decoder reports URL back on missing subscription
- GUI invokes Web browser with URL (public key part of HTTP GET/POST)
- E-Commerce server
 - **Content key shop!**
 - Delivers **key file** (application/dab-capk)

DAB CA: B-2-C with backchannel

- ECMs contain URL of e-commerce server offering content keys
- Rx decoder reports URL back on missing subscription
- GUI invokes Web browser with URL (public key part of HTTP GET/POST)
- E-Commerce server
 - **Content key shop!**
 - Delivers **key file** (application/dab-capk)
- Receiver: KeyFile helper application stores downloaded key file

DAB CA: B-2-C without backchannel

- Same as B-2-C CA with backchannel...

DAB CA: B-2-C without backchannel

- Same as B-2-C CA with backchannel...
- Receiver stores URLs in memory

DAB CA: B-2-C without backchannel

- Same as B-2-C CA with backchannel...
- Receiver stores URLs in memory
- Once connected to downloading device, scheme works as described for B-2-C/mobile case

DAB CA: B-2-C without backchannel

- Same as B-2-C CA with backchannel...
- Receiver stores URLs in memory
- Once connected to downloading device, scheme works as described for B-2-C/mobile case
- Or: download via separate purchasing device (PC, mobile phone)
 - Store on IBM Microdrive® (150 yrs of subscriptions)
 - Store content keys for all registered receivers
 - Use Microdrive with any of the receivers

DAB CA: Current IBM Offering

- **Tx system**
 - Encryption (scrambling) module
 - Key management system

DAB CA: Current IBM Offering

- **Tx system**
 - Encryption (scrambling) module
 - Key management system
- **Clearing house**
 - Key encapsulation system

DAB CA: Current IBM Offering

■ Tx system

- Encryption (scrambling) module
- Key management system

■ Clearing house

- Key encapsulation system

■ Rx system

- Decryption (descrambling) module
- Key wallet
- Key agent (dealing with clearing house)

DAB CA: Current IBM Offering

■ Tx system

- Encryption (scrambling) module
- Key management system

■ Clearing house

- Key encapsulation system

■ Rx system

- Decryption (descrambling) module
- Key wallet
- Key agent (dealing with clearing house)

Any Questions? :=)

