**DRM with D3R**

목차

[**1** **목적** 2](#_Toc527360891)

[**2** **GStreamer에 관하여** 2](#_Toc527360892)

[**3** **Block Diagram** 2](#_Toc527360893)

[**4** **ECDH** 3](#_Toc527360894)

[4.1 모듈의 정의 3](#_Toc527360895)

[4.2 Block Diagram 3](#_Toc527360896)

[4.3 API 4](#_Toc527360897)

[**5** **키생성 방식** 5](#_Toc527360898)

[**6** **Demo** 5](#_Toc527360899)

[**7** **Git Address** 5](#_Toc527360900)

1. **목적**

네오와인의 칩 DORCA3\_RIM(D3R)을 응용한 Rasberry pi DRM(Digital rights management) platform 을 GStreamer을 이용하여 구성 및 시행.

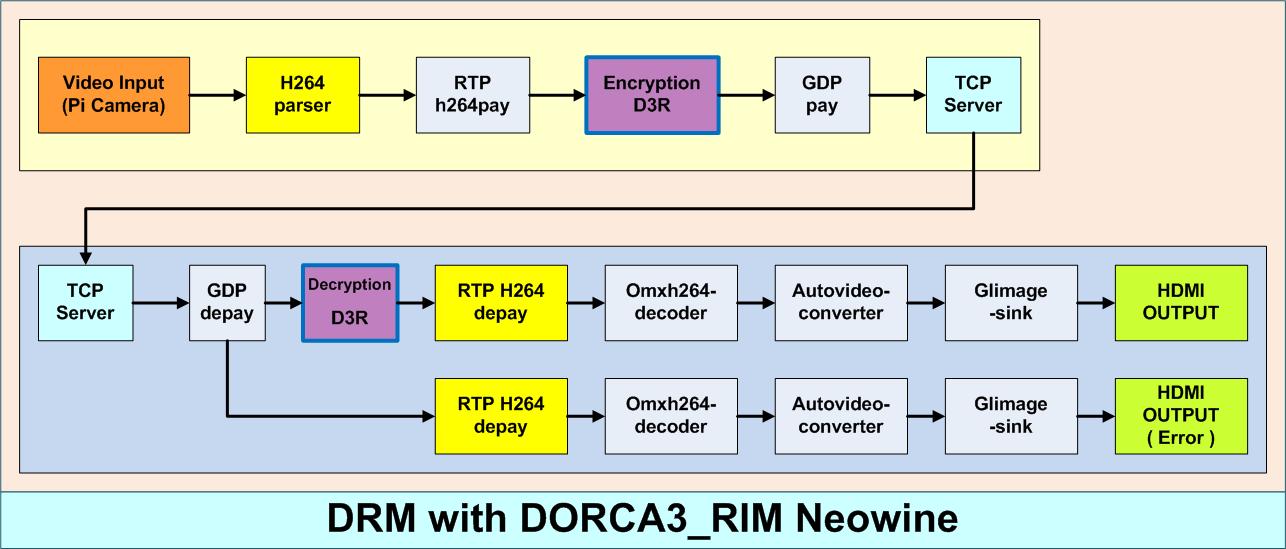
1. **GStreamer에 관하여**

GStreamer는 GObject 기반 자료형 체계와 더불어 C 프로그래밍 언어로 작성한 파이프라인 기반 멀티미디어 프레임워크이다.

GStreamer는 단순 오디오 재생, 오디오 및 비디오 재생, 녹음, 스트리밍 및 편집을 비롯하여 프로그래머가 다양한 미디어 관리 구성 요소를 만들 수 있게 해 준다. 파이프라인 디자인은 영상 편집기, 스트리밍 미디어 브로드캐스터, 미디어 플레이어와 같은 수많은 종류의 멀티미디어 응용 프로그램을 만드는 역할을 한다.

https://ko.wikipedia.org/wiki/GStreamer

1. **Block Diagram**



Gstreamer 파이프라인을 이용한 video streaming 방식이며, 동영상을 보낼땐 GDPpay의 source pad를 프로빙이란 방법으로 data를 encoding하였으며 받을땐 GDPdepay의 sink pad를 똑같은방똑 같은 data를 decoding 하였다.

두개의 라즈베리파이를 이용한 DRM 시스템이다. 각각 server 그리고 receiver 역할을 하며 두 매체 간에 이루어지는 연결은 TCP/IP로 진행이 된다. Server이 제공하는 data는 Pi camera의 input이며 receiver은 이 받은 data를 HDMI를 이용해 output을 한다. Encryption과 Decryption은 D3R로 적용하는데 이때 D3R의 작동여부는 Raspberry Pi의 GPIO를 이용해서 ON/OFF 여부를 제어할 수 있다.

1. **ECDH**

4.1 모듈의 정의

ECDH P256

4.2 Block Diagram



두 사용자가 안전하게 비밀이 보장된 공통키를 만드는데 사용한다

위 그림에서 A USER는 자신의 private key DA 를 만들고 DA \* G하여 HA를 만들고 B USER에 전달한다. B USER는 자신의 private key DB 를 만들고 DB \* G하여 HB를 만들고 A USER에 전달한다

최종단계로, A USER는 S = DAHB = DADBG의 공통키를 만든다.

B USER는 S = DBHA = DADBG의 공통키를 만든다.

4.3 API

**함수명 : EcdhGenPubKey**

**설명 :** 입력한 비밀키로부터 공개키를 만듬

**파라메터:**

uint8\_t\* sk[in] 비밀키 length: 32

struct \_point\* p1[out] 공개키 struct \_point {

uint8\_t x[32];

uint8\_t y[32];

}

**함수명 : EcdhGenPubKeyPuf**

**설명 :** 랜덤값을 만들고 이를 비밀키로 사용하여 공개키를 만듬

**파라메터:**

struct \_point\* p1[out] 공개키 struct \_point {

uint8\_t x[32];

uint8\_t y[32];

}

**함수명 : EcdhGenSessionKey**

**설명 :** 입력한 비밀키와 공개키로부터 최종 공통키를 만듬

**파라메터:**

uint8\_t\* sk[in] 비밀키 length: 32

struct \_point\* p1[out] 공개키 struct \_point {

uint8\_t x[32];

uint8\_t y[32];

}

**함수명 : EcdhGenSessionKeyPuf**

**설명 :** 이전에 생성한 비밀키로 사용하여 최종 공통키를 만듬 사용전 ecdh\_gen\_pub\_key\_puf를 미리 호출하여 비밀키를 만들어야함

**파라메터:**

struct \_point\* p1[out] 공개키 struct \_point {

uint8\_t x[32];

uint8\_t y[32];

}

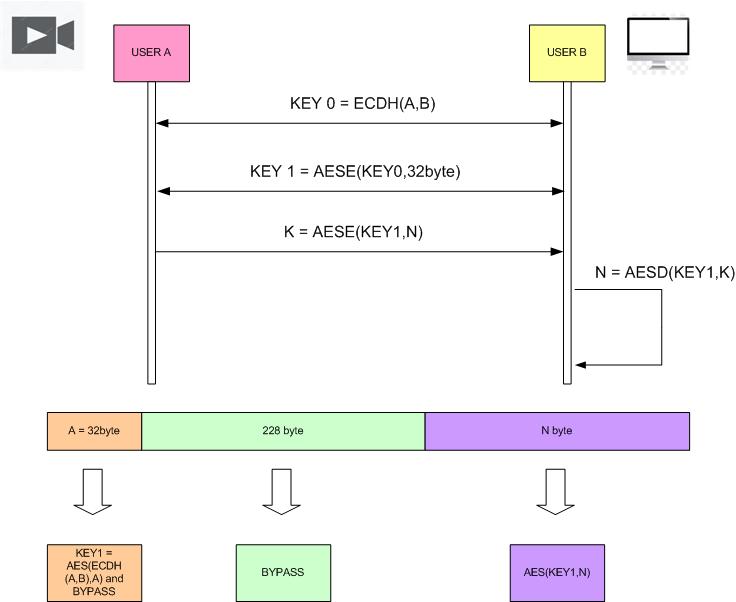
**함수명 : KeyFromEcdh**

**설명 :** ECDH를 통해 생성한키를 AES\_X0에 탑재시킨다.

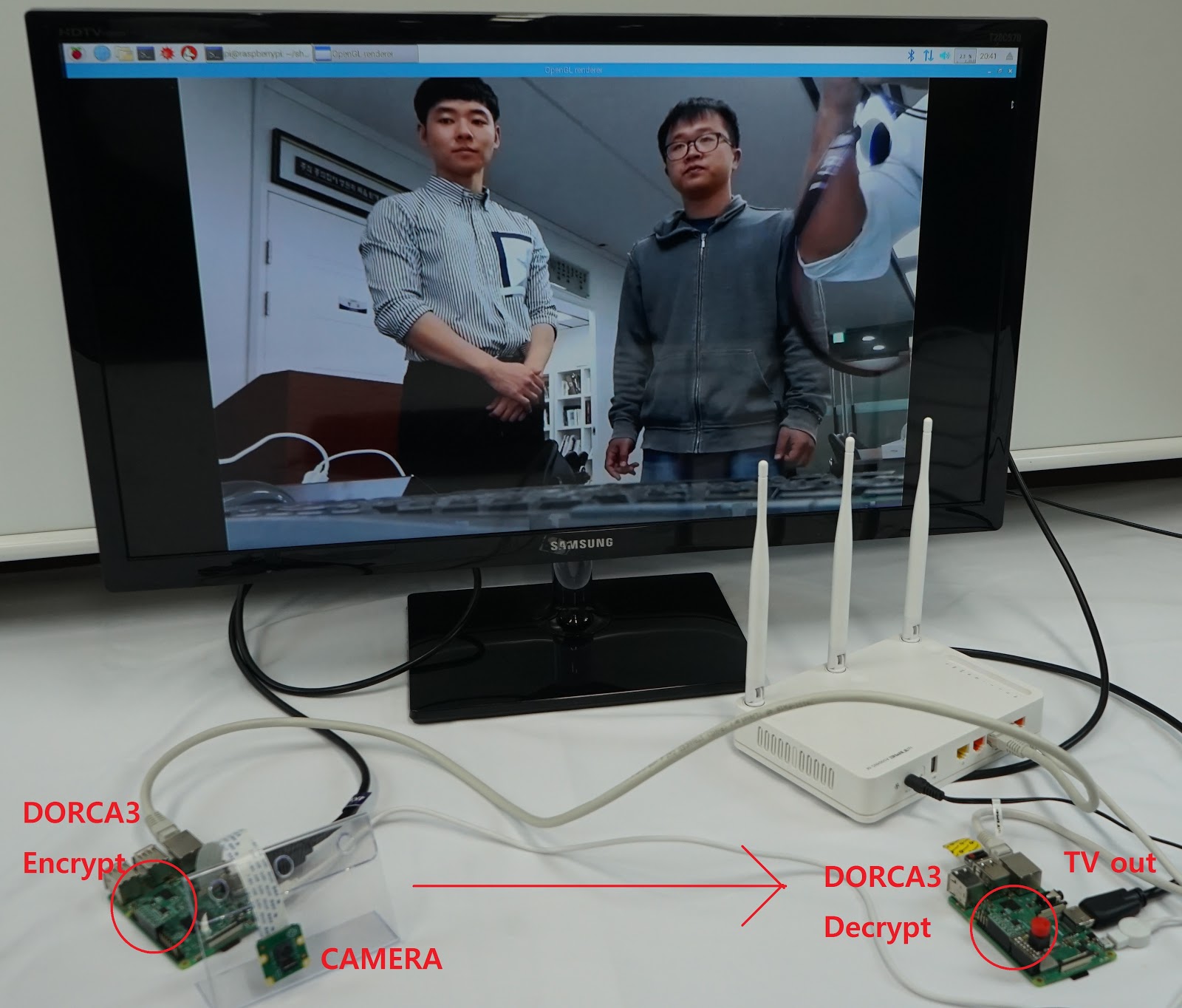
**파라메터:**

없음

1. **키생성 방식**

****

1. **Demo**

****

1. 라즈베리 파이 두대를 준비하고 하나는 server 그리고 하나는 receiver로 지정한다.
2. Git clone으로 <https://github.com/neowine/DRM_D3R>에서 필요한 코드를 두 라즈베리 파이에 다운받는다.
3. Server는 gst-sender의 function folder에서 raspi\_streaming.c, receiver는 gst-receiver의 function folder에서 raspi\_streaming\_player.c에서 ip주소들을 sender pi의 ip주소로 바꿔둔다.
4. 만약 switch로 D3R을 제어하길 원한다면 raspi\_streaming\_player.c의 cb\_have\_data에서 digitalRead function의 input을 원하는 GPIO핀에 세팅해준다.
5. Server와 receiver 둘다 D3R을 올바른 GPIO핀 위치에 장착시키고 server에는 pi camera도 설치한다
6. Server는 gst-sender의 폴더에서, 그리고 receiver는 gst-receiver 폴더에서 make를 해주고 ./test커맨드를 이용해서 실행시킨다. (단, sender을 먼저 실행시켜야 한다.)
7. 한번 make를 한 상태에서 추후에 DRM\_D3R을 실행시키고 싶을땐 DRM\_D3R 폴더에 들어가 해당 라즈베리 파이에 맞는 ./sender.sh 아니면 ./receiver.sh커맨드로 손쉽게 실행시킬 수 있다.
8. **Git Address**

http://github.com/neowinepub/DRC\_D3R