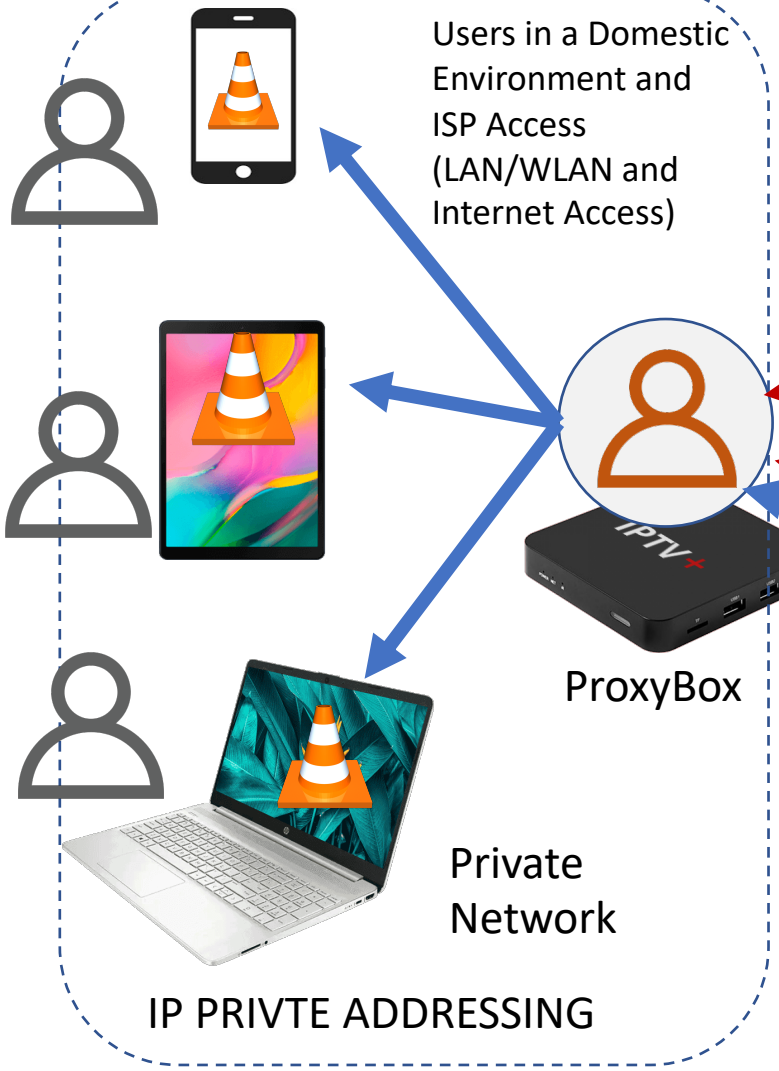
PA#1

A secure “pay-per-view” real-time media streaming system

System Model

Cloud-Provided Pay-Per-View Streaming as a Service



SAPKDP PROTOCOL

SRTSP Protool

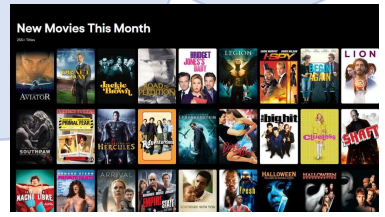
Signaling Server

Registered Users & ProxyBoxes

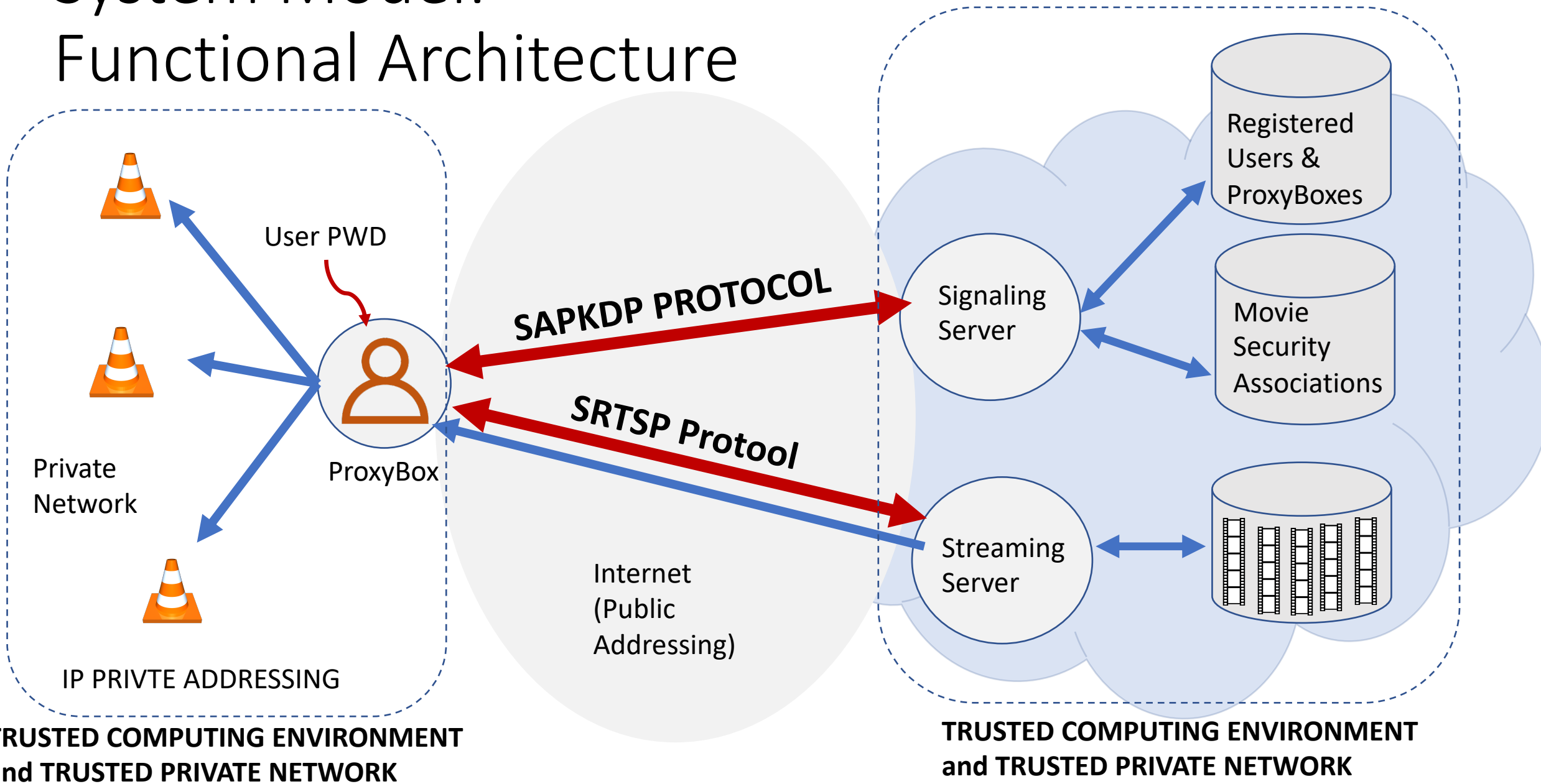
Movie Security Associations

Streaming Server

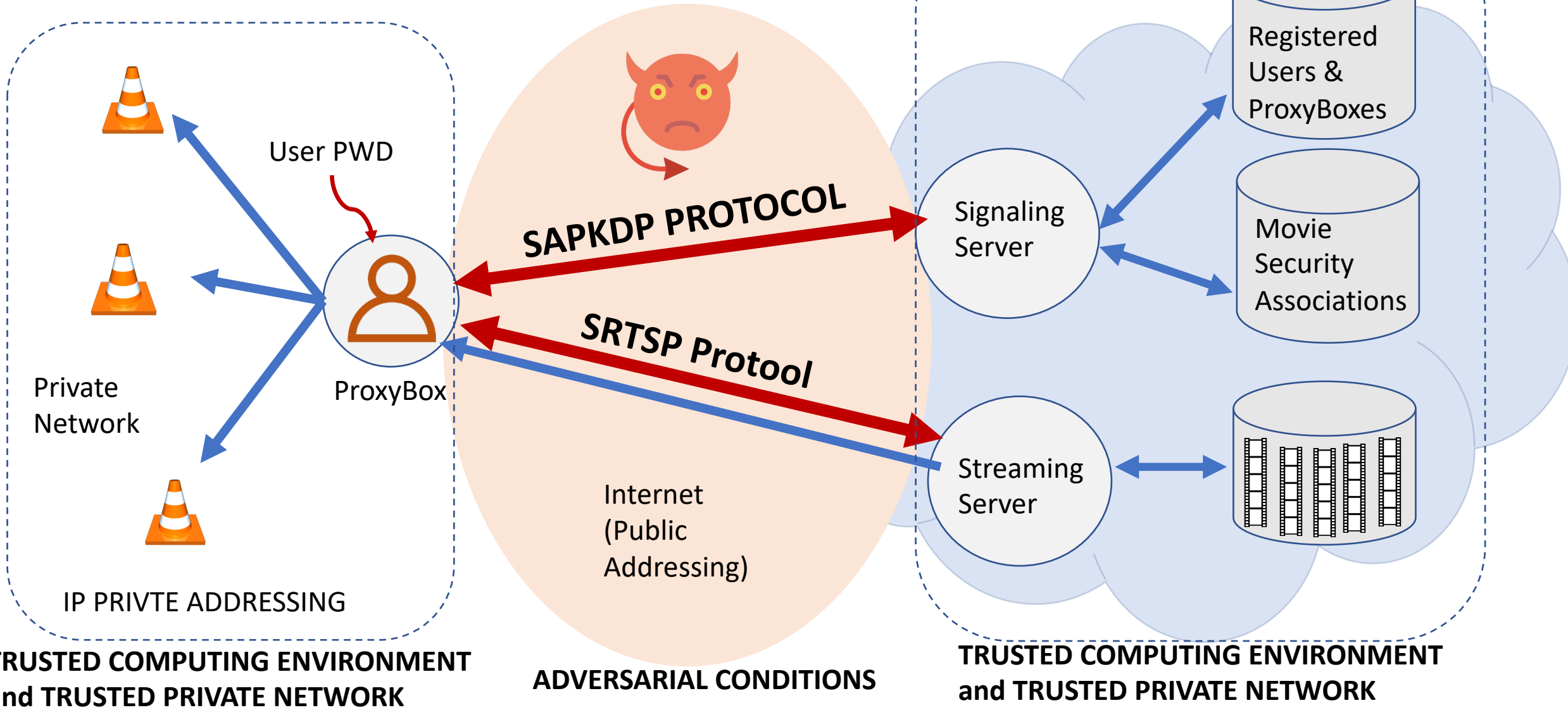
Internet (Public Addressing)



System Model: Functional Architecture



System Model and Trust Model



TRUSTED COMPUTING ENVIRONMENT and TRUSTED PRIVATE NETWORK

ADVERSARIAL CONDITIONS

TRUSTED COMPUTING ENVIRONMENT and TRUSTED PRIVATE NETWORK

Adversary Model

Communication Endpoints

X509 Framework Attack Types Considered:

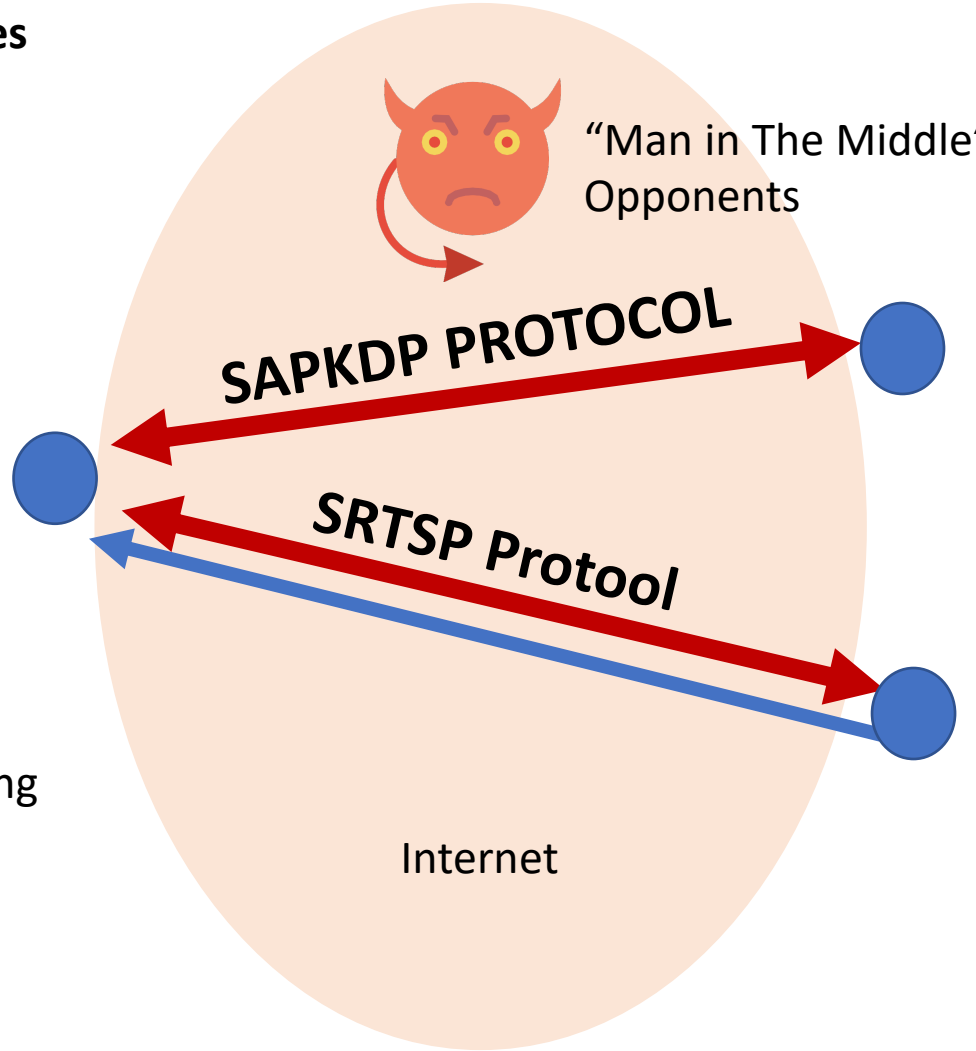
Identity and IP Spoofing or Maquerading
(Peer-Authenticity Breaks)

Data-Leakage
(Confidentiality Breaks)

Message/DataFlows Authenticity Breaks

Message/DataFlows Tampering
(Integrity Breaks)

Can do Traffic Analysis



ADVERSARIAL CONDITIONS

Out of the Adversary Model Scope

DoS, DDoS

Ex:
- Network Congestion and/or Saturation
- Availability and Correctness of Endpoints

The protocols involved

- **SAPKDP**

  Secure Authentication, Payment and Key-Distribution Protocol

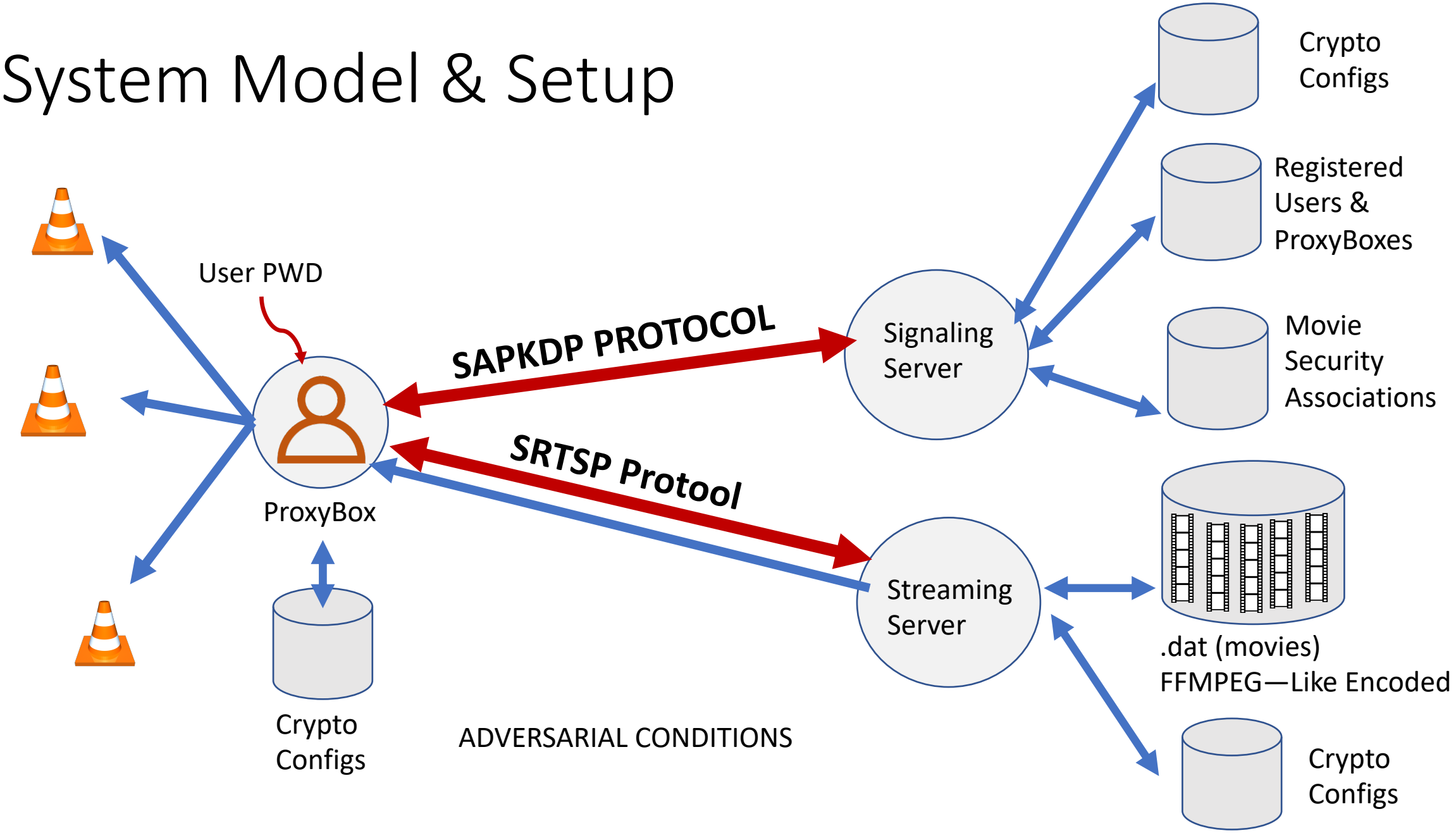
  - Can be implemented over UDP, TCP or HTTP (you can choose the encapsulation in your design and implementation)
  - Important: for the PA#1 delivery you will not use TLS or HTTPS – anyway the protocol will be secure by design and implementation
  - Optionally and later on … you can support it over TLS, DTLS or HTTPS if you want !

- **SRTSP**

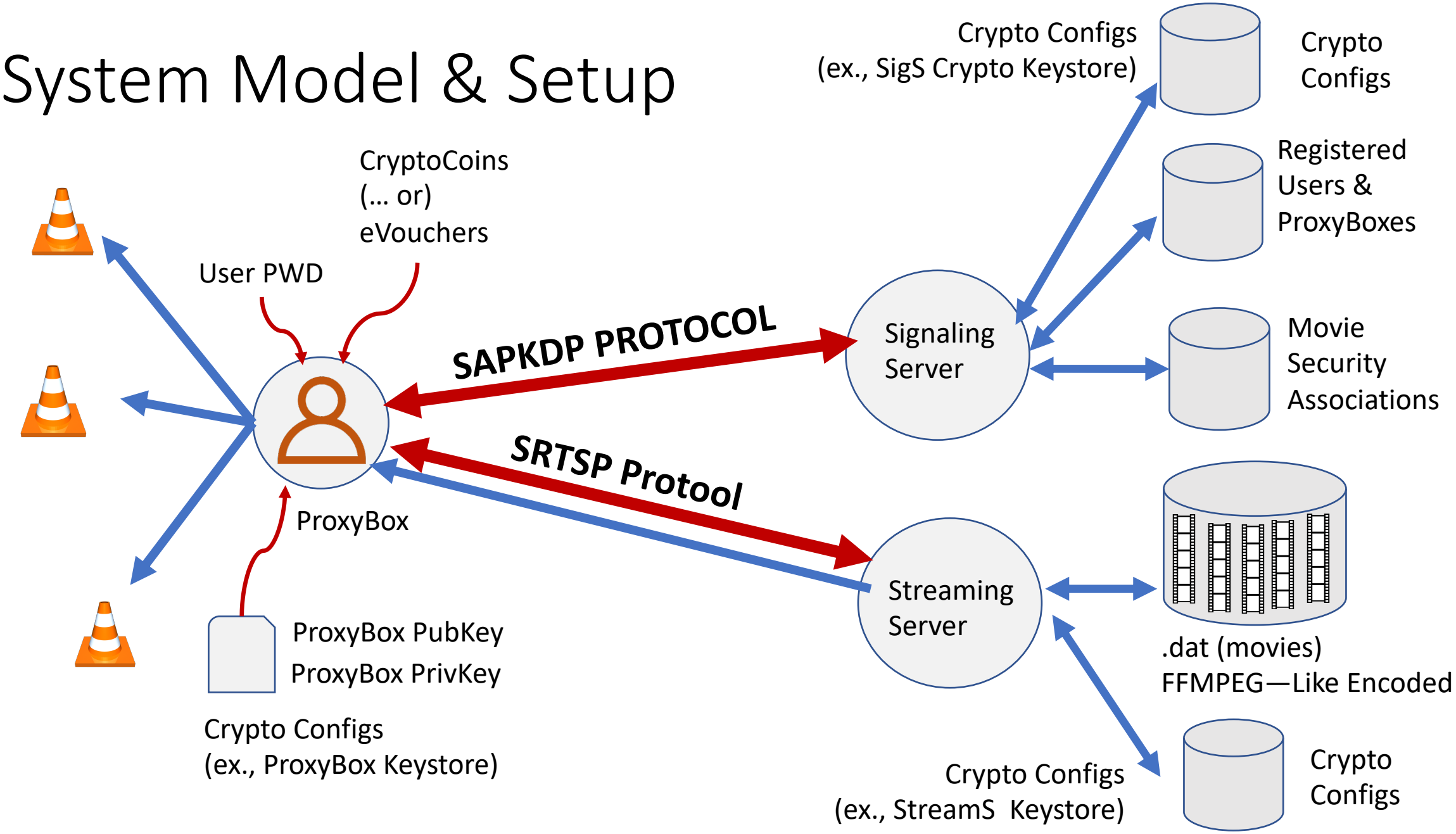
  Secure Real Time Streaming Protocol

  - Must be implemented over UDP !

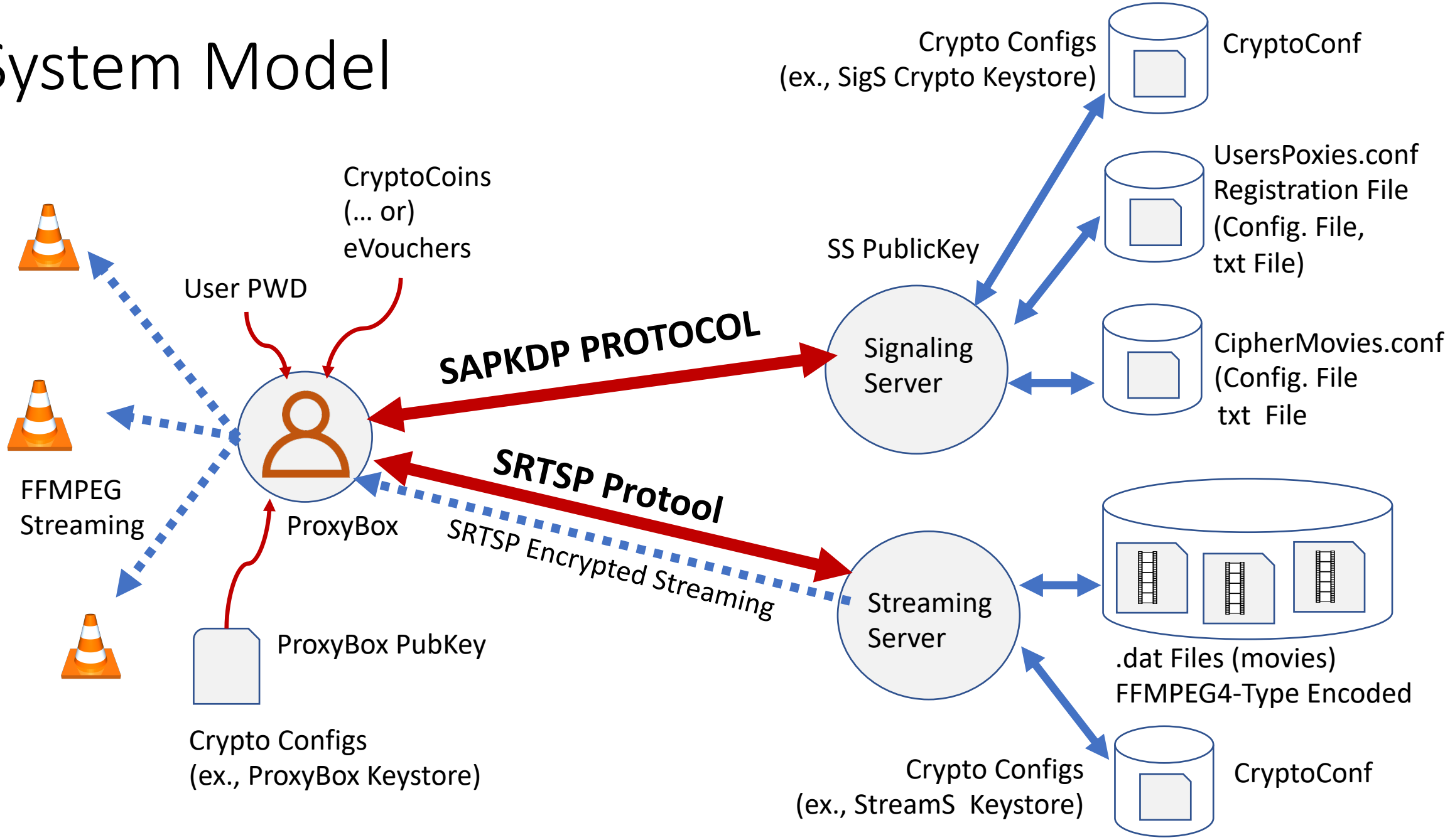
System Model & Setup



System Model & Setup



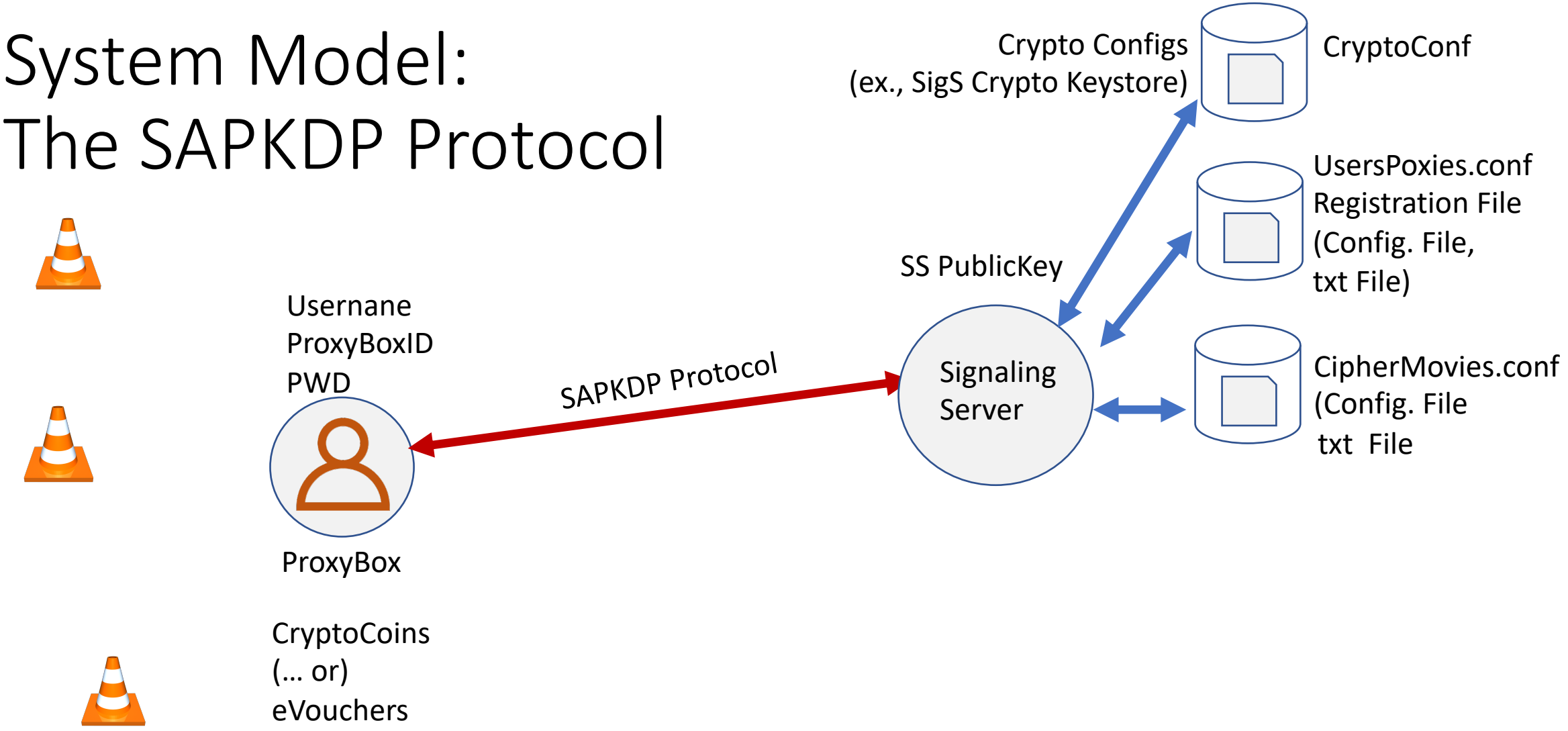
System Model



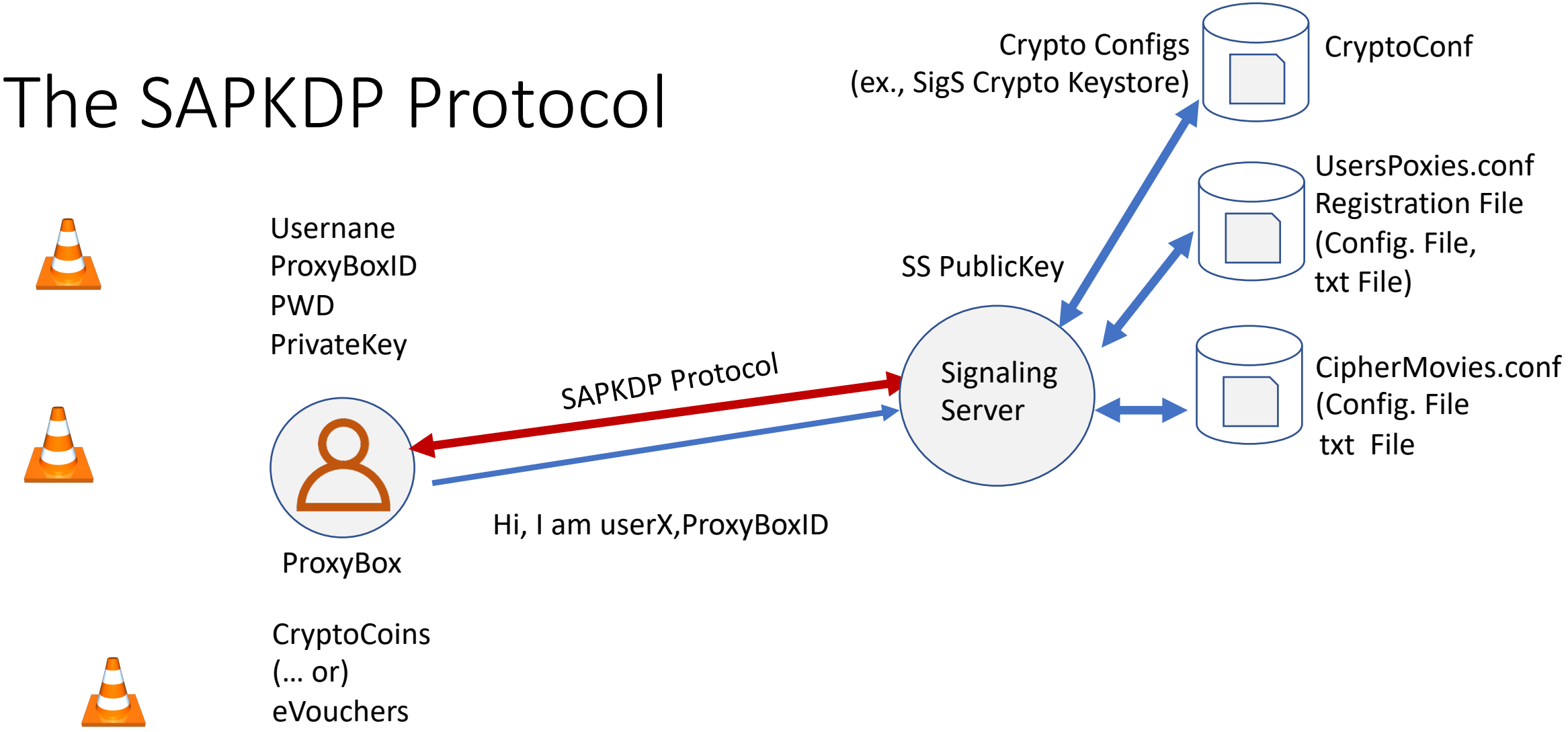
SADKDP Functional Discussion

(Specification formalized in a specific reference doc)

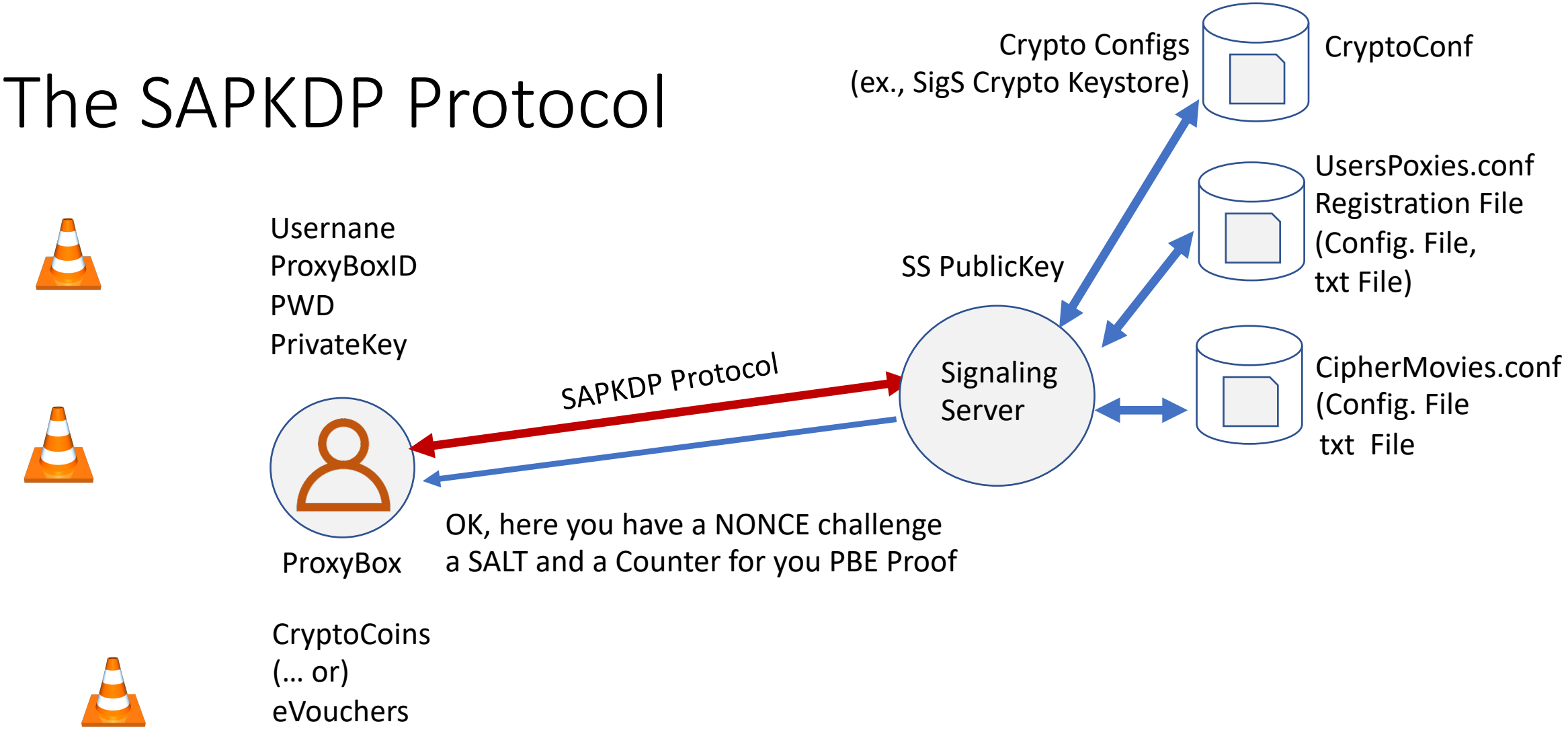
System Model:
The SAPKDP Protocol



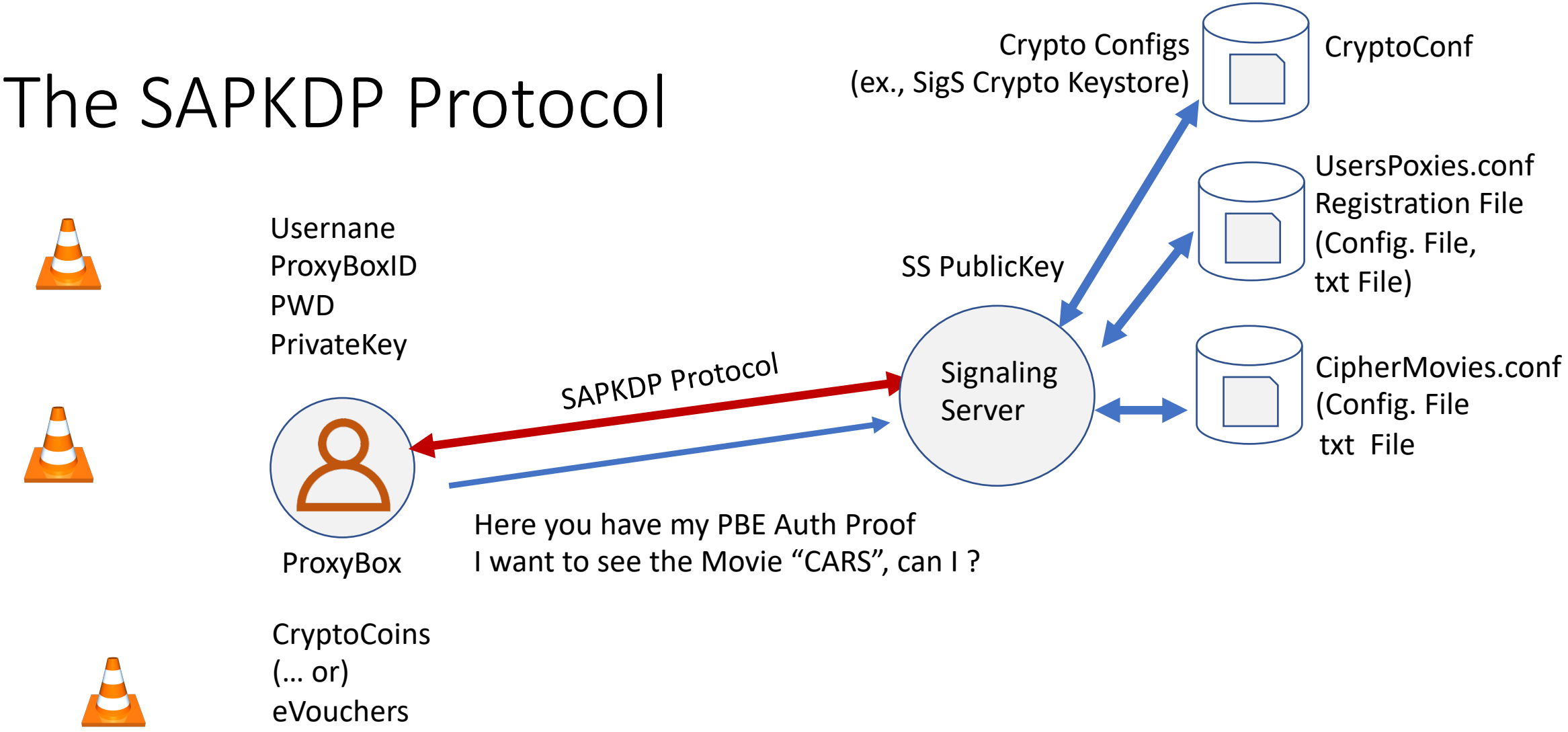
The SAPKDP Protocol



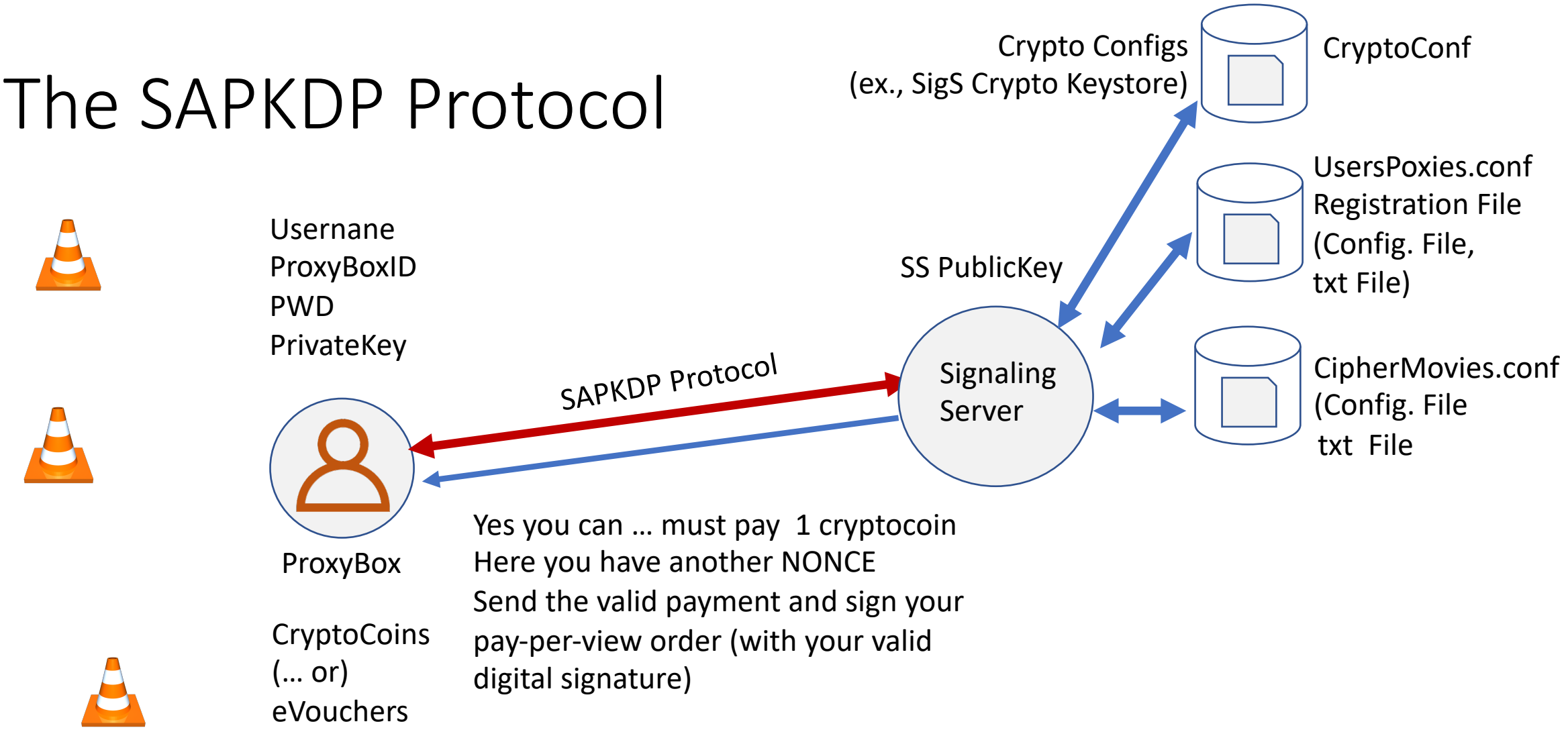
The SAPKDP Protocol



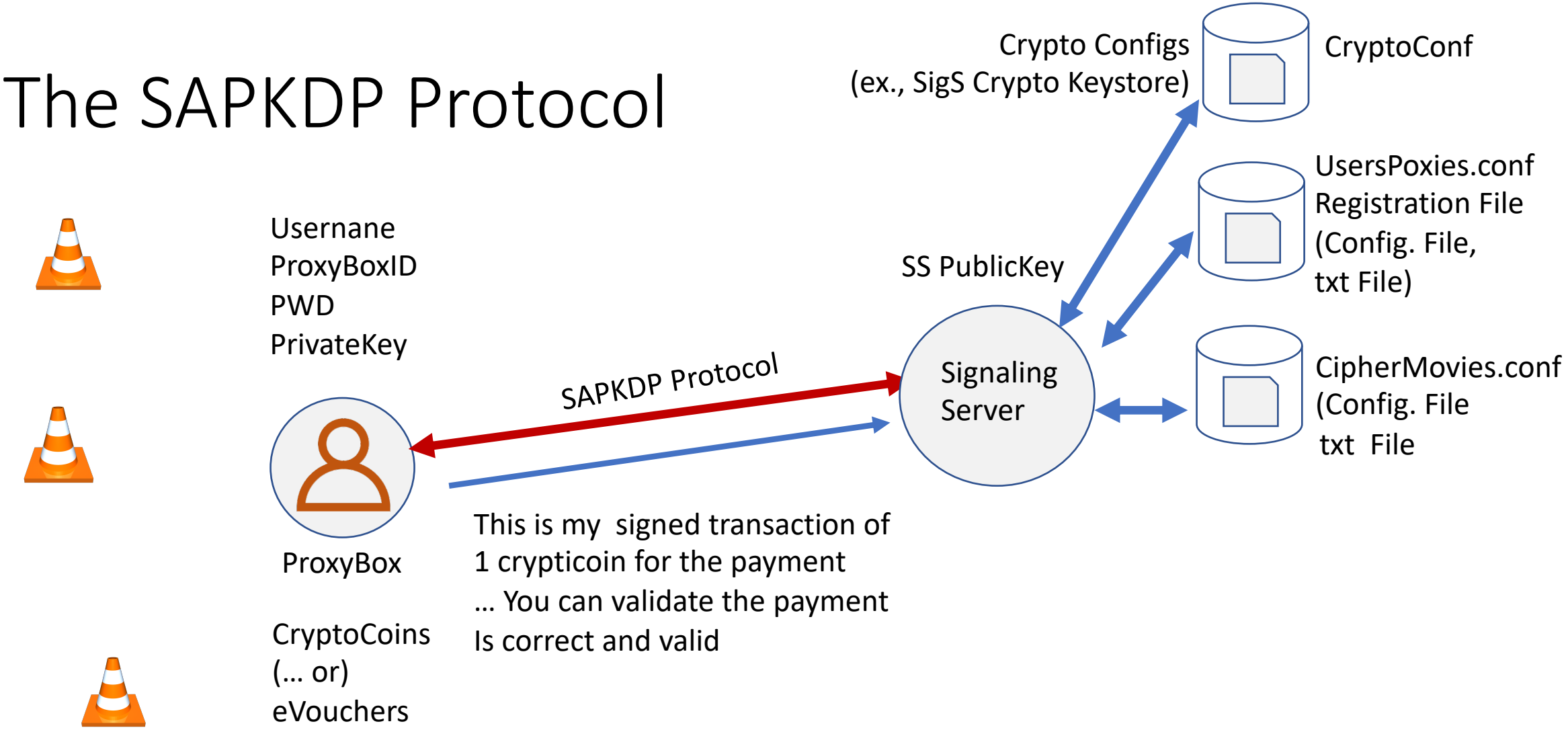
The SAPKDP Protocol



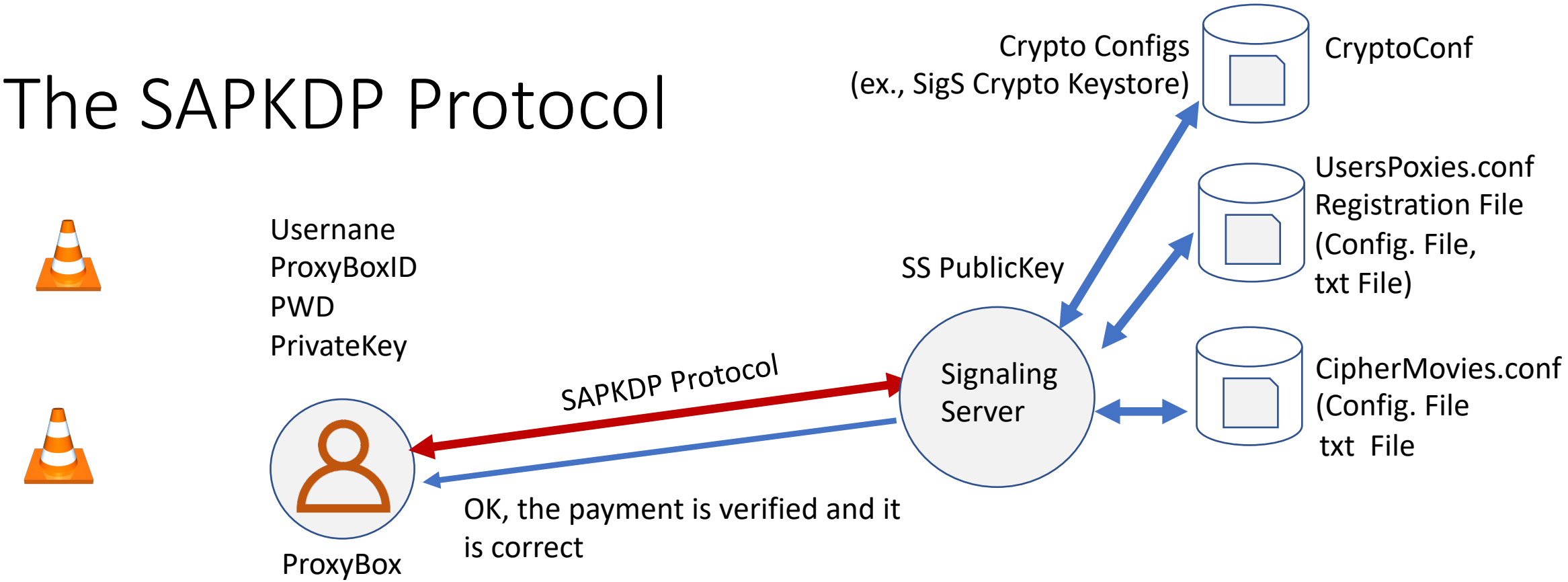
The SAPKDP Protocol



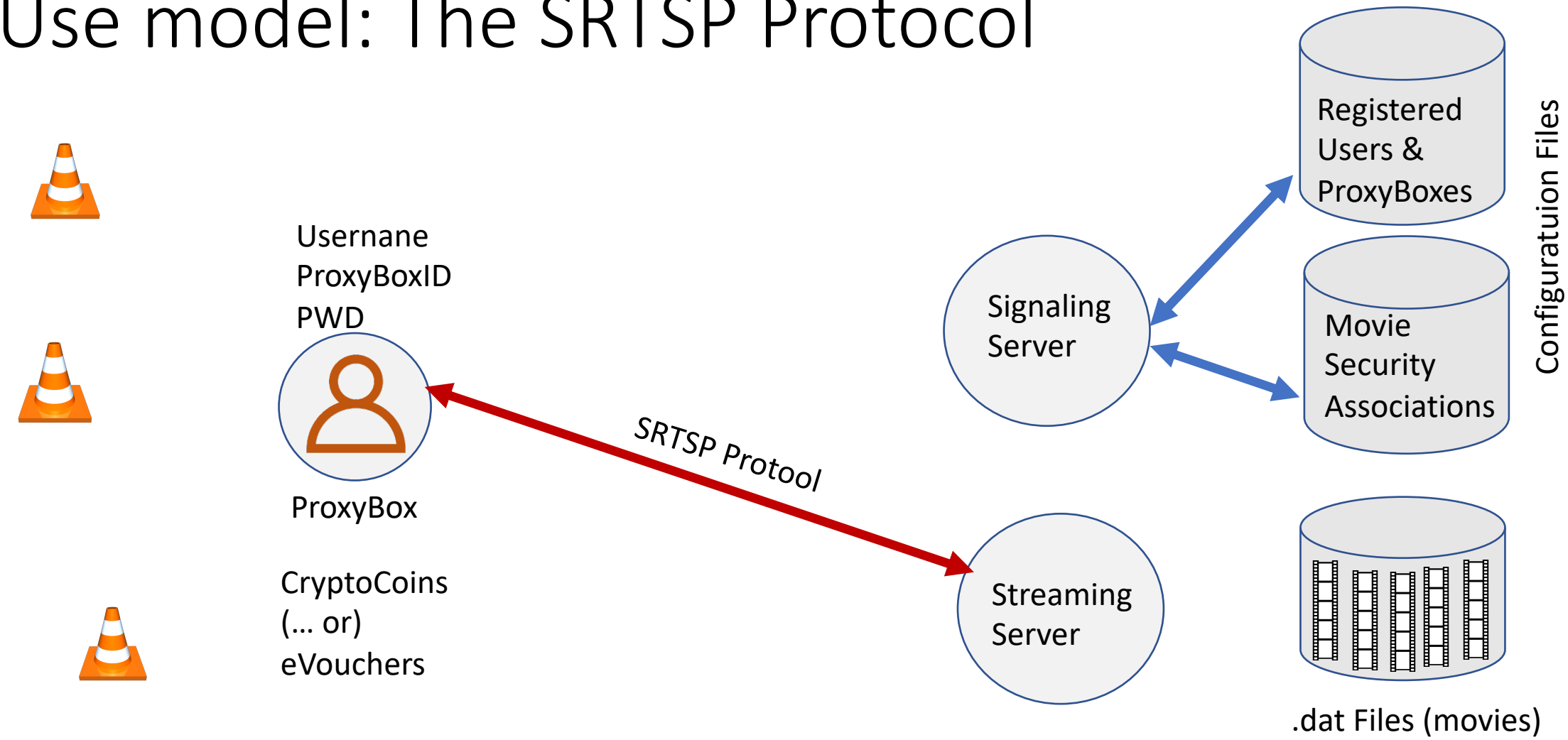
The SAPKDP Protocol



The SAPKDP Protocol



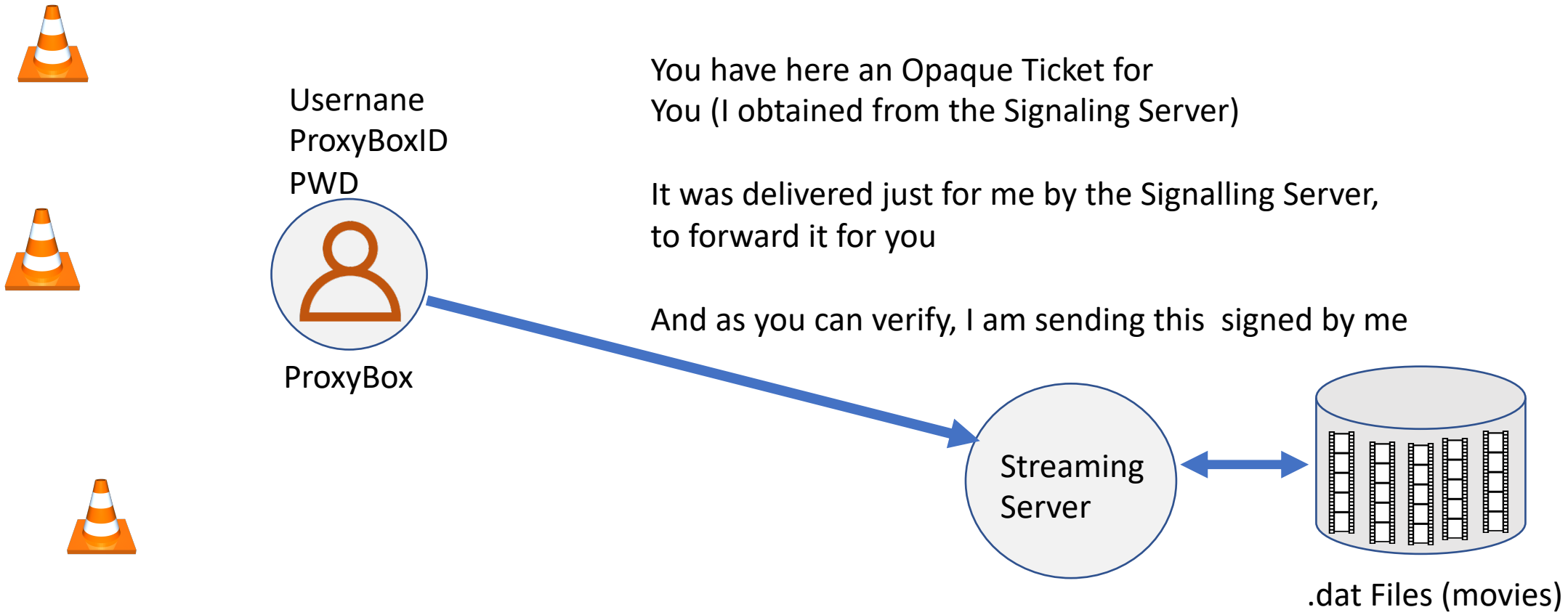
Use model: The SRTSP Protocol



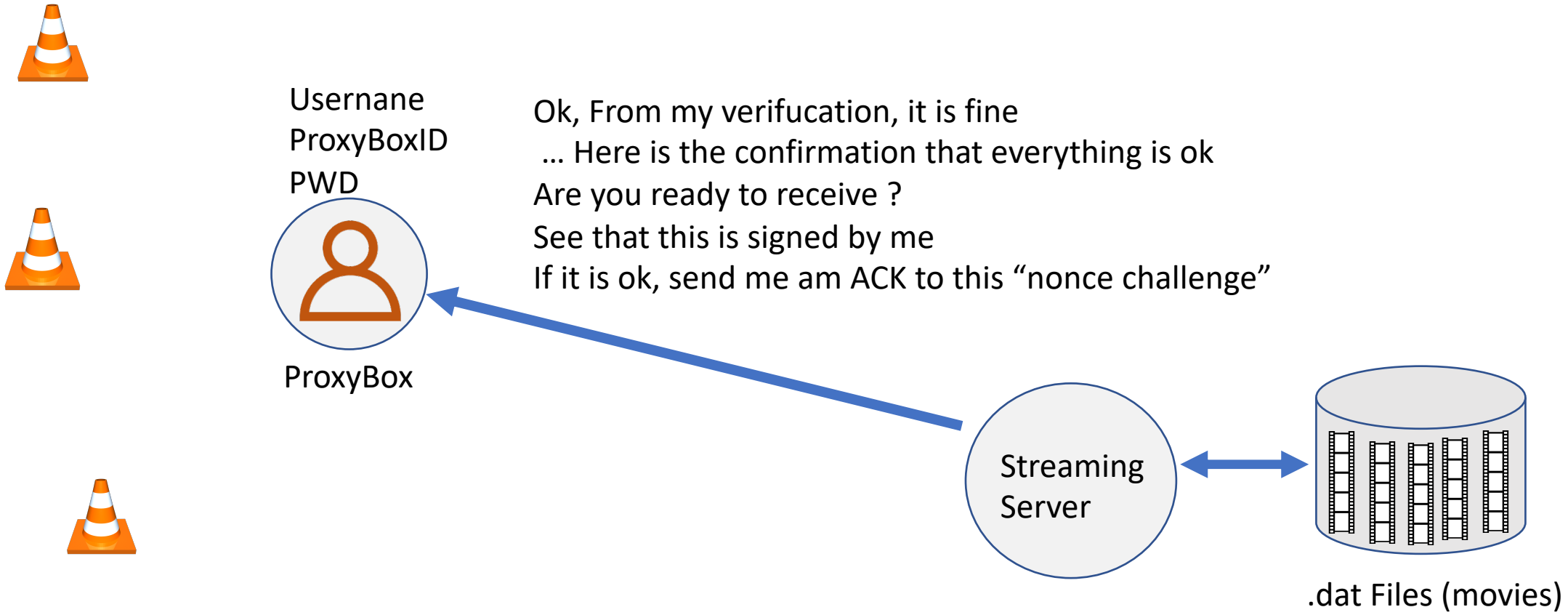
SRTSP Functional Discussion

(Specification formalized in a specific reference doc)

The SRTSP Protocol

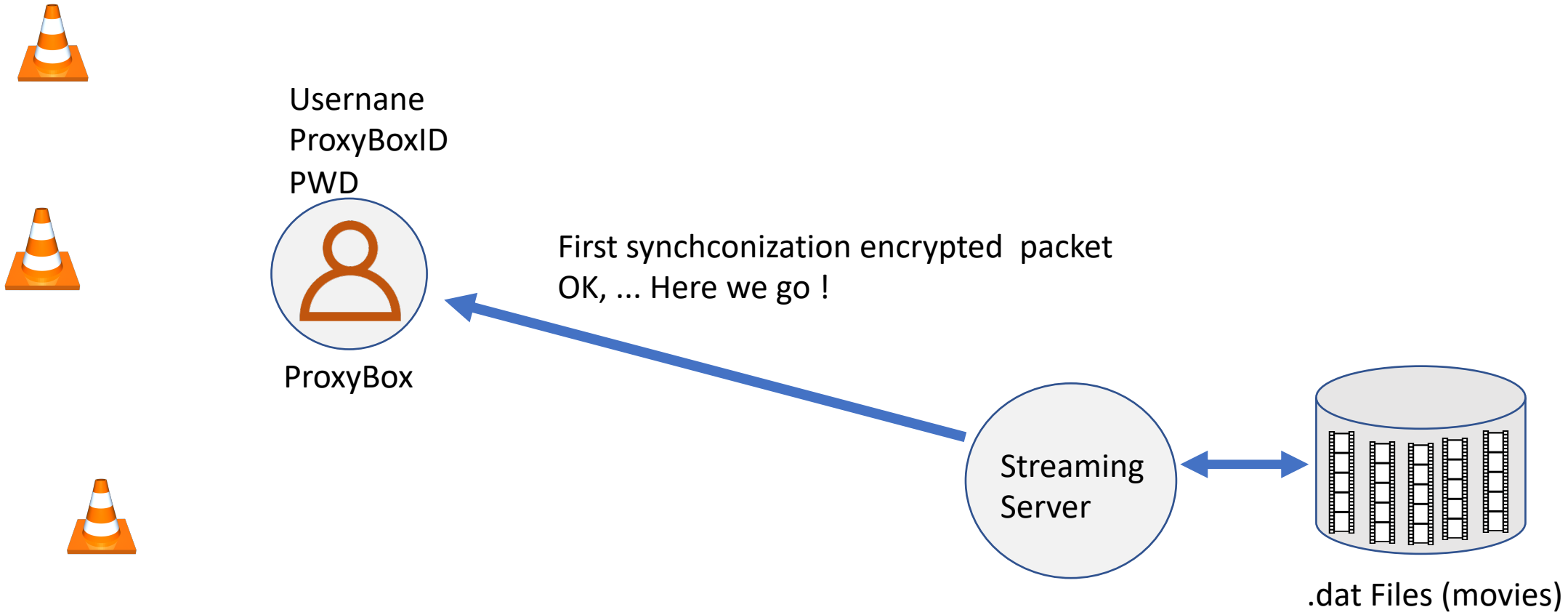


The SRTSP Protocol

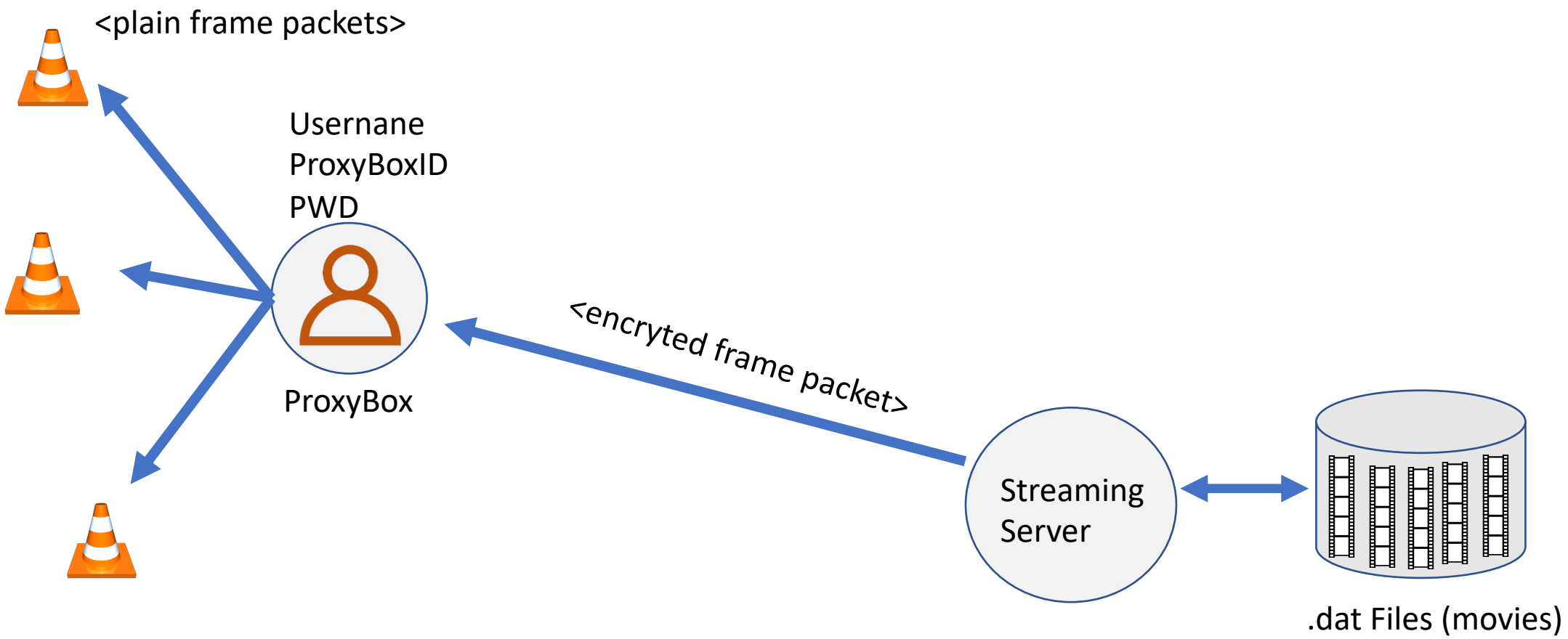


The SRTSP Protocol

The SRTSP Protocol



The SRTSP Protocol and Real-Time Playing



Implementation

PA#1 developed in 3 Stages:
Step 1 (or STAGE 1)
Initial Approach to a “simplified” version of SRTSP Protocol
No Signaling Server, No SAPKDP Protocol, no Coins and no PayPerView

Step 2 (or subsequent STAGE 2)
Signaling Server
SAPKDP
SRTSP
(SAPKDP and SRTSP Complete Integration and everything else)

Ref. Evaluation

12/20 Points
~8-12 / OCT
(Prelim. Demos on LABs
15-17 / OCT)

20/20 Points

Delivery + Quiz:
Ref: 24/NOV

“Hands On”: Here you go: Stage 1 !
(Simple SRTSP Version and simplified vision of the desired system)

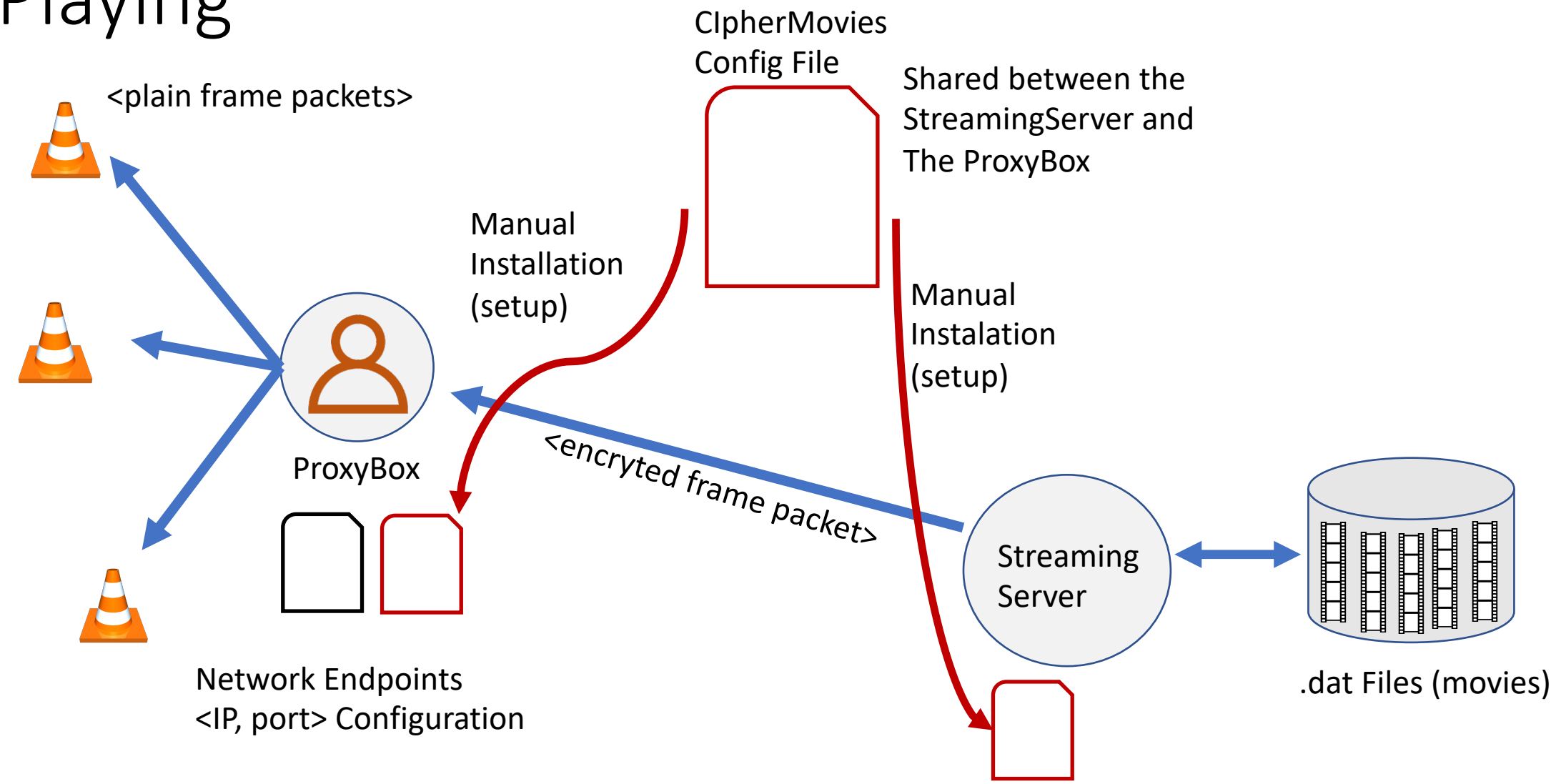
STEP 1 Implementation

Your first step challenge (suggestion)

- Start with the SRTSP Protocol
- A direct “step” forward to the provided material (Lab 3, Streaming and Materials)
  - You have Movies (.dat) and the initial (unsecure) StreamingServer and Proxy implementation
  - Ready to be used (unsecurely)!
- You can develop the ProxyBox and StreamingServer extending the provided implementations
  - You have the base materials and of course the VLC tool
- Can start by using “static” configurations
  - Configuration files manually installed in ProxyBOX and Streaming Server
- The SRTSP for the streaming phase only requires Symmetric Crypto and MACs
  - BUT the USABLE CRYPROGRAPHY MUST BE CONFIGURABLE !!!!

Later on, the static configurations must be removed after the complete implementation of the SRTSP and SAPKDP protocols, as well as the Signaling Server

The “simplified” SRTSP Protocol for Real-Time Playing



Initial static configuration for the SRTSP Protocol and its implementation in the Streaming Server and in the ProxyBox

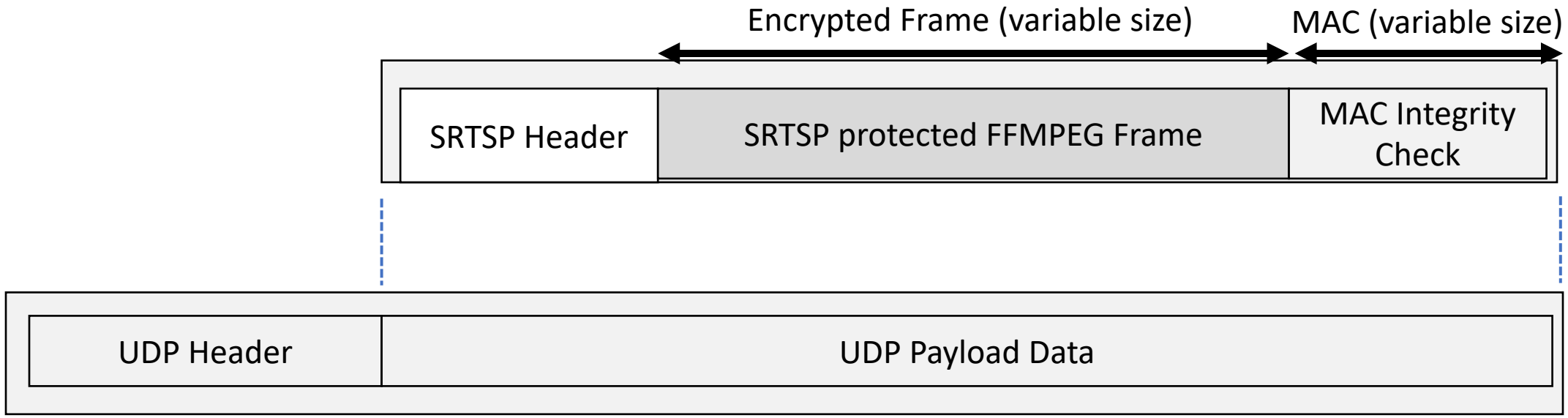
- Can choose any valid “static” configurations
  - CRYPTOGRAPHICALLY CONFIGURABLE !!!!

Later on (Next Steps), this file will be not shared between the Streaming Serve and the ProxyBox

The setup will be dynamic after the complete implementation of the SRTSP and SAPKDP protocols, as well as the Signaling Server

Simplified version of SRTSP and SRTSP encapsulation over UDP and Datagram Packets

SRTSP Header contains:
4 bits: contains the version id of the Simplified SRTSP protocol: 0001
4 bits: contains an indication that the setup is manual: 0000 means manual configuration of endpoints
16 bits: integer, contains the size in bytes of the encapsulated encrypted frame



For this week ! (1)

- Try to address the first proposed challenge (STEP 1) for he SRTSP
- You only need to start form the materials you already have

  StreamingServer

  Proxy
- Try to avoid complexity in the base StreamingServer and Proxy
  - You must implement with a modular approach !
  - Extend the DatagramSocket or Datagram Packet Classes
  - Ex: MySRTSPDatagramSocket ….. MySRTSPDatagramPacket
  - The minimum number of lines changed in the StreamingServer and Proxy Class, the better and more moduar will be you first solution
  - VERY IMPORTANT FOR THE NEXT PHASE !

For this week (2)

Try to avoid complexity in the base StreamingServer and Proxy

- You must implement with a modular approach !
- Extend the DatagramSocket or Datagram Packet Classes
- Ex: MySRTSPDatagramSocket ….. MySRTSPDatagramPacket
- The minimum number of lines changed in the StreamingServer and Proxy Class, the better and more moduar will be you first solution
- VERY IMPORTANT FOR THE NEXT PHASE !

How Many LoC do you have in your Secure Proxy and Secure Stream Server implementation ? +/- 10% diff max. comparing w/ the original code ?

Is your configuration neutral ? Can we use any cuphersuite configuration ?