SADKDP Protocol Specification: Generic Functional Description Overview

Protocol runs in 6 rounds

Protocol Identification Type: 1

Message Types: 1,2,3, … etc

Entities involved: ProxyBox (PB) , Signaling Server (SS)

Generic Functional Description Overview

Ent.Flow	Message description	Functional Description	M. Type
PB > SS Round 1	PB-Hello	Hi, I am userX,ProxyBoxID	1
SS>PB Round 2	SS-AuthenticationRequest	OK, here you have a NONCE challenge a SALT and a Counter for you PBE Proof	2
PB > SS Round 3	PB-Authentication	Here you have my PBE Auth Proof I want to see the Movie “CARS”, can I ?	3
SS>PB Round 4	SS-PaymentRequest	Yes you can ... must pay 1 cryptocoin Here you have another NONCE Send the valid payment and sign your pay-per-view order (with your valid digital signature)	4
PB>SS Round 5	PB-Payment	This is my signed transaction of 1 crypticoin for the payment ... You can validate the payment Is correct and valid	5
SS>PB Round 6	SS-TicketCredentials	OK, the payment is verified and it is correct I am sending all the info your need for the movie you want, protected and just for you and signed by me : • ENDPOINT (IP & Port) • ciphersuite conf • cryptograhic materials ad keys • Opaque Info (encrypted ticket> you must send to the stream server	6

SADKDP - Protocol Initial Specification: Rounds and Message Types

Ent.Flow	Message description	M. Type	Functional Description
PB > SS Round 1	PB-Hello	1	UseID, ProxyBoxId
SS > PB Round 2	SS-AuthenticationRequest	2	N1, Salt, Counter
PB > SS Round 3	PB-Authentication	3	PBE$_{\text{UserPwd,Salt,Counter}}$ (N1’,N2,MovidID), IntCheck3
SS > PB Round 4	SS-PaymentRequest	4	ECDSASignature$_{\text{KprivSS}}$ (Price, N2’, N3), IntCheck4
PB > SS Round 5	PB-Payment	5	ECDSASignature$_{\text{KpubPBOX}}$ (N3’, N4, PaymentCoin), IntCheck5
SS > PB Round 6	SS-TicketCredentials	6	{IP, Port, MovieID, ciphersuiteConf, CryptoSA, SessionKey, MacKey, N4’ }$_{\text{KpubPB}}$, {IP, Port, MovieID, ciphersuiteConf, CryptoSA, SessionKey, MacKey, NC1 }$_{\text{KpubRTSS}}$, ECDSASignature$_{\text{KprivSS}}$ (Payloads), IntCheck6

Alert/Error Protocol Message Type

Sent by each specific endpoint if any verification of SADKDP Message Types fail in the cryptographic and content processing

PB SS	PBErrorAlert	90	MType, ErrorCode, IntCheck90
SS > PB	SSErrorALert	91	MType, ErrorCode, IntCheck91

SADKDP Guarantees:

- Traffic Floc Integrity
- Message Integrity of all relevant message Types: 3,4,5,6
- Message Authentication and Integrity Guarantees on IntChecks
- PWD-based authentication and confidentiality in Message Type 5
- Peer-Authentication of payloads in Message Types 5 and 6
- Confidentiality and Peer-Authentication Guarantees in message type 6

SRTSP Protocol Specification: Generic Functional Description Overview

Protocol runs in 4 handshake rounds followed by the secure real-time multimedia streaming for playing

Protocol Identification Type: 2

Message Types: 1,2,3, … etc

Entities involved: ProxyBox (PB) , RealTimeStreamingServer (RTSS)

Generic Functional Description Overview

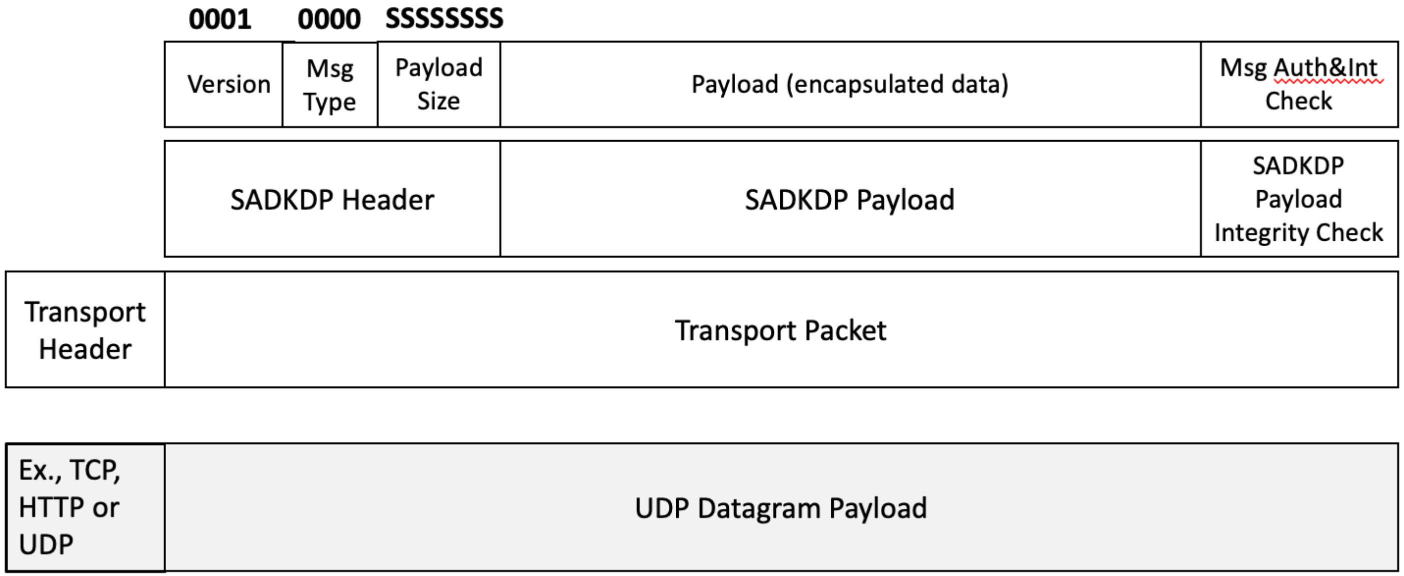
Ent.Flow	Message description	Functional Description	M. Type
PB > RTSS Round 1	PB-RequestAndCredentials	Hy Streaming Server ... I am requesting to see the movie "movie name/id" I send you an Opaque Ticket for (I obtained from the Signaling Server) It was delivered just for me by the Signalling Server after my payment, to forward it for you As you can verify, I am also sending this request signed by me You have here an Opaque Ticket for You (I obtained from the Signaling Server)	1
SS > RTSS Round 2	RTSS-Verification	Ok, From my verification, it is fine and the ticket is valid … ... Here is the confirmation that everything is ok Are you ready to receive ? See that this is signed by me If it is ok, send me am ACK to this "nonce challenge" showing me that you also have the required cryptographic credentials to receive/decrypt and play the movie …	2
PB > RTSS Round 3	PB-AckVertification	Yep, I recognize your signature ... I send the answer to your challenge ... s you see I ready with the right ciphersuites and credentials … So Yep we are now eager, ready, with our "popcorns" ready to start playing!	3
RTSS > PB Round 4	RTSS-SynkInitialFrame	First synchconization encrypted meta-packet OK, ... Here we go next with the movie (frames)!	4
RTSS > PB	EncryptedStreamData	Encrypted Stream Data (Media Frames)	5

RTSTP - Protocol Initial Specification: Rounds and Message Types

Ent.Flow	Message description	M. Type	Specification
PB > RTSS Round 1	PB-RequestAndCredentials	1	{ IP, Port, ciphersuiteConf, CryptoSA, SessionKey, MacKey, NC1 }$_{KpubS}$, Na1 , ECDSASignature$_{KprivSS}$ (Payloads), Intcheck1
RTSS > PB Round 2	RTSS-Verification	2	{ Na1’, Na2, TickeyValidityConfirmation}$_{KS}$, Intcheck2
PB > RTSS Round 3	PB-AckVertification	3	{ Na2, Na3}$_{Ks}$, Intcheck3
RTSS > PB Round 4	RTSS-SynkInitialFrame	4	{ Na3, initmark-frame, …}$_{Ks}$, IntCheck4
RTSS > PB Rounds i	EncryptedStreamData	i	Encrypted Stream Data (Media Frames) { SequenceNumber, Frame }$_{Ks}$, InitCheckF
Round N	RTSS-SynkFinalFrame	N	{endmark-fram}$_{Ks}$, IntCheckN

SADKDP and SRTSP: Protocol Encapsulations

Simplified SRTSP Encapsulation Format



0001 0000 SSSSSSSS

Version | Msg Type | Payload Size | Payload (encapsulated data) | Msg Auth&Int Check

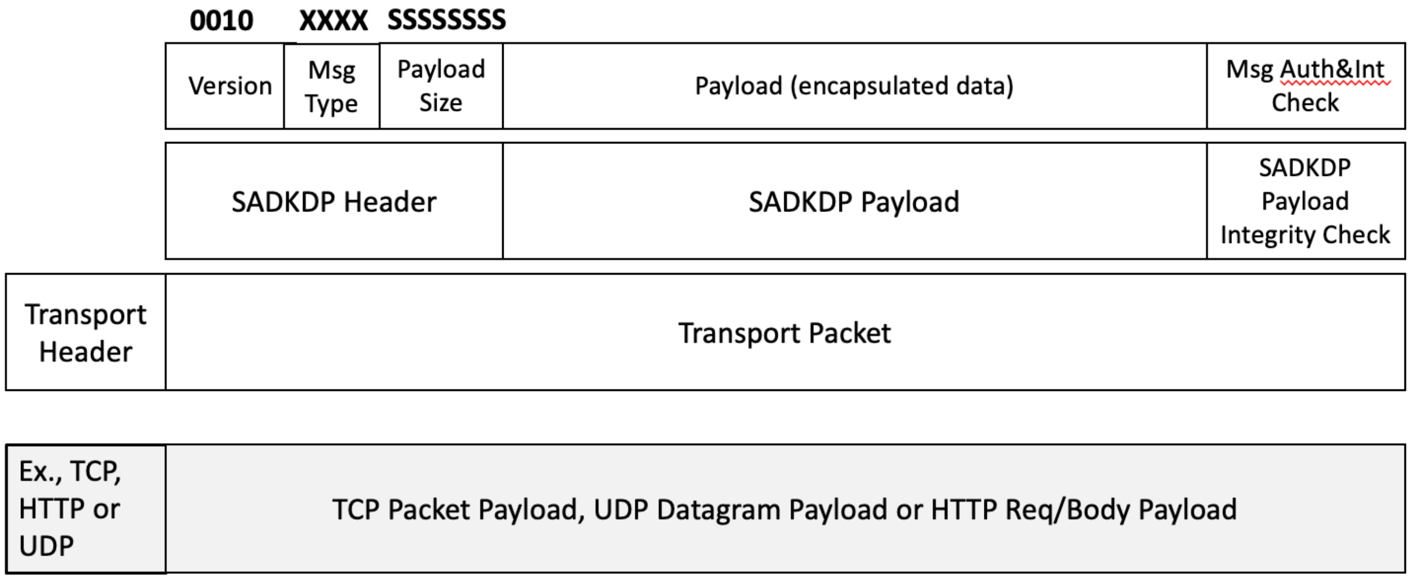
SADKDP Header | SADKDP Payload | SADKDP Payload Integrity Check

Transport Header | Transport Packet

Ex., TCP, HTTP or UDP | UDP Datagram Payload

SADKDP and SRTSP: Protocol Encapsulations

SADKDP Encapsulation Format



0010 XXXX SSSSSSSS

Version | Msg Type | Payload Size | Payload (encapsulated data) | Msg Auth&Int Check

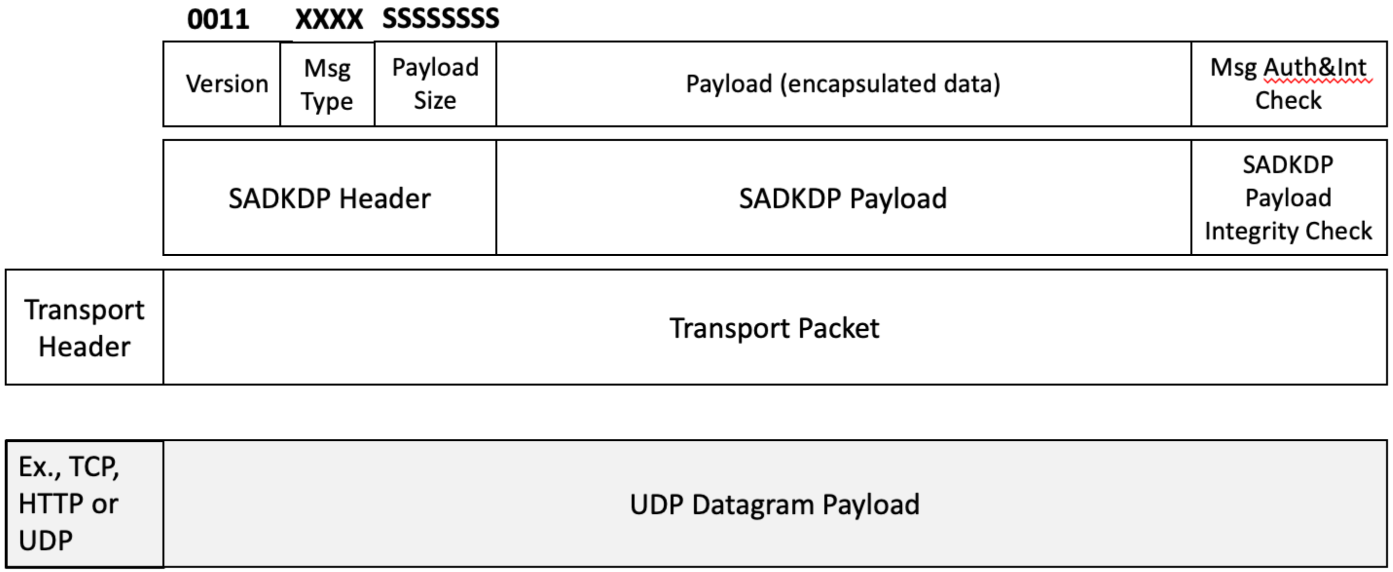
SADKDP Header | SADKDP Payload | SADKDP Payload Integrity Check

Transport Header | Transport Packet

Ex., TCP, HTTP or UDP | TCP Packet Payload, UDP Datagram Payload or HTTP Req/Body Payload

SADKDP and SRTSP: Protocol Encapsulations

SRTSP Encapsulation Format



0011 XXXX SSSSSSSS

Version | Msg Type | Payload Size | Payload (encapsulated data) | Msg Auth&Int Check

SADKDP Header | SADKDP Payload | SADKDP Payload Integrity Check

Transport Header | Transport Packet

Ex., TCP, HTTP or UDP | UDP Datagram Payload