# Semplice attacco al protocollo attraverso MITM

In questa sezione riportiamo un primo esempio di attacco al protocollo di provisioning. L’attacco è stato eseguito utilizzando un telefono Pixel 6 su cui è stato preventivamente eseguito il rooting, che ha permesso di manipolare la directory in cui sono memorizzati i certificati delle autorità di certificazione attendibili.

A computer with a polar proxy logo

Description automatically generated

Figura 16 Cattura di una sessione di provisioning.

La sessione di provisioning di una eSIM Very Mobile è stata eseguita collegando lo smartphone ad un Access Point Wi-Fi creato con hostapd su un portatile Linux. Su quest’ultimo è stato utilizzato il tool iptables per intercettare tutto il traffico TCP indirizzato alla porta 443 e inviarlo al tool PolarProxy [11], un proxy TLS locale eseguito sul portatile stesso. Questo tool si è poi collegato al server SM-DP+ il cui indirizzo era contenuto nel campo destination IP address dei pacchetti intercettati dal kernel del portatile. Ciò ha permesso di creare due sessioni TLS distinte: una a sinistra nella figura tra il Pixel 6 e PolarProxy, l’altra a destra tra PolarProxy e il server SM-DP+. Nel mezzo il tool di proxy è stato in grado di decrittare il traffico ricevuto da un end-point e ricriptarlo verso l’altro end-point, salvando il traffico decriptato nelle due direzioni su tracce pcap locali. Chiaramente il traffico salvato non contiene il profilo in chiaro ma i pacchetti della run di autenticazione tra l’eUICC e il server SM-DP+.

Interessante notare che l’app di provisioning ha “accettato” di collegarsi ad un server TLS il cui certificato non è firmato da alcuna CA riconducibile a GMSA in quanto generato dal tool PolarProxy. Altro punto interessante: si è ripetuto l’attacco contro uno smartphone Apple (un iPhone 12 Pro) ottenendo la stessa traccia di traffico pcap. Si conferma che l’intero procedimento ha portato al provisioning del profilo e all’attivazione della eSIM ricaricabile: il Pixel 6 si è infatti collegato alla rete dati dell’operatore.

Si riporta nel seguito il dissezionamento della traccia pcap.

I messaggi del protocollo RSP sono incapsulati in JSON trasmessi tramite HTTP. I valori corrispondenti alle diverse chiavi del JSON sono perlopiù codificati in Base64.

Facendo riferimento alla Figura 11, è stato possibile catturare tutti i messaggi scambiati tra LPAd e SM-DP+. Il primo messaggio analizzato è il messaggio 6 in Figura 11, ovvero initiateAuthentication. Questo contiene:

1. euiccChallenge: rappresentazione Base64 della challenge randomica generata dall’eUICC per autenticare il server;
2. euiccInfo1: Base64 corrispondente all’ASN.1 della struttura EUICCInfo1, la quale contiene un campo VersionType e le liste degli identificatori delle chiavi pubbliche delle RootCA supportate dall’eUICC per verificare le firme e per generarle;
3. smdpAddress: stringa indicante l’indirizzo del server SM-DP+, nel caso specifico “sys.prod.ondemandconnectivity.com”;

Il messaggio di risposta al precedente è il messaggio 9 in Figura 11. Esso contiene:

1. functionExecutionStatus: riporta la stringa “Executed-Success”;
2. transactionId: stringa rappresentante l’ID della transazione;
3. serverSigned1: Base64 corrispondente all’ASN.1 della struttura ServerSigned1. I valori che saranno firmati dal server in questo caso sono: TransactionId, euiccChallenge, serverAddress, serverChallenge;
4. serverSignature1: Base64 della firma del server;
5. euiccCiPKIdToBeUsed: Base64 della chiave pubblica della CA da usare;
6. serverCertificate: Base64 del certificato del server in formato DER, nel caso specifico è illustrato in Figura 17;

Segue il messaggio authenticateClient (messaggio 15 in Figura 11) che contiene l’ID della transazione e il Base64 corrispondente all’ASN.1 della struttura AuthenticateServerResponse, di tipologia AuthenticateResponseOk. Essa contiene:

1. euiccSigned1: transactionId, serverAddress, serverChallenge e euiccInfo2, ovvero una struttura ASN.1 complessa esposta in [1] a pagine 429-430;
2. euiccSignature1: firma dell’eUICC;
3. euiccCertificate: rappresentazione DER del certificato dell’eUICC (CERT.EUICC.SIG), nel caso specifico è illustrato in Figura 18;
4. nextCertInChain: rappresentazione DER del certificato che certifica quello dell’eUICC, nel caso specifico è illustrato in Figura 19;

Per il messaggio seguente occorre far riferimento al messaggio 6 di Figura 14. Esso contiene:

1. functionExecutionStatus: Executed-Success;
2. transactionId;
3. profileMetadata: Base64 corrispondente all’ASN.1 della struttura StoreMetadataRequest, definita in [1] a pagina 288. Nello specifico, è possibile leggere l’iccid della sim, il nome del provider (Very) ed il nome del profilo (VERY Mobile);
4. smdpSigned2: Base64 corrispondente all’ASN.1 della struttura SmdpSigned2 definita in [1] a pagina 321;
5. smdpSignature2;
6. smdpCertificate: come sopra;

Per gli ultimi due messaggi occorre far riferimento ai messaggi 5 e 11 di Figura 20. Il primo, ovvero getBoundProfilePackage, contiene:

1. prepareDownloadResponse: Base64 corrispondente alla struttura ASN.1 PrepareDownloadResponse nella forma PrepareDownloadResponseOk, ovvero contenente euiccSigned2 ([1] pagina 434) ed euiccSignature2;
2. transactionId;

Il secondo contiene:

1. functionExecutionStatus: Executed-Success;
2. transactionId;
3. boundProfilePackage: Base64 corrispondente alla struttura ASN1 definita in [1], pagina 49;

A diagram of a program

Description automatically generated

Figura 11 Diagramma che descrive la Common Mutual Authentication.

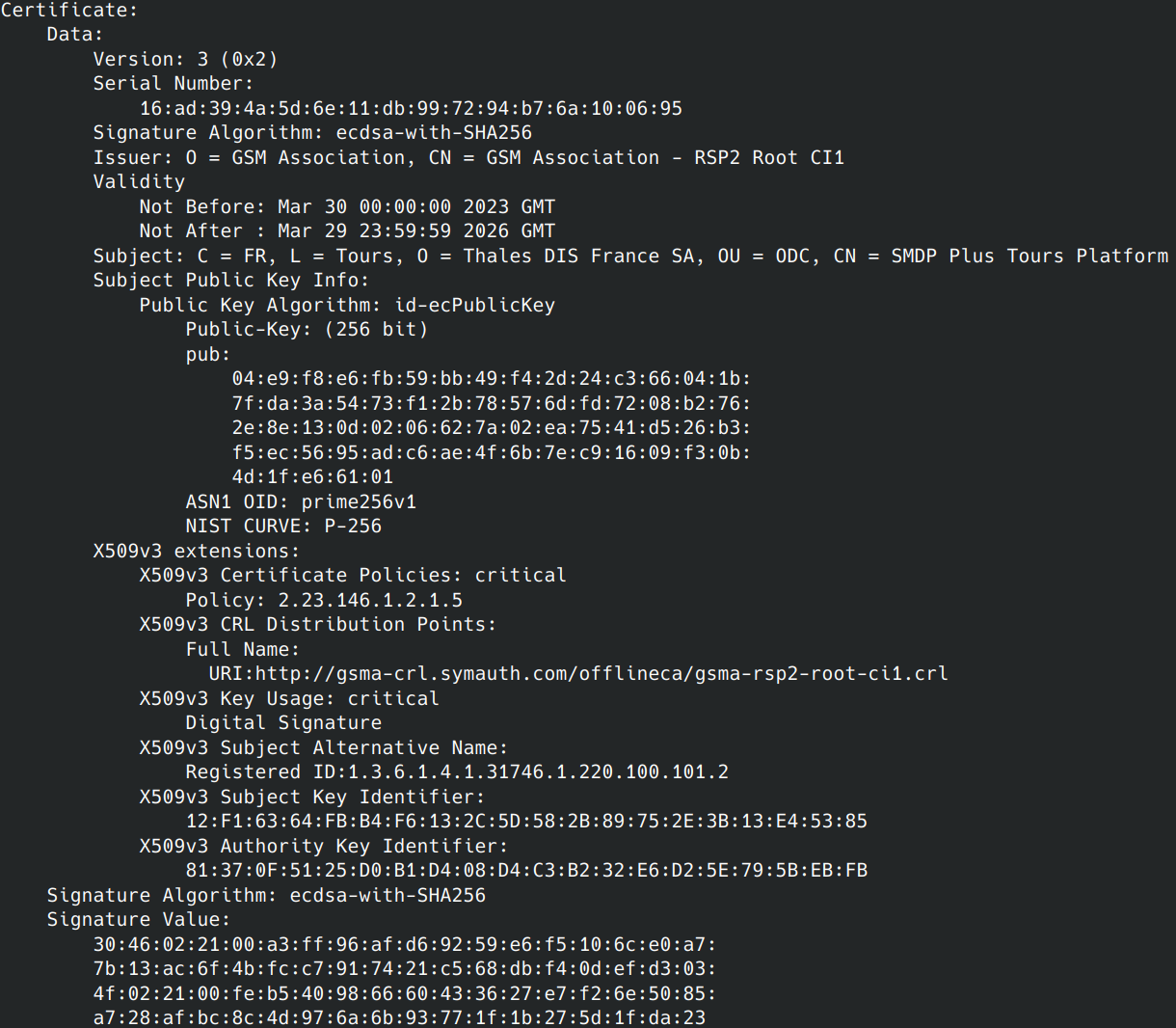


Figura 17 Certificato del server SM-DP+.

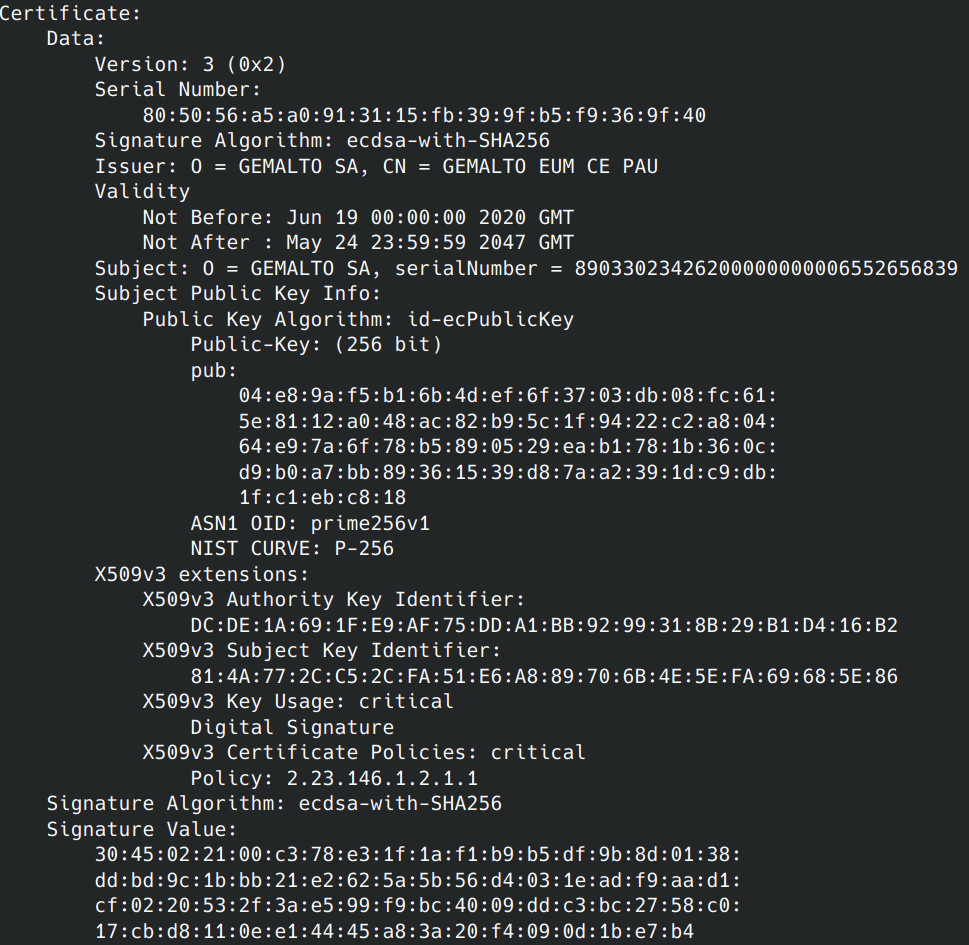


Figura 18 Certificato dell'eUICC.

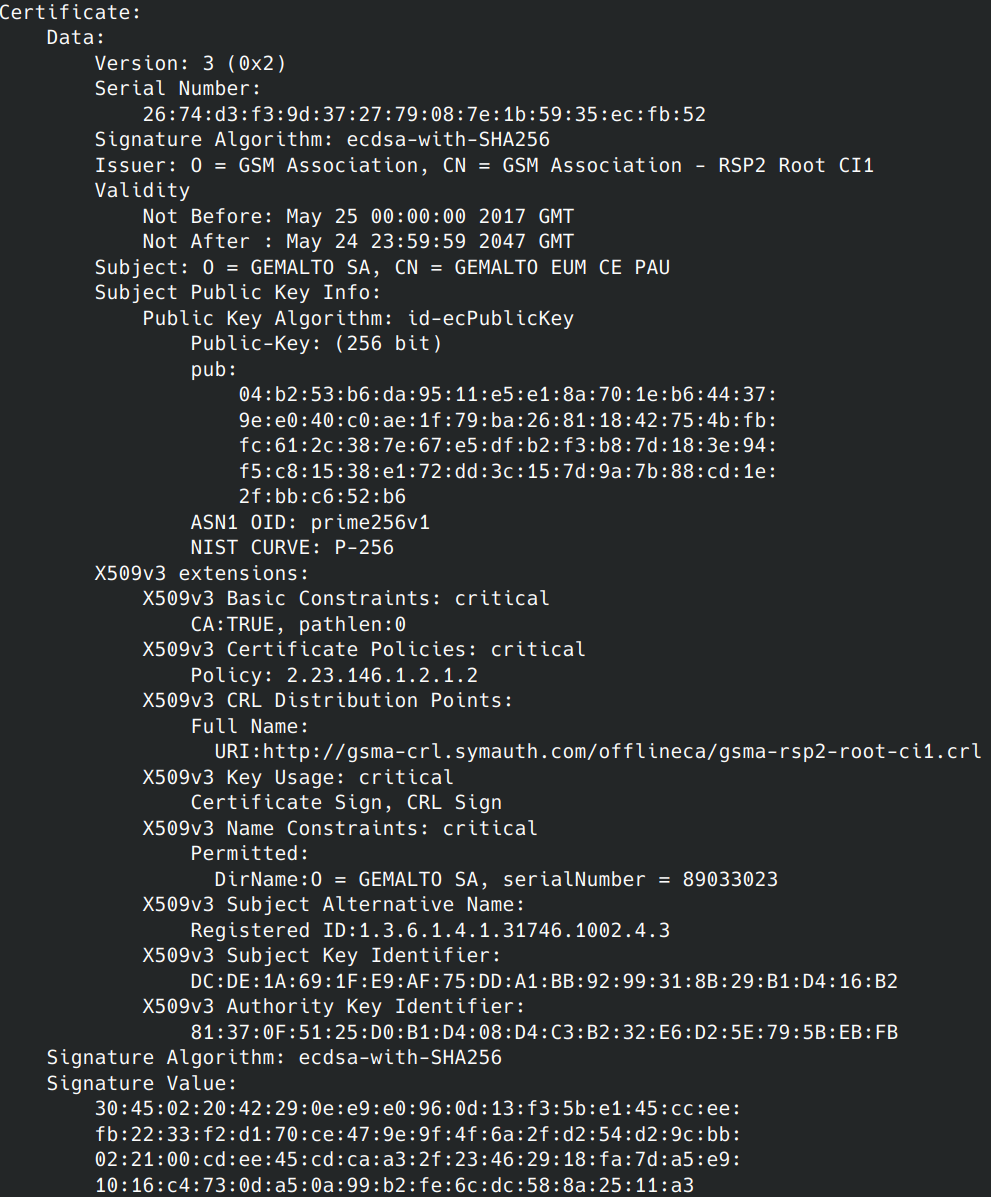


Figura 19 Certificato della CA che valida il certificato dell'eUICC.

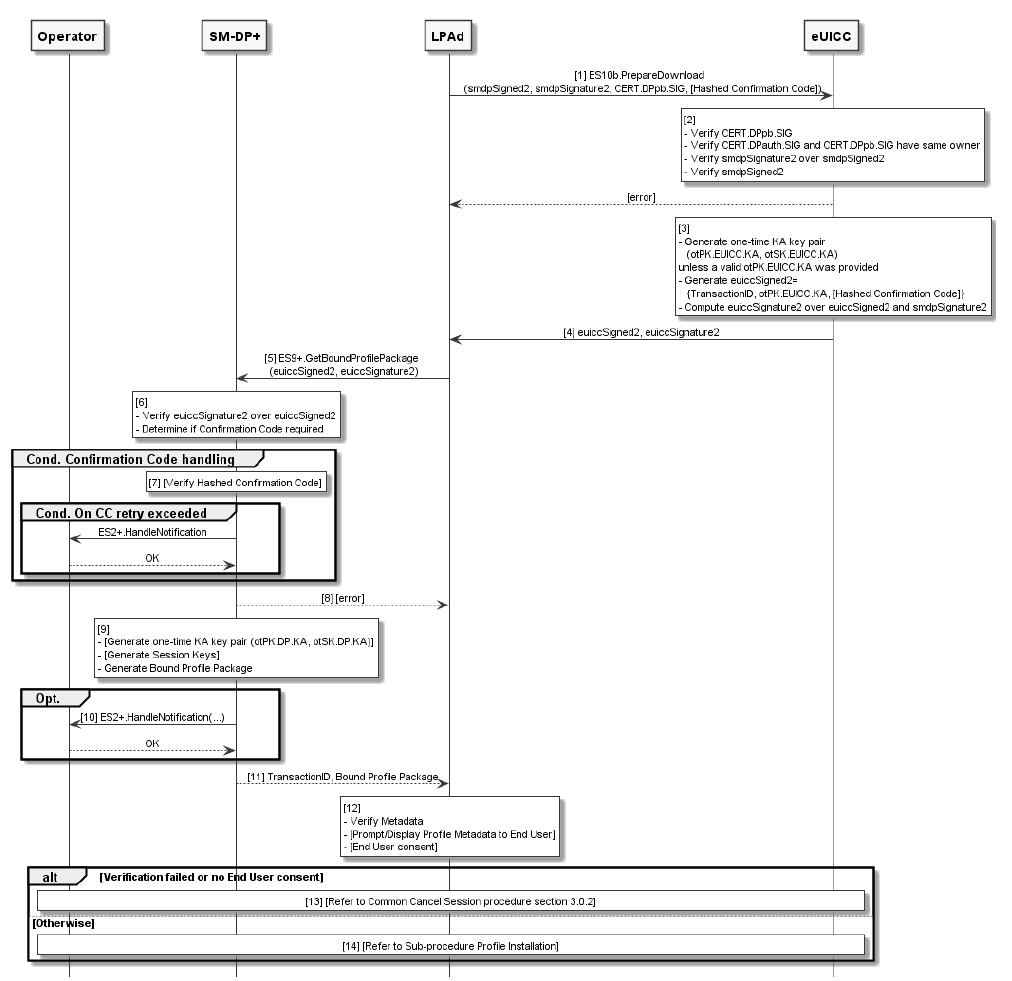


Figura 20 Sotto-procedura di download ed installazione del profilo.

# BIBLIOGRAFIA

1. GSM Association. “RSP Technical Specification Version 3.0”, 2022. https://gsma.com/esim/ wp-content/uploads/2022/10/SGP.22-v3.0-1.pdf.