



EMV Concept - Offline Data Authentication| How an Static Data Authentication Works | SDA | Application of Cryptography in Cards & Payments

Sivasailam Sivagnanam
Ideating solutions that moves millions of money digitally everyday
Veröffentlicht: 11. Sept. 2021

+ Folgen

INTRODUCTION:

The application of cryptography is wide spread in this digital banking era. In this article, we will look into its application in Cards and Payments Industry. More specifically, we will look into how EMV contact transaction applies cryptography concepts to securely validate and process the transaction. Offline authentication would be the best suited candidate to discuss such concepts.

WHAT IS OFFLINE AUTHENTICATION?

Consider yourself in a super market and the cashier now requests you to dip in your card at the point of sale (POS) terminal. The moment you dip in your card. Both terminal and the card starts talking to each other (nothing romantic, lol) . A lot of information will be processed in a matter of few seconds. One such process is offline authentication wherein the terminal verifies whether the card is legitimate or not. The outcome of such processes decides the fate of the transaction.

There are 3 different forms of offline authentication namely static data authentication (SDA), Dynamic Data Authentication (DDA) and combined data authentication (CDA). In this article, we will cover the concept of SDA. But before we look into SDA, it is important for us to understand certain basics concepts especially four important ones like the asymmetric cryptography, hash functions, digital signatures and public key infrastructure. SDA uses all these four concepts in its implementation.

Modern banking solutions involve the use of both symmetric and asymmetric cryptography. Hence, we will see about symmetric cryptography as well.

SYMMETRIC CRYPTOGRAPHY:

In symmetric cryptography, if Alice and Bob decide to communicate securely, first they need to share a secret key between them. Alice then encrypts her message with this shared secret key between herself and bob. Further, bob after receiving the message decrypts with the shared secret key and reads the message. This seems to be simple and straight forward solution until you bring in couple of more people into the channel.

Now, for all 4 of them to interact with each other, we need 6 keys in total. Scaling up this solution for an organization of 10,000 employees will need 49,995,000 keys to be generated for them to communicate securely among each other. To solve this problem, implementation of asymmetric cryptography is widely used.

ASYMMETRIC CRYPTOGRAPHY:

Asymmetric cryptography uses a concept called key pairs wherein a pair of keys is generated per user; a public key and a private key. The public key is distributed to everyone in the network and private is stored secretly.

If Alice wishes to send a message to Bob, she encrypts the message with Bob's public key. Bob receives this message and decrypts with his own private key to get the clear message. Note that, no one other than bob will be able to decrypt the message since the message was encrypted by Bob's public key. In this way, an organization with 10,000 employees will only need 20,000 keys pairs to achieve the desired outcome.

HASH FUNCTIONS:

Hash functions are one-way function i.e it is not reversible. If you have a hash value it is impossible to find out the input of the hash value.

Secondly, it takes input of any length and produces an output of fixed length. You type in one letter or a whole book, the length of the output remains constant (20 bits in case of hash function used in EMV).

Lastly, it gives a unique output for different input. In the below example, just by adding a dot to the end of the sentence changes the whole hash value. This proves that even if the input is altered slightly, then entire hash value changes.

Currently SHA-2 and SHA-3 are the considered secure at present. Hash functions play a critical part in Static Data Authentication (SDA). We will discuss more in detail on its implementation when we reach there.

DIGITAL SIGNATURES:

In asymmetric cryptography section, we discussed that when Alice wants to sends a message to Bob, she encrypts the message with Bob's public key and sends it through the channel. Bob then decrypts with private key to get that message. Whereas in case of digital signature, Alice encrypts the contents with her private key. YES! You read it right, private key. That is because our goal to achieve here is different. In case of asymmetric cryptography, our goal was to pass a message secretly from Alice to Bob but now with digital signature, we are trying to prove that the sender of the message is Alice herself and not anyone else. Let us see this through with an example.

AIM: Alice sends a message to Bob. After receiving the message, Bob should be confident that the message indeed was sent by Alice and not an attacker. Secondly, Bob should be confident that the message was not alerted. Lastly, Bob also wants to ensure that nobody eavesdrop this message. Let us see how digital signature and asymmetric keys works hand in hand to achieve this.

Step 1: Alice and Bob create an asymmetric key pair for themselves. Alice has her public and private key. Likewise, Bob also has his own public and private key. As usual, public key is shared to everyone who is intended to communicate and private key is stored secretly.

Step 2: Alice creates a message and encrypts it under Bob's public key. This results in an encrypted message (Green Box)

Step 3: Alice then passes her clear message into a hash function like SHA-3 which results in a fixed length output hash value (let us consider 20 bits hash output)

Step 4: Taking the output hash value from Step 3, Alice further encrypts the hash with her private key. This step can also be coined as digital signing which results to an output called "Digital Signature" (Another Green Box)

Step 5 : Bob receives the encrypted message together with the digital signature (2 green boxes). First, he decrypts the encrypted message using his private key which recovers the message in clear format

Step 6 : Bob then takes the digital signature and decrypts it with Alice public key which obtains a hash value. But you see the data is still not compromised because even if the digital signature ends up in the hands of someone not intended to, all they can do is to decrypt with Alice public key and get the hash value which is of no use to compute the input message. Remember we discussed hash functions are not reversible and impossible to get the input from hash value.

Step 7 : But until now our question remains unanswered. Bob still needs to ensure the sender is really Alice and not someone who pretends to be Alice. For that, Bob will now apply the same hash function which Alice used onto the clear message obtained from Step 5. This will result in a hash value.

Step 8 : If the output of Step 6 = Step 7; then he can be sure of two things. One that the message was not altered. If altered, the resulting hash would not be the same. Two, he can be sure that the certificate was indeed sent by Alice since he used Alice public key to decrypt it. If someone else altered the certificate then Alice's public key won't be able to decrypt it.

Additionally, in this whole channel, eavesdropping problem is also ruled out since asymmetric keys are used to exchange messages. If an attacker attempted to read this message, he won't be able to because only Bob's private key can decrypt the message.

PUBLIC KEY INFRASTRUCTURE:

One last stop, before we start discussing about SDA, is public key infrastructure (PKI) where all the concepts we reviewed above will be implemented. PKI uses digital certificates so let's understand the use of it.

One real life example, consider a student perusing his under-graduate course from a certified institution. Upon successful completion of the course, the student will be provided with certificate of completion. This certificate can be used by the student to prove his/her educational qualification to whom so ever.

Similarly, in PKI, Certificate Authorities (CA) is a trusted organization which issues digital certificates to organization/individuals to prove their credibility. When someone wants to obtain a digital certificate, they prepare a certificate signing request (CSR) which contains the requesting entity's public key and few other particulars about the requestor. They send it to the Certificate Authority. The CA performs necessary checks and validates the credibility of the requestor. Once all checks are satisfactory, the CA then:

Creates a digital certificate which contains the requestor public key sent in the CSR and Signs the certificate with CA's private key and returns to the certificate requestor. The requestor then can issue this certificate to anyone as proof of their public key.

STATIC DATA AUTHENTICATION:

In payments world, the schemes act as Certificate Authorities. It is responsible for providing the Issuer public key certificate to the card issuer and also to distribute its public keys to the acquirer which then passed to the terminal.

HOW SDA WORKS?

The terminal attempts to authenticate the static data present inside the chip. This static data is embossed into the chip by the issuer. By verifying this static data, the terminal can be sure that the card (also referred to as ICC) is not affected by unauthorized alteration of data after personalization.

PROCESS FLOW

The entire process of terminal authenticating the static data happens in 3 steps:

- RETRIVAL OF CERTIFICATE AUTHORITY PUBLIC KEY
- RETRIVAL OF ISSUER PUBLIC KEY
- VERIFICATION OF SIGNED STATIC APPLICATION DATA(SSAD)

RETRIVAL OF CERTIFICATE AUTHORITY PUBLIC KEY

If a terminal accepts visa card, it is expected to have all the Visa CAPKs. It is usually supplied by the schemes to acquirer and passed on to the terminal. ICC also supplies the RID (first 5 bytes of AID) and CAPK index to the terminal. Terminal selects the relevant CAPK using the CAPK and RID.

RETRIVAL OF ISSUER PUBLIC KEY

Terminal now needs the Issuer public key (PK) in order to validate the signed static data. It recovers this information from Issuer PK Certificate which was signed by the CA Private Key.

As we know digital signature works on the concept of asymmetric cryptography, that is any message signed/encrypted by a key can only be decrypted by its corresponding key pair. In case of Issuer PK certificate, it is signed by CA Private Key and can be verified only by CA Public key. Luckily, we got this info from previous step.

Next is to apply the recovery function to the Issuer PK certificate together with the extracted CAPK which will open up the contents of the certificate (12 values shown above). Terminal then concatenates from point 2 to point 10 and then passes it to a hashing algorithm indicated by point 6 which results in a hash value of 20 bytes. If the hash value computed above matches with the hash value present in the certificate then the Issuer Public key is considered valid and chosen to do further operation.

VERIFICATION OF SIGNED STATIC APPLICATION DATA (SSAD)

The concept to see the for the next step. We have a public key which then can be used to see the contents of the certificate. Inside the certificate there are several values like the hash values, hashing algorithm indicator and several others which will act as input to hashing algorithm. Hash value is computed and matched against the reference hash result in the SSAD. If matches, then SDA has passed. Terminal then sets the relevant TVR bit to indicate that the SDA Verification has passed.

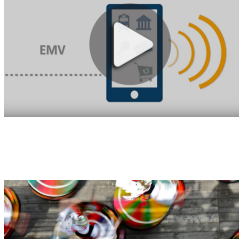
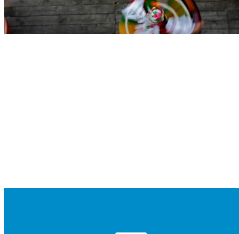
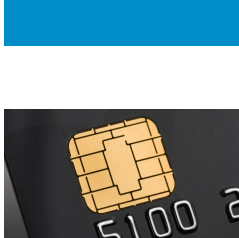

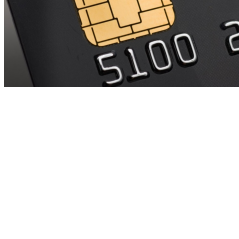



CONCLUSION:

I hope this article was useful to you and would like to hear your thoughts, suggestions and corrections if any in the comments. Thanks for reading through. Stay hungry to learn.

EMV Concept - How ARQC is generated | Visa CVN 1...
8. Aug. 2022

How is your PIN validated
???
31. Aug. 2021

Ebenfalls angesehen

- **Full EMV transaction**
Binoy Baby · 4 Jahre
- **Certifying (m)POS terminals for Level 3: as easy as it looks?**
Steve Lacourt · 6 Jahre
- **CDA EMV**
Mahmoud Elshafey · 5 Jahre
- **EMV Application Specification :: Offline Data Authentication (ODA) - part I**
Ahmed Hemdan Farghaly · 1 Jahr
- **EMV Application Specification :: Offline Data Authentication (ODA) - part II**
Ahmed Hemdan Farghaly · 1 Jahr
- **Everything EMV.**
Binoy Baby · 4 Jahre
- **EMV Application Specification :: Read Application Data**
Ahmed Hemdan Farghaly · 1 Jahr
- **Offline Data Authentication (ODA)**
Binoy Baby · 4 Jahre

EMV QuickChip Transaction

Binoy Baby · 4 Jahre

PINBlock Explained:

Iftakhar AL Haque · 4 Jahre

152 · 17 Kommentare

Gefällt mir	Kommentieren	Teilen
<div>Norhan Abdalla8 Monate</div> <div>Thank you very much for your efforts, This article was really helpful. Could you suggest references for EMV process and security for someone who wants to go a little in depth?</div> <div>Gefällt mir · Antworten 1 Gefällt mir</div>		
<div>Sivasailam Sivagnanam8 Monate</div> <div>Norhan Abdalla - Thanks for your acknowledgment</div> <div>Gefällt mir · Antworten</div>		
<div>Norhan Abdalla8 Monate</div> <div>Sivasailam Sivagnanam I have skimmed through them and I think they work more as a guide or technical manual but the way you explained SDA was pretty methodological.</div> <div>Gefällt mir · Antworten</div>		
<div>ARJUN V1 Jahr</div> <div>Excellent read! Well done.. Really appreciate the effort.. The complex concepts made real simple! Expecting more such articles from you.. 🙏</div> <div>Gefällt mir · Antworten 1 Gefällt mir</div>		
<div>Sivasailam Sivagnanam1 Jahr</div> <div>ARJUN V - Thanks for your words of encouragement.</div> <div>Gefällt mir · Antworten</div>		
<div>Ghaffour Jamal1 Jahr</div> <div>Very interesting how you explain the emv authentication. Very complex and sensitive concept but explained in simple and easy way. Good job Sivasailam Sivagnanam, CSM®, ISTQB®</div> <div>Gefällt mir · Antworten 1 Gefällt mir</div>		
<div>Sivasailam Sivagnanam1 Jahr</div> <div>Thanks for your kind words Ghaffour Jamal</div> <div>Gefällt mir · Antworten</div>		
<div>Dhrubajyoti (Dhrub) Mukherjee1 Jahr</div> <div>Thanks for putting down these concepts in a simple way!!</div> <div>Gefällt mir · Antworten 1 Gefällt mir</div>		
<div>Sivasailam Sivagnanam1 Jahr</div> <div>Dhrubajyoti (Dhrub) Mukherjee 🙏👍</div> <div>Gefällt mir · Antworten</div>		
<div>Charanya Nagarajan1 Jahr</div> <div>Good job 🙏</div> <div>Gefällt mir · Antworten 1 Gefällt mir</div>		
<div>Sivasailam Sivagnanam1 Jahr</div> <div>Charanya Nagarajan - Thank you</div> <div>Gefällt mir · Antworten</div>		
<div>Bhagyashree Upadhyay1 Jahr</div> <div>This is a great</div> <div>Gefällt mir · Antworten 1 Gefällt mir</div>		
<div>Raed Bou Fakheredine8 Monate</div> <div>Very interesting, informative and clear for everyone interested in the combination between Card Payments and Security.</div> <div>Gefällt mir · Antworten 1 Gefällt mir</div>		
<div>Rakkeshvarma Rajendran1 Jahr</div> <div>Thanks for sharing Sivasailam Sivagnanam, CSM®, ISTQB®</div> <div>Gefällt mir · Antworten 1 Gefällt mir</div>		
<div>Pradeep Parihar1 Jahr</div> <div>Thanks for posting</div> <div>Gefällt mir · Antworten 1 Gefällt mir</div>		

Weitere Kommentare anzeigen

Zum Anzeigen oder add a comment [einloggen](#)

Weitere Artikel von dieser Person