Interpreting the EMV Tokenisation Specification

Francisco Corella, PhD
fcorella@pomcor.com

Karen Lewison, MD
kplewison@pomcor.com

October 19, 2014

Contents

1 Why It Matters **1**

2 Issues with the Specification **2**
2.1 Issues Related to the Token Cryptogram 2
2.2 Issuing Bank as Token Service Provider 4
2.3 Underspecification of the E-Commerce Use Case 5
2.4 Inadequacy of the Card-on-File E-Commerce Use Case 5
2.5 Impossibility of the Scan at Point of Sale Use Case 6
2.6 No Mention of Offline Data Authentication 6

3 Observations **6**

4 Conclusions **7**

Comments **8**

References **8**

1 Why It Matters

The recent Apple Pay announcement [1], and simultaneous announcements by MasterCard [2] and Visa [3], have brought attention to the concept of *tokenization*, which refers to the replacement of a credit card number (also known as a PAN, which stands for Primary Account Number) with an alternative, similarly formatted number called a *payment token*. The concept is not new: it has been used for years by payment processors[1] and acquiring banks to help merchants avoid security exposure by storing the PAN on their behalf [4]. But Apple Pay uses a new kind of tokenization, where the token is stored within the cardholder's device and sent to the payment network in the course of a transaction, instead of being a mere reference to a PAN kept in a secure database, a reference that is only seen by the merchant, while the PAN is sent to the payment network.

[1]The term *payment processor* has multiple meanings; here we use it to refer to an entity to which the merchant outsources some its payment processing tasks, including interactions with the acquiring bank.

While the earlier kind of tokenization increases security by storing the PAN more securely, the new kind increases security by preventing *cross-channel fraud*. Payment tokens are used in dynamic-data transactions where the cardholder's device generates a *cryptogram*, which is a symmetric-key digital signature on data that includes a cryptographic nonce[2] generated by the merchant; they cannot be used in traditional static-data transactions that rely on static card data communicated to the merchant via a magnetic stripe reader or a web form.[3] If tokens were not used, a fraudster might be able to sniff card data from a dynamic transaction and use it in a static transaction. This is called "cross-channel fraud" because different kinds of transactions are sometimes referred to as taking place through different channels, such as the "NFC channel". Tokenization will no longer be needed when static-data transactions have been phased out. But in the meantime the new kind of tokenization, in conjunction with dynamic-data transactions, has the potential to greatly increase payment security.

2 Issues with the Specification

The *EMV Tokenisation Specification* [5] is concerned with the new kind of tokenization, while acknowledging the existence of the earlier kind: "*This specification does not preclude existing Acquirer or other third-party implemented Payment Token solutions in which these entities generate Payment Tokens and perform Payment Token to PAN mapping within their ecosystem*" [5, Section 2.1, second bullet].

Unfortunately, the specification, or at least the portion that has been published so far, is labeled as a *Technical Framework* and is not a real specification that can be used to develop and test interoperable implementations. Most of the operational details are left unspecified, the reader being referred instead to the *message specification* of each payment network (presumably MasterCard, Visa and American Express).

Worse, the document has serious issues, including ambiguities, inconsistencies, omissions, and portions that simple do not make sense. In this section we discuss the issues and propose an interpretation of the document that resolves some of them. (Others cannot be resolved without taking a different approach to some of the use cases.) Then, in the rest of the paper we make observations based on the proposed interpretation and draw a few conclusions.

2.1 Issues Related to the Token Cryptogram

The document introduces the concept of a *token cryptogram* without specifying what it consists of or how it is generated. Section 1.6 of the specification defines a token cryptogram as follows:

> *A cryptogram generated using the Payment Token and additional transaction data to create a transaction-unique value. The calculation and format may vary by use case.*

[2] In cryptography, a nonce is a number only used once in a given context.

[3] Dynamic data transactions include EMV chip-and-PIN transactions, EMV contactless transactions in EMV mode, and EMV contactless transactions in mag-stripe mode. In a mag-stripe-mode contactless transaction, the cryptogram is a *dynamic security code* that replaces the static security code of magnetic track data.

But then Section 4.1 says that it is "*uniquely generated by the Token Requestor to validate authorised use of the Token*", and Section 3.8 says that

> *Potential Token Requestors include, but are not limited to: Card-on-file Merchants; Acquirers, Acquirer Processors, and payment gateways on behalf of Merchants; Payment enablers, such as original equipment manufacturer (OEM) device manufacturers; Digital wallet providers; Card Issuers.*

Generation of the cryptogram by the token requestor would imply that the token cryptogram is generated when the token is requested and provisioned to the cardholder's device, contradicting the definition of a token cryptogram as a "*transaction-unique value*".

The token cryptogram is validated by the Token Service Provider, whereas cryptograms used in authorization requests according to earlier EMV specifications, viz. the ARQC cryptogram of the EMV 4.3 specification also used in the EMV mode of the contactless specification, and the dynamic security code of the mag-stripe mode of the contactless specification, are validated by the issuer. Is the token cryptogram used in addition to an earlier-specification cryptogram or instead of it?

Paragraph 1c of Section 9.2 says that

> *Token Cryptogram will be generated based on the Token data elements and will be passed in the Chip Cryptogram field. (The cryptogram may be a full chip cryptogram, or an abbreviated Track 2 equivalent cryptogram.)*

The term "*chip cryptogram*" is not defined or mentioned anywhere else in the specification or in any other EMV specification, but the parenthetical remark seems to refer to the ARQC cryptogram ("*full chip cryptogram*") and the dynamic security code ("*abbreviated Track 2 equivalent cryptogram*"). The paragraph can be interpreted as saying that the token cryptogram is like the cryptograms of earlier specifications, now called "*chip cryptograms*", but calculated using the token and token expiration date instead of the PAN and PAN expiration date.[4] And the fact that the token cryptogram is passed in the chip cryptogram field suggests that it is meant to replace the chip cryptogram instead of coexisting with it.

In all earlier EMV specifications, and presumably in this one, a cryptogram is a digital signature computed with a symmetric key, and can only by validated by a party who shares the key with the cardholder's device. In earlier EMV specifications that party is always the issuer. In this specification, however, that party seems to be the Token Service Provider. The following is included in the descriptions of all the use cases:

> *The Payment Network will interface with the Token Service Provider to . . . Validate the Token Cryptogram and validate the Token Domain Restriction Controls for that Payment Token (alternatively the Card Issuer may validate the cryptogram if it has the necessary keys).*

But the parenthetical remark is inconsistent with the flow diagrams shown in the figures that illustrate the use cases (Figures 4–7). The card issuer and the token service provider can be one and the same, in which case cryptogram validation by the issuer is the same

[4]The PAN and PAN expiration date do not seem to be included in default cryptogram calculations according to Tables 26 and CCD-3 of Book 2 of the EMV 4.3 specification, but may be included in calculations of dynamic security codes, whose details are left out of the EMV Contactless Specification.

as validation by the token service provider rather than an alternative. They can also be different, but in that case the issuer cannot validate the token cryptogram even if it has the symmetric key, because it does not receive the cryptogram.

Cryptogram generation requires a cryptographic nonce generated by the merchant and sent to the cardholder's device, which makes the cryptogram transaction-specific. Other EMV specifications describe the use of the nonce,[5] but the tokenization specification never mentions it.

2.2 Issuing Bank as Token Service Provider

As mentioned above, the issuer can be the token service provider. This is indicated by a note in each of Figures 4–7: "*The Token Service Provider may be located at the issuer, Payment Network, or a third party*". Those figures, and the descriptions of the corresponding uses cases 1–4 in Sections 9.2–5, do not make a distinction based on where the token service provider is located, and therefore, if taken literally, should be applicable to the case where the token service provider is the issuer. That would mean that the following interactions would take place in that case: the payment network would ask the issuer to validate the cryptogram and map the token and token expiration date to the PAN and PAN expiration date; the issuer would return the PAN and PAN expiration date to the payment network; the payment network would replace the token and token expiration date with the PAN and PAN expiration date in the authorization request before forwarding the request to the issuer; the issuer would return the PAN to the payment network in the authorization response; and the payment network would replace the PAN with the token in the authorization response before forwarding it to the acquirer, either by asking the issuer for the PAN-to-token mapping, or by reversing the token-to-PAN mapping used earlier to modify the authorization request.

Such interactions are wasteful, and unnecessarily expose the PAN, by sending it from the issuer to the payment network and including it in the authorization request and the authorization response. But there is a hint in Section 8.3 of the specification that the figures and descriptions should not be taken literally:

> *The Payment Network SHOULD use the Token Service Provider to map the Payment Token to the PAN in the incoming authorisation message prior to sending the message to the Card Issuer, and SHALL always map the PAN back to the Payment Token in any response messages sent back to the Acquirer (except in cases where the Card Issuer is acting as the Token Service Provider).*

The exception in parenthesis hints that, contrary to what is literally shown in the figures and stated in the descriptions, where the issuer always returns the PAN in the authorization response, it is the token rather than the PAN that the issuer places in the authorization response when acting as token service provider; hence the payment network need not map the PAN to the token in that case.

Therefore we assume that the case where the token service provider is the issuer is meant to be implemented in a straightforward way, in which the payment network forwards the authorization request from the acquirer to the issuer and the authorization response from

[5] They refer to the nonce as an "*unpredictable number*", although what matters is not unpredictability but uniqueness.

the issuer to the acquirer without replacing the token with the PAN, and hence does not need to obtain the PAN from the issuer.

2.3 Underspecification of the E-Commerce Use Case

This use case (Use Case 2, Figure 5, Section 9.3) is concerned with payment transactions using a mobile device connected to the Internet.

The software components used on the mobile device to perform the transaction are imprecisely and inconsistently defined. An introductory paragraph on page 70 mentions a "*mobile / digital wallet*" that can transfer data to "*a Merchant*" through a "*wallet API*", while the description of step 1 on page 72 mentions a "*Merchant application / digital wallet*" that interacts with a "*payment application*" and passes data elements to a "*Merchant platform*".

(An Apple Pay demo [6] shows in-app payments where the cardholder taps a Pay-with-Apple-Pay button displayed by a merchant-provided native app on the mobile device. This causes the Apple Pay app to prompt the cardholder to touch the fingerprint sensor in order to make the payment. We conjecture that the merchant app and the Apple Pay app communicate through an API exposed by the Apple Pay app, and the merchant app communicates with a merchant back-end (the "*merchant platform*") over the Internet through a private API.)

The transaction flow in this use case is the same as in the "*NFC at Point of Sale*" use case (Use Case 1, Figure 4, Section 9.2), and in particular it involves a token cryptogram, which the mobile device sends to the merchant's back-end. (This is not shown in Figure 4, but it is mentioned in the description.) The merchant-generated nonce needed to compute the cryptogram is not mentioned. Implementors might be tempted to let the merchant app generate the nonce, which would be insecure because the merchant app runs on the mobile device and should be considered to be under adversarial control. The nonce should be generated instead by the merchant back-end.

2.4 Inadequacy of the Card-on-File E-Commerce Use Case

This use case (Use Case 3, Figure 6, Section 9.4) refers to "*scenarios where an e-commerce Merchant that has payment card data on file in a database seeks to remove the underlying security exposure of storing card data by replacing the PANs with Payment Tokens*". But it does not make sense as described in the specification.

A card-on-file arrangement is in essence a delegation from the cardholder to the merchant of authority to initiate payment transactions on the cardholder's behalf. In the traditional non-cryptographic setting, delegation is accomplished by the cardholder's consent for the merchant to retain the card data. It is possible to implement delegation cryptographically, but not with a simple cryptogram implemented as a mere symmetric-key signature.

Figure 6 shows the merchant sending the acquirer a cryptogram, which the description in Section 9.4 says is "*optional*". The specification does not say what generates the cryptogram. Is it the cardholder's device? If so the cryptogram cannot be transaction specific because the cardholder's device is not involved in a card-on-file transaction by definition. Is it the merchant's equipment? If so, what key is the cryptogram generated with? If it is generated with the key stored in the cardholder's device, that key must be given to the merchant,

degrading security for the cardholder. If it is generated with a key owned by the merchant, it serves no useful security purpose.

2.5 Impossibility of the Scan at Point of Sale Use Case

In this use case (Use Case 4, Figure 7, Section 9.5) the point of sale reads a QR code displayed by the user's mobile device. But the use case cannot work as described because there is no way for the point of sale to send the nonce required to generate the cryptogram to the mobile device.

If a QR code is to be used in a transaction where a mobile device generates a cryptogram, the mobile device should read the QR code from the point of sale rather than the point of sale from the mobile device. The QR code can then include the nonce. After the mobile device reads the QR code, it can complete the transaction over the Internet, or locally through a Bluetooth or WiFi connection.

2.6 No Mention of Offline Data Authentication

The specification does not discuss the impact of tokenization on the offline data authentication that takes place between the user's device and the point of sale in EMV 4.3 transactions and EMV-mode contactless transactions. Offline data authentication is not used in contactless mag-stripe mode transactions, but there is no indication that the specification is only applicable to the mag-stripe mode.

In contexts other than mag-stripe mode, offline data authentication comes in three flavors: SDA, DDA, and CDA. We believe that DDA and CDA are compatible with the new kind of tokenization, but SDA is not.

3 Observations

Assuming that the above interpretation is the intended one, we can make the following observations to summarize the specification and some of its implications.

1. The specification is concerned with a new kind of tokenization that prevents cross-channel fraud by replacing card data (PAN and PAN expiration date) with token data (token and token expiration date). The token data is provisioned to the user's device, together with a symmetric key that the device uses at transaction time to generate a cryptogram by signing transaction data. The cryptogram is computed as in a non-tokenized transaction, except that token data is used in the computation instead of card data. The symmetric key is shared with a token service provider that validates the cryptogram by verifying the signature and translates back-and-forth between token data and card data in the course of a transaction.

2. The user's device may be a mobile device such as a phone or a tablet, but it can also be a chip card, in which case the token and token expiration date are stored in the chip while the PAN and PAN expiration date are embossed on the card and encoded in the magnetic stripe.

3. The token service provider may be the payment network or, equivalently, a third party to which the payment network outsources the task of providing the service. In that case the payment network or third party validates the cryptogram on behalf of the issuer. The issuer does not see the cryptogram, but is provided with both the PAN and the token. Alternatively, the token service provider may be the issuer. In that case only the issuer sees the PAN in the course of a transaction.

4. In a non-tokenized transaction, the issuer derives a form of end-to-end security from validating the cryptogram generated by the user's device. That end-to-end security is lost in a tokenized transaction if the token service provider is the payment network or a third party. It is preserved if the token service provider is the issuer.

5. Whether or not the issuer is the token service provider, it needs to know and record the token if it wants to report to the user which of the user's devices was used to perform each transaction.

6. Instead of acting as a token service provider, the issuer can achieve essentially the same level of security by allocating a secondary account number (SAN) from a card number BIN range rather that a token from a token BIN range.[6] The issuer can prevent cross-channel fraud by flagging the SAN in its internal database as to be used exclusively in EMV transactions, which require cryptogram verification. The only difference between a non-tokenization transaction with a SAN and a tokenized transaction is that, in a tokenized transaction, the acquirer and the merchant receive from the issuer the last four digits of the PAN, plus a "*Token Assurance Level*" that tells them how confident the issuer is that the token and shared key were provisioned to the correct user rather than an impostor.

7. An e-commerce transaction as described in Use Case 2 differs from an NFC transaction as described in Use Case 1 only in the transmission medium between the user's device and the merchant. The e-commerce use case is much more secure than a traditional e-commerce payment that relies on transmission of static card data over a secure channel, as the cryptogram prevents same-channel fraud and the token prevents cross-channel fraud.

8. An e-commerce transaction as described in Use Case 2 can be used for in-store payments (as shown in the above-mentioned Apple Pay demonstration, where a consumer is seen making in-app payments while at the Apple store and at a Panera restaurant [6]). An in-store e-commerce use case could be turned into an in-store Bluetooth or WiFi use case simply by changing the transmission medium.

4 Conclusions

The following conclusions can be drawn from the above observations.

Apple Pay use case. In the use case that is probably implemented by Apple Pay for both in-store and in-app transactions, a token service provider provisions a token and

[6] BIN stands for Bank Identification Number. The first 6 digits of a credit card number are a BIN.

a shared key to the mobile device. When it comes to making a payment, the merchant sends a cryptographic nonce to the device and the device generates a cryptogram, which is a symmetric digital signature computed with the shared key on data that includes the nonce. The merchant includes the token and the cryptogram in the authorization request, which travels via the acquirer to the payment network. The payment network asks the token service provider to validate the cryptogram on behalf of the issuer and map the token to the PAN; then it forwards to the issuer a modified authorization request that includes both the token and the card number but not the cryptogram. The role of payment service provider can be fulfilled by the payment network itself without essentially altering the use case.

Alternative use case with end-to-end security. As an alternative, the issuer itself can play the role of token service provider and provision the token and shared key to the mobile device, just as it provisions a shared key to a chip card in a non-tokenized EMV transaction. (The issuer may also provision a token to a chip card; the token is then stored in the chip while the PAN is embossed on the card.) In that case the payment network forwards the authorization request to the issuer without replacing the PAN with the token. The transaction flow is essentially the same as in a non-tokenized transaction. The cryptogram is validated by the issuer, preserving the end-to-end security that is lost when the cryptogram is validated by the payment network or a third party playing the role of token service provider.

Alternative to tokenization. Instead of provisioning a token to a mobile device (or a chip card), the issuer can achieve essentially the same level of security by provisioning a secondary account number and flagging it in its own database as being intended exclusively for use in EMV transactions, which require cryptogram validation.

Comments

If you have any comments on this white paper, please post them to the blog post that announces and summarizes the paper [7].

References

[1] Apple Inc. Apple Announces Apple Pay. September 9, 2014. http://www.apple.com/pr/library/2014/09/09Apple-Announces-Apple-Pay.html.

[2] James Anderson. MasterCard Digital Enablement Service (MDES): Making Digital Payments Happen. September 10, 2014. http://newsroom.mastercard.com/2014/09/10/mastercard-digital-enablement-service-mdes-making-digital-payments-happen/.

[3] Business Wire. Visa Launches Innovative Token Service. September 09, 2014. www.businesswire.com/news/home/20140909006526/en/Visa-Launches-Innovative-Token-Service.

[4] First Data Corporation. EMV and Encryption + Tokenization: A Layered Approach to Security. 2012. http://www.firstdata.com/downloads/thought-leadership/EMV-Encrypt-Tokenization-WP.PDF.

[5] EMVCo. EMV Payment Tokenisation Specification—Technical Framework. http://www.emvco.com/specifications.aspx?id=263.

[6] Darrell Etherington. Here's Apple Pay In Action. September 9, 2014. http://techcrunch.com/2014/09/09/heres-apple-pay-in-action/.

[7] Francisco Corella. Making Sense of the EMV Tokenisation Specification. Blog Post. October 19, 2014. http://pomcor.com/2014/10/19/making-sense-of-the-emv-tokenisation-specification/.