Merchant-Initiated Transaction and Credential-on-File Support

Overview

CyberSource supports Credential-on-File (COF) and Merchant-Initiated Transaction (MIT) Mandates for for Visa, Mastercard, American Express and Discover on these gateway processors:

- AIBMS
- American Express Direct
- Barclays
- Chase Paymentech Solutions
- Elavon
- Elavon Americas
- FDC Compass
- FDC Nashville Global
- FDMS Nashville
- HBoS
- HSBC
- Ingenico ePayments
- Lloyds-OmniPay
- LloydsTSB Cardnet
- Moneris
- OmniPay Direct
- Rede
- SIX
- Streamline
- Vero
- Visa Platform Connect
- Worldpay VAP

The COF mandate applies to new business models with numerous transactions for which the merchant keeps payment credentials on file. This mandate enables issuers to identify such transactions and treat them accordingly.

The COF mandate also enables a merchant to identify any pre-existing relationship between a cardholder and an issuer by using a common identifier.

The MIT mandate ensures that merchants, acquirers, and issuers understand the transaction processing cycle. The MIT framework introduces a global standard for identifying transaction intent and whether a transaction is merchant-initiated (without participation of cardholder). Under the terms of the Payment

Service Directive 2 in Europe Customer Initiated eCommerce transactions must be strongly authenticated. To avoid declines for Secure Customer Authentication whilst storing a credential on file, you must ensure the customer is fully authenticated through the Step Up Challenged 3D Secure experience. The MIT framework facilitates exemptions from Strong Customer Authentication (SCA) using 3D-Secure depending on the transaction context, e.g. recurring transactions, Mail-Order/Telephone-Order (MOTO) transactions.

Benefits of Complying with the COF Mandate

- Recognizing stored credential transactions helps the merchant understand the transaction risk and enables robust processing, which result in differential treatment
- Helps issuers better understand transaction risk levels
- Higher authorization approval rates and more completed sales
- Fewer customer complaints for false-positive declines

Benefits of Complying with the MIT Mandate

- Merchants, acquirers, and issuers can link a series of related transactions
- Consistent MIT processing:
    - Merchants' MIT transactions—token-based transactions in particular—have better approval rates when the issuer can identify them
    - MIT and token-based transactions can be processed without strong customer authentication (these transactions might otherwise fail, especially in regions complying with PSD2)
- Transaction transparency, which results in higher authorization rates and improved cardholder experience

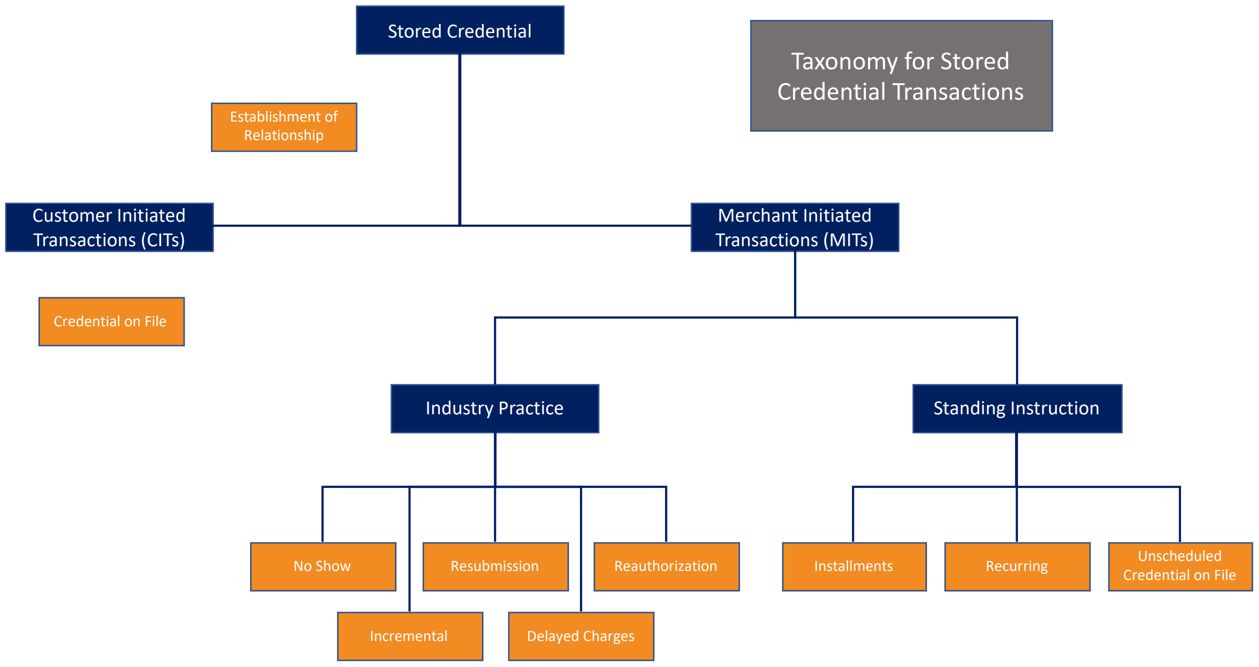
How to Comply

For merchants that use CyberSource's Token Management System (TMS), Recurring Billing services, or any of the legacy token services, most of the complexity is handled on behalf of the merchant. See the TMS MIT and COF FAQ for an overview and compliance checklist. If you do not use TMS, speak to your CyberSource account manager about its benefits such as increasing conversion rates, easing MIT adoption and reducing compliance burden.

If you initiate Merchant-Initiated Transactions but do not use CyberSource Tokenization, continue reading to determine how you can become compliant.

Merchants must evaluate their business to determine whether their transactions correspond to the network-defined use cases discussed below. New APIs have been developed to identify the transactions according to Industry Specific and Standing Instruction use cases.

Misusing any of the fields can lead to declines. All of these fields are required in order to support COF and MIT mandates, unless otherwise noted.


Stored Credential
Taxonomy for Stored Credential Transactions
Establishment of Relationship
Customer Initiated Transactions (CITs)
Merchant Initiated Transactions (MITs)
Credential on File
Industry Practice
Standing Instruction
No Show
Resubmission
Reauthorization
Incremental
Delayed Charges
Installments
Recurring
Unscheduled Credential on File

COF Transactions

For a COF transaction, the cardholder does not need to enter card details because the merchant uses payment credentials previously stored by the cardholder. Examples include a transaction using a customer's merchant profile or staged digital wallet.

The COF mandate applies to token and primary account number (PAN) transactions and applies to Visa, Mastercard, and Discover.

Establishment of Relationship

The initial transaction must be a cardholder present Customer Initiated Transaction (CIT) which indicates that the credential is being stored on file for the first time (whether a zero-dollar authorization or first transaction). Under the terms of PSD2 this initial transaction must be strongly authenticated using 3D-Secure including step-up challenge, unless it is a Mail-Order or Telephone-Order (MOTO) transaction.

- SCMP API: subsequent_auth_first = Y
- Simple Order API: subsequentAuthFirst = true
- REST: processingInformation.authorizationOptions.initiator.credentialStoredOnFile = true

- SCMP API: e_commerce_indicator = internet, MOTO or payer authentication
- Simple Order API: ccAuthService_commerceIndicator = internet, MOTO or payer authentication
- REST: processingInformation.commerceIndicator = internet, MOTO or payer authentication

Note: Using this field when the cardholder is NOT present for subsequent transactions might cause declines because the issuer is expecting authentication data associated with the cardholder being present.

Merchants must request that the issuer performs step-up challenge in the 3DS enroll transaction by setting the following field:

- SCMP API: pa_challenge_code = 04
- Simple Order API: payerAuthEnrollService_challengeCode= 04
- REST: consumerAuthenticationInformation.challengeCode=04

Establishing a Relationship for Recurring Payments

In addition to the above fields, merchants must indicate when a credential is being stored for use on future recurring transactions by setting the following field during establishment of the relationship.

- SCMP API: auth_first_recurring_payment=Y
- Simple Order API: firstRecurringPayment=true
- REST: processingInformation.recurringOptions.firstRecurringPayment=true

Establishing a Relationship for Instalment Payments

In addition to the above fields, merchants must indicate when a credential is being stored for use on future instalment transactions by setting the following field during establishment of the relationship.

- SCMP API: installment_sequence=1

- Simple Order API: installment_sequence=1
- REST: installmentInformation.sequence=1

Store the Transaction Identifier for future MITs

Merchants storing credentials with the intention of performing merchant-initiated transactions in the future must also store the network transaction identifier from the authorization response. Note that for Visa the value from an MIT can be stored and used in subsequent MIT requests, but for other schemes future MIT requests must include the transactionID from the transaction when establishing the relationship. If using TMS this is automatically managed on behalf of merchants.

- SCMP API: auth_payment_network_transaction_id
- Simple Order API: paymentNetworkTransactionID
- REST: processorInformation.networkTransactionId

Consumer-Initiated Transaction (CIT) using COF

A card holder present CIT that uses payment information previously provided by the cardholder. The cardholder orders an item online and instructs you to use the payment information that is saved in your system.

- SCMP API: subsequent_auth_stored_credential = Y
- Simple Order API: subsequentAuthStoredCredential = true
- REST: processingInformation.authorizationOptions.initiator.storedCredentialUsed = true

Note: All other CIT use cases are considered business-as-usual. See the credit card guides on CyberSource.com for more information.

MIT—Subsequent Transactions

Merchants must evaluate their businesses to determine whether their transactions correspond to the network-defined use cases discussed below. New APIs have been developed to identify the transactions according to the Industry-Specific and Standing Instruction use cases.

Industry-Specific Business Practice MITs:

These MITs are known as 'Industry Practice" and are performed as a follow-up to an original cardholder-merchant interaction that could not be completed in a single transaction. Merchant-initiated transactions use stored credentials. The following transaction types are industry-specific transactions:

- Incremental
- Resubmission
- Reauthorization
- Delayed Charges
- No Show

Incremental

A merchant performs an initial card holder present pre-authorization storing the credentials-on-file for future use. The merchant then submits one or more incremental authorizations via the incremental authorization service linked to the initial pre-authorization. Incremental authorizations are only supported on Visa Platform Connect, FDI Global and Barclays. An example merchant performing this transaction type is a lodging merchant performing an initial pre-authorization at check-in, and incremental authorizations for purchases that exceed the originally pre-authorized amount, such as meals or drinks.

- SCMP API: ics_incremental_auth including auth_request_id
- Simple Order API: CCIncrementalAuthService including authRequestID
- REST: Patch to https://apitest.cybersource.com/pts/v2/payments/{id}

Resubmission

A merchant resubmits transactions that request authorization but were declined due to insufficient funds while the goods or services were already delivered to the cardholder. A merchant in this situation can resubmit the request to recover outstanding debt from cardholders. Merchants must include the network transaction if from the initial cardholder present CIT transaction on resubmission transactions.

- SCMP API:
  - subsequent_auth = Y
  - subsequent_auth_reason = 1
  - subsequent_auth_original_amount = <amount of original CIT transaction – Discover only>
  - subsequent_auth_stored_credential = Y
  - subsequent_auth_transaction_id = <auth_payment_network_transaction_id>
- Simple Order API:
  - subsequentAuth = true

    - subsequentAuthReason = 1
    - subsequentAuthOriginalAmount = <amount of original CIT transaction – Discover only>
    - subsequentAuthStoredCredential = true
    - subsequentAuthTransactionID = < paymentNetworkTransactionID>
- REST:
    - processingInformation.authorizationOptions.initiator.type = merchant
    - processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.reason = 1
    - processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction. originalAuthorizedAmount = < amount of original CIT transaction – Discover only >
    - processingInformation.authorizationOptions.initiator.storedCredentialUsed = true
    - processingInformation.authorizationOptions.initiator.merchantInitiated Transaction.previousTransactionID = < processorInformation.networkTransactionId >

Reauthorization

A merchant initiates a reauthorization when the original order or service is not complete or fulfilled within the authorization validity limit set by the scheme. Common instances that require reauthorizations include delayed shipments, split shipments, extended stays, and extended rentals.

- SCMP API:
    - subsequent_auth = Y
    - subsequent_auth_reason = 3
    - subsequent_auth_original_amount = <amount of original CIT transaction – Discover only>
    - subsequent_auth_stored_credential = Y
    - subsequent_auth_transaction_id = <auth_payment_network_transaction_id>
- Simple Order API:
    - subsequentAuth = true
    - subsequentAuthReason = 3
    - subsequentAuthOriginalAmount = <amount of original CIT transaction – Discover only>
    - subsequentAuthStoredCredential = true
    - subsequentAuthTransactionID = < paymentNetworkTransactionID>
- REST:
    - processingInformation.authorizationOptions.initiator.type = merchant
    - processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.reason = 3
    - processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction. originalAuthorizedAmount = < amount of original CIT transaction – Discover only >
    - processingInformation.authorizationOptions.initiator.storedCredentialUsed = true
    - processingInformation.authorizationOptions.initiator.merchantInitiated Transaction.previousTransactionID = < processorInformation.networkTransactionId >

Delayed Charge

A delayed charge is associated with an agreement between you and the cardholder for services rendered. Merchants might use delayed charges after providing services such as lodging, travel, or auto rental.

- SCMP API:
    - subsequent_auth = Y
    - subsequent_auth_reason = 2
    - subsequent_auth_original_amount = <amount of original CIT transaction – Discover only>
    - subsequent_auth_stored_credential = Y
    - subsequent_auth_transaction_id = <auth_payment_network_transaction_id>
- Simple Order API:
    - subsequentAuth = true
    - subsequentAuthReason = 2
    - subsequentAuthOriginalAmount = <amount of original CIT transaction – Discover only>
    - subsequentAuthStoredCredential = true
    - subsequentAuthTransactionID = < paymentNetworkTransactionID>
- REST:
    - processingInformation.authorizationOptions.initiator.type = merchant
    - processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.reason = 2
    - processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction. originalAuthorizedAmount = < amount of original CIT transaction – Discover only >
    - processingInformation.authorizationOptions.initiator.storedCredentialUsed = true
    - processingInformation.authorizationOptions.initiator.merchantInitiated Transaction.previousTransactionID = < processorInformation.networkTransactionId >

No-Show Transaction

A no-show transaction occurs when you and a cardholder have an agreement for a purchase, but the cardholder does not meet the terms of the agreement. No-show transactions are often used in hotels or restaurants where bookings are not honored despite the agreement entered into by the cardholder.

- SCMP API:
    - subsequent_auth = Y
    - subsequent_auth_reason = 4
    - subsequent_auth_original_amount = <amount of original CIT transaction – Discover only>
    - subsequent_auth_stored_credential = Y
    - subsequent_auth_transaction_id = <auth_payment_network_transaction_id>
- Simple Order API:
    - subsequentAuth = true
    - subsequentAuthReason = 4
    - subsequentAuthOriginalAmount = <amount of original CIT transaction – Discover only>
    - subsequentAuthStoredCredential = true
    - subsequentAuthTransactionID = < paymentNetworkTransactionID>
- REST:
    - processingInformation.authorizationOptions.initiator.type = merchant
    - processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.reason = 4
    - processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction. originalAuthorizedAmount = < amount of original CIT transaction – Discover only >
    - processingInformation.authorizationOptions.initiator.storedCredentialUsed = true

    - processingInformation.authorizationOptions.initiator.merchantInitiated Transaction.previousTransactionID = < processorInformation.networkTransactionId >

Standing-Instruction MITs

Standing-Instruction MITs are performed in order to follow agreed-upon instructions from the cardholder for the provision of goods or services. The following transaction types are standing-instruction transactions:

Installment Payment

An installment payment is one transaction in a series of transactions that uses stored credentials. This type of payment requires a cardholder agreement so that the merchant can initiate a series of fixed amount charges on a fixed schedule with a defined end date for a single purchase.

- SCMP API:
    - e_commerce_indicator = install
    - subsequent_auth = Y
    - subsequent_auth_stored_credential = Y
    - subsequent_auth_original_amount = <amount of original CIT transaction – Discover only>
    - subsequent_auth_transaction_id = < auth_payment_network_transaction_id >
- Simple Order API:
    - ccAuthService_commerceIndicator = install
    - subsequentAuth = true
    - subsequentAuthStoredCredential = true
    - subsequentAuthOriginalAmount = <amount of original CIT transaction – Discover only>
    - subsequentAuthTransactionID = < paymentNetworkTransactionID>
- REST:
    - processingInformation.commerceIndicator = install
    - processingInformation.authorizationOptions.initiator.type = merchant
    - processingInformation.authorizationOptions.initiator.storedCredentialUsed = true
    - processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction. originalAuthorizedAmount = < amount of original CIT transaction – Discover only >
    - processingInformation.authorizationOptions.initiator.merchantInitiated Transaction.previousTransactionID = < processorInformation.networkTransactionId >

Recurring Payment

A recurring payment is a transaction in a series of transactions that uses a stored credential and that is processed for a fixed amount at fixed, regular intervals (not to exceed one year). This type of transaction requires a cardholder agreement in order for the merchant to initiate future transactions at regular intervals for the purchase of goods or services.

- SCMP API:
    - e_commerce_indicator = recurring
    - subsequent_auth = Y
    - subsequent_auth_stored_credential = Y

    - subsequent_auth_original_amount = <amount of original CIT transaction – Discover only>
    - subsequent_auth_transaction_id = < auth_payment_network_transaction_id >
- Simple Order API:
    - ccAuthService_commerceIndicator = recurring
    - subsequentAuth = true
    - subsequentAuthStoredCredential = true
    - subsequentAuthOriginalAmount = <amount of original CIT transaction – Discover only>
    - subsequentAuthTransactionID = < paymentNetworkTransactionID>
- REST:
    - processingInformation.commerceIndicator = recurring
    - processingInformation.authorizationOptions.initiator.type = merchant
    - processingInformation.authorizationOptions.initiator.storedCredentialUsed = true
    - processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction. originalAuthorizedAmount = < amount of original CIT transaction – Discover only >
    - processingInformation.authorizationOptions.initiator.merchantInitiated Transaction.previousTransactionID = < processorInformation.networkTransactionId >

Unscheduled Credential on File (COF)

This type of transaction uses a stored credential for a fixed or variable amount and does not occur on a scheduled or regularly occurring transaction date. The cardholder must provide consent for the merchant to initiate one or more future transactions. An example of such a transaction is an account auto-top-up transaction.

- SCMP API:
    - e_commerce_indicator=internet
    - subsequent_auth=Y
    - subsequent_auth_stored_credential = Y
    - subsequent_auth_transaction_id = < auth_payment_network_transaction_id >
- Simple Order API:
    - ccAuthService_commerceIndicator = internet
    - subsequentAuth = true
    - subsequentAuthStoredCredential = true
    - subsequentAuthTransactionID = < paymentNetworkTransactionID>
- REST:
    - processingInformation.commerceIndicator = internet
    - processingInformation.authorizationOptions.initiator.type = merchant
    - processingInformation.authorizationOptions.initiator.storedCredentialUsed = true
    - processingInformation.authorizationOptions.initiator.merchantInitiated Transaction.previousTransactionID = < processorInformation.networkTransactionId >

Network Transaction IDs

Visa Transaction IDs

The ID of the original CIT transaction must be included for the following MIT scenarios:

- Resubmission
- Delayed Charges
- Reauthorization
- No Show
- Incremental

A transaction ID from the original or a previous successful transaction in the chain is required for the following MIT scenarios:

- Installment
- Recurring
- Unscheduled COF

Mastercard, Discover, American Express Transaction ID

The ID of the original CIT transaction must be included for all MIT scenarios:

- Resubmission
- Delayed Charges
- Reauthorization
- No Show
- Incremental
- Installment
- Recurring
- Unscheduled COF

Additionally, Discover requires that the transaction amount of the original CIT transaction is included for all MIT scenarios.