Authentication processes

The **3DS Requestor** performs three processes during an authentication:

1. **Initialise Authentication** - the 3DS Requestor sends a request to **ActiveServer** to initialise the authentication, preparing **ActiveServer** for the authentication.
2. **Execute Authentication** - **ActiveServer** executes the authentication. There are two main authentication flows in 3DS2, **frictionless flow** and **challenge flow**, which are described in the Process Flows section.
3. **Get Authentication Result** - the authentication result is returned to the 3DS Requestor.

Process 1: Initialise Authentication

In this step the **3DS web adapter** collects the information that the cardholder has entered at the front end and passes it to the **3DS Requestor** at the back end. The **3DS Requestor** then passes all the information required by 3DS2 to **ActiveServer** to start the authentication.

Using our example in the integration overview, when the customer selects ***Checkout***, the **3DS web adapter** sends an `initialise authentication` message to the 3DS Requestor.

The 3DS Requestor receives the `initialise authentication` message, formats it according to the **ActiveServer** API, and generates and adds a unique **3DS Requestor transaction ID** (`threeDSRequestorTransID`) to the message. Once the message is populated, it will be sent to **ActiveServer**.

When **ActiveServer** receives the `initialise authentication` message it will return the callback URLs, i.e. `threeDSServerCallbackUrl` for the **3DS Requestor** to setup page forwarding mechanisms on the checkout page (hidden iframes are used for current implementation of the callbacks in `3DS Web Adapter`). Once the iframes are setup, **ActiveServer** will be able to start the browser information collecting process and be ready for the authentication process.

Note

ActiveServer and the ACS automatically collect the browser information, and this process is **NOT** part of the **3DS Requestor**.

The only thing that may be related to this process in the **3DS Requestor** environment is that the `3DS web adapter` sets up the required hidden iframes automatically in the checkout page in order for the callback pages to be executed.

Requestor Browser Collection (RBC)

In some scenarios clients may want to use the requestor to collect and provide the browser information, instead of by ActiveServer automatically. ActiveServer can adapt to different requestor environments and support receiving the browser information, giving the requestor more control of the page flow.

To bypass the default browser information collecting process, an optional parameter `skipAutoBrowserInfoCollect` is available in the `/auth/brw/init` request. When this parameter is set to `true`, ActiveServer will **not** collect the browser information automatically.

The 3DS requestor must implement the browser info collecting through the user's browser with Javascript and collect the server side browser info fields, such as `browserUserAgent` and `browserIP`. Please check the requestor demo for details.

When `skipAutoBrowserInfoCollect` is set to `true`, providing the `acctNumber` in the `/auth/brw/init` request becomes **optional**. If the `acctNumber` is not provided, the 3DS Method process will be skipped.

The parameter `threeDSMethodAvailable` is introduced to the `/auth/brw/init` response, which will indicate whether the process is available for the provided `acctNumber`, allowing the client to choose whether they want to execute `threeDSServerCallbackUrl` or not. When `acctNumber` is not provided in the `/auth/brw/init` call, `threeDSMethodAvailable` is set to `false`, regardless of the card range that will be provided in the `/auth/brw` request.

Process 2: Execute Authentication

Once the browser information collection is complete, authentication can be executed. This will trigger **ActiveServer** to initiate the 3DS2 messaging process.

What if browser information collection failed?

It is not uncommon that the browser information collection process may fail or not be started at all. In this scenario, to improve user experience, **ActiveServer** will return an `InitAuthTimedOut` event (15 seconds after `InitAuth` request is called and no `brw` call thereafter) to the `3ds web adapter` to indicate that **Process 1** has failed. The checkout page can then handle this error accordingly.

Without this timeout mechanism, the checkout process could be stuck in the middle of processing as there won't be any callbacks sent by **ActiveServer**.

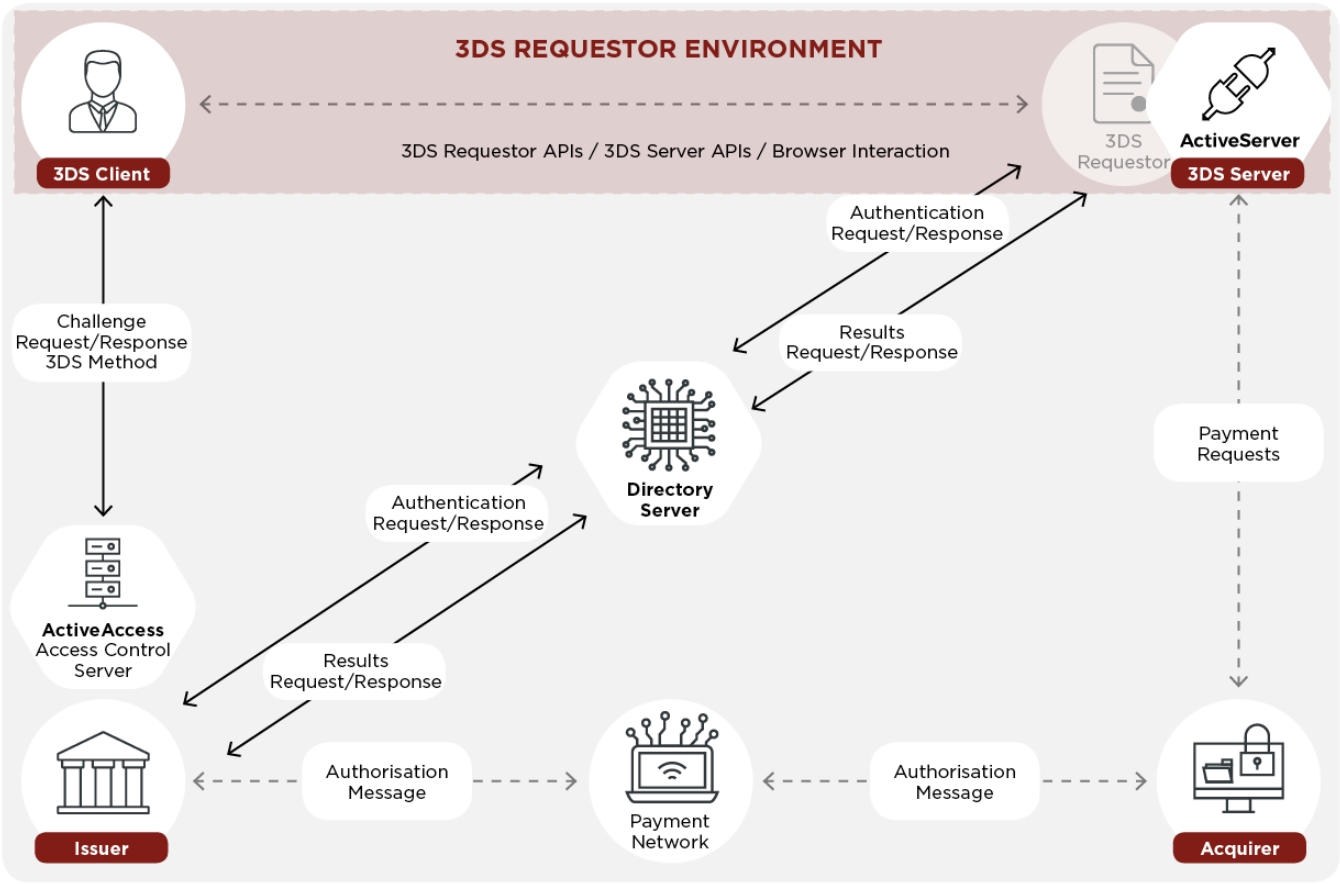
Requestor Browser Collection (RBC)

The requestor must pass the collected browser information as a parameter argument for the `browserInfoCollected` field as part of the `/auth/brw` request. Please check the requestor demo for details.

There are two main authentication flows in 3DS2, **frictionless flow** and **challenge flow**.

- **Frictionless flow** - initiates a 3D Secure authentication flow, which consists of the **AReq**/**ARes** authentication messages. If the ACS determines the transaction is low risk based on the information provided, authentication is approved immediately.
- **Challenge flow** - if the ACS determines that the transaction is high risk, above certain thresholds, or subject to regulatory mandates which requires further Cardholder interaction, the **frictionless flow** transitions into the **challenge flow**. In addition to the **AReq** and **ARes** messages that comprise **frictionless flow**, the **challenge flow** consists of the **CReq**/**CRes** challenge messages and the **RReq**/**RRes** result messages.

This diagram shows the challenge flow:



> *The dotted lines indicate messaging outside the scope of the 3DS2 protocol, including communication between the Client/3DS Requestor and Authorisation*

Process 3: Get Authentication Result

Once the 3DS2 process is complete, the merchant will get the authentication result. The authentication result (from the ARes or RRes depending on challenge status) contains information such as as the ECI, Authentication Value (e.g. CAVV) and final Transaction Status.

Note that the authentication result is also available in the response of **Process 2** if the authentication process is frictionless. However, the merchant site can always get the authentication result by calling the `Get Authentication Result` API.

What's next?

Select **Next** to learn more about the **Authentication sequence**.