Global Gateway 2.6

August 1

2014

This file describes the web services provided by version 2.6 of the ACHeck21 Global Gateway.

Table of Contents

Revision History

9/8/2009	CDW	Added *PostingDate* to *CreateCheck*, *CreateValidatedCheck*, *GetCheck* and *Find…ChecksDetails* web services. Added documentation of *CheckInfo* structure to *GetCheck* service
1/22/2010	CDW	Added *Addenda* to *CreateCheck* and *CreateValidatedCheck* web services. Added sample POST in Appendix 2.
2/11/2010	CDW	Added documentation of *CreateRCC* service.
03/28/2010	CDW	Remove query language documentation into a separate document.
06/07/2010	CDW	Fixed typo in *AuthorizeCheck* documentation. Changed Result Codes header to match field name.
7/20/2010	CDW	Fixed typo in *CreateCheck* stating that credit transactions have a transaction code of 26/36.
8/8/2010	CDW	Added missing specification for parameters to *CreateCheck*.
8/13/2010	CDW	Added explanation of 10011 error from *SendBatch* and the filename restrictions on that service.
11/26/2010	BTH	Updated *CreateCheck* service to support explicit images with BOC & ARC types. Added *C21* code for direct Check 21 utilization.
11/30/2010	BTH	Added new method *CreateCheckWithIR* for using ACH to Check 21 intelligent routing models.
1/14/2011	BTH	Added *Code* and *Message* to HTTP header for REST requests for better error handling on complex systems.
2/6/2011	BTH	Added new method *CreateValidatedCheckWithIR.* Added missing existing method *ValidateCheck.* Added missing *PhoneNumber* parameter to request on both *ValidateCheck* and *AuthorizeCheck* methods.
3/23/2011	BTH	New web services version 2.4. *CreateCheckWithIR* and *CreateValidatedCheckWithIR* has new parameter of *EntryClass* to set default SEC code.
12/9/2011	CDW	Added *SendRCKBatch* service.
12/23/2011	BTH	Upated *CreateRCC* API specification to match WSDL specification.

4/2/2012	CDW	Added *ToFedDate* to *FindChecks* query language.
4/2/2013	BTH	Added *PayerAddr4* to *CreateRCC* and *CreateValidatedRCC* methods
8/8/2013	BTH	Corrected a typo on the URL for version 2.5. Added HTML entity requirements to any XML query to be sent as a parameter.
4/10/2014	BTH	Added new Memo parameter to *CreateRCC* and *CreateValidatedRCC* with version 2.6
6/10/2014	BTH	Added new *CreateReturn* method

Documentation Suggestions

At ACHeck21, our goal is to provide payment processing solutions that help to make your business a success. If you have trouble with this document, or have suggestions for improvements, please email them to documentation@dcsdeposits.com. Please reference the publication date from the title page in any correspondence about the documentation.

Service address

The WSDL for the web services can be found at:

https://gateway.acheck21.com/GlobalGateway/WebServices/Gateway/2.6/Service.asmx?WSDL

Basic entities

The web services are classified according to *entities*, a concept that should be familiar to users of REST services. An entity is a conceptual class of web services, representing the item on which the service operates. Thus, the *CreateCheck* service is considered as part of the *Check* entity, because it creates a new check item in the system. The SOAP services do not have an explicit relationship with their entities, although where reasonable the name of the service should give some clue. The relationship is most explicit with the REST services, where the entity name is part of the URI for the service.

Whether explicit or implicit, this classification of services by entity can help in understanding how services interact and in identifying the appropriate service to use.

The table below lists the entities exposed by the web services, along with the operations performed on them. The remainder of this document describes the details of each entity and service.

Batch (entity)	*/batch*
ApproveBatch	PUT */<ClientID>/#/approve*
DeleteBatch	DELETE */<ClientID>/#*
GetBatch	GET */<ClientID>/#*
SendBatch	POST
	/batches
FindBatches	GET *?query*
Check (entity)	*/check*
CreateCheck	POST
CreateCheckWithIR	POST /IR
ValidateCheck	POST /validate
CreateValidatedCheck	POST /validated
CreateValidatedCheckWithIR	POST /validatedIR
CreateRCC	*Not Supported*
CreateIAT	POST /IAC
DeleteCheck	DELETE /#
GetCheck	GET/#
RepresentCheck	PUT */#/represent*
AuthorizeCheck	POST /authorize
	/checks
GetChecks	*unavailable*
FindChecks	GET*/<ClientID>?query*
FindChecksDetails	GET*/<ClientID>/details?query*
FindPendingChecks	GET*/<ClientID>/pending?query*
FindPendingChecksDetails	GET*/<ClientID>/pending/details?query*
FindReturns	GET*/<ClientID>/returns?query*
FindReturnsDetails	GET*/<ClientID>/returns/details?query*
Client (entity)	*/client*
GetClient	GET */#*
Clients (entity collection)	*/clients*
FindUserClients	GET */username*
User (entity)	*/user*
GetUser	GET */username*
DeleteUser	DELETE */username*
UpdateUser	PUT */username*
CreateUser	POST */username*
Users (entity collection)	*/users*
FindClientUsers	GET */ClientID*
Roles (entity collection)	*/roles*
GetRoles	GET
	Unavailable
Login	

	Authenticate	
	ExtendSessionToken	
	Logout	
RTN (entity)		*/rtn*
	VerifyTransitNumber	GET /TransitNumber

ACHeck21 Web Services – Basic Concepts

Authentication

Session authentication

Both service classes (SOAP and REST) provide a built-in mechanism for authentication. SOAP applications that wish to implement single-point of login coordinated with the security system built-into the gateway can use either the *Authenticate* or *Login* service to verify login credentials. *Login* returns a security token that provides access to a session with duration of 20 minutes. This session can be extended by using the *ExtendSessionToken* service, or will be automatically extended by SOAP services called within the 20 minute window using this token. *Authenticate* does not create a session, merely verifies the credentials, so it is appropriate for applications which will not be able to asynchronously maintain the session token.

Roles

The web services use the additive, role-based authentication model used in the Global Gateway web site. Some web services are only available to logins that have the correct role. The required role or roles are listed with each service.

All of the web services except for those that are part of the *Authenticate* entity require that the user have the *Gateway.WebService* role, in addition to whatever role is required for the service itself.

SOAP service

All SOAP services include authentication information. This allows client programs to run independently of any concept of a session. REST methods are different, and will be discussed in the next section. The authentication parameters are the *username* and *password*. Because all communication to the gateway is via https, there is no security risk here – the XML SOAP packets are encrypted in transit. Further, because usernames and passwords are fairly short, it may require less traffic to send the individual components than to send a session key. This also means that every web service can return an error code indicating that the authentication failed. This too is advantageous, as it allows finer control over authentication, allowing the system administrators to rapidly cut off access to the system in the event of security breaches.

SOAP applications which do not wish to keep login information for their duration can use the *Login*, *ExtendSessionToken*, and *Logout* services to maintain a session concept. Those applications should pass an empty *Password* and include the session token in the *Username* field. Using this approach, any SOAP calls made with the token in the *Username* field will extend the life of that token, just as if the *ExtendSessionToken* service had been called. Applications using this approach need to be able to guarantee that they will either extend the session within the 20 minute window or respond

appropriately to the possible expiration of their session token (possibly by requesting that the user login again, or by operating at a lower level of service, or by cleanly exiting).

REST service

The REST service layer supports Basic HTTP authentication to control access to the services. Thus, any time a REST service is called, if there is not an *Authorization* header, the service will return a 401 status code. The ACHeck21 gateway uses basic authentication as all communication is via https and therefore secure.

Because the REST services use this approach, there is no concept of a session (in fact, a session token will never be created for a REST client).

The *Code* and *Message* informational fields normally supplied in a SOAP response will only be available from within the HTTP header itself in order to maintain REST simplicity. All server based response codes should be ignored if the *Code* and *Message* fields have values within as these header fields will contain much more detailed responses as to why the error occurred.

Client IDs

Version 2 of the web services implements the single point of login feature first introduced in the *Micro* client. Using this feature, it is possible for a single user to be authorized to act on behalf of many clients. For this reason, most of the services have a parameter for the ACHeck21 *client ID* on whose behalf the action should be taken. This implementation allows a consolidating application to mimic the single point of login feature. To retrieve the list of clients on whose behalf the user is authorized to operate, use the *FindUserClients* service.

Queries

Services that provide searches of the back-end system (such as *FindBatches*) accept their queries in a parameter named *Query* that is represented as an untyped string of XML that is to be HTML entity encoded according to specification http://www.w3.org/TR/REC-xml/#syntax. If a language such as .NET is being employed, the automatically generated proxy stub will often times encode this for you automatically. This approach allows for changes to the gateway search system without invalidating the proxy stubs that existing client applications are using to reference the web service. Complete query language syntax can be found in the companion document "**ACHeck21 Query Language.pdf"**.

No changes will be made that result in a previously-legal query producing an error, although it may happen that the results of a query change across revision. Where this happens, we will attempt to document the change in the header for the web service.

Query parameters are generally very flexible, allowing the specification of the operation (equality, similarity, etc.) and the values to be searched. Programmers familiar with SQL will find the syntax comfortable, even though the syntax is it not exactly SQL. Where there are multiple search parameters, you can specify only those parameters that you care about for your search. Any unspecified parameters will simply not impact the search operation. For a full specification of the general query language see the document: *ACHeck21 Search Language*.

Updates

Services that update single entities in the system (such as *UpdateCheck*) accept new values as either named XML entries (SOAP) or as named form elements (REST). The services will only update the fields that are explicitly mentioned in the service call, leaving all other values untouched.

Return values

Each service returns a numeric value indicating the success or failure of the request. SOAP calls return this value in the XML response, along with any other requested information. REST calls return primarily an HTTP status code; however, the content of the response will be an error code, rather than any requested data, if an error occurs. For example, a PUT to gateway.acheck21.com/batch/1001?approve will return HTTP status code 204 (No content) when the operation succeeds, 404 if the batch could not be found, 403 if the batch is not in the *Pending* state, or 401 if the user is not authorized. When a 403 error is returned, the content of the response will include the ACHeck21 error code as well. The HTTP status codes are given in parenthesis after the ACHeck21 error code in the return codes section of each operation.

All services may return a 500 (Internal Server Error) if something unexpected happens. In those cases, the return data (either in the SOAP packet *Message* or in the body of the REST response, will contain more details on the actual error.

Batch (entity)

The *Batch* entity refers to batches of checks in the system. Batches are assigned unique identifiers by the ACH system, these identifiers are the keys returned from the *GetBatch* service, and are used as the reference key in all other *Batch* services.

ApproveBatch

This service provides for programmatic approval of a batch of transactions. Approval releases the transactions in the batch into the ACH or Check21 system for collection. Batches which have been approved cannot be edited or deleted. The operation will fail if the batch is not in the *Pending* state.

Security

The user must have the `Gateway.Deposit` role and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<ApproveBatch xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <BatchNbr>string</BatchNbr>
</ApproveBatch>
```

Response

```
<ApproveBatchResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</ApproveBatchResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/batch/*<ClientID>*/#/approve

Operation: PUT

Response: *empty*

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)
10002	Batch not in *Pending* state (403)

DeleteBatch

This service allows for deleting of a batch before approval. The batch is actually moved into the *Deleted* state, but it is not possible to do anything further with it. The operation will not fail if the batch was already deleted.

Security

The user must have the `Gateway.Deposit` role and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<DeleteBatch xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <BatchNbr>string</BatchNbr>
</DeleteBatch>
```

Response

```
<DeleteBatchResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</DeleteBatchResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/batch/*<ClientID>*/#

Operation: DELETE

Response: *empty*

Possible values for <Code>

0	No error (204)

10000	Not authorized (401)
10001	Item not found (404)

GetBatch

This service allows for the retrievable of batch-specific information. Note that, unlike the version 1 service, you cannot retrieve the transactions themselves with this service (for that, use the *FindTransactions* service). Searching is provided using either the batch number (assigned by ACHeck21), the sequence number (assigned by the client). Searches need only specify one of these identifiers (although specifying more than one will not cause any negative side-effects, assuming that the different identifiers refer to a single item).

You can only retrieve a single batch with this service. To retrieve information about multiple batches, you must make multiple calls.

Security

SOAP Service

Request

```
<GetBatch xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <BatchNbr>string</BatchNbr>
  <SeqNbr>string</SeqNbr>
</GetBatch>
```

Response

```
<GetBatchResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <BatchInfo>
    <BatchNbr>int</BatchNbr>
    <SeqNbr>int</SeqNbr>
    <Filename>string</Filename>
    <AcceptedCount>int</AcceptedCount>
    <RejectedCount>int</RejectedCount>
    <DuplicateCount>int</DuplicateCount>
    <AcceptedAmount>double</AcceptedAmount>
    <UploadDate>string</UploadDate>
    <ApprovedDate>string</ApprovedDate>
    <ApprovedBy>string</ApprovedBy>
    <DeletedDate>string</DeletedDate>
    <DeletedBy>string</DeletedBy>
    <BatchStatus>string</BatchStatus>
    <BatchTypeDescription>string</BatchTypeDescription>
    <PendingDisbursment>double</PendingDisbursment>
    <DisburseDate>string</DisburseDate>
    <Returns>
      <Count>int</Count>
      <Total>double</Total>
      <Date>string</Date>
```

```
      </Returns>
    </BatchInfo>
  </GetBatchResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/batch/*ClientID*? *search_criterion*

or

https://gateway.acheck21.com/GlobalGateway/REST/batch/ClientID/batchNbr

where *search_criterion* is one of:

batchnbr=*batch_nbr*

seqnbr=*seq_nbr*

Operation: GET

Response: Same as the SOAP response.

Possible values for <Code>

0	No error (200)
10000	Not authorized (401)
10001	Item not found (404)
10005	Parameter error (400)

SendBatch

Accepts a Base64 encoded string of a batch file. This batch must follow the XML, CSV or XFR specification in ASCII character set. Unicode is not accepted. Documentation of these formats is provided separately.

Security

The user must have the `Gateway.Upload` role and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<SendBatch xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <Base64EncodedBatch>string</Base64EncodedBatch>
  <Filename>string</Filename>
</SendBatch>
```

Response

```
<SendBatchResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</SendBatchResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/batch/*<ClientID>*

Operation: POST

Form contents: The form must contain elements corresponding to the *Base64EncodedBatch* and *Filename* elements in the XML. These form elements must have the same names as in the XML. Alternately, you can POST the XML, CSV or binary file directly, using multipart encoding, placing the file into a form element named *BatchFile*. This should be uploaded as if sent from an <input type="FILE" name="BatchFile" /> tag. In this latter case, the *Filename* field in the XML will be ignored (and need not be sent) and the name from the FILE element will be used in its place.
Response: *empty*

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10003	Base 64 string not valid (403)
10004	File format error (403)
10005	Parameter error (400)
10011	Duplicate item (403)

Notes

If the web service returns 10011 (Duplicate item), it is because a file with the same name was already uploaded, and the ACHeck21 system has not yet processed that file. Most of the time, waiting a couple of minutes and trying again will resolve the issue. However, once the original file is processed, uploading a second file with the same name will result in a duplication error on the Global Gateway. As a rule, different batches must have different filenames. Be aware that *SendBatch* will not normally return this error, as the back-end processing systems are usually quite fast.

SendRCKBatch

Accepts a Base64 encoded string of a batch file. This batch must follow the XML, CSV or XFR specification in ASCII character set. Unicode is not accepted. Documentation of these formats is provided separately. The items in the file will be converted to RCK items and presented for payment via the gateway. A PPD item for a processing fee, as determined by the merchant service agreement and state of origin will be created as well.

Security

The user must have the `Gateway.Upload` role and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<SendRCKBatch xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <Base64EncodedBatch>string</Base64EncodedBatch>
  <Filename>string</Filename>
</SendRCKBatch>
```

Response

```
<SendRCKBatchResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</SendRCKBatchResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/batch/*<ClientID>*/rck

Operation: POST

Form contents: The form must contain elements corresponding to the *Base64EncodedBatch* and *Filename* elements in the XML. These form elements must have the same names as in the XML. Alternately, you can POST the XML, CSV or binary file directly, using multipart encoding, placing the file into a form element named *BatchFile*. This should be uploaded as if sent from an <input type="FILE" name="BatchFile" /> tag. In this latter case, the *Filename* field in the XML will be ignored (and need not be sent) and the name from the FILE element will be used in its place.

Response: *empty*

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10003	Base 64 string not valid (403)
10004	File format error (403)
10005	Parameter error (400)
10011	Duplicate item (403)

Notes

If the web service returns 10011 (Duplicate item), it is because a file with the same name was already uploaded, and the ACHeck21 system has not yet processed that file. Most of the time, waiting a couple of minutes and trying again will resolve the issue. However, once the original file is processed, uploading a second file with the same name will result in a duplication error on the Global Gateway. As a rule, different batches must have different filenames. Be aware that *SendRCKBatch* will not normally return this error, as the back-end processing systems are usually quite fast.

Batches (entity collection)

The *Batches* entity refers to a collection of batches. This list contains no transactions, just a collection of batch information. Successive requests to services from the *Check* entity must be made to retrieve the actual transactions.

FindBatches

This service retrieves a list of batches that match search criteria. The for the query DTD is found at: https://gateway.acheck21.com/GlobalGateway/dtd/batches.dtd.

Security

The user must have either the `Gateway.User` role and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<FindBatches xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <Query>XML</Query>
</FindBatches>
```

Response

```
<FindBatchesResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <BatchCount>int</BatchCount>
  <Batches>
    <BatchNbr>int</BatchNbr>
    ...
    <BatchNbr>int</BatchNbr>
  </Batches>
</FindBatchesResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/batches/*<ClientID>*

Operation: GET

Form body: The form must contain a single INPUT element named *Query* that contains the XML definition of the query.

Response:

```
<uri>uri-to-batch</uri>
```

The first element is a direct URI for this batch; the second represents the URI that can be used to retrieve the list of transactions in this batch.

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10005	Parameter error (400)

Query-able fields

UploadDate:	The date the batch was originally uploaded.
Status:	One of *Pending*, *Rejected*, *Approved*, *Deleted*.
BatchNbr:	The system-assigned batch number.
SeqNbr:	The client-assigned batch number.

Check (entity)

The *Check* entity refers to a single check in the system. This can either be a new transaction being uploaded, or an existing transaction on which the return and disbursement status is being checked.

CreateCheck

Create a new transaction in the system. This service can be used to create check with or without images. If images are sent, the transaction will be routed via the Check21 system if possible.

Security

The user must have the `Gateway.E-Check` role (for transactions without images) or the `Gateway.Check21` role (for transactions with images); and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<CreateCheck xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <ClientTag>Conditional - string</ClientTag>
  <IndividualName>string</IndividualName>
  <CheckNumber>string</CheckNumber>
  <TransitNumber>string</TransitNumber>
  <DDANumber>string</DDANumber>
  <AccountType>Checking or Savings</AccountType>
  <CheckAmount>string</CheckAmount>
  <EntryClass>TEL, PPD, ARC, RCK, CCD, WEB, BOC, POP, C21</EntryClass>
  <FrontImage>Base64-encoded TIFF G4 image @200bpi</FrontImage>
  <RearImage>Base64-encoded TIFF G4 image @200bpi</RearImage>
  <MICR>Optional – string</MICR>
  <PostingDate>Optional – string</PostingDate>
  <Addenda>
    <Addendum>Optional - value</Addendum>
  </Addenda>
</CreateCheck>
```

Response

```
<CreateCheckResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <CheckID>integer</CheckID>
</CreateCheckResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/check

Operation: POST

Form contents: The form must contain elements corresponding to the *ClientID*, *ClientTag*, *IndividualName*, *CheckNumber*, *TransitNumber*, *DDANumber*, *AccountType*, *CheckAmount*, and *EntryClass* elements in the XML, unless sending a raw MICR line, in which case the *MICR* field must be present in lieu of the components. These form elements must have the same names as in the XML. If

sending a BOC, ARC, or Check21 transaction, the images must either be in Base64-encoded strings with the appropriate names in the form, or as attachments with the correct name in a multi-part form. If sending a posting date or addenda, these elements will be represented in appropriately-named form elements. Transactions with images must **not** have *ClientTag* set as this field is set by the Global Gateway for specialized handling.

Response:

```
<uri>uri-to-new-check</uri>
```

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10003	Base 64 string not valid (403)
10005	Parameter error (400)
10006	Client not authorized (401)
10011	Duplicate item (403)
10012	Transaction exceeds client transaction limit (403)
10013	Check amount causes daily total to exceed daily limit (403)
10014	Check amount causes monthly total to exceed monthly limit (403)
10015	RDFI not qualified to participate (403)
10016	Corporate customer advises not authorized (403)
10017	CheckNotPreviouslyAuthorized (403)
10019	Error in addenda sent (400)
10020	Addenda not supported for entry class (403)
10022	Auxiliar On-Us detected in MICR line for ACH item (ineligible) (403)

Notes

The *ClientTag* and *IndividualName* fields are optional, the rest of the fields must be provided. No checking is done to verify that the *TransitNumber* or *DDANumber* fields are legal values, only that the *TransitNumber* is the correct length (9 characters) and internally consistent (that the check digit is correct). The DDA number is limited to 15 characters, the check number to 15 characters.

If the *AccountType* is not provided, the system assumes *Checking*.

The C*heckAmount* field must be numeric, have no more than 2 digits after the decimal place, and no dollar sign or internal commas and not be equal to $0.00. The *ClientTag*, if provided, can have any format needed by the caller. The service does not require or check that the value be unique. *ClientTag* must **not** be used in the event images are to be sent.

To send a credit, the *CheckAmount* should be negative. This triggers an ACH transaction code of either 22 or 32.

If sending a raw MICR line, the value will be accepted only if it conforms to an acceptable format, and if the transit number meets the criteria above for the *TransitNumber* field. Note that sending a raw MICR

precludes sending values for *TransitNumber*, *DDANumber*, and *CheckNumber*. If both are sent, error code 10005 will be returned.

The 10006 return code indicates that the client is not authorized to submit transactions of the type indicated by the *EntryClassCd*.

The C21 *EntryClassCd* can only be used for Check21 transactions. This will later be converted to a 937 type for later reporting access.

The *PostingDate* is optional. If not provide, the check will be posted the next time the client's e-checks are batched. If not empty, it cannot specify a date in the past. The format of the *PostingDate* is mm/dd/yyyy.

At this time, addenda records are only supported for PPD transactions. There can only be a single addendum, and it can contain no more than 80 characters. ACHeck21 does no validation of the contents of the addendum, that is the responsibility of the sender. Of course, the addendum is optional even for PPD transactions.

The *CheckID* (or URI) returned is a unique identifier for the new check. It can be used in successive calls to retrieve information about the document. The identifier will remain valid for the lifetime of the document within the ACHeck21 system.

Check21 Notes

Images will be decoded and their format verified. No IQA checking will be done, that is the responsibility of the sender. Both front and rear images must be sent. Errors in the image files will result in error code 10003.

The *AccountType* field is not required for Check21 transactions. If sent, it will be changed to Checking automatically.

The *EntryClassCd* must be C21 for Check21 transactions. This will be reassigned to 937 later automatically. C21 is used as various financial institutions may use a format other than X9.37 but is still considered a Check 21 based transaction.

CreateCheckWithIR

Create a new transaction in the system utilizing the ACH to Check 21 Intelligent Routing model. This service must have images. This service differs from *CreateCheck* in that the *EntryClass* submitted will default to *C21* if an item is not able to be processed through the ACH network and is then sent using the Check 21 network. This will limit the kinds of ACH items that can be processed and imposes greater restrictions on information that needs to be supplied.

Security

The user must have the `Gateway.E-Check` and `Gateway.Check21`; and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<CreateCheckWithIR xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <IndividualName>string</IndividualName>
  <CheckNumber>string</CheckNumber>
  <TransitNumber>string</TransitNumber>
  <DDANumber>string</DDANumber>
  <AccountType>Checking or Savings</AccountType>
  <CheckAmount>string</CheckAmount>
  <EntryClass>ARC, BOC, POP</EntryClass>
  <FrontImage>Base64-encoded TIFF G4 image @200bpi</FrontImage>
  <RearImage>Base64-encoded TIFF G4 image @200bpi</RearImage>
  <MICR>Optional – string</MICR>
  <PostingDate>Optional – string</PostingDate>
</CreateCheckWithIR>
```

Response

```
<CreateCheckWithIRResult
xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <CheckID>integer</CheckID>
  <CheckInfo>
    <CheckID>integer</CheckID>
    <UploadDate>date-time</UploadDate>
    <IndividualName>string</IndividualName>
    <CheckNumber>string</CheckNumber>
    <TransitNumber>string</TransitNumber>
    <DDANumber>string</DDANumber>
    <AccountType>string</AccountType>
    <CheckAmount>string</CheckAmount>
    <EntryClass>string</EntryClass>
    <ClientTag>string</ClientTag>
    <SentToFed>Boolean</SentToFed>
    <ReturnStatus>
      <Return>
        <UploadDate>yyyy-mm-dd</UploadDate>
        <ReturnDate>yyyy-mm-dd</ReturnDate>
        <ReturnCode>string</ReturnCode>
      </Return>
    </ReturnStatus>
  </CheckInfo>

</CreateCheckWithIRResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/check/IR

Operation: POST

Form contents: The form must contain elements corresponding to the *ClientID*, *IndividualName*, *CheckNumber*, *TransitNumber*, *DDANumber*, *AccountType, EntryClass*, and *CheckAmount* elements in the XML, unless sending a raw MICR line, in which case the *MICR* field must be present in lieu of the

components. These form elements must have the same names as in the XML. The images must either be in Base64-encoded strings with the appropriate names in the form, or as attachments with the correct name in a multi-part form. If sending a posting date, these elements will be represented in appropriately-named form elements.

Response:

```
<uri>uri-to-new-check</uri>
```

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10003	Base 64 string not valid (403)
10005	Parameter error (400)
10006	Client not authorized (401)
10011	Duplicate item (403)
10012	Transaction exceeds client transaction limit (403)
10013	Check amount causes daily total to exceed daily limit (403)
10014	Check amount causes monthly total to exceed monthly limit (403)
10017	CheckNotPreviouslyAuthorized (403)

Notes

The *IndividualName* field is optional, the rest of the fields must be provided. No checking is done to verify that the *TransitNumber* or *DDANumber* fields are legal values, only that the *TransitNumber* is the correct length (9 characters) and internally consistent (that the check digit is correct). The DDA number is limited to 15 characters, the check number to 15 characters.

EntryClass must be ARC, BOC or POP. Any other SEC code submitted will return a 10005 Parameter error.

If the *AccountType* is not provided, the system assumes *Checking*.

The C*heckAmount* field must be numeric, have no more than 2 digits after the decimal place, and no dollar sign or internal commas and not be equal to $0.00.

Credits are not supported.

If sending a raw MICR line, the value will be accepted only if it conforms to an acceptable format, and if the transit number meets the criteria above for the *TransitNumber* field. Note that sending a raw MICR precludes sending values for *TransitNumber*, *DDANumber*, and *CheckNumber*. If both are sent, error code 10005 will be returned.

The *PostingDate* is optional. If not provide, the check will be posted the next time the client's e-checks are batched. If not empty, it cannot specify a date in the past. The format of the *PostingDate* is mm/dd/yyyy.

The *CheckID* (or URI) returned is a unique identifier for the new check. It can be used in successive calls to retrieve information about the document. The identifier will remain valid for the lifetime of the document within the ACHeck21 system.

The *CheckInfo* field is returned to assist the user in identifying how a transaction will be processed whether that be using ACH or Check 21. Evaluating this field is necessary to know the routing model used.

Check21 Notes

Images will be decoded and their format verified. No IQA checking will be done, that is the responsibility of the sender. Both front and rear images must be sent. Errors in the image files will result in error code 10003.

ValidateCheck

This service presents a check for validation against a sequence of check validation engines, including but not limited to RTN verification, bad check authorization, etc. The actual sequence of validations performed if configured when the client is initially boarded. To simply present a check to a check authorization company, use the *AuthorizeCheck* service.

Security

The user must have either the `Gateway.AuthorizeCheck` role.

SOAP Service

Request

```
<ValidateCheck xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <MICR>string</MICR>
  <TransitNumber>string</TransitNumber>
  <DDANumber>string</DDANumber>
  <CheckNumber>string</CheckNumber>
  <CheckAmount>money</CheckAmount>
  <DLNumber>money</DLNumber>
  <PhoneNumber>string</PhoneNumber>
</ValidateCheck>
```

Response

```
<ValidateCheckResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <Validation>
    <Response>Authorized OR Declined OR Warning OR Error</Response>
    <AuthorizationCode>string</AuthorizationCode>
    <DetailLines>
      <string>line</string>
      …
      <string>line</string>
    </DetailLines>
  </Validation>
```

```
</ValidateCheckResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/check/validate

Operation: POST

Form contents: The form must contain elements corresponding to the *ClientID, MICR*, *TransitNumber*, *DDANumber*, *CheckNumber*, *CheckAmount*, *and DLNumber* elements in the XML. These form elements must have the same names as in the XML.

Response: The return value is the *Authorization* portion of the SOAP response.

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)

Notes

The request should contain either the *MICR* or the separate components (*TransitNumber*, *DDANumber*, *CheckNumber*) from the check. If both are provided, the software will use the *MICR* value. The *MICR* must be expressed using the TOAD encoding, where the transit symbol is encoded as a 'T', the on-us as an 'O', the amount symbol as an 'A', and the dash as a 'D'. Note that the characters must be upper-case, using lower case letters will result in a parameter error.

The *DLNumber* is optional. If provided, it must contain the 2 character state abbreviation followed by a hyphen, then the actual license number; for example:

AZ-D123456

means Arizona drivers license number D123456. Failure to conform to this format will result in an error from the check verification system.

The authorization code, if successful, will be stored on the ACHeck21 Global Gateway, where it can be retrieved once the actual check is uploaded.

CreateValidatedCheck

Create a new transaction in the system. This service can be used to create check with or without images. If images are sent, the transaction will be routed via the Check21 system if possible. The check will first be run through the check verification system (see *ValidateCheck* for more information).

Security

The user must have the `Gateway.E-Check` role (for transactions without images) or the `Gateway.Check21` role (for transactions with images); and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<CreateValidatedCheck
    xmlns="https://gateway.acheck21.com/GlobalGateway/">
```

```
      <Username>string</Username>
      <Password>string</Password>
      <ClientID>string</ClientID>
      <ClientTag>string</ClientTag>
      <IndividualName>string</IndividualName>
      <CheckNumber>string</CheckNumber>
      <TransitNumber>string</TransitNumber>
      <DDANumber>string</DDANumber>
      <AccountType>Checking or Savings</AccountType>
      <CheckAmount>string</CheckAmount>
      <EntryClass>TEL, PPD, ARC, RCK, CCD, WEB or BOC</EntryClass>
      <FrontImage>Base64-encoded TIFF G4 image @200bpi</FrontImage>
      <RearImage>Base64-encoded TIFF G4 image @200bpi</RearImage>
      <MICR>Optional – string</MICR>
      <DLNumber>string</DLNumber>
      <PhoneNumber>string</PhoneNumber>
      <PostingDate>Optional – date</PostingDate>
      <Addenda>
        <Addendum>value</Addendum>
      </Addenda>
    </CreateValidatedCheckResult>
```

Response

```
    <CreateValidatedCheckResult
xmlns="https://gateway.acheck21.com/GlobalGateway/">
      <Code>int</Code>
      <Message>string</Message>
      <ValidationData>
        <Response>Authorized or Declined or Warning or Error</Response>
        <AuthorizationCode>string</AuthorizationCode>
        <DetailLines>
          <string>string</string>
          <string>string</string>
        </DetailLines>
      </ValidationData>
      <CreateData>
        <CheckID>int</CheckID>
      </CreateData>
    </CreateValidatedCheckResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/check/authorized

Operation: POST

Form contents: The form must contain elements corresponding to the *ClientID*, *ClientTag*, *IndividualName*, *CheckNumber*, *TransitNumber*, *DDANumber*, *AccountType*, *CheckAmount*, *EntryClass*, *DLNumber*, and *PhoneNumber* elements in the XML, unless sending a raw MICR line, in which case the *MICR* field must be present in lieu of the components. These form elements must have the same names as in the XML. If sending a Check21 transaction, the images must either be in Base64-encoded strings with the appropriate names in the form, or as attachments with the correct name in a multi-part form.

Response: Still undefined.

Possible values for <Code>

The service can return any code returned from either the *CreateCheck* or *AuthorizeCheck* web services.

Notes

This service combines the behaviors of *ValidateCheck* followed by *CreateCheck*. As such, it requires the parameters of each. For more details in the meaning and requirements of each parameter and the return values, see the descriptions for those 2 services.

CreateValidatedCheckWithIR

Create a new transaction in the system. This service can be used to create check with images intended to be used with the Intelligent Routing model. The check will first be run through the check verification system (see *ValidateCheck* for more information).

Security

The user must have the `Gateway.E-Check` role and the `Gateway.Check21` role; and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<CreateValidatedCheck
    xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <IndividualName>Optional - string</IndividualName>
  <CheckNumber>string</CheckNumber>
  <TransitNumber>string</TransitNumber>
  <DDANumber>string</DDANumber>
  <AccountType>Checking or Savings</AccountType>
  <CheckAmount>string</CheckAmount>
  <EntryClass>ARC, BOC, POP</EntryClass>
  <FrontImage>Base64-encoded TIFF G4 image @200bpi</FrontImage>
  <RearImage>Base64-encoded TIFF G4 image @200bpi</RearImage>
  <MICR>Optional – string</MICR>
  <DLNumber>Optional - string</DLNumber>
  <PhoneNumber>Optional - string</PhoneNumber>
  <PostingDate>Optional – date</PostingDate>
</CreateValidatedCheck>
```

Response

```
<CreateValidatedCheckResult
xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <ValidationData>
    <Response>Authorized or Declined or Warning or Error</Response>
    <AuthorizationCode>string</AuthorizationCode>
    <DetailLines>
```

```
        <string>string</string>
        <string>string</string>
      </DetailLines>
    </ValidationData>
    <CheckInfo>
      <CheckID>integer</CheckID>
      <UploadDate>date-time</UploadDate>
      <IndividualName>string</IndividualName>
      <CheckNumber>string</CheckNumber>
      <TransitNumber>string</TransitNumber>
      <DDANumber>string</DDANumber>
      <AccountType>string</AccountType>
      <CheckAmount>string</CheckAmount>
      <EntryClass>string</EntryClass>
      <ClientTag>string</ClientTag>
      <SentToFed>Boolean</SentToFed>
      <ReturnStatus>
        <Return>
          <UploadDate>yyyy-mm-dd</UploadDate>
          <ReturnDate>yyyy-mm-dd</ReturnDate>
          <ReturnCode>string</ReturnCode>
        </Return>
      </ReturnStatus>
    </CheckInfo>
  </CreateValidatedCheckResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/check/authorizedIR

Operation: POST

Form contents: The form must contain elements corresponding to the *ClientID*, *CheckNumber*, *TransitNumber*, *DDANumber*, *AccountType*, *CheckAmount*, *DLNumber, EntryClass*, and *PhoneNumber* elements in the XML, unless sending a raw MICR line, in which case the *MICR* field must be present in lieu of the components. These form elements must have the same names as in the XML.

Possible values for <Code>

The service can return any code returned from either the *CreateCheckWithIR* or *ValidateCheck* web services.

Notes

This service combines the behaviors of *ValidateCheck* followed by *CreateCheckWithIR*. As such, it requires the parameters of each. For more details in the meaning and requirements of each parameter and the return values, see the descriptions for those 2 services.

CreateRCC

This service creates an RCC transaction using images generated by the ACHeck21 web service. These images are designed to maximize the acceptability of the image.

Security

The user must have the `Gateway.CreateRCC` role, and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<CreateRCC
    xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <PayerName>string</PayerName>
  <PayerAddr1>string</PayerAddr1>
  <PayerAddr2>string</PayerAddr2>
  <PayerAddr3>string</PayerAddr3>
  <CheckDate>string</CheckDate>
  <PayeeName>string</PayeeName>
  <Amount>string</Amount>
  <TransitNumber>string</TransitNumber>
  <DDANumber>string</DDANumber>
  <CheckNumber>string</CheckNumber>
  <SignatureText>string</SignatureText>
  <Endorse1>string</Endorse1>
  <Endorse2>string</Endorse2>
  <Endorse3>string</Endorse3>
  <Endorse4>string</Endorse4>
</CreateRCC>
```

Response

```
<CreateRCCResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <CheckID>integer</CheckID>
</CreateRCCResult>
```

CreateReturn

This service creates a Return transaction for a *CheckID*. The item must exist in the system and be a valid reject style return code (R01, R02, etc).

Security

The user must have the `Gateway.Returns.Upload` role, and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<CreateReturn
    xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <CheckID>string</CheckID>
  <PayerName>string</PayerName>
  <ReturnCode>string</ReturnCode>
</CreateReturn>
```

Response

```
<CreateReturnResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</CreateReturnResult>
```

Possible values for <Code>

0	No error (204)
10001	Item Not Found (404)
10000	Not authorized (401)
10005	Parameter error (400)
10006	Client not authorized (401)
10011	Duplicate item (403)

Notes

Please refer to Appendix 3 for valid return codes for use with *ReturnCode* parameter

DeleteCheck

This service deletes an existing document from the system. This is only possible until the item is submitted into the ACH system, at which point there is nothing that can be done to the document.

Security

The user must have the `Gateway.E-Check` role (or `Gateway.CreateRCC` if deleting an RCC transaction) and the *ClientID* that created the check must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<DeleteCheck xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <CheckID>string</CheckID>
</DeleteCheck>
```

Response

```
<DeleteCheckResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</DeleteCheckResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/check/#

Operation: DELETE

Response: *empty*

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)

10001 Item not found (404)

Notes

The return code of 10001 (Item not found) can result from either a programming error (passing an incorrect *CheckID*) or from attempting to delete a check once it has been submitted to the ACH system. At this moment, the only way to distinguish would be to follow a 10001 return code with a call to *GetCheck* with the same *DocumentID*. If the check has been submitted, the item will be found by that service.

GetCheck

This service retrieves a single document. To retrieve more than one document, use the services of the *Checks* entity. *CheckID*s are returned by the *FindChecks* and *CreateCheck* services.

Security

The user must have either the `Gateway.User` role and the *ClientID* that created the check must be in the *ReportsTo* tree for the authenticated user. Return information will only be provided if the user also has the `Gateway.Raturns` role.

SOAP Service

Request

```
<GetCheck xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <CheckID>string</CheckID>
</GetCheck>
```

Response

```
<GetCheckResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <CheckInfo>
    <CheckID>integer</CheckID>
    <UploadDate>date-time</UploadDate>
    <IndividualName>string</IndividualName>
    <CheckNumber>string</CheckNumber>
    <TransitNumber>string</TransitNumber>
    <DDANumber>string</DDANumber>
    <AccountType>string</AccountType>
    <CheckAmount>string</CheckAmount>
    <EntryClass>string</EntryClass>
    <ClientTag>string</ClientTag>
   <SentToFed>Boolean</SentToFed>
    <ReturnStatus>
      <Return>
        <UploadDate>yyyy-mm-dd</UploadDate>
        <ReturnDate>yyyy-mm-dd</ReturnDate>
        <ReturnCode>string</ReturnCode>
      </Return>
    </ReturnStatus>
  </CheckInfo>
</GetCheckResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/check/#

Operation: GET

Response: Same as the SOAP response.

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

Notes

The *SentToFed* flag returned by the service will be true if the item has been sent via either the ACH or Check21 systems. If this flag is *false*, then the item can still be deleted or updated.

RepresentCheck

This service represents a single check specified by a *CheckID*. The service accepts new values for the *TransitNumber*, *DDANumber*, and *CheckNumber* fields. As of this moment, the check amount cannot be changed at this point. This service can only be used for checks that have already been submitted and returned from the ACH or Check21 systems. To fix values in a posted but not submitted e-check, use the *UpdateCheck* service.

Security

The user must have the `Gateway.E-Check` role and the *ClientID* that created the check must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<RepresentCheck xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <CheckID>string</CheckID>
  <CheckNumber>string</CheckNumber>
  <TransitNumber>string</TransitNumber>
  <DDANumber>string</DDANumber>
</RepresentCheck>
```

Response

```
<RepresentCheckResult
xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</RepresentCheckResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/check/#/represent

Operation: PUT

Response: *empty*

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)
10007	Check has not been returned, so cannot be re-presented (403)
10008	Check has reached re-presentment limit (403)

UpdateCheck

There is no distinct *UpdateCheck* service. To modify a check created using *CreateCheck* but which has not been submitted to the ACH or Check21 systems, delete then re-create the check. To re-present a check that has been returned, use the *RepresentCheck* service.

AuthorizeCheck

This service submits a check for authorization against a check verification service, such as ECHO.

Security

The user must have either the `Gateway.AuthorizeCheck` role.

SOAP Service

Request

```
<AuthorizeCheck xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <MICR>string</MICR>
  <TransitNumber>string</TransitNumber>
  <DDANumber>string</DDANumber>
  <CheckNumber>string</CheckNumber>
  <CheckAmount>money</CheckAmount>
  <DLNumber>money</DLNumber>
</AuthorizeCheck>
```

Response

```
<AuthorizeCheckResult
xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <Authorization>
    <Response>Authorized OR Declined OR Warning</Response>
    <AuthorizationCode>string</AuthorizationCode>
    <DetailLines>
      <string>line</string>
      …
      <string>line</string>
    </DetailLines>
  </Authorization>
</AuthorizeCheckResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/check/authorize

Operation: POST
Form contents: The form must contain elements corresponding to the *ClientID, MICR*, *TransitNumber*, *DDANumber*, *CheckNumber*, *CheckAmount*, *and DLNumber* elements in the XML. These form elements must have the same names as in the XML.
Response: The return value is the *Authorization* portion of the SOAP response.

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)

Notes

The request should contain either the *MICR* or the separate components (*TransitNumber*, *DDANumber*, *CheckNumber*) from the check. If both are provided, the software will use the *MICR* value. The *MICR* must be expressed using the TOAD encoding, where the transit symbol is encoded as a 'T', the on-us as an 'O', the amount symbol as an 'A', and the dash as a 'D'. Note that the characters must be upper-case, using lower case letters will result in a parameter error.

The *DLNumber* is optional. If provided, it must contain the 2 character state abbreviation followed by a hyphen, then the actual license number; for example:

AZ-D123456

means Arizona drivers license number D123456. Failure to conform to this format will result in an error from the check verification system.

The authorization code, if successful, will be stored on the ACHeck21 Global Gateway, where it can be retrieved once the actual check is uploaded.

Checks (entity collection)

The *Checks* entity refers to a collection of checks. Services here provide mechanisms to retrieve more than one document at a time and for retrieving return information.

GetChecks

This service returns check detail information (see the *GetCheck* service) for one or more checks identified by *CheckID*.

Security

The user must have either the `Gateway.User` role and the *ClientID* that created the check must be in the *ReportsTo* tree for the authenticated user. Return information will only be provided if the user also has the `Gateway.Returns` role.

SOAP Service

Request

```
<GetChecks xmlns="https://gateway.acheck21.com/GlobalGateway/">
```

```
    <Username>string</Username>
    <Password>string</Password>
    <CheckIDs>
      <CheckID>int</CheckID>
      ...
      <CheckID>int</CheckID>
    </CheckIDs>
  </GetChecks>
```

Response

```
  <GetChecksResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
    <Code>int</Code>
    <Message>string</Message>
    <Checks>
      <CheckInfo>...</CheckInfo>
      ...
      <CheckInfo>...</CheckInfo>
    </Checks>
  </GetChecksResult>
```

REST Service

There is no REST version of this service at this time.

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

Notes

The *CheckInfo* structured returned by the web service has the following C# definition:

```
public class CheckInfo {
    public int CheckID { get; set; }

    public string UploadDate { get; set; }
    public string IndividualName { get; set; }
    public string CheckNumber { get; set; }
    public string TransitNumber { get; set; }
    public string DDANumber { get; set; }
    public AccountType AccountType { get; set; }
    public Decimal CheckAmount { get; set; }
    public string ClientTag { get; set; }
    public string EntryClass { get; set; }
    public DateTime PostingDate { get; set; }
    public bool SentToFed { get; set; }
    public Return[] ReturnStatus { get; set; }
} // class CheckInfo
```

FindReturns

This service returns a list of *CheckID*s for returned documents that match the specified criteria.

Security

The user must have the `Gateway.Returns` role and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<FindReturns xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <Query>XML</Query>
</FindReturns>
```

Response

```
<FindReturnsResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <Checks>
    <CheckID>int</CheckID>
    ...
    <CheckID>int</CheckID>
  </Checks>
</FindReturnsResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/checks/*<ClientID>*/returns?*query*

Operation: GET

Response: A series of the following:

```
<uri>uri-to-returned-check</uri>
```

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

Query-able fields

UploadDate:	The date the item was originally uploaded.
ReturnDate:	The date the item was returned.
ReturnCode:	The NACHA return code to search for. A list of recognized NACHA codes (including several provided by ACHeck21® for image returns) can be found in the appendix under ACHeck21/NACHA return code list.
SeqNbr:	The user-assigned batch number for the originally uploaded document.

FindReturnsDetails

This service combines the *FindReturns* and *GetChecks* services to retrieve the details of returned documents. For the internal format of the *CheckInfo* returned, see the *GetCheck* service description.

SOAP Service

Request

```
<FindReturnsDetails xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <Query>XML</Query>
</FindReturnsDetails>
```

Response

```
<FindReturnsDetailsResult
    xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <Checks>
    <CheckInfo>...</CheckInfo>
    ...
    <CheckInfo>...</CheckInfo>
  </Checks>
</FindReturnsDetailsResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/checks/*<ClientID>*/returns/details?*query*

Operation: GET

Response: Same as the SOAP response.

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

Query-able fields

Same as *FindReturns*.

FindChecks

This service provides a mechanism for searching the system for deposited documents that match specified criteria. To search for items that have been submitted via the *CreateCheck* service but have not yet been released, use the *FindPendingChecks* service.

Security

The user must have either the `Gateway.User` role and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<FindChecks xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
```

```
  <Query>XML</Query>
</FindChecks>
```

Response

```
<FindChecksResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <Checks>
    <CheckID>int</CheckID>
    ...
    <CheckID>int</CheckID>
  </Checks>
</FindChecksResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/checks/*<ClientID>?query*

Operation: GET

Response: A series of the following:

```
<uri>uri-to-returned-check</uri>
```

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

Query-able fields

UploadDate:	The date the batch was originally uploaded
ToFedDate:	The date the item was deposited
Amount:	The check amount
SeqNbr:	The user-assigned sequence number for the batch in which the item was originally uploaded
Name:	The individual name on the item
TransitNbr:	The ABA routing/transit number
AccountNbr:	The bank account number
CheckNbr:	The actual check number
ClientTag:	The original document client tag sent when the item was uploaded
EntryClass:	The ACH entry class code. Legal values are: 937, ARC, BOC, PPD, TEL, WEB, RCC or CCD

FindChecksDetails

This service combines the *FindChecks* and *GetChecks* services to retrieve the details of documents matching a specified criterion with a single round-trip. For the internal format of the *CheckInfo* returned, see the *GetCheck* service description.

SOAP Service

Request

```
<FindChecksDetails xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <Query>XML</Query>
</FindChecksDetails>
```

Response

```
<FindChecksDetailsResult
    xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <Checks>
    <CheckInfo>...</CheckInfo>
    ...
    <CheckInfo>...</CheckInfo>
  </Checks>
</FindChecksDetailsResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/checks/*<ClientID>*/details*?query*

Operation: GET

Response: The same as the SOAP response.

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

Query-able fields

Same as *FindChecks*.

FindPendingChecks

This service provides a mechanism for searching the system for pending documents (documents that were created using the *CreateCheck* service but which have not yet been released for processing) that match specified criteria. To search for items that were uploaded in a batch, or that have been released for processing, use the *FindChecks* service.

Security

The user must have either the `Gateway.User` role and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<FindPendingChecks xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
```

```
  <ClientID>string</ClientID>
  <Query>XML</Query>
</FindPendingChecks>
```

Response

```
<FindPendingChecksResult
    xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <Checks>
    <CheckID>int</CheckID>
    ...
    <CheckID>int</CheckID>
  </Checks>
</FindPendingChecksResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/checks/*<ClientID>*/pending*?query*

Operation: GET

Response: A series of the following:

```
<uri>uri-to-returned-check</uri>
```

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

Query-able fields

UploadDate:	The date the batch was originally uploaded
Amount:	The check amount
SeqNbr:	The user-assigned sequence number for the batch in which the item was originally uploaded
Name:	The individual name on the item
TransitNbr:	The ABA routing/transit number
AccountNbr:	The bank account number
CheckNbr:	The actual check number
ClientTag:	The original document client tag sent when the item was uploaded
EntryClass:	The ACH entry class code. Legal values are: 937, ARC, BOC, PPD, TEL, WEB, RCC or CCD

FindPendingChecksDetails

This service combines the *FindPendingChecks* and *GetChecks* services to retrieve the details of pending documents matching a specified criterion with a single round-trip. For the internal format of the *CheckInfo* returned, see the *GetCheck* service description.

Security

The user must have either the `Gateway.User` role and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<FindPendingChecksDetails
    xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <Query>XML</Query>
</FindPendingChecksDetails>
```

Response

```
<FindPendingChecksDetailsResult
    xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <Checks>
    <CheckInfo>...</CheckInfo>
    ...
    <CheckInfo>...</CheckInfo>
  </Checks>
</FindPendingChecksResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/checks/*<ClientID>*/pending/details*?query*

Operation: GET

Response: The same as the SOAP response.

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

Query-able fields

Same as the *FindPendingChecks* service.

Client (entity)

The *Client* entity refers to an individual ACHeck21 client or ODFI. Services provided on this entity allow for retrieving information about individual clients.

GetClient

This service returns basic control information about an individual client from the client ID.

Security

The user must have either the `Gateway.User` role and the *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<GetClient xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
</GetClient>
```

Response

```
<GetClientResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <ClientInfo>
    <ClientID>string</ClientID>
    <Active>boolean</Active>
    <City>string</City>
    <State>string</State>
    <Zip>string</Zip>
    <ClientName>string</ClientName>
    <ContactEmail>string</ContactEmail>
    <ContactPerson>string</ContactPerson>
    <Phone1>string</Phone1>
    <Phone2>string</Phone2>
    <Fax>string</Fax>
    <Website>string</Website>
    <AchCompanyName>string</AchCompanyName>
  </ClientInfo>
</GetClientResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/client/*<ClientID>*

Operation: GET

Response: The return value is the XML content of the *ClientInfo* portion of the SOAP response.

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

Clients (entity collection)

This entity refers to a collection of clients, specified by client ID.

FindUserClients

This service returns a collection of *ClientIDs* specifying those clients whose data the user has access to. To simplify the expected, typical use of the service, it returns not only the client ID, but the client name and the last four digits of the depository account.

Security

If the user has either the `Gateway.Manage` roles, the base client for *SearchUser* must be in the *ReportsTo* tree for the authenticated user. Otherwise, the user must have either the `Gateway.User` and the *SearchUser* must be the same as the authenticated user.

SOAP Service

Request

```
<FindUserClients xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <SearchUser>string</SearchUser>
</FindUserClients>
```

Response

```
<FindUserClientsResult
xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <ClientCount>integer</ClientCount>
  <Clients>
    <ShortClientInfo>
      <ClientID>string</ClientID>
      <ClientName>string</ClientName>
      <ClientLast4>string</ClientLast4>
    </ShortClientInfo>
    ...
    <ShortClientInfo>
      <ClientID>string</ClientID>
      <ClientName>string</ClientName>
      <ClientLAst4>string</ClientLAst4>
    </ShortClientInfo>
  </Clients>
</FindUserClientsResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/clients/<*username*>

Operation: GET

Response: Same as SOAP service *Clients* tag.

Possible values for <Code>

0	No error (200)
10000	Not authorized (401)
10001	Item not found (404)

User (entity)

A *User* is the security-control point for the system. The services provided here allow applications to retrieve and update information about individual users.

CreateUser

This service creates a new association between a user and a client. If the user does not exist, it will be created with the given password. Successive calls to *UpdateUser* can be made to define other fields for the user.

Security

The user must have either the `Gateway.Manage` role, and *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<CreateUser xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
  <NewUsername>string</NewUsername>
  <NewPassword>string</NewPassword>
</CreateUser>
```

Response

```
<CreateUserResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</CreateUserResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/user

Operation: POST

Form contents: The form must contain an element corresponding to the *ClientID*, *NewUsername*, and *NewPassword* elements in the XML. These form elements must have the same names as in the XML.

Response:

```
<uri>uri-to-new-user</uri>
```

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)
10009	The user name does not meet requirements (400)
10010	The password does not meet requirements (400)
10011	Duplicate item (403)

GetUser

This service retrieves basic information about a user. The service does not return the current password. If a user has lost their password, it can be reset from the login screen of the web site, or it can be reset using the *UpdateUser* service.

Security

The user must have either the `Gateway.Manage` role, and the base client for the user to be retrieved must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<GetUser xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <SearchUser>string</SearchUser>
</GetUser>
```

Response

```
<GetUserResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <UserInfo>
    <Active>boolean</Active>
    <ClientID>string</ClientID>
    <FirstName>string</FirstName>
    <LastName>string</LastName>
    <Roles>
      <Role>string</Role>
      ...
      <Role>string</Role>
    </Roles>
  </UserInfo>
</GetUserResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/user/<*username*>

Operation: GET

Response: The same as the *User* tag of the SOAP response.

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

DeleteUser

This service is used to delete a user. In order to call this service, the user making the request must have access to the root client associated with the user.

This service can be called by a user to "self-delete," in which case the user's credentials will be invalidated. If authentication was via session key, the session will be expired as well.

Security

The user must have either the `Gateway.Manage` role, and the base client for the user to be deleted must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<DeleteUser xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <SearchUser>string</SearchUser>
</DeleteUser>
```

Response

```
<DeleteUserResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</DeleteUserResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/user/<*username*>

Operation: DELETE

Response: *none*

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

UpdateUser

This service allows for modification of information about an individual user. The only required fields are the authentication fields and the *SearchUser*. Only those fields that are included in the request are actually updated, all other fields are left untouched. If there is a *Roles* tag, then the user's existing roles will be replaced; therefore, if the calling application wishes to add a new Role, they must first retrieve the current role list using the *GetUser* service. The *Roles* tag contains a comma-separated list of roles to be assigned to the user. The roles in the list are not validated; to retrieve a list of currently defined roles, you can use the *GetRoles* service to retrieve a list of roles and explanations.

Security

The user must have either the `Gateway.Manage` role, and the base client for the user to be updated must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<UpdateUser xmlns="https://gateway.acheck21.com/GlobalGateway/">
```

```
  <Username>string</Username>
  <Password>string</Password>
  <SearchUser>string</SearchUser>
  <NewPassword>string</NewPassword>
  <Active>boolean</Active>
  <ClientID>string</ClientID>
  <FirstName>string</FirstName>
  <LastName>string</LastName>
  <Roles>string</Roles
</UpdateUser>
```

Response

```
<UpdateUserResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</UpdateUserResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/user/<*username*>

Operation: PUT

Response: *none*

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)
10009	The user name does not meet requirements (400)
10010	The password does not meet requirements (400)

Users (entity collection)

The *Users* entity defines a collection of *User* entities.

FindClientUsers

This service allows applications to retrieve the list of user match a specified query.

Security

The user must have either the `Gateway.Manage` role, and *ClientID* must be in the *ReportsTo* tree for the authenticated user.

SOAP Service

Request

```
<FindClientUsers xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
  <ClientID>string</ClientID>
</FindClientUsers>
```

Response

```
 <FindClientUsersResult
xmlns="https://gateway.acheck21.com/GlobalGateway/">
   <Code>int</Code>
   <Message>string</Message>
   <UserCount>integer</UserCount>
   <Users>
     <User>username</User>
     ...
     <User>username</User>
   </Users>
 </FindClientUsersResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/users/*<ClientID>*

Operation: GET

Response: A list of URIs to the retrieved users:

```
<uri>uri-to-user</uri>
...
<uri>uri-to-user</uri>
```

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)
10001	Item not found (404)

Roles (entity collection)

The *Roles* collection is provided to allow authenticating applications to retrieve a list of system-defined roles.

GetRoles

This service retrieves the list of system-known roles.

Security

The user must have either the `Gateway.Manage` role.

SOAP Service

Request

```
 <GetRoles xmlns="https://gateway.acheck21.com/GlobalGateway/">
   <Username>string</Username>
   <Password>string</Password>
 </FindClientUsers>
```

Response

```
 <GetRolesResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
   <Code>int</Code>
   <Message>string</Message>
   <RoleCount>integer</RoleCount>
```

```
  <Roles>
    <Role>username</Role>
    ...
    <Role>username</Role>
  </Roles>
</GetRolesResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/role

Operation: GET

Response: A list of roles:

```
<role>rolename</role>
...
<role>rolename</role>
```

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)

Authenticate (entity)

The *Authenticate* entity encapsulates the services provided for authenticating users and maintaining session information.

Login

This service authenticates a username / password combination and provides a session token that can be used in other service calls. These session tokens expire after 20 minutes, unless explicitly extended using the *ExtendSessionToken* service, or implicitly extended by using the token in another service call.

Security

There are no a priori restrictions on the roles of the user to be authenticated. Note however, that if there are no Roles set up for the user, they will not be able to successfully access any services outside of the *Authenticate* entity.

SOAP Service

Request

```
<Login xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
</Login>
```

Response

```
<LoginResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <SessionToken>string</SessionToken>
</LoginResult>
```

REST Service

This entity does not have REST services. Access to REST is via Basic HTTP authentication.

Possible values for <Code>

0	No error (200)
10000	Not authorized (401)

Authenticate

This service authenticates a username / password combination. It does not create a session nor return a session token. This service can be used to implement a scheme where a user is authenticated at startup, and the username/password combination retained for later service calls.

Security

There are no a priori restrictions on the roles of the user to be authenticated. Note however, that if there are no Roles set up for the user, they will not be able to successfully access any services outside of the *Authenticate* entity.

SOAP Service

Request

```
<Authenticate xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Username>string</Username>
  <Password>string</Password>
</Authenticate>
```

Response

```
<AuthenticateResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
  <SessionToken>string</SessionToken>
</AuthenticateResult>
```

REST Service

This entity does not have REST services. Access to REST is via Basic HTTP authentication.

Possible values for <Code>

0	No error (200)
10000	Not authorized (401)

ExtendSessionToken

This service can be used to explicitly extend a session token. This can be useful in applications which do not predictably call other services, but which can schedule intermittent calls to this service to guarantee that the session token remain valid when needed by the main execution thread.

Security

The token must not be expired.

SOAP Service

Request

```
<ExtendSessionToken xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <SessionToken>string</SessionToken>
</ExtendSessionToken>
```

Response

```
<ExtendSessionTokenResult
xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</ExtendSessionTokenResult>
```

REST Service

This entity does not have REST services. Access to REST is via Basic HTTP authentication.

Possible values for <Code>

0	No error (204)
10000	Not authorized (401)

Logout

This service can be used applications to explicitly invalidate a session token when it is no longer required. While not technically necessary, as sessions will expire after 20 minutes if not extended; this service can allow well-behaved applications to increase their overall security by eliminating the token when the application exits.

When calling the REST service, the token in the URI must be the same as the value of the *Authorization* header.

Security

The call never returns an error code, even if passed an invalid or expired session key.

SOAP Service

Request

```
<Logout xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Token>string</Token>
</Logout>
```

Response

```
<LogoutResult xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</LogoutResult>
```

REST Service

This entity does not have REST services. Access to REST is via Basic HTTP authentication.

Possible values for <Code>

0	No error (204)

RTN (entity)

The *RTN* entity encapsulates the services provided for verifying routing/transit numbers.

VerifyTransitNumber

This service accepts a 9 digit transit number and verifies whether it is a known legal transit number. This method uses the FedACH list as provided via *webservicex*, and is subject to the limitations of that list.

Security

There are no role restrictions on the use of this service; in fact, the SOAP service does not even require a username/password pair. Be aware that for architectural reasons, the REST service does require that the caller be authenticated.

SOAP Service

Request

```
<VerifyTransitNumber xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <TransitNumber>string</TransitNumber>
</VerifyTransitNumber>
```

Response

```
<VerifyTransitNumberResult
    xmlns="https://gateway.acheck21.com/GlobalGateway/">
  <Code>int</Code>
  <Message>string</Message>
</VerifyTransitNumberResult>
```

REST Service

URL: https://gateway.acheck21.com/GlobalGateway/REST/rtn/<*transitnumber*>

Operation: GET

Response: *none*

Possible values for <Code>

0	No error (204)
10001	Item not found (404)

Appendix 1 – POST sample

POST Sample

The following html page demonstrates how to use a web page to access the REST services, in this case the *SendBatch* service. Simply change the URI in the *action* attribute to include the correct client ID, place the Base-64 encoded file contents in the *value* of the *Base64EncodedBatch* element, and the correct file name in the *value* of the *Filename* element. Once done, pressing the *Submit* button on the page will upload the batch. The other REST services can be called in analogous ways.

```
<html>
<head>
    <title>POST sample</title>
</head>

<body>
<form id="mainform" method="post"
      action="https://gateway.acheck21.com/GlobalGateway/REST/batch/ClientID">
  <input type="text" name="Base64EncodedBatch" value="data" /><br />
  <input type="text" name="Filename" value="filename with extension" /><br />
  <input type="submit" />
</form>
</body>
</html>
```

Appendix 2 – Error Codes

ACHeck21 Error Codes

These are the values that could be returned in the <Code> field from one of the web services.

0	No error (200, 204)
10000	Not authorized (401)
10001	Item not found (404)
10002	Batch not in *Pending* state (403)
10003	Base 64 string not valid (403)
10004	File format error (403)
10005	Parameter error (400)
10006	Client not authorized (401)
10007	Check has not been returned, so cannot be re-presented (403)
10008	Check has reached re-presentment limit (403)
10009	The user name does not meet requirements (400)
10010	The password does not meet requirements (400)
10011	Duplicate item (403)
10012	Transaction exceeds client transaction limit (403)
10013	Check amount causes daily total to exceed daily limit (403)
10014	Check amount causes monthly total to exceed monthly limit (403)
10015	RDFI not qualified to participate (403)
10016	Corporate customer advises not authorized (403)
10017	Check not rreviously authorized (403)
10018	Posting date is in the past (400)
10019	Error in addenda sent (400)
10020	Addenda not supported for entry class (403)

HTTP Error Codes

The following HTTP error codes are returned by the REST service.

200	OK
204	No content
400	Bad request
401	Unauthorized
403	Forbidden
404	Not found
409	Conflict

Code 200 indicates that the operation completed successfully, and the content of the response is the data. Code 204 indicates that the operation completed successfully but that no data is returned. When codes 400 or 403 are returned, the HTTP header will contain two new fields, *Code* and *Message,* of which the error code and message will be returned. For all other HTTP error codes, content is not required by the HTTP specification, and may or may not be returned. The calling application should not assume it will receive any HTTP content for those errors. For a simple primer on HTTP status codes, readers can refer to the page at: http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html.

Appendix 3 – NACHA Return Codes

ACHeck21/NACHA return code list

The following are the standard NACHA return codes:

R01 Insufficient Funds
R02 Account Closed
R03 No Account/Unable to Locate Account
R04 Invalid Account Number
R05 Reserved
R06 Returned per ODFI Request
R07 Authorization Revoked by Customer (adjustment entries)
R08 Payment Stopped or Stop Payment on Item
R09 Uncollected Funds
R10 Customer Advises Not Authorized
R11 Check Truncation Entry Return or State Law Affecting Acceptance of PPD Accounts
R12 Branch Sold to another DFI
R13 RDFI not qualified to participate
R14 Representative Payee Deceased or Unable to Continue in that Capacity
R15 Beneficiary or Account Holder (Other than a Representative Payee) Deceased
R16 Account Frozen
R17 File Record Edit Criteria
R18 Improper effective entry date
R19 Amount field error
R20 Non-Transaction Account
R21 Invalid Company Identification
R22 Invalid Individual ID Number
R23 Credit Entry Refused by Receiver
R24 Duplicate Entry
R25 Addenda Error
R26 Mandatory Field Error
R27 Trace Number Error
R28 Routing Number Check Digit Error
R29 Corporate Customer Advises Not Authorized
R30 RDFI Not Participant in Check Truncation Program
R31 Permissible Return Entry
R32 RDFI Non-Settlement
R33 Return of XCK Entry
R34 Limited Participation DFI
R37 The source document to which a check entry relates has been presented for payment
R38 Stop Payment on Source Document
R39 Improper source document, ineligible ACH item

The following codes are specific to ACHeck21®:

R67	Duplicate return
RED	Exceeds dollar limit
REI	Endorsement irregular
REV	Transaction reverse by ACHeck21
RIR	X9.37 Image Rejection
RRL	Signature irregular
RUP	Unable to process item (e.g. mutilated check, not our item)
RXS	Refer to Maker