Risk Inquiry Service (RIS) Technical Specifications Guide

Data Collector and Risk Inquiry System (RIS)
v. 6.4.5

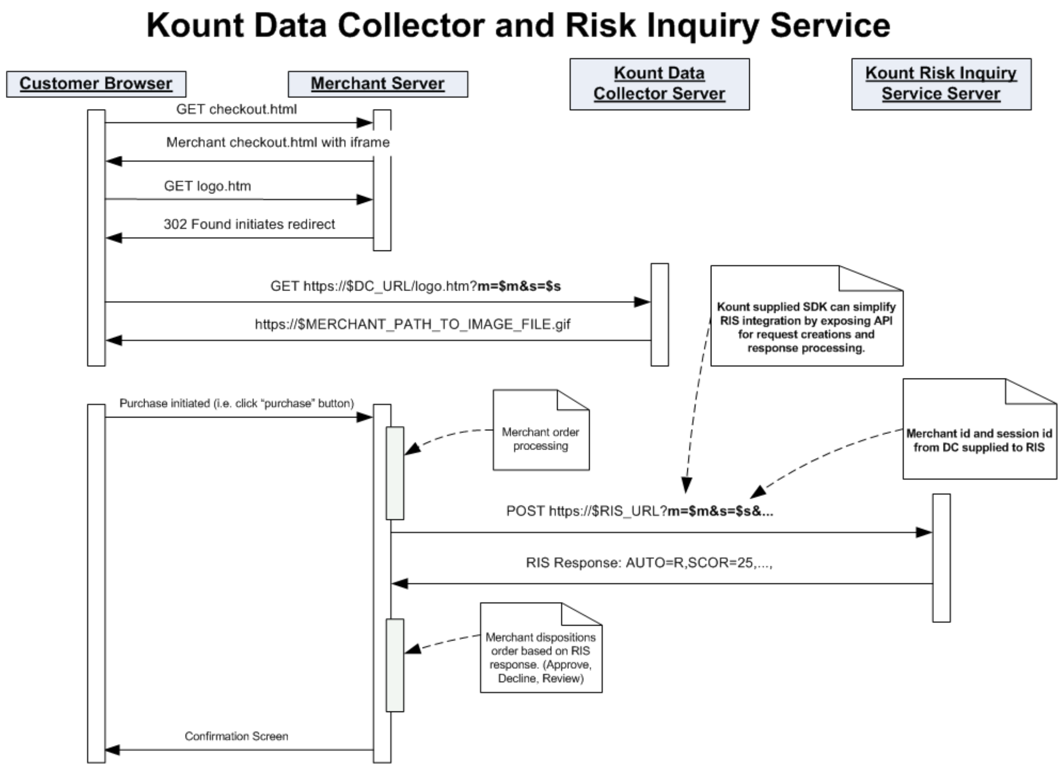
Contents

Overview

Kount aggregates and evaluates data from two primary sources, the **Data Collector (DC)** and the **Risk Inquiry Service (RIS)**. Subsequent to collecting and evaluating customer data, Kount returns a string of key-value pairs to the merchant. Order related data is also displayed in the Agent Web Console (AWC) accessible via a secured merchant login over the Internet.

The Data Collector gathers information from a customer's device by redirecting the device browser momentarily to Kount then back to the merchant. This passive analysis obfuscates Kount's interaction with the customer and does not affect the customer's purchasing experience.

The Risk Inquiry Service evaluates the data provided by the Data Collector and the order-form data submitted from the merchant to create a fraud score. Merchant specified rules are also assessed for each transaction during this evaluation process. Once an order has been evaluated a response string of key-value pairs is returned to the merchant including a score, device fingerprint, and an automated response code. Upon receipt of this response data the merchant can disposition orders based upon specified rules.


Kount Data Collector and Risk Inquiry Service
Customer Browser
Merchant Server
Kount Data Collector Server
Kount Risk Inquiry Service Server
GET checkout.html
Merchant checkout.html with iframe
GET logo.htm
302 Found initiates redirect
GET https://$DC_URL/logo.htm?m=$m&s=$s
https://$MERCHANT_PATH_TO_IMAGE_FILE.gif
Kount supplied SDK can simplify RIS integration by exposing API for request creations and response processing.
Purchase initiated (i.e. click "purchase" button)
Merchant order processing
Merchant id and session id from DC supplied to RIS
POST https://$RIS_URL?m=$m&s=$s&...
RIS Response: AUTO=R,SCOR=25,...,
Merchant dispositions order based on RIS response. (Approve, Decline, Review)
Confirmation Screen

Kount Environments

Kount has separate environments for test and production. Initial integration for both the Data Collector and Risk Inquiry Service will take place in the test environment. A boarding document containing required test environment information will be sent to the merchant.

The test environment is **not** engineered to support load testing; it is designed primarily to verify connectivity and proper data submission. Many features such as order linking, scoring, device location, and persona related information are limited in the test environment.

Test credit cards can be passed into the test environment but will fail in the production environment.

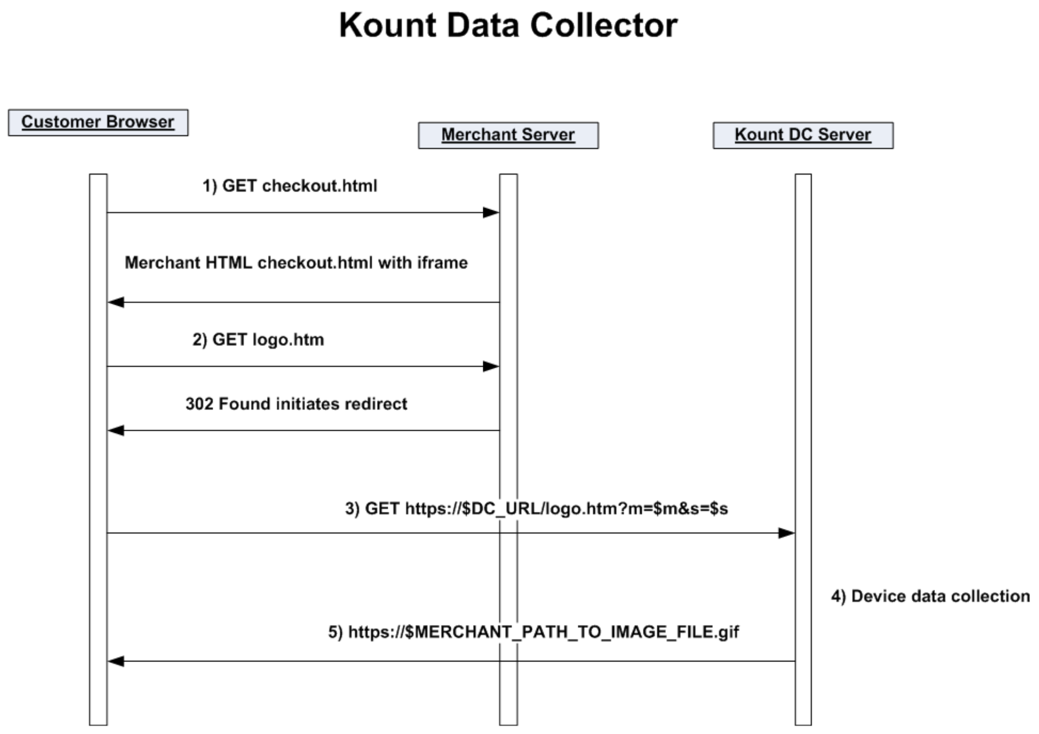
Port 443/HTTPS is required for submission and receipt of Data Collector and Risk Inquiry Service data in both the test and production environments.

Upon certifying that the correct data is being passed for both the Data Collector and Risk Inquiry Service the merchant will be issued a Certification Letter and additional production boarding information. Any customized data created in the test environment will have to be re-created in the production environment, this includes, users, rules, site IDs, user defined fields, and certificates.

The test environment will continue to be available to the merchant for testing purposes but should not be used with production traffic from the merchant. Typically Kount has quarterly releases with new or enhanced features that are backward compatible. These new features will be available in the test environment as well.

Data Collector

The Data Collector process provides data related to the device initiating the transaction, this process is transparent to the customer and sub second. The data collected is used in conjunction with the RIS data.


Kount Data Collector
Customer Browser
Merchant Server
Kount DC Server
1) GET checkout.html
Merchant HTML checkout.html with iframe
2) GET logo.htm
302 Found initiates redirect
3) GET https://$DC_URL/logo.htm?m=$m&s=$s
4) Device data collection
5) https://$MERCHANT_PATH_TO_IMAGE_FILE.gif

The following sequence describes the Data Collector process:

1. Customer browses to merchant order form page containing Data Collector code
2. Customer browser automatically requests redirect on the Data Collector element
3. Customer browser is redirected to Kount servers
4. Kount collects device attributes
5. Kount directs customer browser to display static image hosted by merchant

Data Collector Requirements

1. Port 443 must be available to post and receive data from Kount
2. Merchant adds code to their checkout process.
    - HTML iframe:
        - The iframe should be placed on the order form page where the payment method or credit card information is requested, usually near the bottom of the page.
    - Server side code:
        - The server side code consists of a logo.htm and logo.gif server side executable scripts.
        - The path to the server side code must be a fully qualified path.
    - Code to create a unique session identifier:
        - When the form page is accessed, a session identifier must be created and stored to be used as the session identifier in the subsequent RIS post from the merchant to Kount. This identifier must be unique for at least 30 days and must be unique for every transaction submitted by each unique customer. If a single session ID were to be used on multiple transactions, those transactions would link together and erroneously affect the persona information and Kount score.
3. Merchant supplies static image URL to Kount.
    - The static image can be an image that is currently being displayed on the page or Kount can provide an image if desired.
    - If the path or image requires change by the merchant subsequent to moving into production, Kount must be notified of the new path or filename to avoid possible failure.

The following code examples demonstrate various ways to code for the data collector. Additional details can be found in the FAQ and Troubleshooting section.

DISCLAIMER: All of the code presented in this document is for example purposes only. Any production implementation performed by the customer should follow all internal code quality standards of the customer's organization including security review.

Data	Size	Description	Example
COMPANY_SERVER_URL	N/A	HTTPS URL path to the company's servers provided in boarding documentation from Kount	https://tst.kaptcha.com
MERCHANT_ID	6	Six digit identifier issued by Kount.	999999
SESSION_ID	Min: 1 Max: 32	Alpha-numeric string that must be unique over a thirty (30) day period of time created by merchant.	1BDB721BA17E4A4BB58B21A5460D0B
MERCHANT_URL	N/A	Merchant's website URL	https://www.somecompany.com/

Session creation code example

```
<?php
$sess = session_id();
if (!$sess) {
    // If the session hasn't already been started, start it now and look up the id
    session_start();
    $sess = session_id();
}

// The session id is now available for use in the variable $sess
// For more details and examples on working with sessions in PHP, see:
// http://us2.php.net/manual/en/book.session.php
// http://us2.php.net/session_start
// http://us2.php.net/session_id
?>
```

HTML iframe example

Requirements – The iframe has a minimum width=1 and height=1

```
<iframe width=1 height=1 frameborder=0 scrolling=no src=https://MERCHANT_URL/logo.htm?
     m=merchantId&s=sessionId>
     <img width=1 height=1 src=https://MERCHANT_URL/logo.gif?m=merchantId&s=sessionId>
</iframe>
```

Server side code examples

PHP

logo.htm

```
<?php
     $m=$_GET['m'];
     $s=$_GET['s'];
     header ("HTTP/1.1 302 Found");
     header ("Location: https://COMPANY_SERVER_URL/logo.htm?m=$m&s=$s");
?>
```

logo.gif

```
<?php
     $m=$_GET['m'];
     $s=$_GET['s'];
     header ("HTTP/1.1 302 Found");
     header ("Location: https://COMPANY_SERVER_URL/logo.htm?m=$m&s=$s");
?>
```

JAVA

LogoHtm.java

```
public void doGet (HttpServletRequest request, HttpServletResponse response)
          throws ServletException, IOException {
     String merchantId = request.getParameter("m"); //Supplied Merchant ID
     String sessionId = request.getParameter("s"); //Session ID
     String url = "https://COMPANY_SERVER_URL/logo.htm?m=" + merchantId + "&s=" + sessionId;
     response.setStatus(HttpServletResponse.SC_FOUND);
     response.sendRedirect(url);
}
```

LogoGif.java

```
public void doGet (HttpServletRequest request, HttpServletResponse response)
          throws ServletException, IOException {
     String merchantId = request.getParameter("m"); //Supplied Merchant ID
     String sessionId = request.getParameter("s"); //Session ID
     String url = "https://COMPANY_SERVER_URL/logo.htm?m=" + merchantId + "&s=" + sessionId;
     response.setStatus(HttpServletResponse.SC_FOUND);
     response.sendRedirect(url);
}
```

.NET

Add the following to the appSettings section of the Web.config file:

```
<appSettings> <add key="DC.Url" value="https://tst.kaptcha.com" />
```

logo.aspx

```
public void Page_Load (object sender, System.EventArgs e) {
     Response.Redirect(string.Format("{0}/logo.htm?m={1}&s={2}",
     ConfigurationManager.AppSettings["DC.Url"], Request["m"], Request["s"]));
}
```

logo_gif.aspx

```
public void Page_Load (object sender, System.EventArgs e) {
     Response.Redirect(string.Format("{0}/logo.htm?m={1}&s={2}",
     ConfigurationManager.AppSettings["DC.Url"], Request["m"], Request["s"]));
}
```

Phone-to-Web Order Submissions

When a merchant submits phone orders via the same web page interface as a customer, the data regarding the merchant's device is being sent to Kount, not the customer's device data. This will cause order linking to occur and in time will elevate the score of all orders associated with the persona.

IMPORTANT: Linking will also occur if the browser session ID is used as the transaction session ID and multiple orders are submitted from within the same browser session without closing the browser. To avoid this type of linking, the browser session ID could be incremented appending the UNIX timestamp, choose a different methodology for creating the session ID, or **agents must close the browser between orders to ensure a new session has been created**.

There are two different methods for receiving phone orders.

1. If the customer service agents navigate to **a separate order entry page that does not collect iframe data:** Call Center/Phone Orders will be posted as a Mode=P; hard code the IP address specifically to 10.0.0.1 and provide the phone number within the ANID field (if no phone number is available, pass 0123456789 hard coded).
2. If the customer service agents navigate to **the same page as the customer (iframe data is collected):** Phone-to-Web Order Submission script example is in Appendix A (to exclude merchant owned IP addresses that should not be forwarded to Kount), post Call Center Orders as a Mode=Q; hard code the IP address specifically to 10.0.0.1.

In any of the above circumstances, if the email address is not provided to the agents, the agents will need to input `noemail@kount.com` as the email address in order to prevent linking.

The scripts found in **Appendix A** demonstrate configuration examples of excluding merchant owned IP addresses that should not be forwarded on to the company's server.

Required and Optional fields can be found in **Appendix B**.

Data Collector FAQ

General:

Q: Where do I get the Merchant ID?

A Sandbox Boarding Information document will be sent following the initial kick-off call with the Merchant ID and URLs associated with the DC and RIS processes. A separate document for production will be sent with the production service URLs once the test transaction have been certified.

Q: Why does Kount require a redirect?

The redirect gathers device information without interfering with the order process. The redirect also obfuscates the communication between Kount and the customer.

Q: How do I receive a login to the AWC?

Kount will create the initial administrator user. Once the user has been created an automated e-mail will be sent requesting a password creation.

Q: Should I send production traffic to the test environment to test with?

Production traffic should not be sent to the test environment due to the possibility of skewed scoring from the test orders.

Iframe:

Q: Where should I place the iframe on my website?

If multiple forms of payment methods are available, i.e. Credit Card, PayPal, Bill Me Later, the iframe should be included on the page where the payment method is chosen. If only a single payment method is used, i.e. Credit Card, then the iframe should be included on the page where the customer inputs their credit card information.

IMPORTANT: Place the iframe only on one page. Do not place the image on multiple form pages on your website.

Q: Are there size restrictions on the iframe?

Yes, the iframe must be at least 1x1 pixels in size.

Q: Does it matter where the iframe shows up on the web page?

No, this is left to the merchant to decide.

Q: Why an iframe?

If a user were to source the page there would be no mention of Kount in the source. It also eliminates the possibility of being indexed by various search engines or crawlers.

Q: Why are there two files in the iframe?

The logo.gif file is a fallback in case iframes have been disabled or the browser does not support iframes.

Q: Are both the iframe and image required?

Yes both are required to ensure that a connection is made by the customer device.

Q: Why don't I just use the image, we don't have any iframes anywhere else on our site.

The manner in which iframes are handled by browsers provides greater insight to Kount.

Q: Why do the .htm and .gif files get interpreted as server side code?

By using .htm and .gif file extensions there is less concern from end users that may inspect the source code.

Session Identifier:

Q: What does the session identifier do?

The session identifier is used to join the device data with the order data sent by the RIS process. When the RIS process posts data to Kount it must use the session identifier that was created when the customer initiated the DC HTML.

Q: Does the session identifier need to be unique?

Yes, the session identifier must be unique over a thirty day time period. If there are duplicate session identifiers, device data originating from the DC process may be erroneously connected to RIS order data.

Q: Are there limitations on the session identifier?

Yes, it must be alpha-numeric with a minimum of one character and a maximum of 32 characters long.

Q: What should I use for the session identifier?

The merchant determines, as long as the limitation guidelines are followed. Many merchants use a portion of the application session that is generated when the page is created as the session identifier.

Q: What happens when a user leaves the page after a session identifier has been created then returns to finish the order?

There can be multiple session identifiers created, only the last session will be used to join with the RIS transaction.

Image:

Q: Does it matter where the image is displayed on the page?

It is recommended to display the iframe below the fold of the initial web page but can be located anywhere on the page.

Q: Why does the merchant need to give Kount the path to an image for the redirection?

Kount must have this path to finish the 302 redirect for the image to load. If the path has not been supplied to Kount a broken image icon will be displayed on the merchant page.

Q: Does the image need to be accessible via the Internet?

Yes the image must be publicly available from the Internet.

Q: Why does the path need to be HTTPS?

If the image is not secure, a notification will appear alerting the customer that there are unsecure items on the check out form.

Q: Does it matter what the name of the file is?

If a customer were to inspect the source of the page there should be no indication of interaction from Kount. To alleviate the possibility of a merchant's customer questioning the interaction between Kount and the merchant do not include any reference to Kount in the file name.

Q: Do you have an example of an image.

Yes, it can be provided upon request. Please contact your Technical Account Manager for further details.

Secure
Payments

Troubleshooting the Data Collector

Q: Is the correct Kount URL being used?

Verify that the correct Kount Data Collector URL is being used, test URL or production URL.

Q: Have you provided Kount with the HTTPS URL path to the image?

If the URL path has not been set there will be a broken image displayed on the page.

Q: Is the image available via the Internet?

Test this by pasting the path of the URL in a browser on an external network and verify that the image appears.

Q: Have appropriate DNS entries, NATs, and firewall settings been configured correctly?

Due to the security concerns regarding test environments or production environment the merchant's network operations may need to verify that proper access is available.

Q: Are the logo.htm and logo.gif files being interpreted as server-side code?

If the files are not interpreted as server-side code, when requested the files will serve up the source code instead of performing the redirect. This can be tested by pointing the browser directly to the logo.htm or logo.gif URLs and verify that the static image appears. If source code appears, then the files are not being interpreted correctly. This can also be tested via a UNIX wget command.

Q: Does the redirect contain the correct Merchant ID?

Verify that the redirect Merchant ID is the correct six digit ID supplied by Kount.

Q: Is the Session ID created in the DC process the same session ID being sent with the RIS post?

Ensure that the Session ID being created and stored during the DC process is the correct one being used in the RIS post to Kount and adheres to the session ID requirements.

Q: I enter the Kaptcha URL and my web browser is redirected to the Google Kaptcha page as a 302 redirect error. What went wrong?

The wrong URL was entered. Most likely, the URL https://ssl.kaptcha.com was passed in. /logo.htm should be appended to the end of the URL so it looks like https://ssl.kaptcha.com/logo.htm. Also, the following example https://someplace.verifi.com/logo.html will correct the redirect error, keeping in mind that "someplace.verifi.com" is just sample text.

Q: Only part of the device data is collected, Javascript, Time Zone and other details seem to be missing?

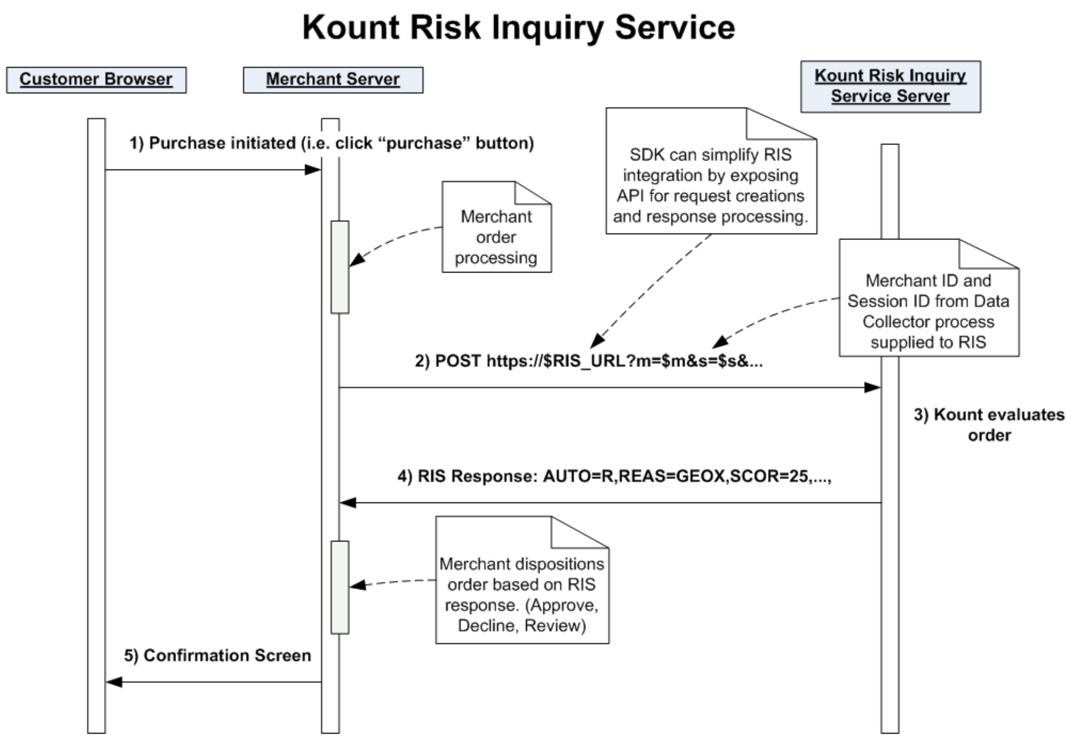
The logo.gif server side script is calling the log.gif instead of the logo.htm. See the "Server Side Code Examples" on page 10 of the Technical Specification.

Q: Why do some of the items within the Extended Variables gadget not display or display as N/A?

A fully qualified path must be used within the scr value of the iFrame.

Risk Inquiry Service (RIS)

The Risk Inquiry Service (RIS) joins device data provided from the data collector process with the customer order data sent from the merchant. Once the device data and the order data are combined, RIS evaluates and scores each transaction. Subsequent to the evaluation RIS returns a response string back to the merchant to be used by the merchant to approve, decline or hold the order for review. Each transaction will continue to be evaluated and dynamically scored for up to fourteen days. The following section describes how to implement the Risk Inquiry Service.


Kount Risk Inquiry Service
Customer Browser
Merchant Server
Kount Risk Inquiry Service Server
1) Purchase initiated (i.e. click "purchase" button)
SDK can simplify RIS integration by exposing API for request creations and response processing.
Merchant order processing
Merchant ID and Session ID from Data Collector process supplied to RIS
2) POST https://$RIS_URL?m=$m&s=$s&...
3) Kount evaluates order
4) RIS Response: AUTO=R,REAS=GEOX,SCOR=25,...,
Merchant dispositions order based on RIS response. (Approve, Decline, Review)
5) Confirmation Screen

The following sequence describes the RIS process:

1. Customer initiates purchase
2. Merchant initiates RIS request to Kount via HTTPS URL encoded post
3. Kount evaluates transaction
4. Kount returns evaluation response to merchant
5. Notification is displayed to customer

Risk Inquiry Service Requirements

RIS data posted to Kount must be URL encoded and submitted as key-value pairs. Much of the work can be simplified by utilizing a Kount provided SDK, including URL encoding. Kount provides a Software Development Kit (SDK) for Java, .NET, PHP, Perl, and Mobile environments.

Recommendations regarding each development environment and their supported versions, configuration, logging, and paths are found in the README file located in the “docs“ directory in each respective SDK, please read the documentation associated with each SDK.

1. Port 443 must be available to post and receive data from Kount.
2. API Keys are used to authenticate the RIS HTTPS submission to Kount, (similar to a password). A single API Key will be used for RIS submissions, the key is not subject to expiration date and does not require re-issuance. To generate an API Key, navigate to the ADMIN tab -> API Keys. See the RIS API Keys section of this document for more detailed instructions. Note: API Keys can only be used with Kount version 0630 and newer, if you are using an older version of Kount, legacy RIS Certificate information can be found in Appendix E of this document.
3. SSL support is required for the RIS process.
4. The session identifier created during the data collector process must be passed as the session identifier for the RIS transaction. This identifier must be unique for at least 30 days. If a single session ID were to be used on multiple transactions, those transactions would link together and erroneously affect the persona information in the risk score.
5. In order to utilize the various SDKs, several required static settings must be configured. Please refer to the README files included in each of the SDKs.
   - PHP settings.ini
   - .NET App.config
   - Perl Company::Ris::Settings
   - Java All settings are included in inquiry

NOTE: There is a 4,000 character limit that makes up an entire RIS post. Should the RIS post exceed the 4k limit, a HTTP standard error will be returned, typically an error code: 413 - Request Entity Too Large.

The table below describes the required static settings found in the SDK:

Data	Size	Description	Example
Merchant_ID	6	Six digit identifier issued by Kount.	999999
COMPANY_SERVER_URL	N/A	HTTPS URL path to the company's servers provided in boarding documentation from Kount	https://risk.text.company.net
LOGGER	N/A	Specifies which logger to use: SIMPLE or NOP.	SIMPLE
SIMPLE_LOG_LEVEL	N/A	If SIMPLE logging is enabled, this lists logging levels in order of decreasing severity:FATAL, ERROR, WARN, INFO, DEBUG	WARN
SIMPLE_LOG_FILE	N/A	Name of the log file for SIMPLE logging	company-sdk-ris.log
SIMPLE_LOG_PATH	N/A	Path to where log file will be written. (Must be a valid path)	/some/path/to/log
APIKEY	Varies	API Key value copied from clipboard - originating from API Key page within Agent Web Console	Alpha/Numeric hashed value provided by Kount
Client Certificate (deprecated field for legacy certificates	N/A	Depending on the SDK environment certain client certificate information will be required.	company-ris-certificate.p12

When to Invoke the Risk Inquiry Service

1. **Pre-Authorization:** Query Kount before attempting an authorization from the payment gateway, below are the considerations regarding pre-authorization
    - Allows the merchant to avoid processing fees on orders set to "decline" by the merchant.
    - All credit card information can be sent to Kount
    - An update RIS call (MODE=U) must be made to update order number and status of payment authorization including AUTH, AVST, AVSZ, and CVVR data provided by payment gateway.
        - For pre-authorization queries set the following required fields as:
            - MACK=Y
            - AUTH=A
        - Once the transaction has been processed, update required and additional fields with MODE=U post (see RIS Service Modes section)
2. **Post-Authorization:** Query Kount after the payment gateway has been contacted, below are the considerations regarding post-authorization

- All payment gateway information can be passed to Kount (Authorization, AVS, CVV) allowing rules to be created regarding AVS and CVV data.
- Some payment gateways do not pass credit card data once they have received it
- Single RIS query, no update is necessary

RIS Data Submission

When submitting data to RIS, it is recommended to send as much data as possible. As part of the RIS post there are required fields and if not populated an error will be returned in the RIS response. Please refer to the SDK documentation with details on how to submit the data for both optional and required fields.

Appendix B lists all fields that are available to post to Kount.

Phone-to-Web Order Submissions

When a merchant submits phone orders via the same web page interface as a customer, the data regarding the merchant's device is being sent to Kount, not the customer's device data. This will cause order linking to occur and in time will elevate the score of all orders associated with the persona.

IMPORTANT: Linking will also occur if the browser session ID is used as the transaction session ID and multiple orders are submitted from within the same browser session without closing the browser. To avoid this type of linking, the browser session ID could be incremented appending the UNIX timestamp, choose a different methodology for creating the session ID, or **agents must close the browser between orders to ensure a new session has been created**.

There are two different methods for receiving phone orders.

1. If the customer service agents navigate to a **separate order entry page that does not collect iframe data:** Call Center/Phone Orders will be posted as a Mode=P; hard code the IP address specifically to 10.0.0.1 and provide the phone number within the ANID field (if no phone number is available, pass 0123456789 hard coded).
2. If the customer service agents navigate to **the same page as the customer (iframe data is collected):** Phone-to-Web Order Submission script example is in Appendix A (to exclude merchant owned IP addresses that should not be forwarded to Kount), post Call Center Orders as a Mode=Q; hard code the IP address specifically to 10.0.0.1.

In any of the above circumstances, if the email address is not provided to the agents, the agents will need to input `noemail@kount.com` as the email address in order to prevent linking.

The scripts found in **Appendix A** demonstrate configuration examples of excluding merchant owned IP addresses that should not be forwarded on to the company's server.

Required and Optional fields can be found in **Appendix B**.

Risk Inquiry Service Modes

Modes are used to specify what type of data is being submitted to Kount.

Mode Q

Initial queries directed from the merchant to Kount that do not originate from a call center environment.

Mode U

Update call to Kount, does not cause a reevaluation of the transaction but will update what is displayed in the Agent Web Console (AWC). This update call does not count towards the number of RIS transactions purchased. Only certain fields can be updated with MODE=U calls.

Name	Description	Character Limit	Valid Values	Required
AUTH	Authorization Status returned to merchant from processor. Acceptable values for the AUTH field are 'A' for Authorized or 'D' for Decline. In orders where AUTH=A will aggregate towards order velocity of the persona while orders where AUTH=D will decrement the velocity of the persona.	1	A or D	No
AVST	Address Verification System Street verification response returned to merchant from processor. Acceptable values are 'M' for match, 'N' for no-match, or 'X' for unsupported or unavailable.	1	M (Match) or N (Not a Match)	No
AVSZ	Address Verification System Zip Code verification response returned to merchant from processor. Acceptable values are 'M' for match, 'N' for no-match, or 'X' for unsupported or unavailable.	1	M (Match) or N (Not a Match)	No
CVVR	Card Verification Value response returned to merchant from processor. Acceptable values are 'M' for match, 'N' for no-match, or 'X' unsupported or unavailable.	1	M (Match) or N (Not a Match)	No
FRMT	Specifies the format of the RIS response if not using SDK, XML, JSON, YAML	4	SDK, XML, JSON, YAML	No
LAST4	Last 4 numbers of Credit Card Value	4	Numeric field, must be 4 numbers	No
MACK	Merchants acknowledgement to ship/process the order. The MACK field must be set as 'Y' if persona data is to be collected to strengthen the score.	1	Y or N	Yes
MERC	Merchant ID assigned to the merchant by Kount.	6	Numeric field, must be 6 numbers	Yes

Name	Description	Character Limit	Valid Values	Required
MODE	Specifies what mode type the RIS post is: MODE=Q MODE=P MODE=X MODE=U	1	Q, P, X, U	Yes
ORDR	Merchant's Order Number	32	Alphanumeric field	No
PENC	RIS parameter for payment encoding.	KHASH	KHASH or MASK	No
PTOK	PTOK value will replace Null value passed in original RIS Post.	32	Value to replace Null value passed in original RIS Post	No
RFCB	Refund/Chargeback status: R = Refund C = Chargeback	1	R (Refund) or C (Chargeback)	No
SESS	Unique Session ID. Must be unique over a 30-day span	32	Alphanumeric field; must mathc original RIS Post	Yes
TRAN	Transaction ID required for update calls to Kount	32	Transaction ID generated by Kount , must match original RIS Post	Yes
VERS	Specifies version of Kount, built into SDK, must be supplied by merchant if not using the SDK	4	Version of Kount being used	Yes

Mode P

Initial queries originating from a call center environment.

- IP Address must be hard coded as "10.0.0.1".
- If customer does not provide an e-mail address, use the value EMAL=noemail@kount.com.
- PayPal is not a valid payment type for MODE=P.

Mode X

Update query made after an initial MODE=Q or P request and/or any MODE=U updates have been made. Updates to certain fields can be made and the transaction will be re-evaluated and return an updated RIS response to the merchant. These updates will be displayed in the AWC. This query will count towards the number of RIS transactions purchased. The same fields listed in the MODE=U section can be changed for MODE=X transactions with the exception of PTYP which is not accepted by MODE=X.

Name	Description	Character Limit	Valid Values	Required
AUTH	Authorization Status returned to merchant from processor. Acceptable values for the AUTH field are 'A' for Authorized or 'D' for Decline. In orders where AUTH=A will aggregate towards order velocity of the persona while orders where AUTH=D will decrement the velocity of the persona.	1	A or D	No
AVST	Address Verification System Street verification response returned to merchant from processor. Acceptable values are 'M' for match, 'N' for no-match, or 'X' for unsupported or unavailable.	1	M (Match) or N (Not a Match)	No
AVSZ	Address Verification System Zip Code verification response returned to merchant from processor. Acceptable values are 'M' for match, 'N' for no-match, or 'X' for unsupported or unavailable.	1	M (Match) or N (Not a Match)	No
CVVR	Card Verification Value response returned to merchant from processor. Acceptable values are 'M' for match, 'N' for no-match, or 'X' unsupported or unavailable.	1	M (Match) or N (Not a Match)	No
FRMT	Specifies the format of the RIS response if not using SDK, XML, JSON, YAML	4	SDK, XML, JSON, YAML	No
LAST4	Last 4 numbers of Credit Card Value	4	Numeric field, must be 4 numbers	No
MACK	Merchants acknowledgement to ship/process the order. The MACK field must be set as 'Y' if persona data is to be collected to strengthen the score.	1	Y or N	Yes
MERC	Merchant ID assigned to the merchant by Kount.	6	Numeric field, must be 6 numbers	Yes

MODE	Specifies what mode type the RIS post is: MODE=Q
	MODE=P
	MODE=X
	MODE=U
		1	Q, P, X, U	Yes

ORDR	Merchant's Order Number	32	Alphanumeric field	No
PENC	RIS parameter for payment encoding.	KHASH	KHASH or MASK	No
PTOK	PTOK value will replace Null value passed in original RIS Post.	32	Value to replace Null value passed in original RIS Post	No

Name	Description	Character Limit	Valid Values	Required
PTYP	Payment Type submitted by merchant: PYPL BLML GDMP GOOG NOTE that PTYP must have been set to actual payment PTYP that will be updated within the initial RIS Post. PTOK must have been set to Null. If the PTYP is set to NONE no update can be performed.	4	PYPL, BLML, GDMP, GOOG	No
RFCB	Refund/Chargeback status: R = Refund C = Chargeback	1	R (Refund) or C (Chargeback)	No
SESS	Unique Session ID. Must be unique over a 30-day span	32	Alphanumeric field; must mathc original RIS Post	Yes
TRAN	Transaction ID required for update calls to Kount	32	Transaction ID generated by Kount , must match original RIS Post	Yes
VERS	Specifies version of Kount, built into SDK, must be supplied by merchant if not using the SDK	4	Version of Kount being used	Yes

NOTE: There are caveats to the PENC, PTOK and PTYP fields. Please contact your Technical Account Manager for further details.

Mode E

This mode designates an error has occurred and is returned to the merchant in the RIS response.

NOTE: If you are a **Payment Processor** using Kount Central, you have access to two additional modes available:

Mode J (for Kount Central merchants only)

This is a simplified RIS call. It only performs Fraud Manager threshold evaluation for the processor's specified customer, and <u>does not include</u> Portfolio Manager Rule evaluation or calculation of a Kount Score.This gives Mode J the advantage of increased performance.

Mode W (for Kount Central merchants only)

This is basically a standard Mode Q with Mode J response appended to the end. Thresholds are evaluated in addition to Portfolio Manager rules. Mode W has the same input requirements as mode Q, with the addition of CUSTOMER_ID. In a Mode W, the threshold decision/response is appended to the Mode Q decision/response. It is the responsibility of the Processor to evaluate both decisions in a Mode W and take appropriate action.

Please see your Merchant Services representative for more information.
NOTE: Appendix C on page 60 lists all warning and error codes.

RIS Payment Types

Kount supports multiple payment types and depending upon the payment type chosen by the customer certain payment tokens are required. If the PayPal Payer ID or Google Checkout Account ID is not sent in the inquiry mode, then it must be sent in the update mode related to the transaction otherwise the order details will not be displayed in the Agent Web Console.

Kount has the ability to add arbitrary payment types rapidly to support an international market. To view the current list of supported payment types, use the API endpoint:

```
https://api.kount.net/rpc/support/payments.html
```

See the **API Specification Guide** for further details.

The content of the endpoint is generated in HTML but changing the extension to `.xml` or `.json` will produce content in those formats.

RIS Payment Encryption Options

When using the Kount SDK all credit card information, by default, uses the KHASH encryption method where the credit card information is irreversibly hashed prior to transmission from the merchant to Kount. When using the JAVA or .Net SDKs (due to the compiled nature of the languages) KHASH is the only option for payment encryption. JAVA or .Net environments must use a direct post (outside of the SDK) method if MASK encryption is chosen.

If not using the SDK the following encryption options are available.

KHASH Kount proprietary hash used to hash the credit card number before passing it to Kount. The hashing algorithm source code can be found in each one of the SDKs or can be requested from Kount.

PTYP=CARD PENC=KHASH

Output - BIN + 14 alpha-numeric characters.
Example - 123456A12C34E56G7DFG

MASK Ability to pass the first six and last four of a credit card filled in with XXXs. PENC=MASK is only valid with PTYP=CARD.

PTYP=CARD PENC=MASK

Output (BIN) + 6 to 9 capital “X” characters + (Last4) of credit card.
Example - 123456XXXXXXXXXX7890

Note: the “X” characters **must** all be capital “X”.

Important: The above example value is just for purposes of illustration. The PTOK should be the same length as the original card number. You can use the card number with the first 6 and last 4 numerals present and the rest of the numbers in the card masked by "Xs" but the number of characters must be the same as those of the actual card number.

Shopping Cart Data

Each RIS request must be submitted with a minimum of one shopping cart item. While other RIS data is entered in as key-value pairs, shopping cart items must be submitted as an array. Each item is an index in the array, and each index must contain the following five attributes.:

Attribute	Type	Size	Description	Example
PROD_TYPE[ ]	String (Plain text no markup or Unicode)	1-255	Shopping cart data array attribute high level or generalized description of the item added to the shopping cart; this value should be free from any markup or Unicode values. This value should be passed as plain text.	TV
PROD_ITEM[ ]	String (Plain text no markup or Unicode)	1-255	Shopping cart data array attribute typically the SKU for an item; this value should be free from any markup or Unicode values. This value should be passed as plain text.m.	SKU-2385-42P
PROD_DESC[ ]	String (Plain text no markup or Unicode)	0-255	Shopping cart data array attribute for a specific description of the item being purchased.	42 Inch Plasma
PROD_QUANT[ ]	Integer	Long	Shopping cart data array attribute signifying the quantity of the item being purchased.	1
PROD_PRICE[ ]	Integer	Long	Shopping cart data array attribute for the price of the single item. Must be a natural number including 0.	75890

User Defined Fields

Kount provides a way for merchants to include additional information related to their business that may not be a standard field in Kount by creating user defined fields. UDFs are created in the Agent Web Console by browsing to the Fraud Control tab and clicking on User Defined Fields. Once you have defined the UDF in the AWC you will be able to pass this data into Kount via an array called UDF as key-value pairs where the label is the key and the data passed in is the value. The maximum number of UDFs that can be created is up to the merchant but response time for evaluating transactions will degrade as more UDFs are added. There are four data types available for user defined fields.

NOTE: UDF labels can be up to 28 characters in length. UDFs cannot begin with a number.

1. Number
2. Alpha-Numeric (can be added to a VIP list)
3. Date
4. Amount

Array [key]=Value	Size	Description	Example
UDF[NUMERIC_LABEL]=value	1-255	Numbers, negative signs, and decimal points.	UDF[FREQUENCY]=107.9
UDF[ALPHA_NUMERIC_LABEL]=value	1-255	Letters, numbers or both.	UDF[COUPON]=BUY11
UDF[DATE_LABEL]=value	1-20	Formatted as *YYYY-MM-DD* or *YYYY-MM-DD HH:MI:SS*	UDF[FIRST_CONTACT] =2012-04-10 17:00:01
UDF[AMOUNT_LABEL]=value	1-255	Integers only, no decimal points, signs or symbols.	UDF[BALANCE]=1100

RIS Response

After a merchant has posted RIS information to Kount, a key-value pair string will be returned back to the merchant. The RIS response format will be the same that was specified in the RIS post, with the default being named pairs. Each data field must be invoked by 'getter' methods on the response object found in the SDK. The merchant can then use the RIS response to automate the order management process by keying off of the **AUTO** field and can utilize any of the additional data returned for internal processing.

An important use of the RIS response is the ability to verify if the Data Collector process was successful and view any warnings or errors that were made during the RIS post from the merchant. The KAPT field is used to determine if the Data Collector process was successful. KAPT=Y means successful, KAPT=N means the process was unsuccessful. All warnings will be displayed in the response and if errors do occur the RIS response will be returned with a MODE=E.

Appendix C lists all warning and error codes.

Kount recommends at the minimum the following methods/functions used in the RIS response:

RIS response Java method examples:

MERC -	getMerchantId()
MODE -	getMode()
TRAN -	getTransactionId()
ORDR -	getOrderNumber()
AUTO -	getAuto()
SCOR -	getScore()
KAPT -	getKaptcha()
SITE -	getSite()
WARNING -	hasWarnings()
|	getWarningCount()
|	getWarnings()
ERROR -	hasErrors()
|	getErrorCount()
|	getErrors()
|	getErrorCode()

Appendix D lists examples of RIS Responses.

Predictive Response - Test environment only

Predictive Response is a mechanism that can be used by Kount merchants to submit test requests and receive back predictable RIS responses. This means that a merchant, in order to test RIS, can generate a particular request that is designed to provide one or more specific RIS responses and/or errors. The predictive response inquiries are not actual RIS inquiries, which means the data will never be submitted to the database and will not be displayed in the Agent Web Console.

The primary reason for having Predictive Response functionality is to diagnose error responses being received from RIS. For instance, a merchant may receive a large number of different error codes after submitting a RIS request. Most of these errors can be reliably reproduced by passing malformed, missing, or additional data in the RIS request. However, some of the errors are extremely difficult or even impossible to reproduce through simple means. There is no way to re-create these errors in a systematic or predictable fashion using RIS request input, rules, and/or Data Collector.

In addition, the merchant may wish to invoke a specific response in order to understand how their respective OMS will manage certain responses or data returned in the response. Predictive Response allows merchants to submit requests designed to return exact responses including errors. An example would be if a merchant wanted to submit a RIS request that would return the very specific responses **SCOR=71**, **AUTO=E**, and **GEOX=CA**.

Predictive Responses are created using the UDF (User Defined Fields) override option. These User Defined Fields do not need to be created through the Agent Web Console, they can be simply passed in as additional fields in the Predictive Response RIS inquiry.

In order to create a Predictive Response RIS Inquiry, the request must contain a specific email parameter in the EMAL field: `predictive@kount.com`.

All other elements of the RIS request you submit must be valid elements and contain the minimum set of required RIS keys.

Below is the hard coded default reply:

```
‘TRAN’ => ‘6V100HV36D98’,
‘AUTO’ => ‘A’,
‘SCOR’ => 50,
‘GEOX’ => ‘US’,
‘BRND’ => ‘VISA’,
‘REGN’ => ‘ID’,
‘NETW’ => ‘A’,
‘CARDS’ => 2,
‘DEVICES’ => 1,
‘EMAILS’ => 3,
‘VELO’ => 4,
‘VMAX’ => 4,
‘SITE’ => ‘DEFAULT’,
‘DEVICE_LAYERS’ => ‘D67BC18BAD.6EF0902E51.8C96FA9E7B.61FD602D96.940A6D1454’,
‘FINGERPRINT’ => ‘00482B9BED15A272730FCB590FFEBDDD’,
‘TIMEZONE’ => 420,
‘REGION’=> ‘ID’,
‘COUNTRY’=> ‘US’,
‘PROXY’ => ‘N’,
‘JAVASCRIPT’ => ‘Y’,
‘FLASH’ => ‘Y’,
‘COOKIES’ => ‘Y’,
‘HTTP_COUNTRY’ => ‘US’,
‘LANGUAGE’ => ‘EN’,
‘MOBILE_DEVICE’ => ‘N’,
‘MOBILE_TYPE’ => ‘’,
‘MOBILE_FORWARDER’ => ‘N’,
‘VOICE_DEVICE’ => ‘N’,
‘PC_REMOTE’ => ‘N’,
‘REASON_CODE’=> ‘’,
‘DDFS’ => ‘2013-07-19’,
‘DSR’ => ‘1080x1920’,
‘UAS’ => ‘Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.31 (KHTML, like Gecko)
 Chrome/26.0.1410.63 Safari/537.31’,
‘BROWSER’ => ‘Chrome 26.0.1410.63’,
‘OS’ => ‘Linux x86_64’,
‘PIP_IPAD’ => ‘’,
‘PIP_LAT’ => ‘’,
‘PIP_LON’ => ‘’,
‘PIP_COUNTRY’ => ‘’,
‘PIP_REGION’ => ‘’,
‘PIP_CITY’ => ‘’,
‘PIP_ORG’ => ‘’,
‘IP_IPAD’ => ‘10.0.0.1’,
‘IP_LAT’ => ‘43.6091’,
‘IP_LON’ => ‘-116.2097’,
‘IP_COUNTRY’ => ‘US’,
‘IP_REGION’ => ‘Idaho’,
‘IP_CITY’ => ‘Boise’,
‘IP_ORG’ => ‘Company Inc.’
```

To pass in a request that will result in a specific response, use one or more UDF overrides to trigger a mock RIS response.

The basic syntax is: **UDF[~K!_label]="foo"**

~K!_ is the prefix, **label** is the desired field for which you want a response, such as **SCOR** or **ERRO**, and after the equal sign (=), enter the specific value you want returned. The **~K!_** prefix is required to trigger the **UDF** to become a predictive response field.

Example 1:

You want to send in a request that will result in a Kount Score of **18**, an Auto Decision of **E**, and a **601 System Error** code.

Request:

UDF[~K!_SCOR]=18

UDF[~K!_AUTO]=E

UDF[~K!_ERRO]=601

Response:

SCOR=18

AUTO=E

ERRO=601

Example 2:

You want to pass in a request that will result in a Kount Score of **42**, an Auto Decision of **Decline** and a **GEOX** of **Nigeria**.

Request:

UDF[~K!_SCOR]=42

UDF[~K!_AUTO]=D

UDF[~K!_GEOX]=NG

Response:

SCOR=42

AUTO=D

GEOX=NG

You can use **UDF** overrides to pass in an unlimited number of mock requests but all of the fields you pass in that are not overrides must be valid. In the response, all of the other elements, besides the **UDF** overrides will be the default values, including **MODE** and **MERC**.

Kount Event Notification System (ENS) and Kount Application Programming Interface (API)

Kount allows a great deal of control over fraud management through a variety of methods including the risk score, rules to determine risk thresholds, VIP lists, and many others that are administered through the Agent Web Console or AWC. Kount also provides the ability for interaction for some functions without the need of the Agent Web Console by utilizing the Event Notification System (ENS) and the Application Programming Interface (API).

Specification guides for the ENS and API can be requested from your account manager or access the API endpoints list by pointing your browser to https://api.kount.net/rpc/list.html.

NOTE: A separate API certificate is required.

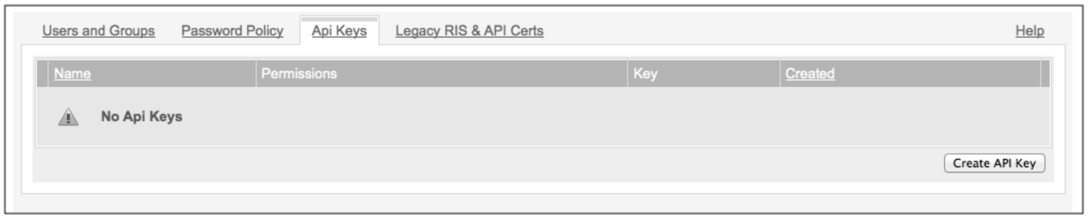
RIS API Keys

To preserve the security of RIS data, merchants must authenticate to Kount using an API Key when submitting RIS requests. API Keys are created within the Agent Web Console. Additional coding examples can be found in the SDK Guide.

API Keys are used to track and control permissions to Kount **APIs** and **RIS**. Rather than creating separate certificates for APIs and RIS, a single API Key can be used to manage both.

NOTE: For **Kount Central** customers, an API Key can manage APIs, RIS, and Kount Central permissions.

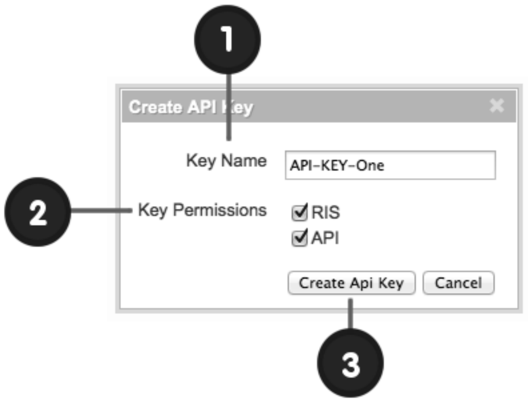
To open the API Key's page, click **Admin** in the main menu and then click **API Keys**.



Name	Permissions	Key	Created
No Api Keys			

By default, no API Keys exist when the API Keys page is opened for the first time. To create a key, click the **Create API Key** button at the lower right of the table.

The **Create API Key** dialog box appears.

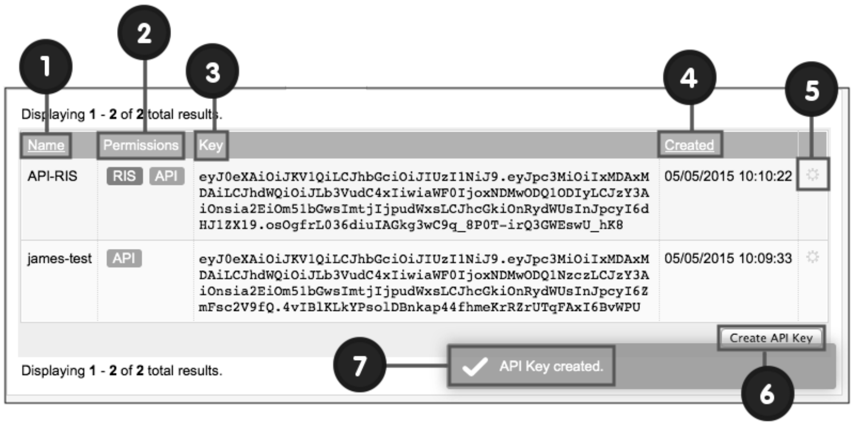


Key Name: API-KEY-One
Key Permissions: ☑ RIS ☑ API

1. **Key Name:** Give the new key a name (the Api-Key is case sensitive).
2. **Key Permissions:** Select RIS, API, or both.

3. **Create API Key:** Click this button to create the key.

NOTE: If the merchant is a **Kount Central** customer, a third check box for Kount Central will be present under **Key Permissions** and available to be selected.



Displaying **1** - **2** of **2** total results.

Name	Permissions	Key	Created	
API-RIS	RIS API	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiIxMDAxMDAiLCJhdWQiOiJLb3VudC4xIiwiaWF0IjoxNDMwODQ1ODIyLCJzY3AiOnsia2EiOm51bGwsImtjIjpudWxsLCJhcGkiOnRydWUsInJpcyI6dHJ1ZX19.osOgfrL036diuIAGkg3wC9q_8P0T-irQ3GWEswU_hK8	05/05/2015 10:10:22	
james-test	API	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiIxMDAxMDAiLCJhdWQiOiJLb3VudC4xIiwiaWF0IjoxNDMwODQ1NzczLCJzY3AiOnsia2EiOm51bGwsImtjIjpudWxsLCJhcGkiOnRydWUsInJpcyI6ZmFsc2V9fQ.4vIBlKLkYPsolDBnkap44fhmeKrRZrUTqFAxI6BvWPU	05/05/2015 10:09:33	

Displaying **1** - **2** of **2** total results.

Once the new key is created, it appears on the **API Keys** page.

1. **Name:** The name of the newly created key appears here.
2. **Permissions:** The permissions given to the key during creation appear here.
3. **Key:** The key value appears here. Copy the key value from this field.
4. **Created:** The date and time the key was created appears here.
5. **Revoke:** Click the "gears" and then click **Revoke** to revoke the key.
6. **Create API Key:** Click this button to create a new key. You can create as many keys as you want.
7. **Success Message:** Like any other success message, click on this green box to close it.

NOTE: Integration languages such as Classic ASP and Cold Fusion do not support the use of API Keys, please refer to Appendix E for information regarding the legacy Certificates.

PHP settings.initial

```
$ch = curl_init();
curl_setopt($ch, CURLOPT_HTTPHEADER,
array("X-Kount-Api-Key: {$this->apiKey}"));
```

.NET App.config

```
HttpWebRequest webReq = (HttpWebRequest)WebRequest.Create(this.url);
webReq.Headers["X-Kount-Api-Key"] = this.apiKey;
```

Java

```
URL url = new URL(this.risServerUrl);
HttpsURLConnection urlConn = (HttpsURLConnection) url.openConnection();
urlConn.setRequestProperty("X-Kount-Api-Key", getApiKeyData());
```

FAQ Risk Inquiry Service

General

Q: Do RIS requests need to be initiated from the same server as the Data Collector process?
No, they can be autonomous, but the RIS session ID must be the same session ID sent from the DC process for that transaction.

Q: How soon does the RIS request need to be sent after the Data Collector?
There is no set time though the sooner the better.

Q: How do I code for the **PayPal** payment type?
There are a couple of options available.

1. On the initial RIS inquiry call (MODE=Q), specify the payment type as PayPal (PTYP=PYPL). With this method of coding for PayPal the order details will not be displayed in the Agent Web Console until after the PayPal Payer ID has been returned to Kount in the subsequent RIS update call (MODE=U).
2. On the initial RIS inquiry call (MODE=Q), specify the payment type as NONE. This method will allow the order details to be displayed in the Agent Web Console. Once the PayPal Payer ID is returned send an update call to Kount, (MODE=U) and update the PTYP, and PTOK fields.

Q: Why am I not seeing any log files in the path I specified?
Check your configuration file and make sure the logger is set to the desired logging level.

How to Populate the EPTOK

EPTOK is a 1024 character RSA encrypted credit card using a public-key provided by Kount. It is recommended that you use the SDK to encrypt credit card numbers if you intend to send them as raw data. However, you can also send the information directly without using the SDK.

If you choose to send the information directly, encrypt the credit card number using Kount's RSA public key using padding PKES#1.

To request our public key, contact your Merchant Services representative.

Troubleshooting Risk Inquiry Service

General

Is the correct URL being used?

- Verify that the correct RIS URL is being used, test URL or production URL.
- Can the URL be resolved?
- Have the RIS responses been checked for errors or warnings?

Appendix A

NOTE: For backward compatibility, the *Device Data Collector* feature of Kount is referenced in the commented code and code samples in this appendix as *Kaptcha*. Any time you see a mention of *Kaptcha* below, it is a reference to the *Device Data Collector*.

PHP example of the Phone-to-Web logo.htm script -

```
<?php
/**
* Example analyzer re-direct script.
*
*
* <p>Expects to be called with 2 GET mode query parameters:
* <dl>
* <dt>m</dt>
* <dd>Company Merchant ID</dd>
* <dt>s</dt>
* <dd>Unique customer session ID</dd>
* </dl>
*
*/
// -- BEGIN CONFIGURATION --
/**
* Hostname of Company endpoint.
* MUST BE SET BY MERCHANT BEFORE USE
*
* @var string
*/
$COMPANY_SERVER = null;
/**
*
* List of ip addresses in dotted quad format (eg "127.0.0.1") that should
* not be redirected to the Company.  These IP addresses are the public facing IP
* addresses that have been assigned by the merchant service provider.
*
     * @var array
*/
$EXCLUDED_IPS = array();
// -- END CONFIGURATION --
function send_empty_page () {
     echo '<html><head></head><body></body></html>';
}
// validate configuration
```

```
if (!isset($COMPANY_SERVER)) {
     error_log("COMPANY_SERVER must be defined in " . __FILE__);
     send_empty_page();
     exit();
}
if (!isset($EXCLUDED_IPS) || !is_array($EXCLUDED_IPS)) {
     error_log("EXCLUDED_IPS must be defined in " . __FILE__);
     send_empty_page();
     exit();
}
// validate input
$MERC = rawurlencode($_GET['m']);
$SESS = rawurlencode($_GET['s']);
// process request
$remoteIP = $_SERVER['REMOTE_ADDR'];
if (false !== array_search($remoteIP, $EXCLUDED_IPS)) {
     // current visitor is in the exclude list
     send_empty_page();
} else {
// Redirect the browser
     header("HTTP/1.1 302 Found");
     header("Location: https://{$COMPANY_SERVER}/logo.htm?m={$MERC}&s={$SESS}");
}
```

C# example of the Phone-to-Web logo.htm script -

```
C#

using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

namespace DataCollectorExample
{
    /// <summary>
    /// DataCollector redirect code sample.
    ///
    /// In addition to the basic redirect functionality required by the
    /// specification this class allows configuration of an "exclude list"
/// of IP addresses that should NOT be forwarded on to the Company’s Data Collector
/// server. This non-typical use case can arise when a Merchant is doing
/// phone-to-web orders in a call center and needs to keep Kaptcha data
/// from those call center orders from reaching the Company.
///
/// 2011 Company, Inc. All Rights Reserved.
/// </summary>
public partial class _Default : System.Web.UI.Page
{
    /// <summary>
    /// Data Collector URL provided by the Company. Check with your
    /// representative for the correct value.
    /// </summary>
    protected const string KaptchaUrl = "https://tst.kaptcha.com";

    /// <summary>
    /// Your merchant ID goes here. Check with your representative
    /// for the correct value.
    /// </summary>
    protected const string MerchantId = "999999";

    /// <summary>
    /// A list of excluded IP addresses. This should be populated with
    /// a list of the IP addresses to be excluded.
    /// </summary>
    protected IList ExcludedIps = new ArrayList();

    /// <summary>
    /// The code to be executed on page load.
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    protected void Page_Load(object sender, EventArgs e)
    {
        this.LoadExcludedIps();
        string ClientIp = Request.UserHostAddress;
        if (this.ExcludedIps.Contains(ClientIp))
        {
            // The client is on the exclusion list. Might be a good
            // idea to log this here.
        }
        else
        {
            Response.Redirect(this.KaptchaUrl + "/logo.htm?m=" + MerchantId +
                "&s=" + HttpContext.Current.Session.SessionID);
        }
        }

        /// <summary>
        /// Load up the IP addresses to be excluded here.
        /// </summary>
        protected void LoadExcludedIps()
        {
            // Add the list of IP addresses here...
            // Ideally these should not be hard coded but read from a
            // config file. For brevity and clarity we'll just do the
            // following. Same with the hard coded merchant ID and
            // Data Collector endpoint URL.
            ExcludedIps.Add("127.0.0.1");
            ExcludedIps.Add("10.0.0.1");
            // etc...
        }
    }
}
```

Java example of the Phone-to-Web logo.htm script -

```
package com.company.example;

import java.io.IOException;
import java.io.PrintWriter;
import java.net.URLEncoder;
import java.util.Arrays;
import java.util.List;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.UnavailableException;

/**
 * Example Data Collector redirect script.
 *
 * <p>In addition to the basic redirect functionality required by the
 * specification this servlet allows configuration of an "exclude list" of ip
 * addresses that should NOT be forwarded on to the Data Collector server. This
 * non-typical use case can arise when a Merchant is doing phone-to-web orders
 * in a call center and needs to keep Data Collector data from those call center
 * orders from reaching the Company.
 *
 * <p>The servlet uses ServletConfig provided configuration parameters to
 * specify these settings:
 * <dl>
 *   <dt>MERC</dt>
 *   <dd>REQUIRED</dd>
 *   <dd>Merchant ID assigned by the Company</dd>
 *
 *   <dt>KAPTCHA_URL</dt>
 *   <dd>REQUIRED</dd>
 *   <dd>URL to the Data Collector endpoint</dd>
 *   <dd>The proper values for testing and production will be provided to you
 *   during the boarding process by your Boarding Manager.</dd>
 *
 *   <dt>EXCLUDE_IPS</dt>
 *   <dd>OPTIONAL</dd>
 *   <dd>Comma separated list of IP addresses in dotted-quad notation to
 *   exclude from the Katpcha redirect.</dd>
 * </dl>
 *
 * <p>See your servlet container documentation for the proper file and format to
 * provide ServletConfig parameters at deploy time.
 *
 * @copyright 2011 Company Inc.
 */
public class LogoHtmServlet extends HttpServlet {

  /**
   * Empty html response body.
   */
  static final String EMPTY_HTML = "<html><head></head><body></body></html>";

  /**
   * Merchant ID assigned by the Company.
   * Set via the MERC parameter in the ServletConfig.
   */
  protected String merc;

  /**
   * URL to the Data Collector endpoint.
   * Set via the KAPTCHA_URL parameter in the ServletConfig.
   */
  protected String kaptchaUrl;

  /**
   * List of ip addresses in dotted quad format (eg "127.0.0.1") that should
   * not be redirected to the real Data Collector server.
   * Set via the EXCLUDE_IPS parameter in the ServletConfig. EXCLUDE_IPS
 * should be a comma separated list of ip addresses to exclude (eg
 * "127.0.0.1,127.0.0.2,127.0.0.3")
 */
protected List<String> excludedIps;


/**
 * Initialize the servlet.
 *
 * @param config the ServletConfig object that contains configutation
 * information for this servlet
 * @throws ServletException if an exception occurs that interrupts the
 * servlet's normal operation
 * @throws UnavailableException if required configuration parameters are not
 * supplied
 */
public void init (ServletConfig config) throws ServletException {
  super.init(config);

  this.merc = config.getInitParameter("MERC");
  if (null == this.merc) {
    throw new UnavailableException(
        "Missing required servlet config parameter MERC");
  }

  this.kaptchaUrl = config.getInitParameter("KAPTCHA_URL");
  if (null == this.merc) {
    throw new UnavailableException(
        "Missing required servlet config parameter KAPTCHA_URL");
  }

  final String exclude = config.getInitParameter("EXCLUDE_IPS");
  if (null != exclude) {
    this.excludedIps = Arrays.asList(exclude.split(","));
  }
} //end init


/**
 * Process an HTTP GET request.
 *
 * @param req HttpServletRequest that encapsulates the request to the
 * servlet
 * @param resp HttpServletResponse that encapsulates the response from the
 * servlet
 * @throws IOException if detected when handling the request
```

```
   * @throws ServletException if the request could not be handled
   */
  public void doGet (HttpServletRequest request, HttpServletResponse response)
      throws ServletException, IOException {

    final String remoteIP = request.getRemoteAddr();
    if (null != this.excludedIps &&
        this.excludedIps.contains(remoteIP)) {
      // current visitor is in the exclude list
      this.getServletContext().log(
          "Excluding " + remoteIP + " from Katptcha processing");

      // return an empty html response instead of redirecting
      response.setStatus(HttpServletResponse.SC_OK);
      response.setContentType("text/html");
      final PrintWriter writer = response.getWriter();
      writer.print(EMPTY_HTML);
      writer.flush();

    } else {
      // normal processing path.
      // get the id of the current session
      final String sess = request.getSession().getId();

      // construct the full URL to the Data Collector server
      final String url = this.kaptchaUrl +
          "?m=" +  URLEncoder.encode(this.merc, "UTF-8") +
          "&s=" + URLEncoder.encode(sess, "UTF-8");

      // Redirect the browser to Data Collector
      response.setStatus(HttpServletResponse.SC_MOVED_TEMPORARILY);
      response.sendRedirect(url);
    }
  } //end doGet

} //end LogoHtmServlet
```

Appendix B

RIS Required Inquiry Keys

RIS Required Inquiry Key	Description	Max Field Length	Source	Required
ANID	Automatic Number Identification (ANI) submitted with order. If the ANI cannot be determined, merchant must pass 0123456789 as the ANID. This field is only valid for MODE=P RIS submissions.	32	Merchant	MODE=P

AUTH	Authorization Status returned to merchant from processor. Acceptable values for the AUTH field are 'A' for Authorized or 'D' for Decline.
	In orders where AUTH=A will aggregate towards order velocity of the persona while orders where AUTH=D will decrement the velocity of the persona.
		1	Merchant	MODE=Q
				MODE=P

CURR	Country of currency submitted on order	3	Merchant	MODE=Q
				MODE=P

EMAL	This is the email address submitted by the customer. If a call center is accepting orders on behalf of customers and the customer does not provide an email address OR the customer does not have an email address, **noemail@kount.com** must be submitted. Kount currently supports 2 bit character sets. Unicode character sets are not supported at this time.	64	Merchant	MODE=Q

IPAD	Dotted Decimal IPv4 address that the merchant sees coming from the customer. If MODE=P or the Phone to Web exclusion is used, the IPAD field should be hard coded to be 10.0.0.1. Other than MODE=P or Phone to Web, the IPAD field should never be an anonymous IP address (i.e. 10.X.X.X or 192.168.X.X). IPv6 Addresses are not supported at this time. Please use the IPv4 dual stack equivalent or pass 10.0.0.1 in the IPAD field if an IPv4 address is not available.	16	Merchant	MODE=Q
				MODE=P

RIS Required Inquiry Key	Description	Max Field Length	Source	Required

MACK	Merchants acknowledgement to ship/process the order. The MACK field must be set as 'Y' if persona data is to be collected to strengthen the score.	1	Merchant	MODE=Q
				MODE=P

MERC	Merchant ID assigned to the merchant by Kount.	6	Merchant	MODE=Q
				MODE=P
				MODE=X
				MODE=U

MODE	Specifies what mode type the RIS post is:
	MODE=Q
	MODE=P
	MODE=X
	MODE=U
		1	Merchant	MODE=Q
				MODE=P
				MODE=X
				MODE=U

PROD_DESC	Shopping cart data array attribute for a specific description of the item being purchased. **NOTE:** Product Descriptions should not contain any markup or unicode values. Non-alpha numeric characters will increment the character count exponentially depending on the special character used. RIS Post Limits: There is a 4,000 character limit that makes up an entire RIS post. Should the RIS post exceed the 4k limit, an HTTP standard error will be returned, typically an error code: 413 - Request Entity Too Large	256	Merchant	MODE=Q
				MODE=P

PROD_ITEM	Shopping cart data array attribute typically the SKU for an item; this value should be free from any markup or Unicode values. This value should be passed as plain text.	256	Merchant	MODE=Q
				MODE=P

PROD_PRICE	Shopping cart data array attribute for the price of the single item. Must be a natural number including 0.	no max	Merchant	MODE=Q
				MODE=P

PROD_QUANT	Shopping cart data array attribute signifying the quantity of the item being purchased.	no max	Merchant	MODE=Q
				MODE=P

PROD_TYPE	Shopping cart data array attribute high level or generalized description of the item added to the shopping cart; this value should be free from any markup or Unicode values. This value should be passed as plain text.	256	Merchant	MODE=Q
				MODE=P

RIS Required Inquiry Key	Description	Max Field Length	Source	Required

PTOK	Payment token submitted by merchant for order (credit card, payer ID, routing/transit, MICR, and account number). When using KHASH the BIN is extracted from the PTOK value, Last4 may be passed independently - it is lost during KHASH. If PTYP is set to None then the PTOK value should be left empty (NULL). If the credit card information is not available and a tokenized value is returned from the payment processor set PTYP=GIFT and send the token returned from processor in the PTOK field.	32	Merchant	MODE=Q
				MODE=P

PTYP	Payment Type submitted by merchant:
	APAY - Apple Pay
	CARD - Credit Card
	PYPL - PayPal
	CHEK - Check
	NONE - None
	GDMP - Green Dot Money Pack
	GOOG - Google Checkout
	BLML - Bill Me Later
	GIFT - Gift Card
	BPAY - BPAY
	NETELLER - Neteller
	GIROPAY - GiroPay
	ELV - ELV
	MERCADE_PAGO - Mercade Pago
	SEPA - Single Euro Payments Area
	INTERAC - Interac
	CARTE_BLEUE - Carte Bleue
	POLI - POLi
	Skrill/Moneybookers - SKRILL
	SOFORT - Sofort
	If the credit card information is not available and a tokenized value is returned from the payment processor set PTYP=GIFT and send the token returned from processor in the PTOK field.
		4	Merchant	MODE=Q
				MODE=P

SESS	Unique session ID	32	Merchant	MODE=Q
				MODE=P
				MODE=X
				MODE=U

SITE	Website identifier of where order originated.	8	Merchant	MODE=Q
				MODE=P

RIS Required Inquiry Key	Description	Max Field Length	Source	Required

TOTL	Total amount in currency submitted in lowest currency factor. e.g. USD = pennies ($1.00 = 100). TOTL must be a natural number including 0.	15	Merchant	MODE=Q
				MODE=P

TRAN	Transaction ID required for update calls to Kount. MODE=U and MODE=X		Kount	MODE=U
				MODE=X

VERS	Specifies version of Kount, built into SDK, must be supplied by merchant if not using the SDK	4	Kount - Merchant	MODE=Q
				MODE=P
				MODE=X
				MODE=U

Kount Central Merchants Only (Payment Processors)

RIS Required Inquiry Key	Description	Max Field Length	Source	Required
CUSTOMER_ID	A unique ID assigned to a Gateway Customer when the customer is created by a Payment Processor	32 (string)	Merchant	MODE=J MODE=W

RIS Optional Keys

RIS Optional Keys	Description	Max Field Length	Source
ANID	Automatic Number Identification (ANI) submitted with order. If the ANI cannot be determined, merchant must pass 0123456789 as the ANID. This field is optional for MODE=Q RIS submissions.	32	Merchant
AVST	Address Verification System Street verification response returned to merchant from processor. Acceptable values are 'M' for match, 'N' for no-match, or 'X' for unsupported or unavailable.	1	Merchant
AVSZ	Address Verification System Zip Code verification response returned to merchant from processor. Acceptable values are 'M' for match, 'N' for no-match, or 'X' for unsupported or unavailable.	1	Merchant
B2A1	Billing street address - Line 1	256	Merchant
B2A2	Billing street address - Line 2	256	Merchant
B2CC	Billing address - Country	2	Merchant
B2CI	Billing address - City	256	Merchant
B2PC	Billing address - Postal Code	20	Merchant
B2PN	Bill-to Phone Number	32	Merchant
B2ST	Billing address - State/Province	256	Merchant
BPREMISE	Bill-to premise address for UK (Required for 192.com)	256	Merchant
BSTREET	Bill-to street address for UK (Required for 192.com)	256	Merchant
CASH	Total fenceable value of goods sold.	15	Merchant
CVVR	Card Verification Value response returned to merchant from processor. Acceptable values are 'M' for match, 'N' for no-match, or 'X' unsupported or unavailable.	1	Merchant
DOB	Date of Birth | Format YYYY-MM-DD	YYYY-MM-DD	Merchant

RIS Optional Keys	Description	Max Field Length	Source
EPOC	This is a timestamp and is expressed as a number such as 1422377956. The timestamp value represents the number of seconds elapsed since midnight 01/01/1970. Reference: http://www.epochconverter.com/.	10	Merchant
FRMT	Specifies the format of the RIS response if not using SDK, XML, JSON, YAML	4	Merchant
GENDER	M or F	1	Merchant
LAST4	Last 4 numbers of Credit Card Value	4	Merchant
NAME	Name submitted with the order. Kount currently supports 2 bit character sets. Unicode character sets are not supported at this time.	64	Merchant.
ORDR	Merchant's Order Number	32	Merchant
S2A1	Shipping street address - Line 1	256	Merchant
S2A2	Shipping street address - Line 2	256	Merchant
S2CC	Shipping address - Country	2	Merchant
S2CI	Shipping address - City	256	Merchant
S2EM	Shipping address - Email address of recipient	64	Merchant
S2NM	Shipping address - Name of recipient	64	Merchant
S2PC	Shipping address - Postal Code	20	Merchant
S2PN	Ship-to Phone Number	32	Merchant
S2ST	Shipping address - State/Province	256	Merchant
SHTP	Shipping type. The following nomenclature is expected for shipping types to be passed to Kount. SHTP: Same Day = SD Next Day = ND Second Day = 2D Standard = ST	2	Merchant
SPREMISE	Ship-to premise address for UK (Required for 192.com)	256	Merchant
SSTREET	Ship-to street address for UK (Required for 192.com)	256	Merchant
UNIQ	Merchant assigned account number for consumer	32	Merchant
UAGT	Customer User-Agent HTTP header	1024	Merchant

RIS Response Keys

RIS Response Keys	Description	Char Limit	Example Data	Source
AUTO Displayed as **STAT** in AWC	Auto-decision response code: • A - Approve • D - Decline • R - Review • E - Escalate	1	R	Kount
BROWSER	Web browser	64	Chrome 26.0.1410.63	Kount
BRND	Brand of credit card used when payment type is 'credit card'	4	DISC	Kount
CARDS	Total number of credit cards associated to persona as seen by Kount	Number	39	Kount
COOKIES	A flag to indicate if the device placing order has 'cookies' enabled or not	1	Y/N	Kount
COUNTERS_TRIGGERED	Number of unique counter names triggered during rules evaluation	Number	1	Kount
COUNTER_NAME_X	Name of counter triggered	64	MYCOUNTER	Kount
COUNTER_VALUE_X	Sum of the number of times a counter was triggered	Number	1	Kount
COUNTRY	Two character ISO country code associated with the physical device	2	US	Kount
DDFS	Date device first seen	10	2013-08-28	Kount
DEVICE_LAYERS	5 device layers representing the operating system, browser, javascript settings, cookie setting and flash settings. Device layers are used to create the device fingerprint.	55	.1A4B4CF8CF .963B6935DF .61FD602D96 .B602984541 (one continuous string with no spaces)	Kount
DEVICES	Total number of unique devices associated to persona as seen by Kount	Devices	1	Kount
DSR	Device screen resolution	10	1080x1920	Kount

RIS Response Keys	Description	Char Limit	Example Data	Source
EMAILS	Total number of unique email addresses associated to persona as seen by Kount	Number	1	Kount
ERROR_COUNT	Number of errors merchant RIS post created	N/A	N/A	Kount
ERROR_N	Error code displayed in RIS response	N/A	N/A	Kount
FINGERPRINT	The unique fingerprint of the device placing the order	32	1679091C5A880FAF6FB5E 6087EB1B2DC (one continuous string with no spaces)	Kount
FLASH	A flag to indicate if the device placing order has 'flash' enabled or not	1	Y/N	Kount
GEOX	Persona related country with highest probability of fraud	2	FJ	Kount
HTTP_COUNTRY	User Home country the device owner has set in the device's Control Panel	2	US	Kount
IP_IPAD	IP address of proxy • IP_COUNTRY - Country of proxy IP address (2, US) • IP_LAT - Latitude of proxy IP address (Number, -90.1922) • IP_LON - Longitude of proxy IP address (Number, 38.6312) • IP_CITY - City of proxy IP address (255, Houston) • IP_REGION - State/Region of proxy IP address (255 character limit) • IP_ORG - Owner of IP address or address block (64, Organization Name)	16	214.43.99.120 (See bullet points in Description column for further data on Char Limit and Example)	Kount

RIS Response Keys	Description	Char Limit	Example Data	Source
JAVASCRIPT	A flag to indicate if the device placing order has 'javascript' enabled or not	1	Y/N	Kount
KAPT	Whether or not device data was collected by the Data Collector process	1	Y/N	Kount
KYCF	Know Your Customer Flag	N/A	N/A	Kount
LANGUAGE	The language the device owner has set in the device's Control Panel	2	EN	Kount
LOCALTIME	The local time the device owner has set in the device's Control Panel	20	2015-02-12 13:41	Kount
MERC	Kount Merchant ID	Number	0555	Merchant
MOBILE_DEVICE	Is the device placing the order of a mobile nature (iPhone; Android; Blackberry; iPad, etc.)	1	Y/N	Kount
MOBILE_FORWARDER	If device is mobile, is it using a forwarder to process the carrier's service	1	Y/N	Kount
MOBILE_TYPE	iPhone; Android; Blackberry; iPad, etc.	32	Blackberry	Kount
MODE	Specifies what mode type the RIS post is, Q, P, X, U, E	1	Q	Merchant
NETW	Riskiest network type associated with persona within the last 14 days • A - Anonymous • H - High School • L - Library • N - Normal • O - Open Proxy • P - Prison • S - Satellite	1	H	Kount
ORDER	Merchant's Order Number	32	b53928bc51	Merchant
OS	Operating System	64	Linux x86_64	Kount

RIS Response Keys	Description	Char Limit	Example Data	Source
PC_REMOTE	Is the device enabled to use PC Remote software	1	Y/N	Kount
PIP_IPAD	Pierced IP address • PIP_COUNTRY - Country of pierced IP address (2, US) • PIP_LAT - Latitude of pierced IP address (Number, -90.1922) • PIP_LON - Longitude of pierced IP address (Number, 38.6312) • PIP_CITY - City of pierced IP address (255, Houston) • PIP_REGION - State/Region of pierced IP address (255 character limit) • PIP_ORG - Owner of pierced IP address or address block (64, Organization Name)	16	214.43.99.120 (See bullet points in Description column for further data on Char Limit and Example)	Kount
PROXY	Was a proxy server detected with this order	1	Y/N	Kount
REASON_CODE	Custom Reason Code associated with Rule Action	16	Will display as NONE if no Reason_Code is defined	Kount
REGN	Region associated to Device Location	2	TX	Kount
REGION	Region associated to GEOX Location	2	TX	Kount
RULES_TRIGGERED	Number of rules triggered by the RIS post	Number	1	Kount
RULE_DESCRIPTION_X	Rule descriptions associated with RULE_ID_X	255	Rule description	Kount
RULE_ID_X	Rule ID associated with merchant created rules	Number	1211986	Kount
SCOR	Kount score	Number	99	Kount

RIS Response Keys	Description	Char Limit	Example Data	Source
SESS	Unique session ID	32	faa6370074b53928bc51ef 913441e0cd (one continuous string with no spaces)	Merchant
SITE	Website identifier of where order originated	8	DEFAULT	Merchant
STAT	Status returned as **AUTO** by Kount: • A - Approve • D - Decline • R - Review • E - Escalate Additional status codes displayed in the AWC. • C - Original transaction was approved but due to dynamic scoring the transaction now has an elevated score and may require reevaluation. • X - Trnasaction was flagged for review but never acted upon and has timed out. • Y - Original transaction was approved but updated with AUTH=D and then timed out.	1	R	Kount
TIMEZONE	The timezone the device owner has set in the device's Control Panel. The value listed represents the number of minutes from Greenwich Meantime. Divide by 60 to get number of hours.	6	360	Kount
TRAN	Kount transaction ID number	12	13KP0ZY4M345	Kount
UAS	User agent string	1024	Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.9a8)	Merchant

RIS Response Keys	Description	Char Limit	Example Data	Source
VELO	Quantity of orders seen from persona within last 14 days. AUTH field must be equal to 'A'.	Number	19	Kount
VERS	Specifies version of Kount, built into SDK, must be supplied by merchant if not using the SDK	4	0555	Merchant-Kount
VMAX	Quantity of orders from persona within the most active 6 hour window in last 14 days. AUTH field must be equal to 'A'.	Number	11	Kount
VOICE_DEVICE	Is the device voice activated (related to mobile devices)	1	Y/N	Kount
WARNING_COUNT	Number of warnings merchant RIS post created	N/A	N/A	Kount
WARNING_N	Warning code displayed in RIS response	N/A	N/A	Kount

Appendix C

RIS Warning and Error Codes

Warning/Error Code	Warning/Error Label	Brief Description
201	MISSING_VERS	Missing version of Kount, this is built into SDK but must be supplied by merchant if not using the SDK
202	MISSING_MODE	The mode type for post is missing Refer to Risk Inquiry Service Modes section of this document
203	MISSING_MERC	The six digit Kount Merchant ID was not sent
204	MISSING_SESS	The unique session ID was not sent
205	MISSING_TRAN	Kount transaction ID number
211	MISSING_CURR	The currency was missing in the RIS submission
212	MISSING_TOTL	The total amount was missing
221	MISSING_EMAL	The email address was missing
222	MISSING_ANID	For MODE=P RIS inqueries the caller ID is missing
223	MISSING_SITE	The website identifier that was created in the Agent Web Console ('DEFAULT' is the default website ID) is missing
231	MISSING_PTYP	The payment type is missing. Refer to the RIS Payment Types section of this document for details
232	MISSING_CARD	The credit card information is missing
233	MISSING_MICR	Missing Magnetic Ink Character Recognition string
234	MISSING_PYPL	The PayPal Payer ID is missing
235	MISSING_PTOK	The payment token is missing. Refer to the RIS Payment Types section of this document for details
241	MISSING_IPAD	The IP address is missing
251	MISSING_MACK	The merchant acknowledgement is missing
261	MISSING_POST	The RIS query submitted to Kount contained no data
271	MISSING_PROD_TYPE	The shopping cart data array attribute is missing. Refer to the Shopping Cart Data section of this document for details
272	MISSING_PROD_ITEM	The shopping cart data array attribute is missing. Refer to the Shopping Cart Data section of this document for details
273	MISSING_PROD_DESC	The shopping cart data array attribute is missing. Refer to the Shopping Cart Data section of this document for details
274	MISSING_PROD_QUANT	The shopping cart data array attribute is missing. Refer to the Shopping Cart Data section of this document for details

Warning/Error Code	Warning/Error Label	Brief Description
275	MISSING_PROD_PRICE	The shopping cart data array attribute is missing. Refer to the Shopping Cart Data section of this document for details
301	BAD_VERS	The version of Kount supplied by merchant does not fit the four integer parameter
302	BAD_MODE	The mode type is invalid. Refer to the RIS Inquiry Service Modes section of this document for details
303	BAD_MERC	The six digit Kount Merchant ID is malformed or wrong
304	BAD_SESS	The unique session ID is invalid. Refer to the Data Collector Requirements and Section creation code example sections of this document for details
305	BAD_TRAN	Kount transaction ID number is malformed
311	BAD_CURR	The currency was wrong in the RIS submission
312	BAD_TOTL	The total amount is wrong. TOTL is the whole number amount charged to customer
321	BAD_EMAL	The email address does not meet required format or is greater than 64 characters in length
322	BAD_ANID	For MODE=P RIS inqueries the caller ID is malformed
323	BAD_SITE	The website identifier that was created in the Agent Web Console ('DEFAULT' is the default website ID) does not match what was created in the AWC.
324	BAD_FRMT	The specified format is wrong. Format options are key value pairs, XML, JSON, YAML
331	BAD_PTYP	The payment type is wrong. Refer to the RIS Payment Types section of this document for details
332	BAD_CARD	The credit card information is malformed or wrong, test cards do not work in the production environment
333	BAD_MICR	Malformed or improper Magnetic Ink Character Recognition string. Refer to the RIS Payment Types section of this document for details
334	BAD_PYPL	The PayPal Payer ID is malformed or corrupt. Refer to the RIS Payment Types section of this document for details
335	BAD_GOOG	Malformed or improper Google Checkout Account ID string. Refer to the RIS Payment Types section of this document for details
336	BAD_BLML	Malformed or improper Bill Me Later account number. Refer to the RIS Payment Types section of this document for details
337	BAD_PENC	The encryption method specified is wrong
338	BAD_GDMP	The GreenDot payment token is not a valid payment token

Warning/Error Code	Warning/Error Label	Brief Description
339	BAD_HASH	When payment type equals 'CARD', [PTYP=CARD] and payment encryption type equals 'KHASH', [PENC=KHASH] the value must be 20 characters in length.
340	BAD_MASK	Invalid or excessive characters in the PTOK field
341	BAD_IPAD	The IP address does not match specifications
342	BAD_GIFT	The Gift Card payment token is invalid due to invalid characters, null, or exceeding character length
351	BAD_MACK	The merchant acknowledgement must be 'Y' or 'N'
362	BAD_CART	There is a discrepancy in the shopping cart key count and the number of items actually being sent in the cart
371	BAD_PROD_TYPE	The shopping cart data array attribute is missing. Refer to the Shopping Cart Data section of this document for details
372	BAD_PROD_ITEM	The shopping cart data array attribute is corrupt or missing. Refer to the Shopping Cart Data section of this document for details
373	BAD_PROD_DESC	The shopping cart data array attribute is corrupt or missing. Refer to the Shopping Cart Data section of this document for details
374	BAD_PROD_QUANT	The shopping cart data array attribute is corrupt or missing. Refer to the Shopping Cart Data section of this document for details
375	BAD_PROD_PRICE	The shopping cart data array attribute is corrupt or missing. Refer to the Shopping Cart Data section of this document for details
399	BAD_OPTN	A UDF has been mistyped or does not exist in the Agent Web Console
401	EXTRA_DATA	RIS keys submitted by merchant were not part of SDK
404	UNNECESSARY_PTOK	When PTYP equals NONE and a PTOK is submitted
413	REQUEST_ENTITY_TOO_LARGE	The RIS Post to Kount exceeded the 4K limit. Refer to the Shopping Cart Data section for further details
501	UNAUTH_REQ	Error regarding certificate • Using test certificate in prod • Using prod certificate in test • Certificate passphrase is wrong • Merchant ID does not match • Certificate has expired

Warning/Error Code	Warning/Error Label	Brief Description
502	UNAUTH_MERC	Invalid Merchant ID has been entered
601	SYS_ERR	Unspecified system error - Contact Merchant Services
602	SYS_NOPROCESS	Kount will not process particular transaction
701	NO_HDR	No header found with merchantId=[XXXXX], sessionId=[htot2kk5khpamo45f777q455], trans=[122347] This error occurs when a RIS request goes to the database and there is no data available in the reply. The Update post had an invalid transaction ID#. Check all required fields for update post and confirm they are being passed correctly.

- **Missing:** When this designation appears, the customer has failed to complete a required field.
- **Bad:** When this designation appears, some data was sent but failed to meet specifications. This could also be explained as malformed data or bad code that did not meet specifications, such as **AVS=W** instead of **AVS=M**.

Appendix D

RIS Response example from recommended methods

```
MERC=900100
MODE=Q
TRAN=6GJX0Y6HVQ72
ORDR=736d473edd
AUTO=A
SCOR=29
KAPT=Y
SITE=DEFAULT
WARNING_COUNT=2
WARNING_0=399 BAD_OPTN Field: [DOB], Value: [1980-00-00]
WARNING_1=399 BAD_OPTN Field: [GENDER], Value: [H]


MODE=E
ERRO=323
ERROR_0=323 BAD_SITE Field: [SITE], Value: [DEFAULT1]
ERROR_1=311 BAD_CURR Field: [CURR], Value: [US]
ERROR_2=341 BAD_IPAD Field: [IPAD], Value: [127.0.0.1234]
ERROR_COUNT=3
WARNING_0=399 BAD_OPTN Field: [DOB], Value: [1980-00-00]
WARNING_1=399 BAD_OPTN Field: [GENDER], Value: [K]
WARNING_COUNT=2
```

RIS Response example using all methods without warnings and with rules triggered

```
VERS=0555
MODE=Q
TRAN=6GJX0Y6HVQ72
MERC=100100
SESS=51b4511430736d473eddc2022a22d556
ORDR=736d473edd
AUTO=A
SCOR=29
GEOX=US
BRND=DISC
REGN=
NETW=A
KYCF=N
KAPT=Y
```

```
CARDS=1
DEVICES=1
EMAILS=1
VELO=0
VMAX=0
SITE=DEFAULT
DEVICE_LAYERS=.877D53E52E.910C7E4A6A.61FD602D96.DFBD320050
FINGERPRINT=00482B9BED15A272730FCB590FFEBDDD
TIMEZONE=420
LOCALTIME=2013-08-30 18:00
REGION=
COUNTRY=US
PROXY=N
JAVASCRIPT=Y
FLASH=Y
COOKIES=Y
HTTP_COUNTRY=US
LANGUAGE=EN
MOBILE_DEVICE=N
MOBILE_TYPE=
MOBILE_FORWARDER=N
VOICE_DEVICE=N
PC_REMOTE=N
RULES_TRIGGERED=2
// If RULES_TRIGGERED=0, then RULE_ID_X and RULE_DESCRIPTION_X lines do not appear.
RULE_ID_0=417436
RULE_DESCRIPTION_0=Custom Counter 3
RULE_ID_1=417488
RULE_DESCRIPTION_1=Custom Counter 1
// If COUNTERS_TRIGGERED > 0, then COUNTER_NAME_X and COUNTER_VALUE_X appear.
COUNTERS_TRIGGERED=2
COUNTER_NAME_0=COUNTER WITH SPACES
COUNTER_VALUE_0=4
COUNTER_NAME_1=MYCOUNTER
COUNTER_VALUE_1=3
REASON_CODE=
DDFS=2013-08-15
DSR=768x1024
UAS=Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
 Chrome/29.0.1547.62 Safari/537.36
BROWSER=Chrome 29.0.1547.62
OS=Windows 7
PIP_IPAD=192.168.0.1
PIP_LAT=13.0788
PIP_LON=-59.5853
PIP_COUNTRY=BB
```

```
PIP_REGION=SM
PIP_CITY=Bridgetown
PIP_ORG=Example Organization
IP_IPAD=10.0.0.1
IP_LAT=43.6091
IP_LON=-116.2097
IP_COUNTRY=US
IP_REGION=ID
IP_CITY=Boise
IP_ORG=Company Inc.
WARNING_COUNT=0
```

RIS Response using all methods with warnings and rules triggered example

```
VERS=0555
MODE=Q
TRAN=6GJX0HJD1RM9
MERC=100100
SESS=51b4511430736d473eddc2022a22d556
ORDR=736d473edd
AUTO=A
SCOR=29
GEOX=US
BRND=DISC
REGN=
NETW=A
KYCF=N
KAPT=Y
CARDS=1
DEVICES=1
EMAILS=1
VELO=0
VMAX=0
SITE=DEFAULT
DEVICE_LAYERS=.877D53E52E.910C7E4A6A.61FD602D96.DFBD320050
FINGERPRINT=00482B9BED15A272730FCB590FFEBDDD
TIMEZONE=420
LOCALTIME=2013-08-30 18:01
REGION=
COUNTRY=US
PROXY=N
JAVASCRIPT=Y
FLASH=Y
COOKIES=Y
HTTP_COUNTRY=US
LANGUAGE=EN
```

```
MOBILE_DEVICE=N
MOBILE_TYPE=
MOBILE_FORWARDER=N
VOICE_DEVICE=N
PC_REMOTE=N
RULES_TRIGGERED=1
RULE_ID_0=1234
RULE_DESCRIPTION_0=Deny all orders originating from Fooland
COUNTERS_TRIGGERED=0
REASON_CODE=
DDFS=2013-08-15
DSR=768x1024
UAS=Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
 Chrome/29.0.1547.62 Safari/537.36
BROWSER=Chrome 29.0.1547.62
OS=Windows 7
PIP_IPAD=192.168.0.1
PIP_LAT=13.0788
PIP_LON=-59.5853
PIP_COUNTRY=BB
PIP_REGION=SM
PIP_CITY=Bridgetown
PIP_ORG=Example Organization
IP_IPAD=10.0.0.1
IP_LAT=43.6091
IP_LON=-116.2097
IP_COUNTRY=US
IP_REGION=ID
IP_CITY=Boise
IP_ORG=Company Inc.
WARNING_0=399 BAD_OPTN Field: [DOB], Value: [1980-00-00]
WARNING_1=399 BAD_OPTN Field: [GENDER], Value: [H]
WARNING_COUNT=2
```

RIS response error without warnings

```
MODE=E
ERRO=323
ERROR_0=323 BAD_SITE Field: [SITE], Value: [DEFAULT1]
ERROR_1=341 BAD_IPAD Field: [IPAD], Value: [127.0.0.1234]
ERROR_COUNT=2
WARNING_COUNT=0
```

RIS response error with warnings

```
MODE=E
ERRO=323
ERROR_0=323 BAD_SITE Field: [SITE], Value: [DEFAULT1]
ERROR_1=311 BAD_CURR Field: [CURR], Value: [US]
ERROR_2=341 BAD_IPAD Field: [IPAD], Value: [127.0.0.1234]
ERROR_COUNT=3
WARNING_0=399 BAD_OPTN Field: [DOB], Value: [1980-00-00]
WARNING_1=399 BAD_OPTN Field: [GENDER], Value: [K]
WARNING_COUNT=2
```

Appendix E

To preserve the security of RIS data, merchants must authenticate to RIS using an X.509 certificate when submitting RIS requests. Certificates are exported via a browser. Additional details regarding the certificate are found in the SDK Guide.

Supported Browsers for certificate exportation:

- Microsoft Internet Explorer version 7 and newer (must be in compatibility mode for IE 10).
- Mozilla Firefox version 5 and newer
- Apple Safari

NOTE: Google Chrome does not support the exportation of certificates.

Converting RIS Certificates to Different Formats

Depending on a variety of factors, you may need to convert your certificate to a different format in order to effectively submit a RIS request.

If you are using a web browser that exports P12 or PFX files, such as Mozilla Firefox, or Internet Explorer, you may need to use the OpenSSL command-line tool to convert the certificate to a PEM formatted certificate file and key file. If you are using a Windows platform, you can download OpenSSL for Windows to perform the conversion.

When converting the exported certificate, verify these two factors:

1. Is OpenSSL the application used to convert the file?
2. Has cURL been compiled with NSS?

If both are true, it is requisite that the version of OpenSSL be 0.9.x. If OpenSSL version 1.x.x is used, the converted certificate will not function and an error will be thrown.

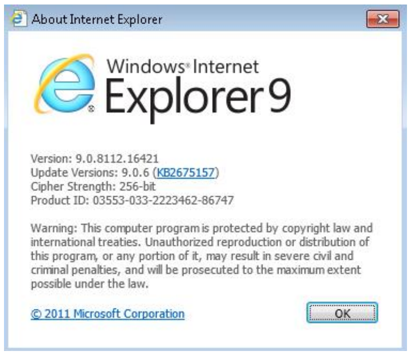
The following pages describe the exportation and conversion of the RIS certificate.

Required certificate format:

```
PHP SDK  .pem
PERL SDK  .p12/.pfx
Java SDK .p12/.pfx
.NET SDK  .p12/.pfx
```

Certificate Exportation and Conversion

This section outlines the required steps for preparing your Internet Explorer 9 web browser to be able to use exported certificates.

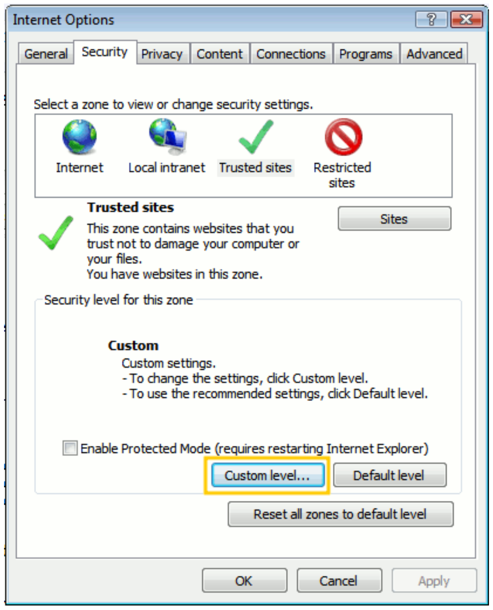


Version: 9.0.8112.16421
Update Versions: 9.0.6 (KB2675157)
Cipher Strength: 256-bit
Product ID: 03553-033-2223462-86747

Warning: This computer program is protected by copyright law and international treaties. Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under the law.

© 2011 Microsoft Corporation

1. Click **Tools** and then click **Internet Options**. Click **Security** tab and then click **Custom level** button.



Select a zone to view or change security settings.

Trusted sites

This zone contains websites that you trust not to damage your computer or your files.
You have websites in this zone.

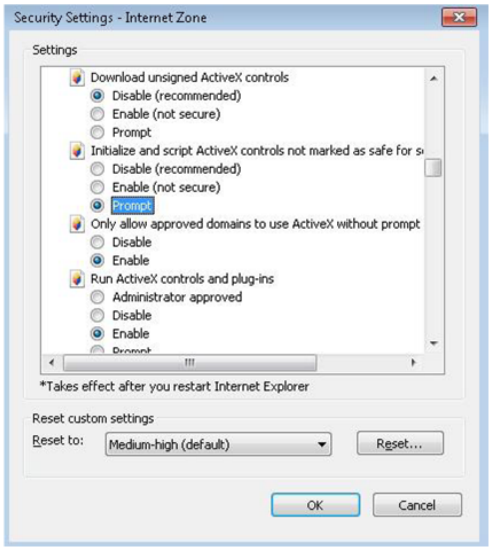
Security level for this zone

Custom

Custom settings.
- To change the settings, click Custom level.
- To use the recommended settings, click Default level.

[ ] Enable Protected Mode (requires restarting Internet Explorer)

2. In the **Settings** box, scroll down and locate **Initialize and script ActiveX controls not marked as safe for scripting (not secure)** and click **Prompt**, then click **OK**.


Security Settings - Internet Zone

Settings

Download unsigned ActiveX controls
- [✓ Disable (recommended)]
- Enable (not secure)
- Prompt

Initialize and script ActiveX controls not marked as safe for s
- Disable (recommended)
- Enable (not secure)
- [✓ Prompt]

Only allow approved domains to use ActiveX without prompt
- Disable
- [✓ Enable]

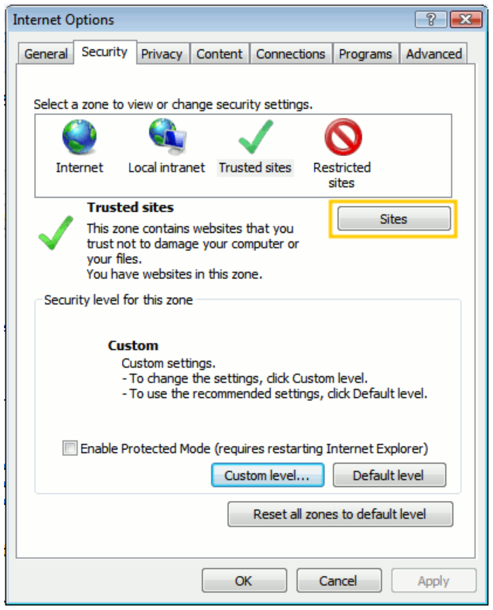
Run ActiveX controls and plug-ins
- Administrator approved
- Disable
- [✓ Enable]
- Prompt

*Takes effect after you restart Internet Explorer

Reset custom settings

Reset to: Medium-high (default)

3. On the **Security** tab in **Internet Options**, select **Trusted Sites**, and then click **Sites**.


Internet Options

Select a zone to view or change security settings.

Trusted sites

This zone contains websites that you trust not to damage your computer or your files.
You have websites in this zone.

Security level for this zone

Custom
Custom settings.
- To change the settings, click Custom level.
- To use the recommended settings, click Default level.

[ ] Enable Protected Mode (requires restarting Internet Explorer)

4. In the **Trusted Sites** dialog, in the **Add this website to the zone** field, type in the URL of the website such as in `https://awc.test.company.net` and `https://awc.company.net` and click **Add**. The URL will be added to the Websites field. Click **Close**.

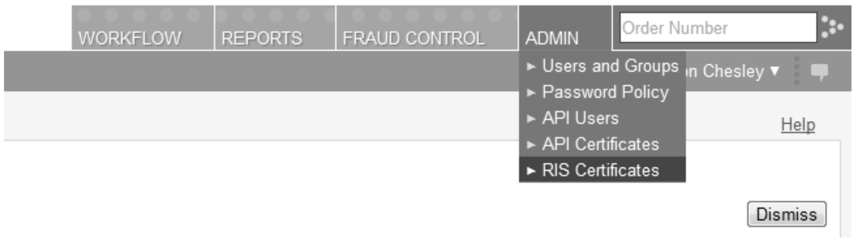
5. On the **Internet Options** dialog, click **OK**.

6. You <u>must</u> restart Internet Explorer to allow your changes to be applied.

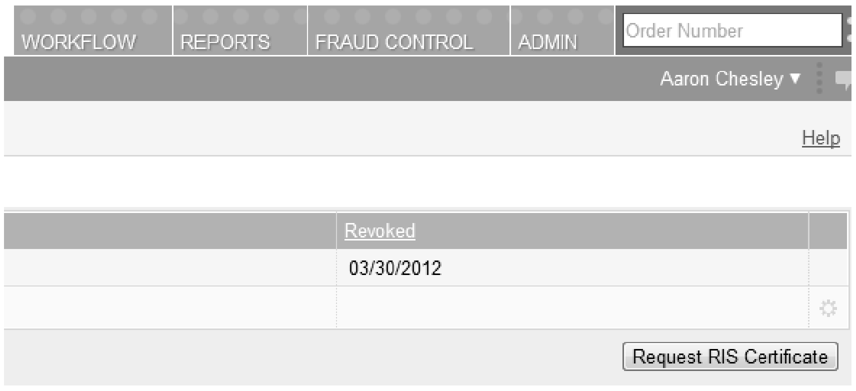
Once IE9 has restarted, you are ready to request the RIS certificate from Kount.

Certificate Request

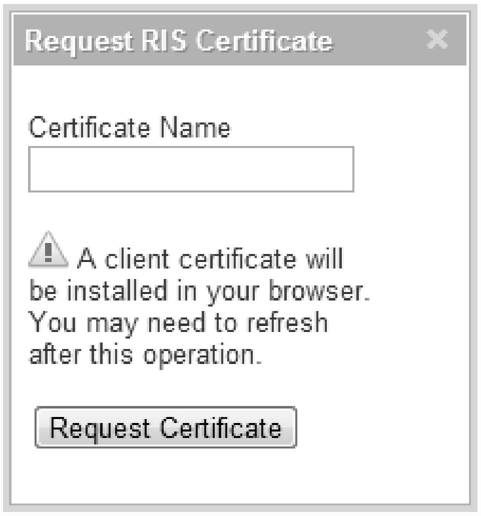
1. Browse to desired console, test or production.
2. Click the **Admin** tab, then click **RIS Certificates**.



3. Click the **Request RIS Certificate** button.



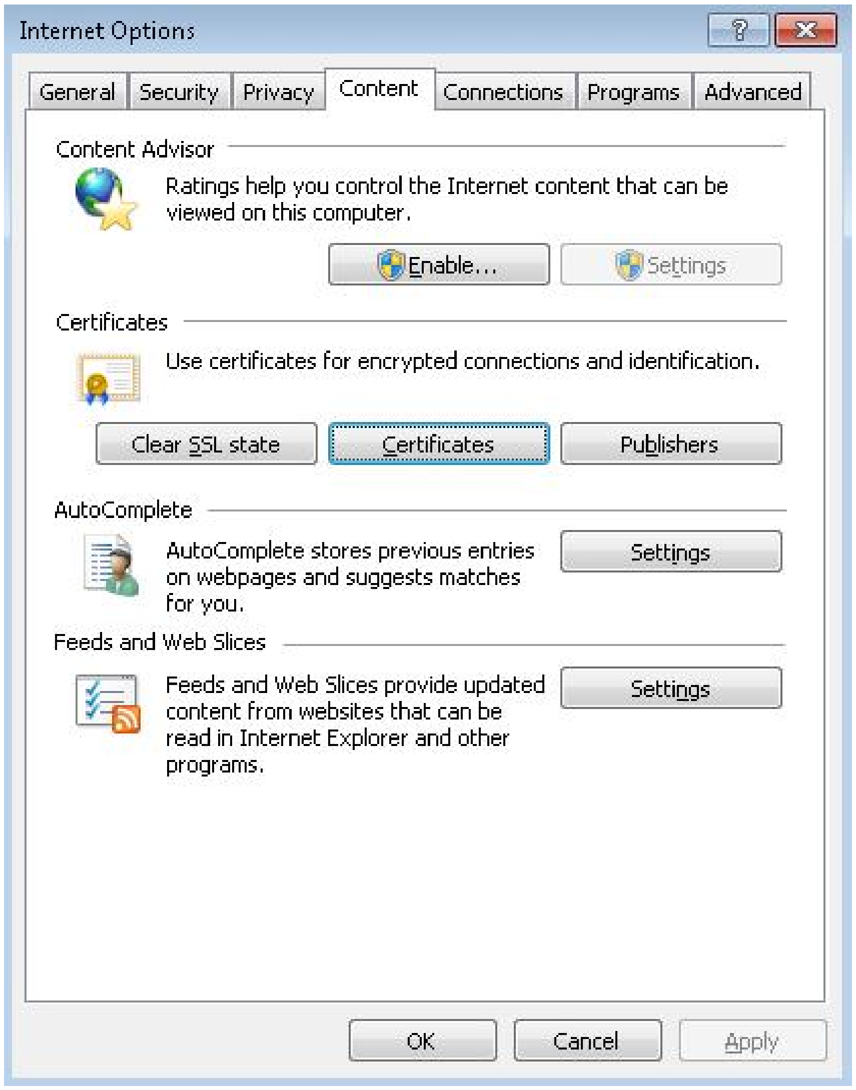
4. Fill in **Certificate Name** then click **Request Certificate**.



A client certificate will be installed in your browser. You may need to refresh after this operation.

Certificate Export in Internet Explorer

1. In IE click **Tools**, **Internet Options**, **Content** tab, **Certificates** button.



Internet Options

General | Security | Privacy | Content | Connections | Programs | Advanced

Content Advisor

Ratings help you control the Internet content that can be viewed on this computer.

Enable... | Settings

Certificates

Use certificates for encrypted connections and identification.

Clear SSL state | Certificates | Publishers

AutoComplete

AutoComplete stores previous entries on webpages and suggests matches for you.

Settings

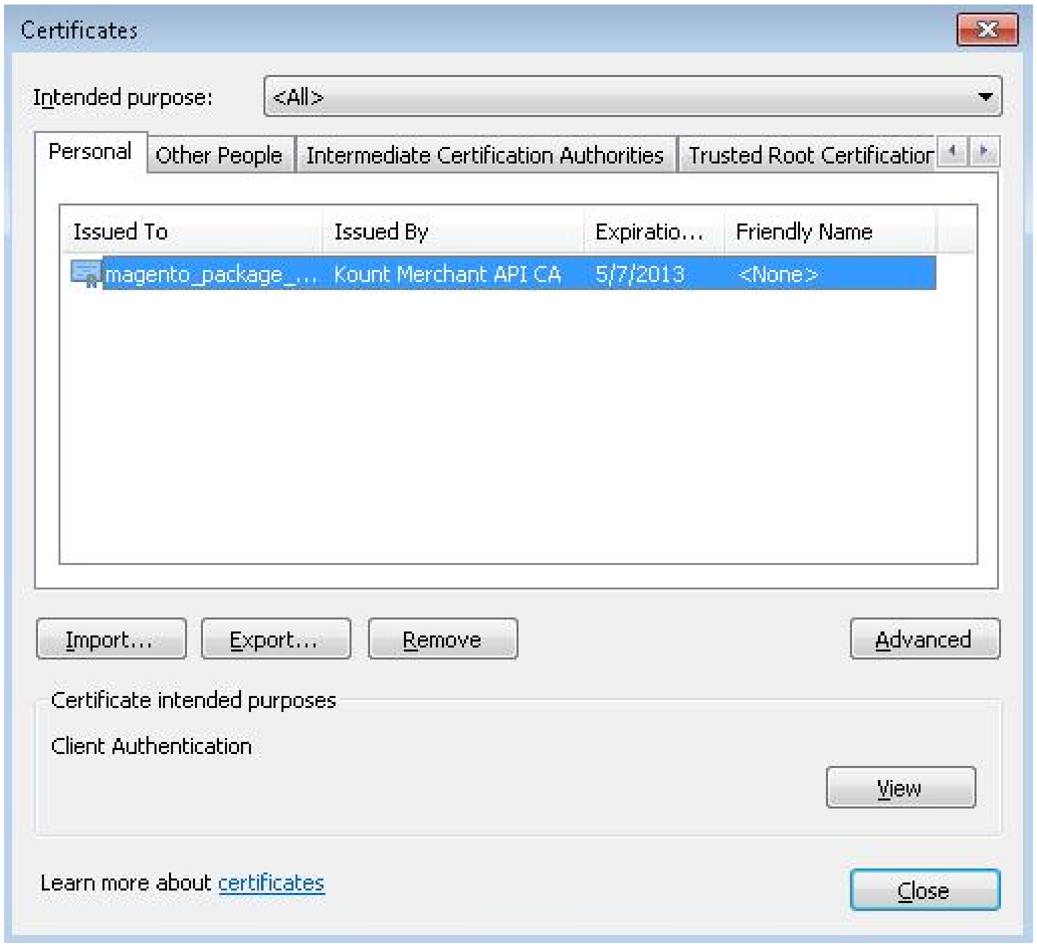
Feeds and Web Slices

Feeds and Web Slices provide updated content from websites that can be read in Internet Explorer and other programs.

Settings

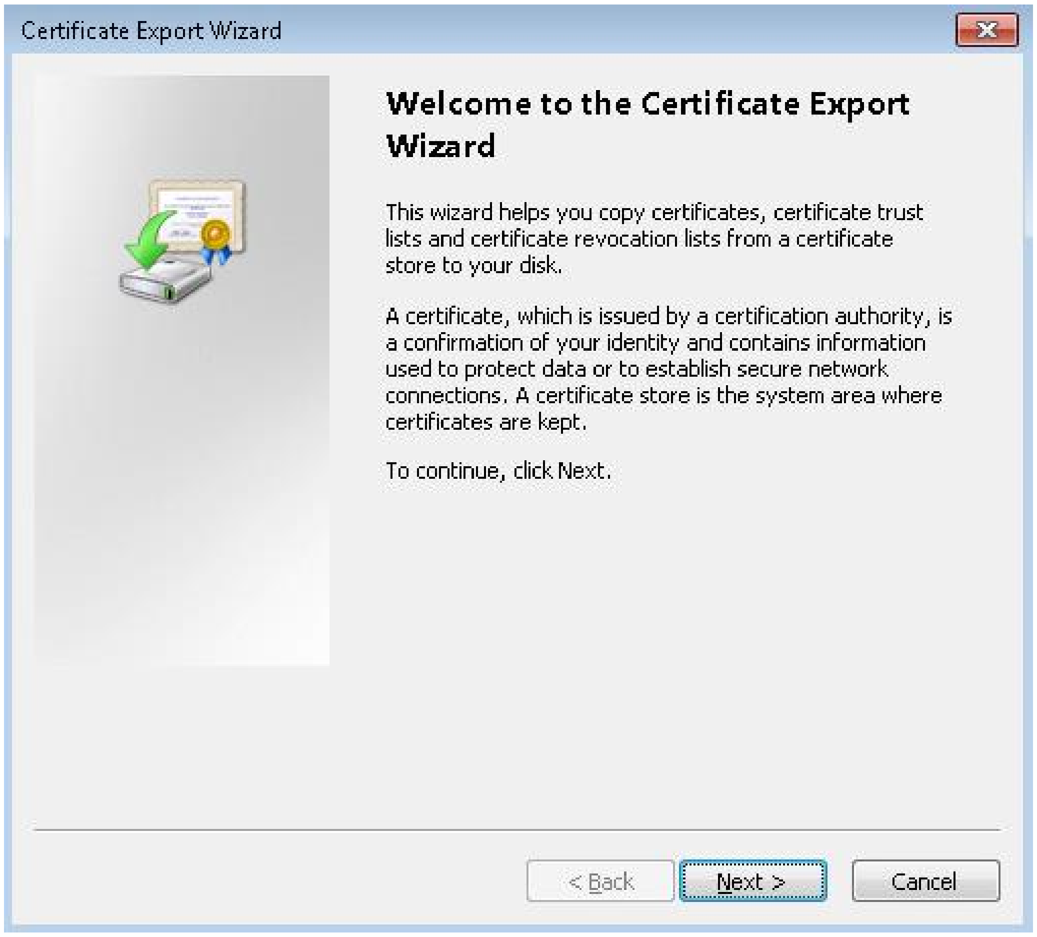
OK | Cancel | Apply

2. On the **Certificates** window, on the **Personal** tab, click the name of the certificate in the list you wish to export and then click the **Export** button.


Certificates
Intended purpose: <All>
Personal
Other People
Intermediate Certification Authorities
Trusted Root Certificatior
Import...
Export...
Remove
Advanced
Certificate intended purposes
Client Authentication
View
Learn more about certificates
Close

Issued To	Issued By	Expiratio...	Friendly Name
magento_package_...	Kount Merchant API CA	5/7/2013	<None>

3. Click **Next** on the Certificate Export Wizard window.



Certificate Export Wizard

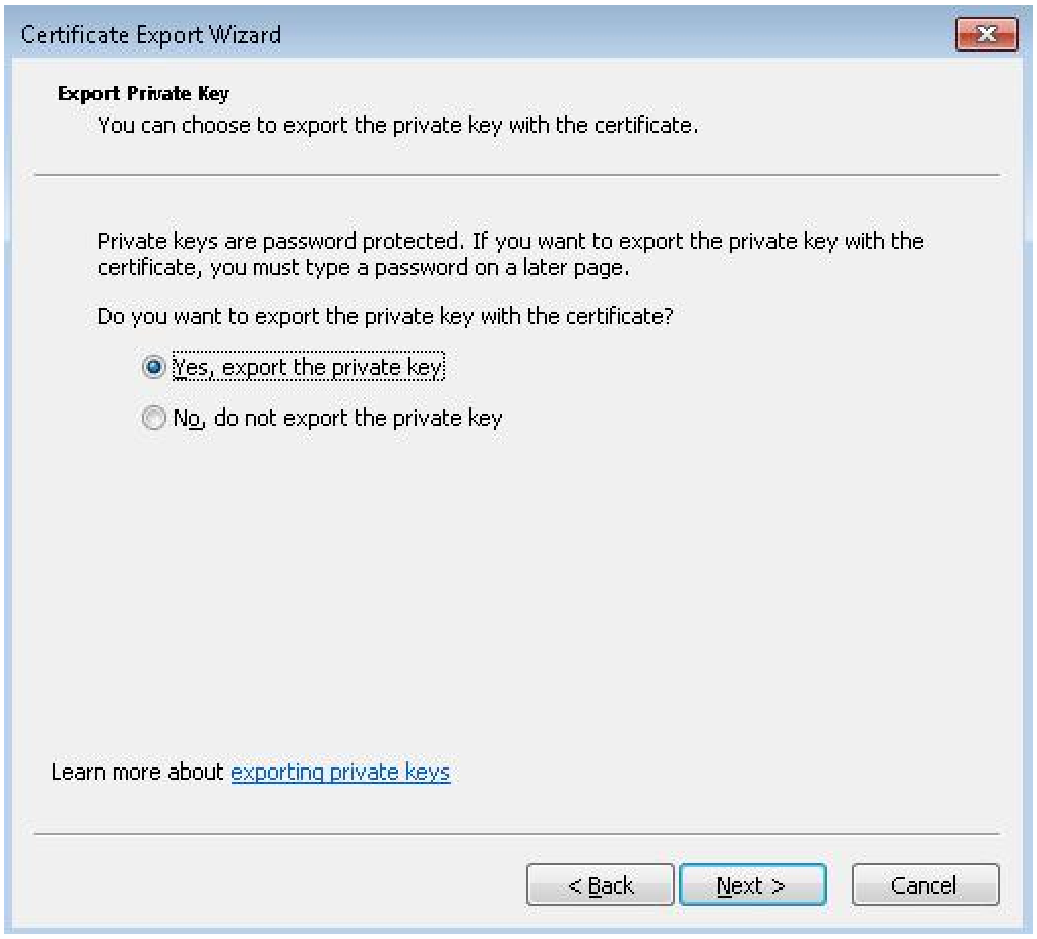
Welcome to the Certificate Export Wizard

This wizard helps you copy certificates, certificate trust lists and certificate revocation lists from a certificate store to your disk.

A certificate, which is issued by a certification authority, is a confirmation of your identity and contains information used to protect data or to establish secure network connections. A certificate store is the system area where certificates are kept.

To continue, click Next.

4. Click **Yes, export the private key radio** button.



Certificate Export Wizard

Export Private Key

You can choose to export the private key with the certificate.

Private keys are password protected. If you want to export the private key with the certificate, you must type a password on a later page.

Do you want to export the private key with the certificate?

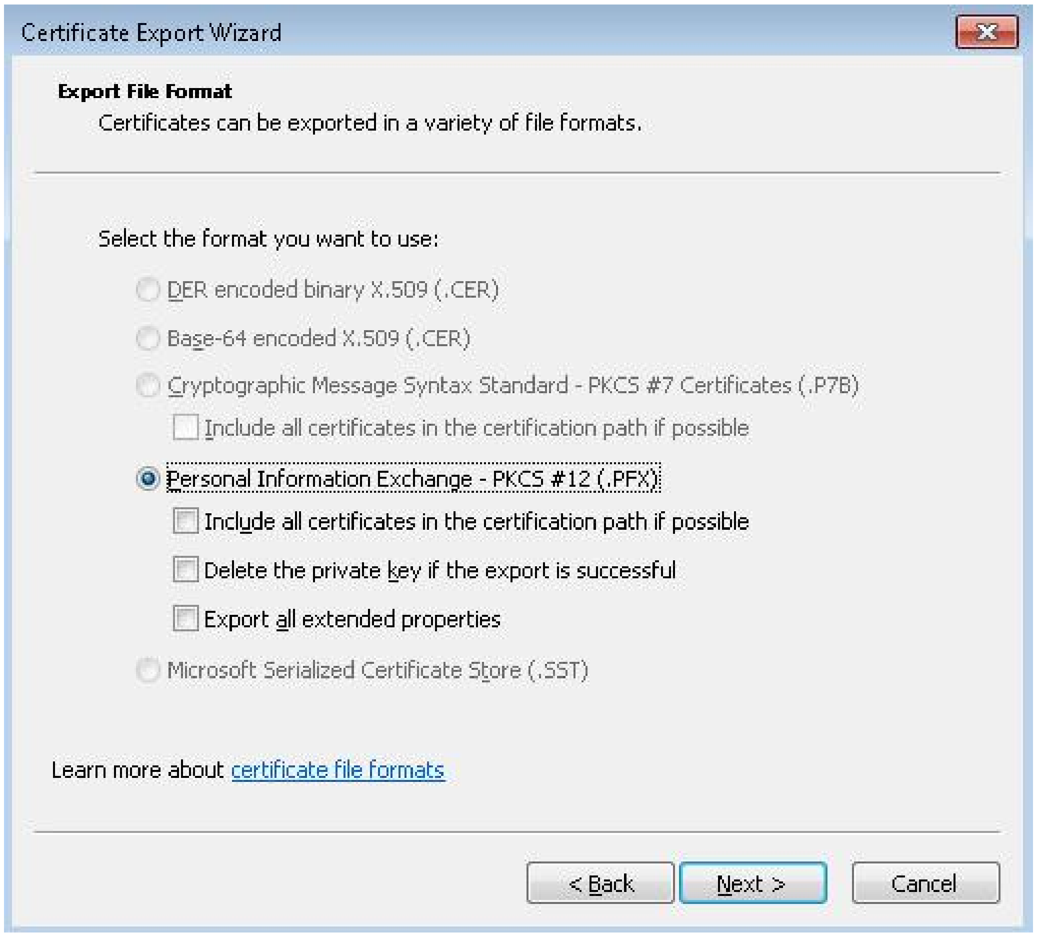
[✓ Yes, export the private key]

No, do not export the private key

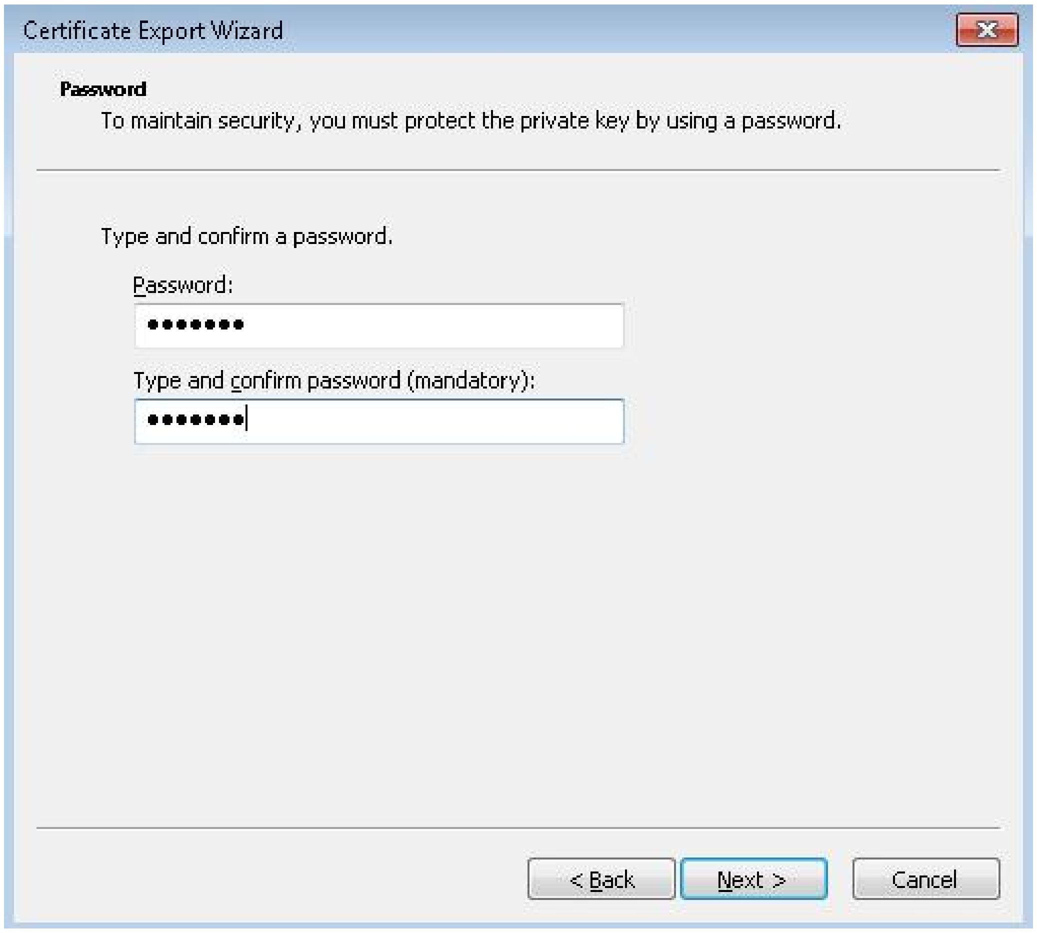
Learn more about exporting private keys

5. On the **Export File Format** page, select the **Personal Information Exchange** radio button, any of the check boxes beneath the radio button that suits your security needs, and then click **Next**.

NOTE While it is entirely up to the merchant as to which, if any, of these options is selected. To be **PCI Standards** compliant, it is recommended that the **Delete the private key if the export is successful** check box is checked. This prevents any unauthorized person who may gain access to a merchant environment from exporting your private key to a remote location for malicious purposes. However, if you select this option and later require access to it, you will need to request a new RIS certificate from Kount.


Certificate Export Wizard
Export File Format
Certificates can be exported in a variety of file formats.
Select the format you want to use:
DER encoded binary X.509 (.CER)
Base-64 encoded X.509 (.CER)
Cryptographic Message Syntax Standard - PKCS #7 Certificates (.P7B)
Include all certificates in the certification path if possible
Personal Information Exchange - PKCS #12 (.PFX)
Include all certificates in the certification path if possible
Delete the private key if the export is successful
Export all extended properties
Microsoft Serialized Certificate Store (.SST)
Learn more about certificate file formats
< Back
Next >
Cancel

6. On the **Password** page, enter the password for the certificate you are exporting in the **Password** field (must be at least six characters), enter it again in the **Type and confirm password (mandatory)** field, and then click **Next**.



Certificate Export Wizard

Password

To maintain security, you must protect the private key by using a password.

Type and confirm a password.

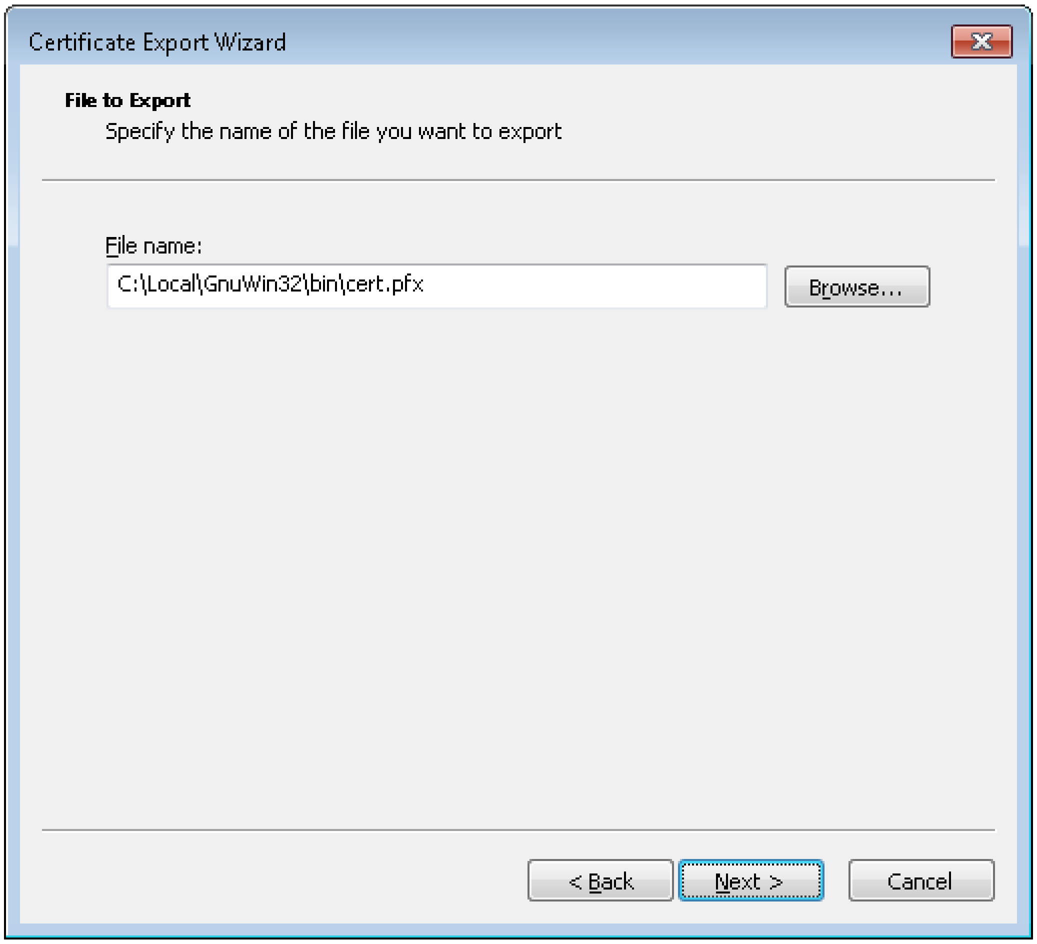
Password:

●●●●●●●

Type and confirm password (mandatory):

●●●●●●●

7. Input path or **Browse** to location where you would like to save the exported file.



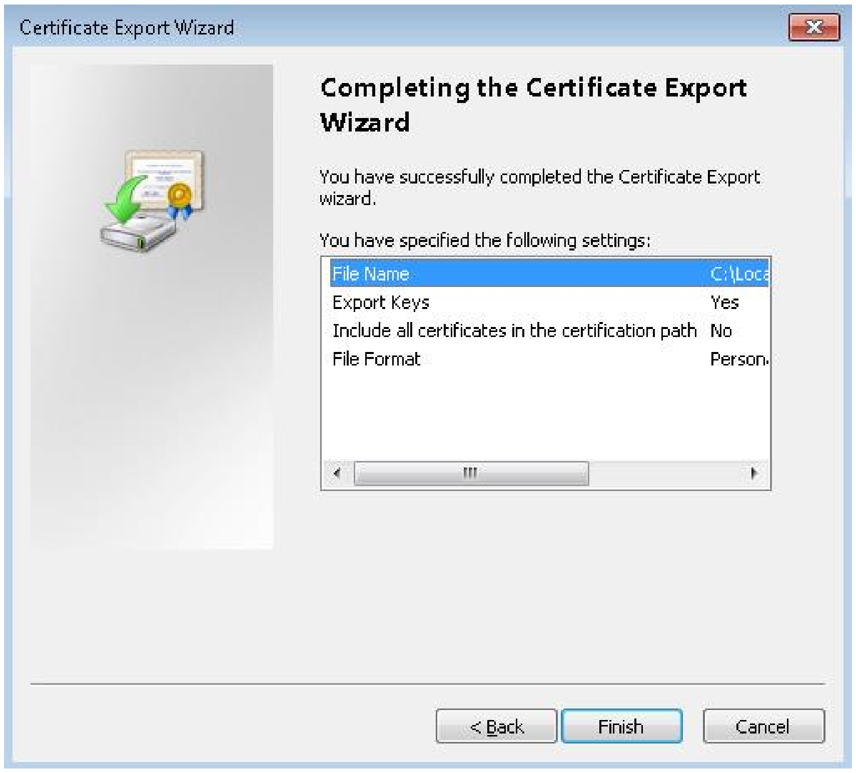
File to Export

Specify the name of the file you want to export

File name:

C:\Local\GnuWin32\bin\cert.pfx

8. In the **Completing the Certificate Export Wizard** page click **Finish**.



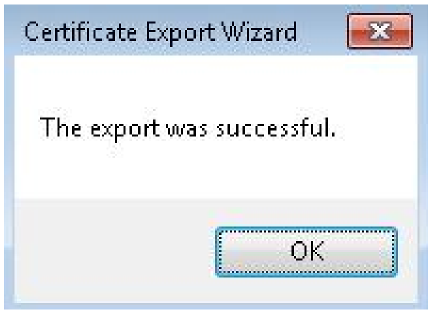
Completing the Certificate Export Wizard

You have successfully completed the Certificate Export wizard.

You have specified the following settings:

File Name	C:\Loca
Export Keys	Yes
Include all certificates in the certification path	No
File Format	Person

9. A success notification should appear.

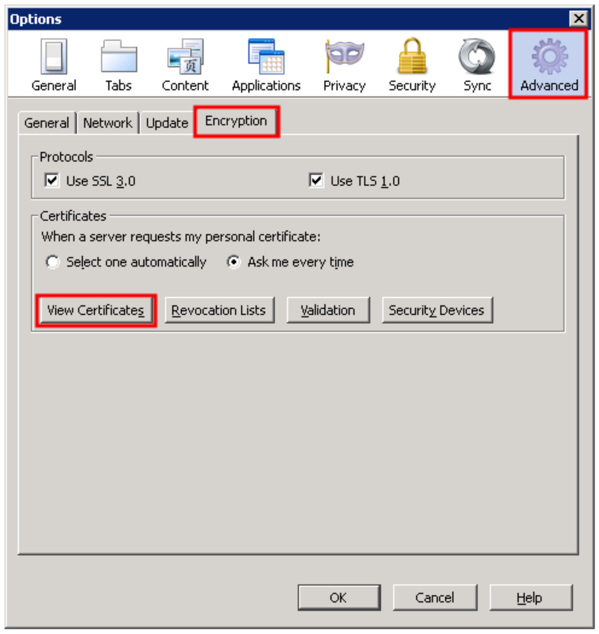


The export was successful.

Certificate Export in Mozilla Firefox

As you did on Internet Explorer, you will need to open the AWC in Firefox, request a RIS certificate, and then proceed with the following steps.

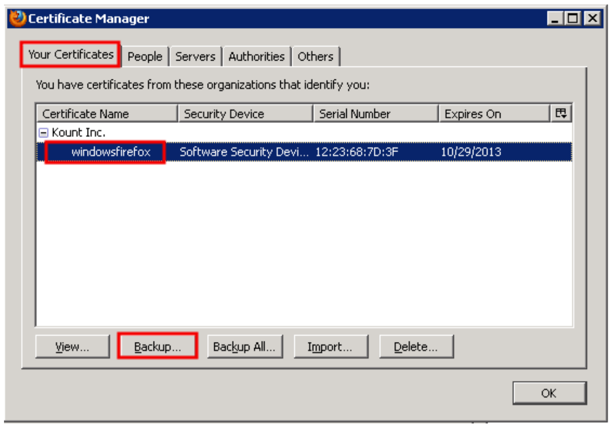
1. In Firefox, click **Tools** and then **Options**.
2. On the Options box, select Advanced, select the Encryption tab, and then select View Certificates.



Protocols
Use SSL 3.0 [✓]
Use TLS 1.0 [✓]

Certificates
When a server requests my personal certificate:
Select one automatically [✓ Ask me every time]

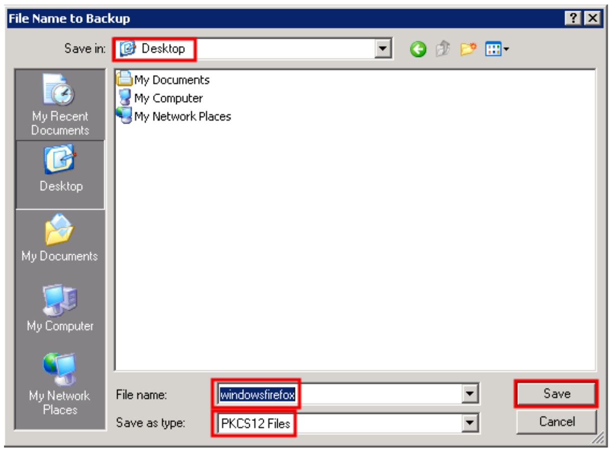
3. On the **Certificate Manager** dialog, on the **Your Certificates** tab, select the desired certificate in the list and then click **Backup**.



You have certificates from these organizations that identify you:

Certificate Name	Security Device	Serial Number	Expires On
Kount Inc.			
windowsfirefox	Software Security Devi...	12:23:68:7D:3F	10/29/2013

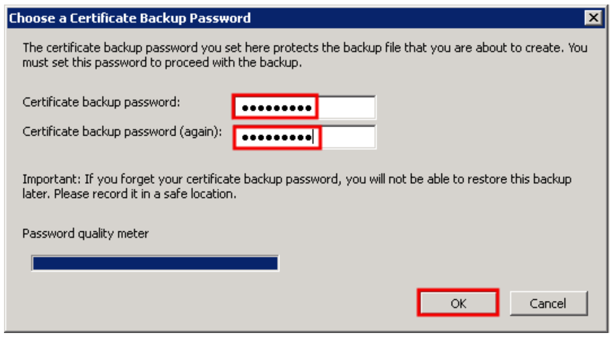
4. When the **File Name to Backup** dialog appears, give the certificate you want to export a name in the **File name** field, select a location to which you want to back up the certificate, such as your desktop, verify that the **PKCS12 Files** file format option is selected, and then click **Save**.
   - The **PKCS12 Files** option should be selected automatically.



File name: windowsfirefox

Save as type: PKCS12 Files

5. On the **Choose a Certificate Backup Password** dialog, enter a password for the certificate backup in the **Certificate backup password** field, enter the same password again in the **Certificate backup password (again)** field, and then click **OK**.
   - Notice that the **Password quality meter** bar shows an indication of the "strength" of the password.



The certificate backup password you set here protects the backup file that you are about to create. You must set this password to proceed with the backup.

Certificate backup password: ●●●●●●●●●

Certificate backup password (again): ●●●●●●●●●

Important: If you forget your certificate backup password, you will not be able to restore this backup later. Please record it in a safe location.

6. When the **Alert** dialog indicating that the certificate and private key have been successfully backed up appears, click **OK**.
7. On the **Certificate Manager** dialog, click **OK** to close.
8. Close any other dialogs that may be open.

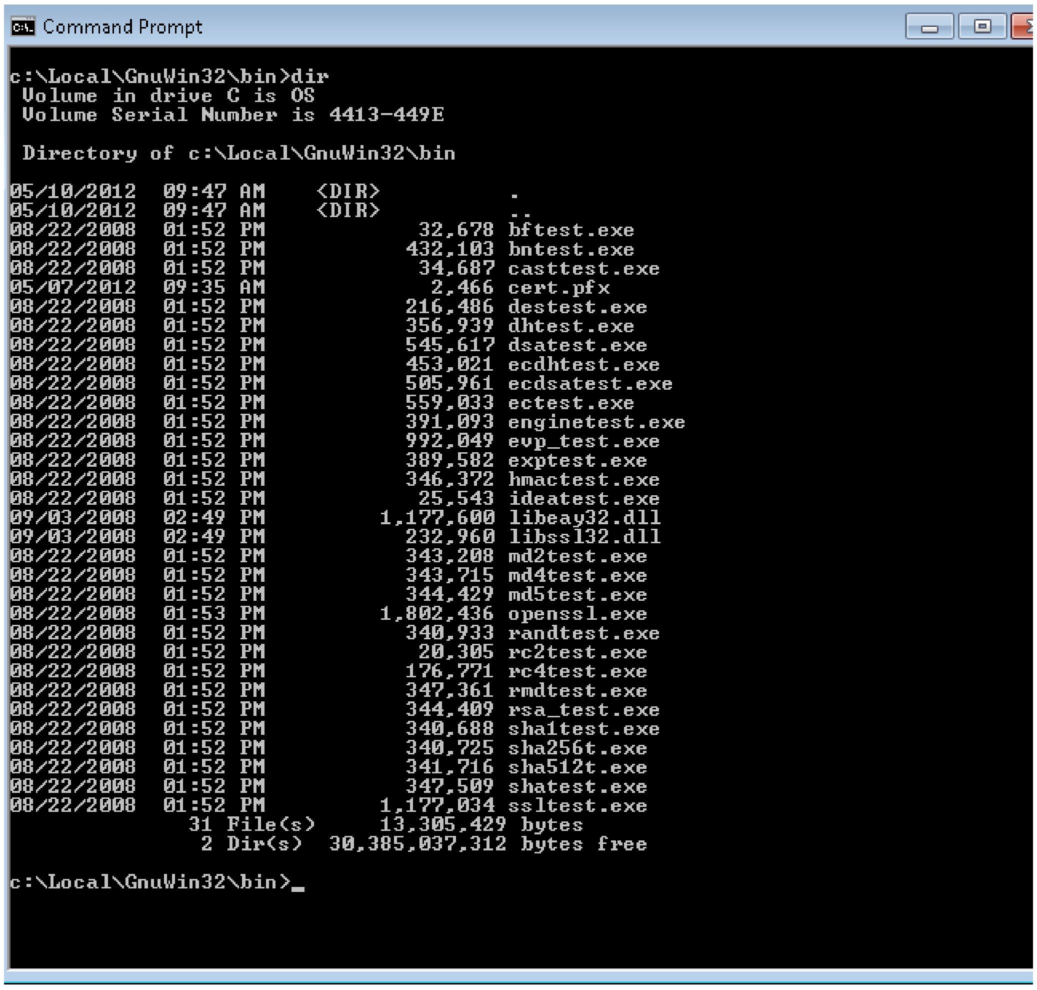
The **RIS certificate** has been successfully exported to your computer.

Certificate Conversion - Prerequisites

For Windows operating systems a separate application may be required to perform the certificate conversion.

X32 - `http://gnuwin32.sourceforge.net/packages/openssl.htm`
X64 - `http://code.google.com/p/openssl-for-windows/downloads/list`

1. Copy or move the exported file to the same directory where the openssl.exe file is located.
2. Open the command prompt and change directory to where the openssl.exe resides.

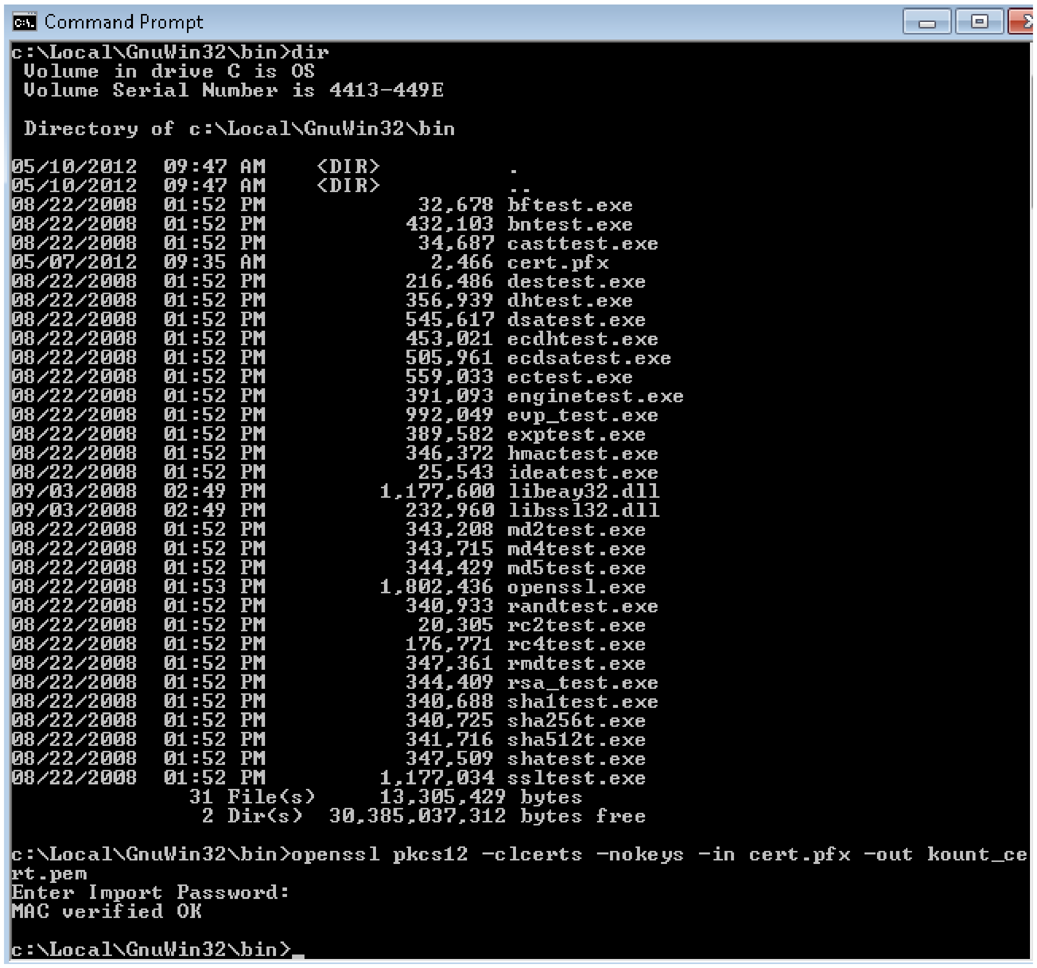


```
c:\Local\GnuWin32\bin>dir
 Volume in drive C is OS
 Volume Serial Number is 4413-449E

 Directory of c:\Local\GnuWin32\bin

05/10/2012  09:47 AM    <DIR>          .
05/10/2012  09:47 AM    <DIR>          ..
08/22/2008  01:52 PM            32,678 bftest.exe
08/22/2008  01:52 PM           432,103 bntest.exe
08/22/2008  01:52 PM            34,687 casttest.exe
05/07/2012  09:35 AM             2,466 cert.pfx
08/22/2008  01:52 PM           216,486 destest.exe
08/22/2008  01:52 PM           356,939 dhtest.exe
08/22/2008  01:52 PM           545,617 dsatest.exe
08/22/2008  01:52 PM           453,021 ecdhtest.exe
08/22/2008  01:52 PM           505,961 ecdsatest.exe
08/22/2008  01:52 PM           559,033 ectest.exe
08/22/2008  01:52 PM           391,093 enginetest.exe
08/22/2008  01:52 PM           992,049 evp_test.exe
08/22/2008  01:52 PM           389,582 exptest.exe
08/22/2008  01:52 PM           346,372 hmactest.exe
08/22/2008  01:52 PM            25,543 ideatest.exe
09/03/2008  02:49 PM         1,177,600 libeay32.dll
09/03/2008  02:49 PM           232,960 libssl32.dll
08/22/2008  01:52 PM           343,208 md2test.exe
08/22/2008  01:52 PM           343,715 md4test.exe
08/22/2008  01:52 PM           344,429 md5test.exe
08/22/2008  01:53 PM         1,802,436 openssl.exe
08/22/2008  01:52 PM           340,933 randtest.exe
08/22/2008  01:52 PM            20,305 rc2test.exe
08/22/2008  01:52 PM           176,771 rc4test.exe
08/22/2008  01:52 PM           347,361 rmdtest.exe
08/22/2008  01:52 PM           344,409 rsa_test.exe
08/22/2008  01:52 PM           340,688 sha1test.exe
08/22/2008  01:52 PM           340,725 sha256t.exe
08/22/2008  01:52 PM           341,716 sha512t.exe
08/22/2008  01:52 PM           347,509 shatest.exe
08/22/2008  01:52 PM         1,177,034 ssltest.exe
              31 File(s)     13,305,429 bytes
               2 Dir(s)  30,385,037,312 bytes free

c:\Local\GnuWin32\bin>_
```

3. Run the following command at the command prompt to convert the .PFX file into a .PEM certificate file.

NOTE The import password is the password that was set during the certificate export.

```
openssl pkcs12 -clcerts -nokeys -in filename.pfx -out file_cert.pem
```



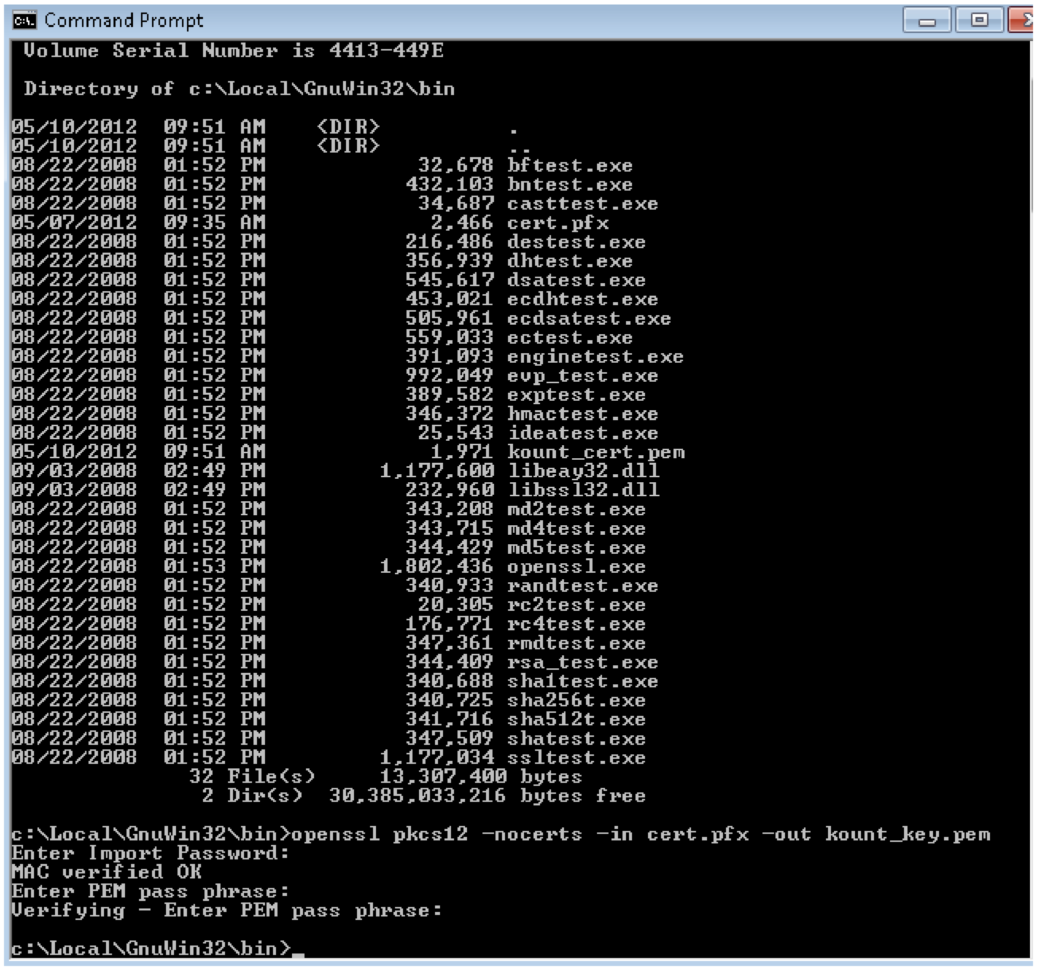
```
c:\Local\GnuWin32\bin>dir
 Volume in drive C is OS
 Volume Serial Number is 4413-449E

 Directory of c:\Local\GnuWin32\bin

05/10/2012  09:47 AM    <DIR>          .
05/10/2012  09:47 AM    <DIR>          ..
08/22/2008  01:52 PM            32,678 bftest.exe
08/22/2008  01:52 PM           432,103 bntest.exe
08/22/2008  01:52 PM            34,687 casttest.exe
05/07/2012  09:35 AM             2,466 cert.pfx
08/22/2008  01:52 PM           216,486 destest.exe
08/22/2008  01:52 PM           356,939 dhtest.exe
08/22/2008  01:52 PM           545,617 dsatest.exe
08/22/2008  01:52 PM           453,021 ecdhtest.exe
08/22/2008  01:52 PM           505,961 ecdsatest.exe
08/22/2008  01:52 PM           559,033 ectest.exe
08/22/2008  01:52 PM           391,093 enginetest.exe
08/22/2008  01:52 PM           992,049 evp_test.exe
08/22/2008  01:52 PM           389,582 exptest.exe
08/22/2008  01:52 PM           346,372 hmactest.exe
08/22/2008  01:52 PM            25,543 ideatest.exe
09/03/2008  02:49 PM         1,177,600 libeay32.dll
09/03/2008  02:49 PM           232,960 libssl32.dll
08/22/2008  01:52 PM           343,208 md2test.exe
08/22/2008  01:52 PM           343,715 md4test.exe
08/22/2008  01:52 PM           344,429 md5test.exe
08/22/2008  01:53 PM         1,802,436 openssl.exe
08/22/2008  01:52 PM           340,933 randtest.exe
08/22/2008  01:52 PM            20,305 rc2test.exe
08/22/2008  01:52 PM           176,771 rc4test.exe
08/22/2008  01:52 PM           347,361 rmdtest.exe
08/22/2008  01:52 PM           344,409 rsa_test.exe
08/22/2008  01:52 PM           340,688 sha1test.exe
08/22/2008  01:52 PM           340,725 sha256t.exe
08/22/2008  01:52 PM           341,716 sha512t.exe
08/22/2008  01:52 PM           347,509 shatest.exe
08/22/2008  01:52 PM         1,177,034 ssltest.exe
              31 File(s)     13,305,429 bytes
               2 Dir(s)  30,385,037,312 bytes free

c:\Local\GnuWin32\bin>openssl pkcs12 -clcerts -nokeys -in cert.pfx -out kount_ce
rt.pem
Enter Import Password:
MAC verified OK

c:\Local\GnuWin32\bin>_
```

4. Run the following command at the command prompt to convert the .PFX file into a .PEM key file.

NOTE The import password is the password that was set during the certificate export.

5. The PEM pass phrase is a separate password but may be the same as the certificate export password. The pass phrase password must be at least six characters in length.

```
openssl pkcs12 -nocerts -in filename.pfx -out file_key.pem
```



Command Prompt

```
 Volume Serial Number is 4413-449E

 Directory of c:\Local\GnuWin32\bin

05/10/2012  09:51 AM    <DIR>          .
05/10/2012  09:51 AM    <DIR>          ..
08/22/2008  01:52 PM            32,678 bftest.exe
08/22/2008  01:52 PM           432,103 bntest.exe
08/22/2008  01:52 PM            34,687 casttest.exe
05/07/2012  09:35 AM             2,466 cert.pfx
08/22/2008  01:52 PM           216,486 destest.exe
08/22/2008  01:52 PM           356,939 dhtest.exe
08/22/2008  01:52 PM           545,617 dsatest.exe
08/22/2008  01:52 PM           453,021 ecdhtest.exe
08/22/2008  01:52 PM           505,961 ecdsatest.exe
08/22/2008  01:52 PM           559,033 ectest.exe
08/22/2008  01:52 PM           391,093 enginetest.exe
08/22/2008  01:52 PM           992,049 evp_test.exe
08/22/2008  01:52 PM           389,582 exptest.exe
08/22/2008  01:52 PM           346,372 hmactest.exe
08/22/2008  01:52 PM            25,543 ideatest.exe
05/10/2012  09:51 AM             1,971 kount_cert.pem
09/03/2008  02:49 PM         1,177,600 libeay32.dll
09/03/2008  02:49 PM           232,960 libssl32.dll
08/22/2008  01:52 PM           343,208 md2test.exe
08/22/2008  01:52 PM           343,715 md4test.exe
08/22/2008  01:52 PM           344,429 md5test.exe
08/22/2008  01:53 PM         1,802,436 openssl.exe
08/22/2008  01:52 PM           340,933 randtest.exe
08/22/2008  01:52 PM            20,305 rc2test.exe
08/22/2008  01:52 PM           176,771 rc4test.exe
08/22/2008  01:52 PM           347,361 rmdtest.exe
08/22/2008  01:52 PM           344,409 rsa_test.exe
08/22/2008  01:52 PM           340,688 sha1test.exe
08/22/2008  01:52 PM           340,725 sha256t.exe
08/22/2008  01:52 PM           341,716 sha512t.exe
08/22/2008  01:52 PM           347,509 shatest.exe
08/22/2008  01:52 PM         1,177,034 ssltest.exe
              32 File(s)     13,307,400 bytes
               2 Dir(s)  30,385,033,216 bytes free

c:\Local\GnuWin32\bin>openssl pkcs12 -nocerts -in cert.pfx -out kount_key.pem
Enter Import Password:
MAC verified OK
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:

c:\Local\GnuWin32\bin>_
```

Appendix F

Standard Test Scenarios

Approve Test Scenarios

1. Add JohnDoeApprove@Acme.com email address to the VIP Approve list in the Test environment.
2. Place a test order using any item, via the front end of the test website.
   (a) When filling out the customer information, please use JohnDoeApprove@Acme.com as the email address.
   (b) Fill in necessary Test Credit Card information.
3. Confirm that after the order has been placed that the proper customer experience message is displaying.
4. Confirm that appears as approved in the Agent Web Console.

Decline Test Scenario

1. Add JohnDoeDecline@Acme.com email address to the VIP Decline list in the Test environment.
2. Place a test order using any item, via the front end of the test website.
   (a) When filling out the customer information, please use JohnDoeDecline@Acme.com as the email address.
   (b) Fill in necessary Test Credit Card information.
3. Confirm that after the order has been placed that the proper customer experience message is displaying.
4. Confirm that appears as declined in the Agent Web Console.

Review Test Scenario

1. Add JohnDoeReview@Acme.com email address to the VIP Decline list in the Test environment.
2. Place a test order using any item, via the front end of the test website.
   (a) When filling out the customer information, please use JohnDoeReview@Acme.com as the email address.
   (b) Fill in necessary Test Credit Card information.
3. Confirm that after the order has been placed that the proper customer experience message is displaying.
4. Confirm that appears as declined in the Agent Web Console.

Note: *Merchants are responsible for checking interaction with their OMS to validate expected behavior.*

Optional Test Scenarios

Shipping/Billing Address Test Scenario

1. Add JohnDoeReview@Acme.com email address to the VIP Review list in the Test environment.
2. Place a test order using any item, via the front end of the test website.
   (a) When filling out the customer information, please use JohnDoeReview@Acme.com as the email address.
   (b) Enter 1234 Main Street, Any Town, ID, USA 83705 for the Shipping Address.
   (c) Enter 5678 Oak Street, Any Town, ID 83705 for the Billing Address.
   (d) Fill in necessary Test Credit Card information.
3. Confirm that after the order has been placed that the proper customer experience message is displaying.
4. Confirm that the order is held as review in the Agent Web Console.
5. Confirm the Shipping Address passed correctly.
6. Confirm the Billing Address passed correctly.

Shipping Phone/Billing Phone Test Scenario

1. Add JohnDoeReview@Acme.com email address to the VIP Review list in the Test environment.
2. Place a test order using any item, via the front end of the test website.
   (a) When filling out the customer information, please use JohnDoeReview@Acme.com as the email address.
   (b) Enter 123-456-7890 for the Shipping Phone Number.
   (c) Enter 102-345-6789 for the Billing Phone Number.
   (d) Fill in necessary Test Credit Card information.
3. Confirm that after the order has been placed that the proper customer experience message is displaying.
4. Confirm that the order is held as review in the Agent Web Console.
5. Confirm the Shipping Phone passed correctly.
6. Confirm the Billing Phone passed correctly.

Order Number Test Scenario

1. Add JohnDoeReview@Acme.com email address to the VIP Review list in the Test environment.

2. Place a test order using any item, via the front end of the test website.
   (a) When filling out the customer information, please use JohnDoeReview@Acme.com as the email address.
3. Confirm that after the order has been placed that the proper customer experience message is displaying.
4. Confirm that the order is held as review in the Agent Web Console.
5. Confirm that the order number has populated correctly.

Note: *Merchants are responsible for checking interaction with their OMS to validate expected behavior.*

Appendix G

Required Elements

The following features should be tested prior to Kount certification. Utilize this checklist to help ensure that all data is posting as expected.

Data Device Collector

- ☐ Merchant_ID
- ☐ Unique Session_ID
- ☐ Merchant_URL
- ☐ Company_Server_URL

Risk Inquiry Service

- ☐ Unique Session_ID is being passed
- ☐ X.509 Certificate is being passed
- ☐ Required Inquiry Keys

Note: Session IDs link the Device Data to the Risk Inquiry Service Data. All Session IDs must be unique for 30 days.

Payment Type Testing

- ☐ Test all Payment Types that will be passed by the Merchant.

Pixel Testing

- ☐ Obfuscate the company's name within the image URL or pathing.
- ☐ The image path must be a HTTPS path.

Optional Elements To Be Tested

When applicable, test the optional features prior to Kount certification. Additionally, all features should be tested for each type of customer available on the merchant website.

Phone Order Testing

1. If the customer service agents navigate to a separate order entry page:

- ☐ Orders posted as a Mode=P
- ☐ Hard code IP address to 10.0.0.1
- ☐ Phone Number within the ANID field is provided

  - If no phone number is available, ANID field is hard coded to 0123456789

- ☐ If no email is provided, EMAIL field is being submitted as noemail@kount.com so that linking does not happen.

2. If the customer service agents navigate to the same order entry page as the customer:

- ☐ Orders posted as a Mode=Q
- ☐ Hard code IP address to 10.0.0.1
- ☐ If no email is provided, EMAIL field is being submitted as noemail@kount.com so that linking does not happen.

Risk Inquiry Service Testing

Confirm that the optional information, listed below, is being passed to Kount during the Risk Inquiry System (RIS) call.

- ☐ Optional Inquiry Keys

User Defined Fields

- ☐ Validate that any user defined fields defined in the Agent Web Console are being passed to Kount during RIS call.

Event Notification System (ENS) Testing

- ☐ Verify the ENS XML file is being received by the URL defined within the Agent Web Console.
- ☐ Validate that ENS is enabled and to the correct URL within the Agent Web Console Website settings page.

Items that *cannot* be tested in the Sandbox environment:

Persona Orders	Address and Phone Number Validation through Melissa Data
Velo Information	Network Information
VMax Information	Device Information
Distance Calculators	External Services

Appendix H

Direct Integration Milestones – High Level

Milestone 1: Install the Device Data Collector image on the order form

Device Data Collector is the method by which Kount collects key identity and network data from the customer's computer. An iFrame is placed onto the order form which references server side scripts and temporarily redirects the customer's browser to Kount. If the merchant is using a multi-page order form, the iFrame image should be placed on the same page where the card number is entered by the customer. The Device Data Collector image may be a 1x1 transparent pixel or another image as desired by the merchant.

Milestone 2: Code the order form data post to the Risk Inquiry System (RIS) specification

Order information is sent from the merchant to Kount when a customer checks out of the cart. Specific required information is sent to Kount, as described in the Technical Specification Guide, such as E-mail, card number, cart item; additional information is desirable. The HTTPS post constitutes a RIS Inquiry to Kount and will result in a real-time reply containing, among other things, a Kount risk score. The merchant needs to code to the Kount Technical Specification Guide in order to properly format the RIS Inquiry and also to accept the real time reply.

Milestone 3: Testing and Certification

Upon completion of internal testing, the merchant should contact a Kount Technical Account Manager and request verification that the RIS inquiries are properly formatted. Using test data, Kount will verify if the merchant's RIS Inquiries are properly formatted and result in the appropriate response from Kount. Upon successful completion of this process, the merchant will achieve certification and can move into production.

Milestone 4: Merchant Training

A Technical Account Manager will schedule training for the personnel who will use the system. Ideally, this training should be done concurrently with the testing and certification phase in Milestone 3. Kount has available printable guides, training videos, and additional documentation.

Milestone 5: Risk Management Calibration

Upon certification, the merchant may begin sending live transactions to Kount. During this phase, the Technical Account Manager will work with the merchant to evaluate the transaction data and fine tune the risk settings to meet the merchant's requirements. Direct action taken on orders may begin upon certification or at the merchant's discretion.

Step 6: Maintenance Milestone

The Technical Account Manager will continue to monitor performance, make periodic recommendations, and be available for consultation as needed.