

ZebraTester

Executing Load Tests with Google Protocol Buffer (protobuf) Messages



Table of Contents

1	Introduction	3
1.1	Full Data Integration.....	3
2	Postprocessing the Recorded Session.....	4
2.1	Automatic Detection of Protobuf Messages.....	4
2.2	Loading the Protobuf FileDescriptorSet (*.desc file) into ZebraTester	5
2.3	Configuring the Message Type for all Protobuf Requests and Responses	7
2.4	Extracting and Assigning Variables from/to Protobuf Message Fields	12
3	Generating the Load Test Program and Executing the Load Test	16
4	Manufacturer, Sales and Support.....	17

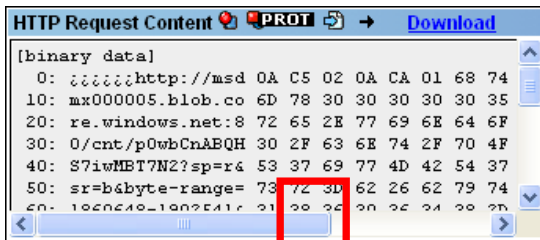
1 Introduction

Wikipedia: "**Protocol Buffers** are a method of serializing structured data, developed by Google Inc.

The design goals for Protocol Buffers emphasized simplicity and performance. In particular, it was designed to be smaller and faster than XML.

Protocol Buffers are widely used at Google for storing and interchanging all kinds of structured information. Protocol Buffers serve as a basis for a custom remote procedure call (RPC) system that is used for nearly all inter-machine communication at Google."

From a tester's view, Protocol Buffers (**protobuf**) is just another way of serializing data, like XML and JSON are. However, in opposite to XML, the "names" of the data fields and the name of the data record type (message type) are not included into the transmitted message, meaning that the (binary) bytes of the data field values are transmitted only – without any metadata information.

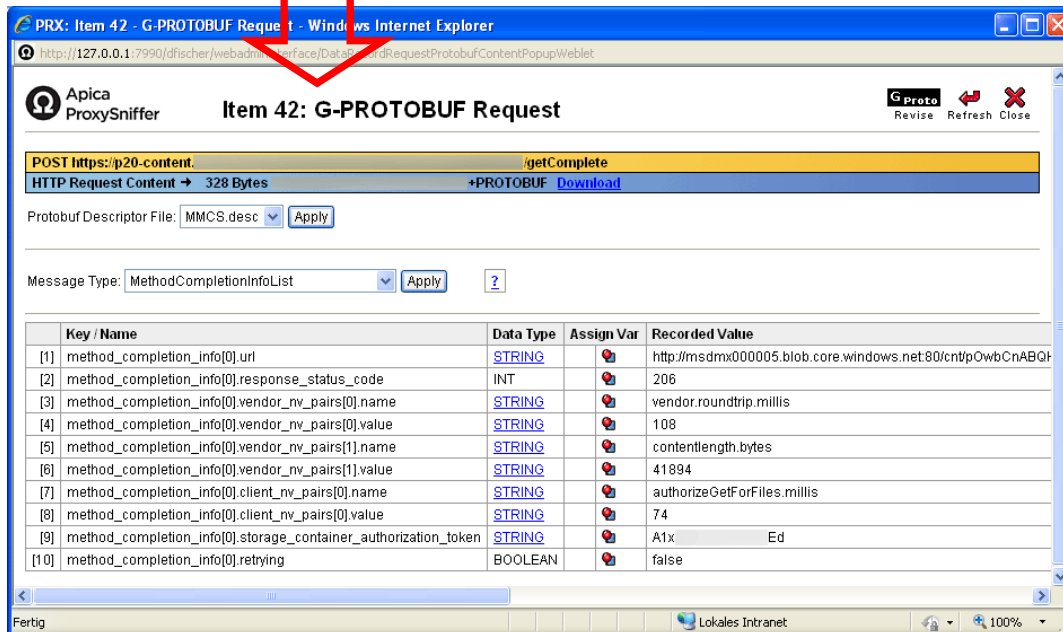


```

[binary data]
0: 00000000http://msd 0A C5 02 0A CA 01 68 74
10: mx000005.blob.co 6D 78 30 30 30 30 30 35
20: re.windows.net:8 72 65 2E 77 69 6E 64 6F
30: 0/cnt/p0wbCnABQH 30 2F 63 6E 74 2F 70 4F
40: S7iwMBT7N2?sp=r4 53 37 69 77 4D 42 54 37
50: sr=b&byte-range= 73 72 3D 62 26 62 79 74
60: 180648-18065416 31 28 26 28 26 24 28 2D
  
```

Therefore, when recording protobuf messages with ZebraTester you will see in the GUI at a first step unstructured binary data only.

Starting from ZebraTester version 5.1, you can load the metadata information for protobuf messages into ZebraTester, by loading one or more Google Protobuf FileDescriptorSets (*.desc files). After that you have to select the corresponding "protobuf message type" and, starting from this point, you can handle and post-process protobuf messages in an easy and convenient way in ZebraTester.



Item 42: G-PROTOBUF Request

POST https://p20-content. getComplete

HTTP Request Content → 328 Bytes +PROTOBUF Download

Protobuf Descriptor File: MMCS.desc Apply

Message Type: MethodCompletionInfoList Apply ?

Key / Name	Data Type	Assign Var	Recorded Value
[1] method_completion_info[0].url	STRING		http://msdmx000005.blob.core.windows.net:80/cnt/p0wbCnABQH
[2] method_completion_info[0].response_status_code	INT		206
[3] method_completion_info[0].vendor_nv_pairs[0].name	STRING		vendor.roundtrip.millis
[4] method_completion_info[0].vendor_nv_pairs[0].value	STRING		108
[5] method_completion_info[0].vendor_nv_pairs[1].name	STRING		contentlength.bytes
[6] method_completion_info[0].vendor_nv_pairs[1].value	STRING		41894
[7] method_completion_info[0].client_nv_pairs[0].name	STRING		authorizeGetForFiles.millis
[8] method_completion_info[0].client_nv_pairs[0].value	STRING		74
[9] method_completion_info[0].storage_container_authorization_token	STRING		A1x Ed
[10] method_completion_info[0].retrying	BOOLEAN		false

1.1 Full Data Integration

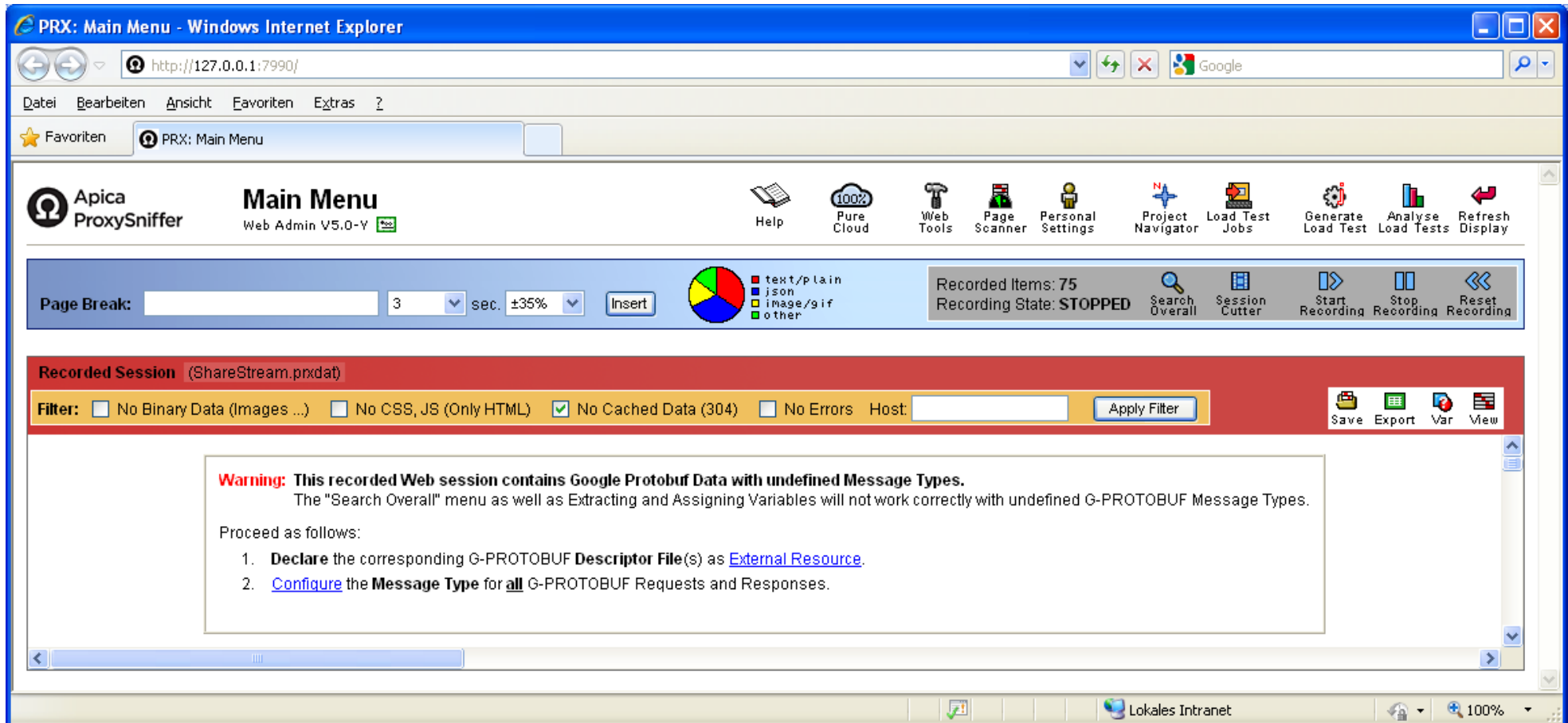
ZebraTester provides full data integration for Protocol Buffers, including:

- Parsing the binary protobuf data and showing them in a human-readable view.
- Extracting and Assigning session variables from/to protobuf data fields, by interpreting and with protecting the binary protobuf message format.
- Searching any ASCII text-fragments over the whole recorded session – and overall error snapshots in the load test result – by interpreting and (temporary) converting the binary protobuf data fields to their corresponding ASCII text presentation.
- Automated distribution of the Protobuf FileDescriptorSets (*.desc files) to the load generators.

2 Postprocessing the Recorded Session

2.1 Automatic Detection of Protobuf Messages

After recording a session with ZebraTester all recorded data are automatically checked if they contain protobuf data. This mechanism checks all recorded HTTP request and response and searches for a matching text fragment containing the string "protobuf" in all HTTP header fields named "Content-Type". If any match is found the following warning text is shown in the ZebraTester GUI:



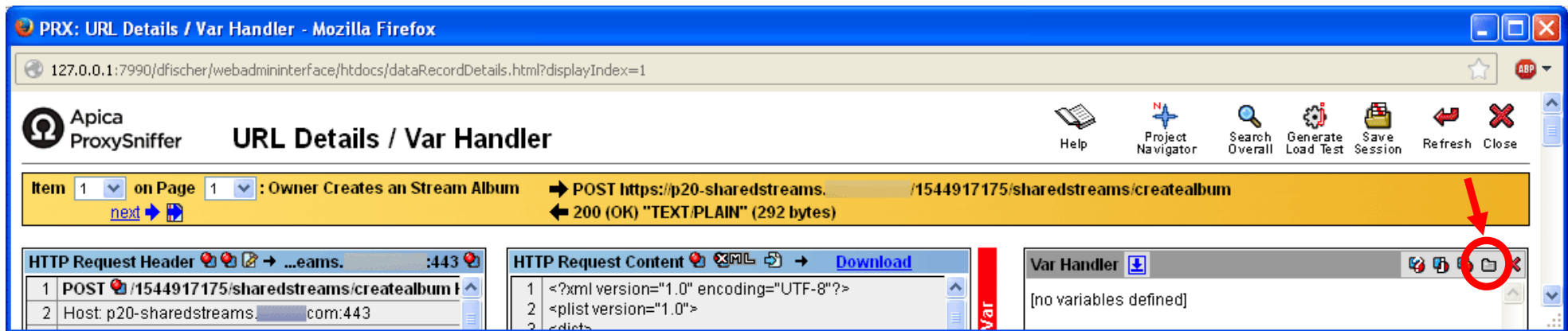
2.2 Loading the Protobuf FileDescriptorSet (*.desc file) into ZebraTester

In order that ZebraTester can interpret the binary protobuf data you have to load the corresponding Protobuf FileDescriptorSet (*.desc file) into ZebraTester, which contains metadata information about the recorded protobuf messages. Such a protobuf *.desc file can be gained from a protobuf source file (*.proto) by using the **protoc** compiler. Example:

```
protoc --descriptor_set_out=addressbook.desc addressbook.proto
```

You have to use the proper *.proto file that matches to the corresponding recorded data. You may ask the development team of the Web application to get the proper *.proto file.

Once you have the protobuf *.desc file you can place it to any directory on your local machine and then register it as an "External Resource" of your recorded session by calling the "Declare External Resources" menu:



PRX: Declare External Resources - Mozilla Firefox

127.0.0.1:7990/dfischer/webadmininterface/PopupConfigExternalResourcesWeblet?action=delete&resourceLocation=local&resourceId=1376637517594

Apica ProxySniffer **Declare External Resources**

Add an External File to Load Test

File Located On: Local System ¹ Remote System (pre installed on Exec Agent) ²

Absolute File Path:

Add File to Java CLASSPATH on the Exec Agent(s) Add File to Java Xbootclasspath/p: on the Exec Agent(s)

¹ Recommended option: Local files are automatically transferred to the load generators when executing a load test.
² Non recommended option: Your load test will not work on cloud-based load generators.

This menu allows you to declare additional, external resources needed when executing a load test. Typically such external resources are **Java library files (*.jar files)** which are used by **self-developed plug-ins**, or **Google Protobuf FileDescriptorSets (*.desc files)** used to parse G-PROTOBUF messages. However, also any other file types can be declared.

Note: Any declarations made in the Var Handler menu such as **Input Files** and **Main Classes of Plug-Ins** are **not external resources** in this context and don't need to be declared because Proxy Sniffer knows already these declarations.

List of all Declared External Resources

Type	Location	CLASSPATH	Xbootclasspath/p:	Absolute File Path
[no external resources declared]				

Note: you should also uncheck the checkbox "Add File to Java CLASSPATH on the Exec Agent(s)"

List of all Declared External Resources

Type	Location	CLASSPATH	Xbootclasspath/p:	Absolute File Path	
	file	local	no	no	D:\ProxySnifferProjectV41\pbuf_lib\MMCS.desc

2.3 Configuring the Message Type for all Protobuf Requests and Responses

In order that the protobuf data can be shown in a human-readable view, and that you can extract and assign variables from/to protobuf messages you have to configure the proper "Message Type" for **all** Protobuf requests and responses. Return to the ZebraTester main menu and click on the second link in the warning text:

Warning: This recorded Web session contains Google Protobuf Data with undefined Message Types.
The "Search Overall" menu as well as Extracting and Assigning Variables will not work correctly with undefined G-PROTOBUF Message Types.

Proceed as follows:

1. **Declare** the corresponding G-PROTOBUF **Descriptor File**(s) as [External Resource](#).
2. **Configure** the **Message Type** for **all** G-PROTOBUF Requests and Responses.

PRX: Revise All G-PROTOBUF Requests and Responses - Mozilla Firefox

127.0.0.1:7990/dfischer/webadmininterface/ProtobufShowAllRequestsAndResponsesWeblet

Apica ProxySniffer **Revise All G-PROTOBUF Requests and Responses** Refresh Close

Item 37: POST https://p20-content.com/ECAEQARoQ67XhT9ucJlza_dwzyMrb0w/authorizeGet		
	Descriptor File	Message Type
HTTP Request	??? [undefined - requires action]	??? [undefined - requires action]
HTTP Response	??? [undefined - requires action]	??? [undefined - requires action]

Item 39: POST https://p20-content.com/ECAEQARoQ67XhT9ucJlza_dwzyMrb0w/getComplete		
	Descriptor File	Message Type
HTTP Request	??? [undefined - requires action]	??? [undefined - requires action]

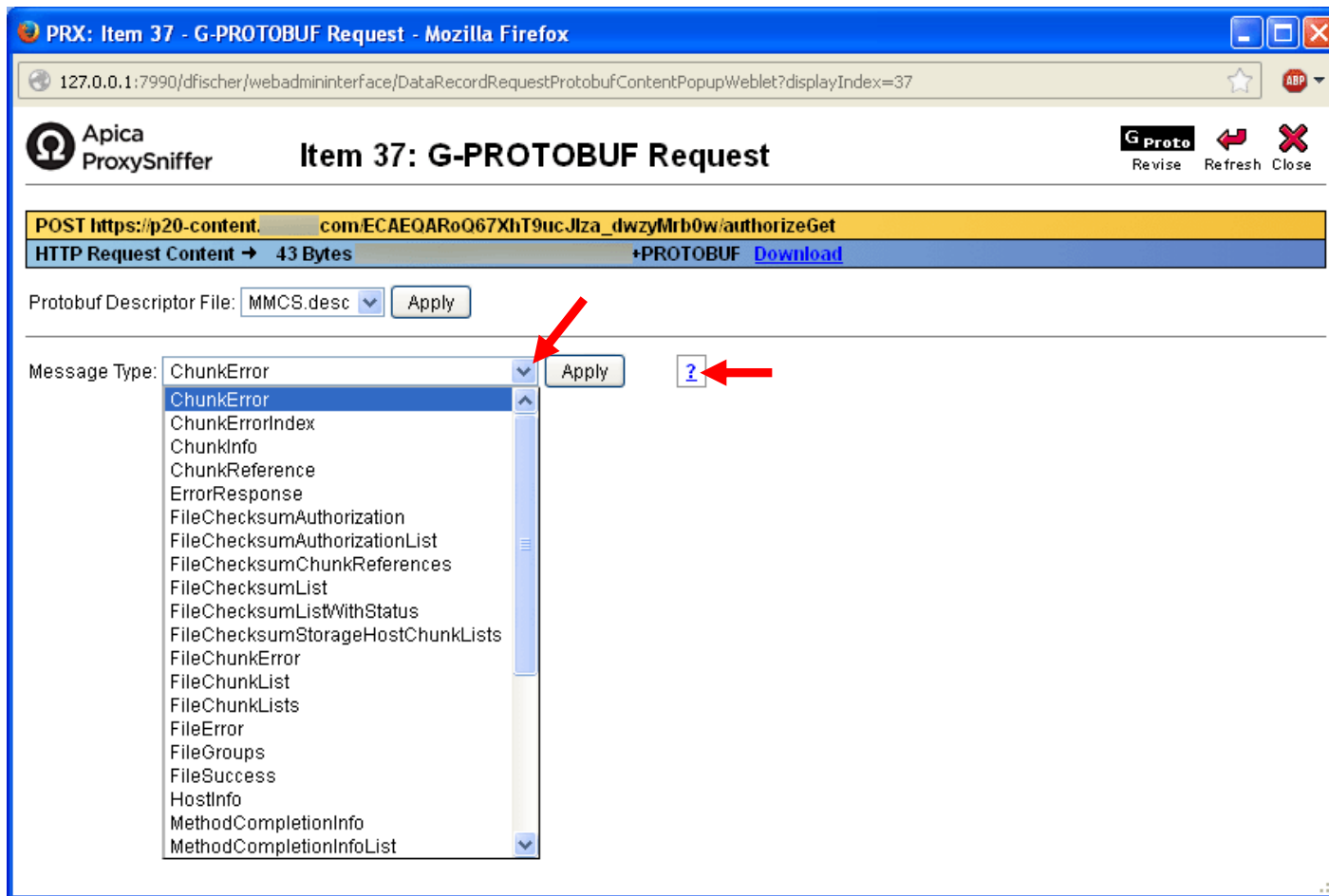
Item 40: POST https://p20-content.com/ECAEQARoQ67XhT9ucJlza_dwzyMrb0w/authorizeGet		
	Descriptor File	Message Type
HTTP Request	??? [undefined - requires action]	??? [undefined - requires action]
HTTP Response	??? [undefined - requires action]	??? [undefined - requires action]

Item 42: POST https://p20-content.com/ECAEQARoQ67XhT9ucJlza_dwzyMrb0w/getComplete		
	Descriptor File	Message Type
HTTP Request	??? [undefined - requires action]	??? [undefined - requires action]

You will see a compilation of all protobuf messages over the whole recorded session.

Click on the magnifier icon of each message to define the "message type".

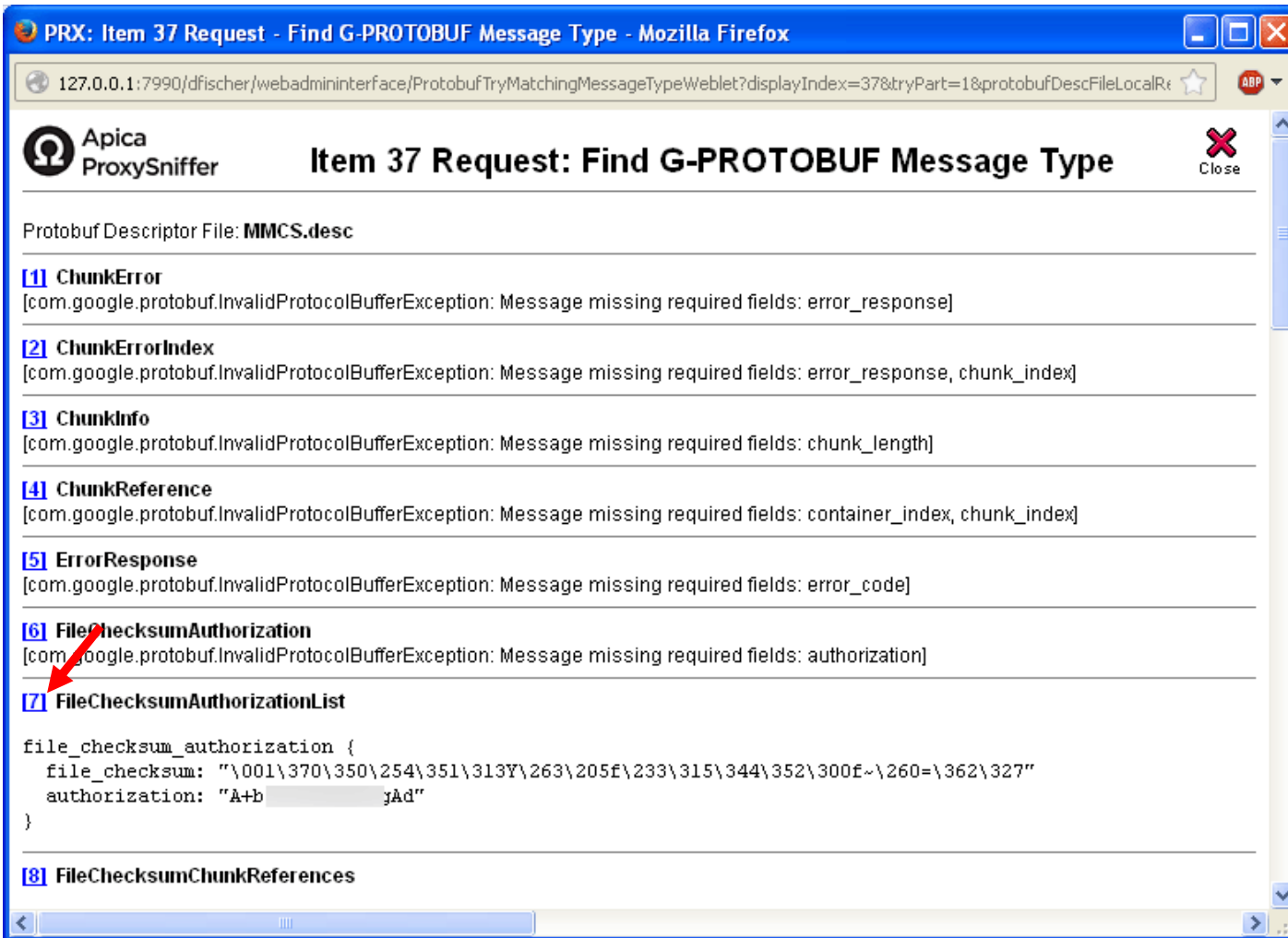
Then you can select the corresponding protobuf "message type" for this HTTP request or response:



If you don't know which "message type" matches to the binary data you can try to guess it by clicking on the ? icon.

However, instead of guessing, best practice is to ask the development team of the Web application for that information – for each HTTP request and response that contains protobuf data.

If you have to guess the "message type" then select the "parsed message-result" that makes the most sense:



PRX: Item 37 Request - Find G-PROTOBUF Message Type - Mozilla Firefox

127.0.0.1:7990/dfischer/webadmininterface/ProtobufTryMatchingMessageTypeWeblet?displayIndex=37&tryPart=1&protobufDescFileLocalR...

Apica ProxySniffer Item 37 Request: Find G-PROTOBUF Message Type Close

Protobuf Descriptor File: **MMCS.desc**

- [1] ChunkError**
[com.google.protobuf.InvalidProtocolBufferException: Message missing required fields: error_response]
- [2] ChunkErrorIndex**
[com.google.protobuf.InvalidProtocolBufferException: Message missing required fields: error_response, chunk_index]
- [3] ChunkInfo**
[com.google.protobuf.InvalidProtocolBufferException: Message missing required fields: chunk_length]
- [4] ChunkReference**
[com.google.protobuf.InvalidProtocolBufferException: Message missing required fields: container_index, chunk_index]
- [5] ErrorResponse**
[com.google.protobuf.InvalidProtocolBufferException: Message missing required fields: error_code]
- [6] FileChecksumAuthorization**
[com.google.protobuf.InvalidProtocolBufferException: Message missing required fields: authorization]
- [7] FileChecksumAuthorizationList**

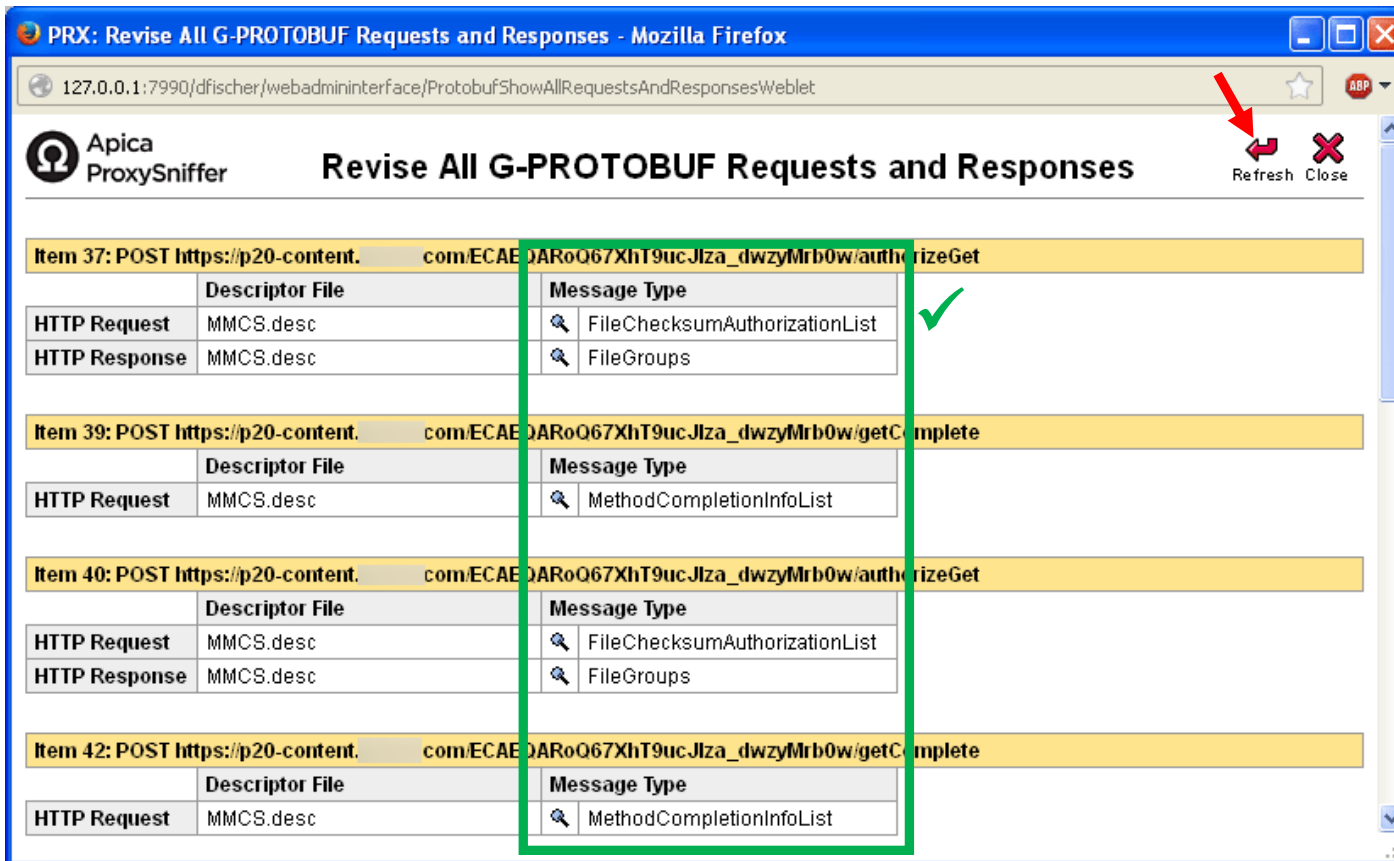
```
file_checksum_authorization {  
  file_checksum: "\001\370\350\254\351\313Y\263\205f\233\315\344\352\300f~\260=\362\327"  
  authorization: "A+b yAd"  
}
```
- [8] FileChecksumChunkReferences**

After you have selected the "message type" you will see the parsed protobuf data:

The screenshot shows the Apica ProxySniffer interface for a G-PROTOBUF Request. The browser window title is "PRX: Item 37 - G-PROTOBUF Request - Mozilla Firefox". The address bar shows the URL: "127.0.0.1:7990/dfischer/webadmininterface/DataRecordRequestProtobufContentPopupWeblet". The main header displays "Item 37: G-PROTOBUF Request" with "G Proto" and buttons for "Revise", "Refresh", and "Close". Below the header, the request details are shown: "POST https://p20-content.com/ECAEQARoQ67XhT9ucJlza_dwzyMrb0w/authorizeGet" and "HTTP Request Content → 43 Bytes +PROTOBUF Download". There are controls for "Protobuf Descriptor File" (set to "MMCS.desc") and "Message Type" (set to "FileChecksumAuthorizationList"). A table below displays the parsed data:

	Key / Name	Data Type	Assign Var	Recorded Value
[1]	file_checksum_authorization[0].file_checksum	BYTE_STRING		[binary data: 21 bytes]
[2]	file_checksum_authorization[0].authorization	STRING		A+b gAd

Return to the "Revise All G-PROTOBUF Requests and Responses" menu and then click on the "Refresh" icon. After that continue to complete the "message types" **for all other** HTTP requests and responses. Finally all "message types" should be defined:



The screenshot shows the Apica ProxySniffer web interface. The title bar indicates the browser is Mozilla Firefox. The address bar shows the URL: 127.0.0.1:7990/dfischer/webadmininterface/ProtobufShowAllRequestsAndResponsesWeblet. The main heading is "Revise All G-PROTOBUF Requests and Responses". There are "Refresh" and "Close" buttons in the top right. A red arrow points to the "Refresh" button. The main content area displays a list of HTTP items, each with a table of "Descriptor File" and "Message Type". A green box highlights the "Message Type" column for items 37, 39, 40, and 42. A green checkmark is visible next to the "FileChecksumAuthorizationList" message type for item 37.

Item	Method	URL	Descriptor File	Message Type
Item 37	POST	https://p20-content.com/ECAE.../authorizeGet	MMCS.desc	FileChecksumAuthorizationList
	HTTP Request		MMCS.desc	FileChecksumAuthorizationList
	HTTP Response		MMCS.desc	FileGroups
Item 39	POST	https://p20-content.com/ECAE.../getComplete	MMCS.desc	MethodCompletionInfoList
	HTTP Request		MMCS.desc	MethodCompletionInfoList
Item 40	POST	https://p20-content.com/ECAE.../authorizeGet	MMCS.desc	FileChecksumAuthorizationList
	HTTP Request		MMCS.desc	FileChecksumAuthorizationList
	HTTP Response		MMCS.desc	FileGroups
Item 42	POST	https://p20-content.com/ECAE.../getComplete	MMCS.desc	MethodCompletionInfoList
	HTTP Request		MMCS.desc	MethodCompletionInfoList

Now you are ready to search any ASCII text fragments overall recorded data and you are ready to extract and assign variables from/to protobuf data fields. At this step it's recommended that you save your recorded session once again.

2.4 Extracting and Assigning Variables from/to Protobuf Message Fields

After you have defined the "message type" for all protobuf messages you can extract and assign variables from/to protobuf message fields – in a standard and convenient way as provided for all other supported data formats:

The screenshot shows the Apica ProxySniffer interface with the following details:

- Item 8**: Content AuthPut → POST https://p20-content.../1544917175/authorizePut
- HTTP Request Content**: G-PROTOBUF Message Type: FileChunkLists


```
file_chunk_list {
  file_checksum: "\001\370\350\254\351\313Y\26
  authorization: "A6g      gAd"
  chunk_info {
    chunk_checksum: "\201p\274\2071\315grFX\314
    chunk encryption key: "\0018+\275\244G9D\2
```
- HTTP Response Content**: G-PROTOBUF Message Type: StorageContainerChunkLists


```
storage_container_chunk_list {
  storage_container_key: "p0v      N2"
  host_info {
    hostname: "msdmx000005.blob.core
    port: 443
```
- Var Handler**: A list of variables including albumName, Content_Token_1, Content_Token_2, ContentServerURL, owner_album_cTag, owner_album_cTag_number, owner_album_cTag_only, and owner_album_location.

Annotations in the image include:

- A red box around the **HTTP Request Content** with a vertical label "Assign var" and an arrow pointing to the "authorization" field.
- A red box around the **HTTP Response Content** with a vertical label "Extract Var" and an arrow pointing to the "storage_container_key" field.

PRX: Item 8 - G-PROTOBUF Response - Mozilla Firefox

127.0.0.1:7990/dfischer/webadmininterf.../DataRecordResponseProtobufContentPopupWeblet?displayIndex=8

Apica ProxySniffer **Item 8: G-PROTOBUF Response** G Proto Revise Refresh Close

POST https://p20-content...m/1544917175/authorizePut

HTTP Response Content ← 587 Bytes •PROTOBUF Download

Protobuf Descriptor File: MMCS.desc Apply

Message Type: StorageContainerChunkLists Apply ?

	Key / Name	Data Type	Extract Var	Recorded Value
[1]	storage_container_chunk_list[0].storage_container_key	STRING		pO N2
[2]	storage_container_chunk_list[0].host_info.hostname	STRING	vendor_hostName	m
[3]	storage_container_chunk_list[0].host_info.port	INT		443
[4]	storage_container_chunk_list[0].host_info.method	STRING		PUT
[5]	storage_container_chunk_list[0].host_info.uri	STRING	vendor_host_uri	/cnt/pOwl iT7N2?sp=w&sr=b&se=2013-08-1
[6]	storage_container_chunk_list[0].host_info.transport_protocol	STRING		HTTP
[7]	storage_container_chunk_list[0].host_info.transport_protocol_version	STRING		1.1
[8]	storage_container_chunk_list[0].host_info.scheme	STRING		https
[9]	storage_container_chunk_list[0].host_info.headers[0].name	STRING		x-ms-client-request-id
[10]	storage_container_chunk_list[0].host_info.headers[0].value	STRING		AD4 \0F
[11]	storage_container_chunk_list[0].host_info.headers[1].name	STRING		x-ms-blob-type
[12]	storage_container_chunk_list[0].host_info.headers[1].value	STRING		BlockBlob
[13]	storage_container_chunk_list[0].host_info.headers[2].name	STRING		x-ms-version
[14]	storage_container_chunk_list[0].host_info.headers[2].value	STRING		2012-02-12
[15]	storage_container_chunk_list[0].host_info.headers[3].name	STRING		Content-Length
[16]	storage_container_chunk_list[0].host_info.headers[3].value	STRING		1902542
[17]	storage_container_chunk_list[0].host_info.headers[4].name	STRING		x-ms-date
[18]	storage_container_chunk_list[0].host_info.headers[4].value	STRING		Thu, 08 Aug 2013 22:27:30 GMT
[19]	storage_container_chunk_list[0].host_info.headers[5].name	STRING		Content-Type
[20]	storage_container_chunk_list[0].host_info.headers[5].value	STRING		application/octet-stream
[21]	storage_container_chunk_list[0].host_info.datacenter_name	STRING		ms_us_dmx
[22]	storage_container_chunk_list[0].chunk_checksum[0].	BYTE_STRING		[binary data: 21 bytes]
[23]	storage_container_chunk_list[0].chunk_checksum[1].	BYTE_STRING		[binary data: 21 bytes]
[24]	storage_container_chunk_list[0].storage_container_authorization_token	STRING	vendor_authToken	Az Ed
[25]	verbosity_level	INT		2

The screenshot shows the Apica ProxySniffer interface with the following details:

- Item 8** on Page 2: Content AuthPut → POST https://p20-content.v/1544917175/authorizePut
- HTTP Request Header:**
 - 1 POST /1544917175/authorizePut HTTP/1.1
 - 2 Host: p20-content 443
 - 3 User-Agent: .
 - 4 : 3.9
 - 5 Dataclass.Share
 - 6 Accept: -protok
- HTTP Request Content:** G-PROTOBUF Message Type: FileChunkLists


```
file_chunk_list {
  file_checksum: "\001\370\350\254\351\313Y\26
  authorization: "A6c Ad"
  chunk_info {
    chunk_checksum: "\201p\274\2071\315gfX\314
    chunk_encryption key: "\0018+\275\244G9\2
```
- HTTP Response Header:**
 - 1 HTTP/1.1 200 OK
 - 2 Date: Mon, 12 Aug 2013 22:53:02 GMT
 - 3 X- 1-3922-4EF5-B6EE-A
 - 4 x- 3.9
 - 5 X-Responding-Instance: content.22000502.nk11p20ic-c
 - 6 Content-Type: application +prc
- HTTP Response Content:** G-PROTOBUF Message Type: StorageContainerChunkLists

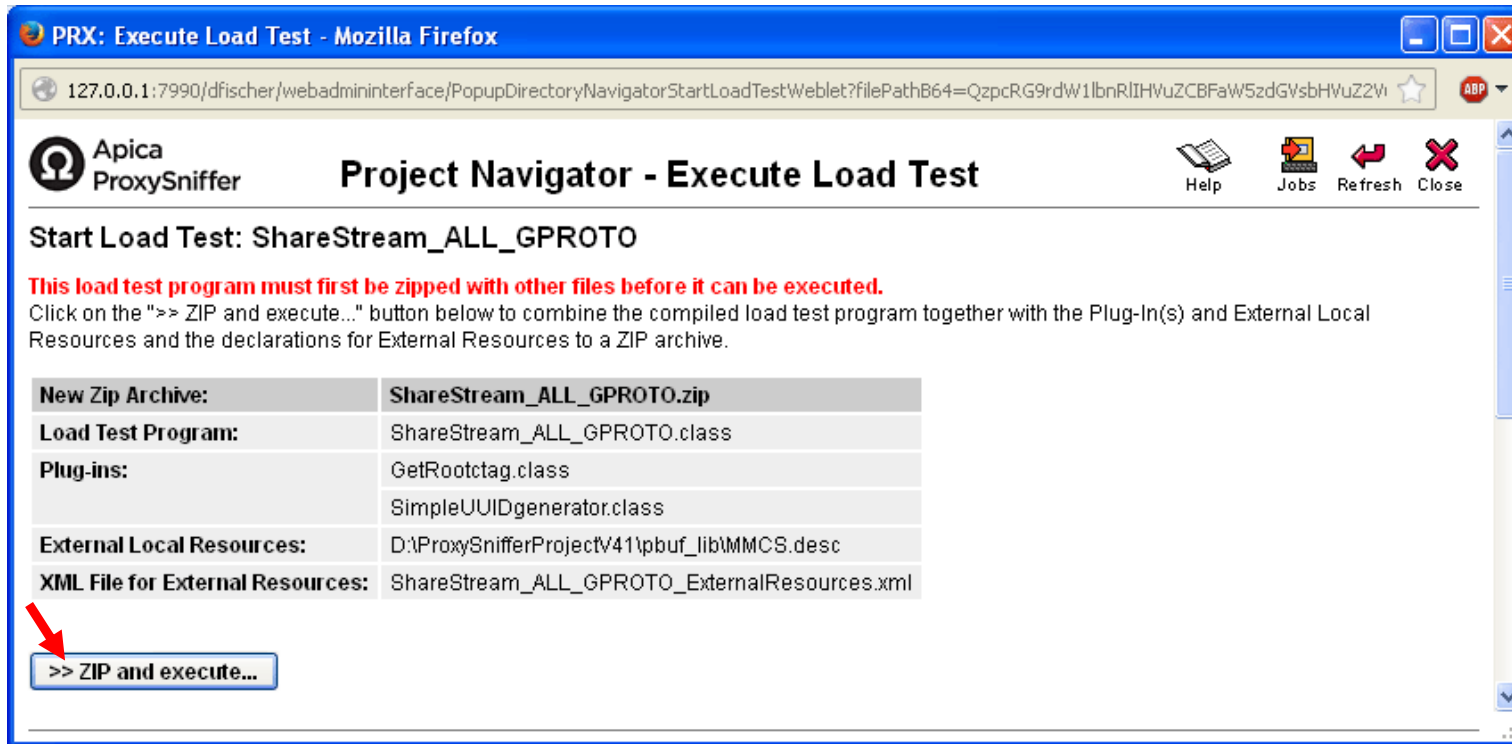

```
storage_container_chunk_list {
  storage_container_key: "p0w 7N2"
  host_info {
    hostname: "m"
    port: 443
```
- Var Handler:**
 - Extract Var from G-PROTOBUF Data
 - Protobuf File: MMCS.desc
 - Message Type: StorageContainerChunkLists
 - Key / Name: storage_container_chunk_list[0].storage_conta
 - Data Type: STRING
 - Recorded Value: 'Az0 Ed'
 - Extract whole Value (selected)
 - Token Delimiters: <>"'&?
 - Extract Token No.: 1
 - Trim Whitespaces: yes
 - Test Extract
 - Extracted Value: 'Az0 Ed'
 - Map to Var Name: vendor_authToken
 - Assign Var automatically to all HTTP requests which contain the same text pattern (full replacement of Extracted Value overall requests)
 - Extract

The screenshot shows the 'Var Handler' window with the following content:

- 38 HTTP Request Header Field Pattern
- 39 G-PROTOBUF Request Data
- [- vendor_authToken [loop var]**
 - ← 8 G-PROTOBUF Response Data: [Az0](#)
 - 16 G-PROTOBUF Request Data
- [- vendor_host_uri [loop var]**
 - ← 8 G-PROTOBUF Response Data: [/cnt/pOwbCnABQHS](#)
- [- vendor_hostName [loop var]**
 - ← 8 G-PROTOBUF Response Data: [m:](#)
- [- vis_album_state_ctag [loop var]**
 - ← 25 XML Response Data: [FT=-@RU=8E](#)
 - 31 HTTP Request Content Pattern
 - 44 HTTP Request Content Pattern

3 Generating the Load Test Program and Executing the Load Test

Generating the load test program and executing the load test can be done as usual – without any special additional configuration. All required resources are automatically combined to a ZIP archive and are automatically distributed to the load generators.



4 Manufacturer, Sales and Support

Ingenieurbüro David Fischer AG, Switzerland | A company of the [Apica Group](#)

Manufacturer's Web Site: <http://www.zebratester.com/>

Apica Support: support@apicasystem.com

Apica Sales: sales@apicasystem.com

Note: All menus provide *context specific help text*, available using the Help Icon:

The screenshot shows a web browser window titled "Project Navigator - Execute Load Test". The address bar shows a long URL. The main content area is titled "Execute Load Test Job: LOAD_20070612". Below this is a "Load Test Input Parameter" section with various settings: "Execute Test from" (Cluster: cluster1), "Number of Concurrent Users" (1), "Load Test Duration" (1 min), "Max. Loops per User" (unlimited), "Startup Delay per User" (200 Milliseconds), "Max. Network Bandwidth per User" (unlimited), "Request Timeout per URL" (60 Seconds), "Max. Error-Snapshots per URL" (30), "Statistic Sampling Interval" (15 Seconds), "Additional Sampling Rate per Page Call" (100%), "Additional Sampling Rate per URL Call" (---), "Debug Options" (none - recommended), and "Additional Options" (SSL | v2/v3/TLS). At the bottom, there is an "Annotation" field and a ">> Continue" button. A red arrow points to the "Help" icon in the top right corner of the window.