The Implications of MRC Measurement Standards on Webcast Metrics ®

Presented By Hugo Martel

May, 2019

Objectives

- Provide an understanding of how Triton Digital's Webcast Metrics defines a session, and why integrating that logic into your player or app workflow is critical for accurate listening measurement

- To create an open dialogue between Triton Digital and Player providers on how to address today's challenges around the measurement of digital audio

Agenda

- Triton Measurement Services and Standards
- Common Challenges in the Digital Audio Industry & How to Address Them
- Triton Measurement Guide (Web Player & Application Guidelines)
- Log-Based vs Client-Side Measurement

Triton Digital's Measurement Services

Webcast Metrics, Webcast Metrics Local, Rankers

- The Webcast Metrics streaming measurement service is the industry standard for online audio consumption data.

Podcast Metrics

- Our Podcast Metrics measurement product provides insightful data into the consumption of podcast content, measuring downloads & listeners

MRC - Digital Audio Measurement Standards

Released in January 2018

- Covers ad and content audience measurement
- Listening session times shall match exactly what appears in session logs
- Stop, Pause & Mute actions shall be taken into consideration
- Prefetch, duplicates & other phantom streams shall be removed from measurement
- Requires user actions: Auto-Play and Auto-Refresh are discouraged
- No players that keep running without human interaction are accepted

Common Challenges in the Digital Audio Industry & How to Address Them

Popular Audio “Streaming” Protocols

Radio / Live streaming - files
- .mp3 or .aac “infinite” files
- Streaming servers are logging the connection start and end time.

Radio / Live Streaming – HLS
- Should be sessions log not chunks log
- matching the opened connections with the m3u8 playlist

On-Demand Music – Song file downloads or progressive downloads
- Sessions shall be created from client side listener tracking

On-Demand (Talk/Music/Radio) – HLS
- Should be sessions log not chunks log
- matching the opened connections with the m3u8 playlist
- Or sessions can be created from client side listener tracking

Podcast – MP3 files
- For WCM: Sessions shall be created from client side listener tracking
- For PCM: Files access logs are perfect.

Common Challenges in the Digital Audio Space

Example 1: “Multi Streams”

- Session starts at 2:00:00 PM, stops at 2:06:00 PM
- Streaming server session logs:
  - A 6-minute session from 2:00:00 to 2:06:00 (An invalid prefetch call)
  - Another 6-minute session, 2:00:01 to 2:06:00 (The valid one)

Why 2 sessions?
The OS takes control of the playback, but the browser also opens the audio stream (file). Although we have procedures in place to invalidate this type of traffic, we expect players to address the issue as well.

How to fix it? Avoid preload

Preferred:
The **<audio>** element’s **preload** attribute should be set to **none**. It should never be set to **auto** or left **blank**, as those options transfer the decision to the browser itself

```
<audio controls preload="none" height="32" width="300" onpause="location.reload();" src="<PLAYER PAGE URL>" type="audio/mpeg" />
```

Common Challenges in the Digital Audio Space

Example 2: "Stream doesn't stop until the App or browser is closed"

- Session starts at 2:00:00 PM, stops at 2:06:00 PM, App closes at 2:10:00 PM
- Streaming server session logs:

  A 10-minute session from 2:00:00 PM to 2:10:00 PM

Why an extra 4 minutes?
The Stop didn't stop the stream, so the streaming server isn't aware and can't log the information.

How to fix it?

1. Stop instead of Pause (See HTML 5 doesn't have a stop)
2. Don't use the browsers default player

Common Challenges in the Digital Audio Space

Example 3: “Pause and buffering”

- Session starts at 2:00:00 PM, pauses at 2:01:00 PM, resumes at 2:10:00 and stops at 2:12:00 PM

- Streaming server session logs:
  A 12-minute session from 2:00:00 PM to 2:12:00 PM

Why an extra 9 minutes?
The player buffers the stream, and doesn’t inform the streaming server.

How to fix it?

1. Stop instead of Pause (See HTML 5 doesn’t have a stop slide)

2. Listener Tracking (LT)

Common Challenges in the Digital Audio Space

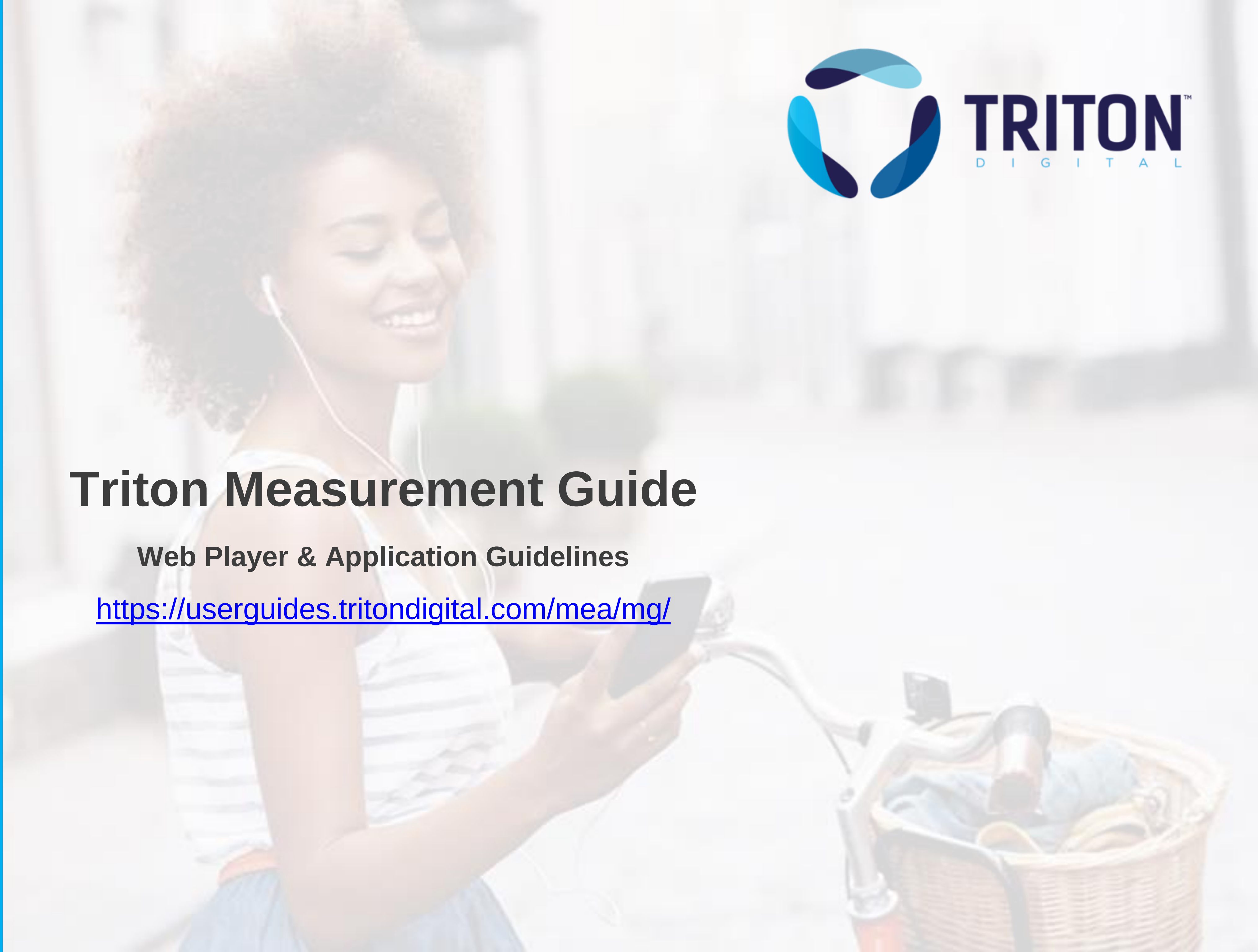
Example 4: “Pause and Re-Start”

- Session starts at 2:00:00 PM, pauses at 2:01:00 PM, resumes at 2:10:00, pauses again at 2:12:00 PM, then the app/browser is closed at 2:14:00
- Streaming server session logs:
  A 10-minute session from 2:00:00 PM to 2:10:00 PM
  A 4-minute session from 2:10:00 PM to 2:14:00 PM

Why an extra 11 minutes?
The player buffers the stream, and doesn’t inform the streaming server.
The player doesn’t resume, but starts a new stream

How to fix it?

1. Stop instead of Pause (See HTML 5 doesn’t have a stop slide)
2. Don’t use the browsers default player



Triton Measurement Guide

Web Player & Application Guidelines

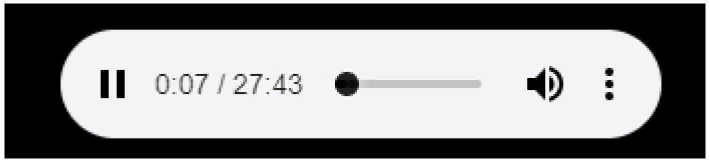
https://userguides.tritondigital.com/mea/mg/

Direct Links on Web Browser

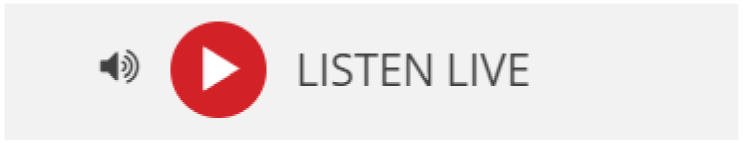
- Web Players must never open streaming MP3 or AAC URLs directly in the web browser internal players.
- Although the browser may use its default player to play the audio content, it is known that these internal players may produce invalid traffic.

https://website.com/files/podcastXYZ.mp3

NO



YES



HTML5 doesn’t have a “Stop”

- The HTML5 <audio> element does not have a true "stop" function, only a "pause.“
- Some live streams – particularly Icecast – are seen by the browser as "a large file”
- Downloading may continue in the background when the stream is paused
- Additional actions to halt the stream/file download might be required

The action can be:

- Audio.src=“” .

  Find the <audio> element: var a = document.getElementsByTagName("audio")[0];
  * Set the src to empty: a.src = ""; * Reload the audio element: a.load();

- Play a short silent audio track.

- Location.reload(). A reload of the page containing the player. (This is and easy solution that works well in all situations but it might not always be appropriate for all)

- Load a blank (""), or "About:blank" value into the source attribute.

No way to pause/stop on Safari 12

The Safari 12 bug

Summary:
The issue is that when you have an <audio> element with a shoutcast/icecast or any other live stream source it keeps on downloading after the pause is clicked, and the stream src is set to empty and the currentTime is set to 0.0 and the <audio> element is actually removed from the DOM.
So there is no way to stop the network resource from downloading the stream, exception to actually do a "window.stop()" command.

Expected Results:
* After pausing the stream and setting the audio src="" and calling load(), then I expect the stream to stop downloading.
* This is the actual result in Safari versions < 12.

Actual Results:
* The stream never stops, it just keeps on downloading.

How to Process Play, Pause, Stop

- Play shall start only one session (Head request, or 0-1 range, 1s pre-fetch are OK)
- The session shall start after the audio/video on-demand pre-roll.
- Pause, Stop should have the same effect on the session: session end
- Resume should start a new session

Auto-play

- Auto-play of audio content is strongly discouraged.
- In some cases, auto-played streams are not audible as the volume may be muted while the stream continues to be active.
- It’s also discouraged by Apple and Google
- Play shall be determined by a user gesture, such as a tap or a click.

Other considerations

Long Sessions
Triton cuts sessions at 24-hours, as it seems many sessions of or exceeding this duration are forgotten streams on desktop computers.
Players should require user interaction after a long period of time. If there is interaction, it should start a new session.

Usage of Registration IDs
When available, please pass a hashed version of an ID (non PII) so that the measurement service can properly use it to de-dup listeners.

Muted/Audibility Requirement
If / when players control the Mute button or receive a callback from the OS, the application should stop the stream.

Validation steps

Series of tests to verify the accuracy of your logs or LT calls

1. Verify if you have any web pages set with auto-play
2. The simple play-stop test: Play 2min, Stop, wait 5min, Close browser or App
3. The pause test: Play 2min, Pause 5min, Play 2min, Stop (or pause), wait 5min, close
4. Repeat the above on Android App / Chrome, iOS App / Safari, Windows Chrome/IE11/Firefox, Mac Safari/Chrome
5. Request Logs to your CDN or to Triton (Keep notes of your IP, Stations tested, exact time of sessions)
6. Analyze & address problems

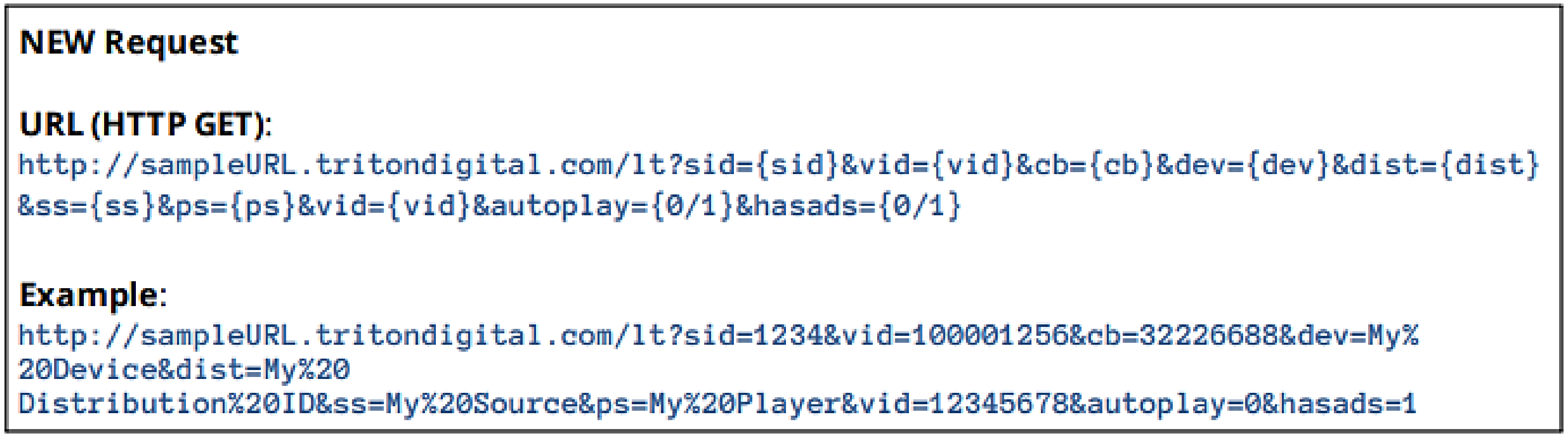
Log-Based vs. Client-Side Measurement

Log-Based vs Client-Side Measurement

Log Collection (LC)	Listener Tracking (LT)
When streaming sessions logs match user actions	When you want to pause, rewind, resume functions.
When Client side measurement isn't possible: Smart Speakers for example	When Client side measurement is possible and preferred
Requires sync between user actions and streaming servers to be accurate	Requires sync between user actions and player implementation of the LT protocol
Doesn't support on-demand content in download or progressive download mode.	Suited for live or on-demand content (music, podcasts, etc.)

Listener Tracking

NEW Request



NEW Request

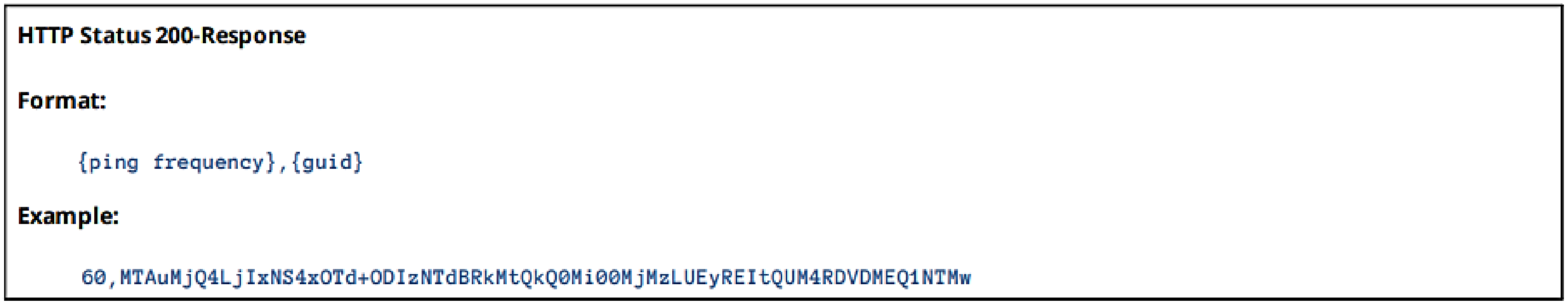
URL (HTTP GET):

```
http://sampleURL.tritondigital.com/lt?sid={sid}&vid={vid}&cb={cb}&dev={dev}&dist={dist}
&ss={ss}&ps={ps}&vid={vid}&autoplay={0/1}&hasads={0/1}
```

Example:

```
http://sampleURL.tritondigital.com/lt?sid=1234&vid=100001256&cb=32226688&dev=My%
20Device&dist=My%20
Distribution%20ID&ss=My%20Source&ps=My%20Player&vid=12345678&autoplay=0&hasads=1
```

LT Response to NEW Request



HTTP Status 200-Response

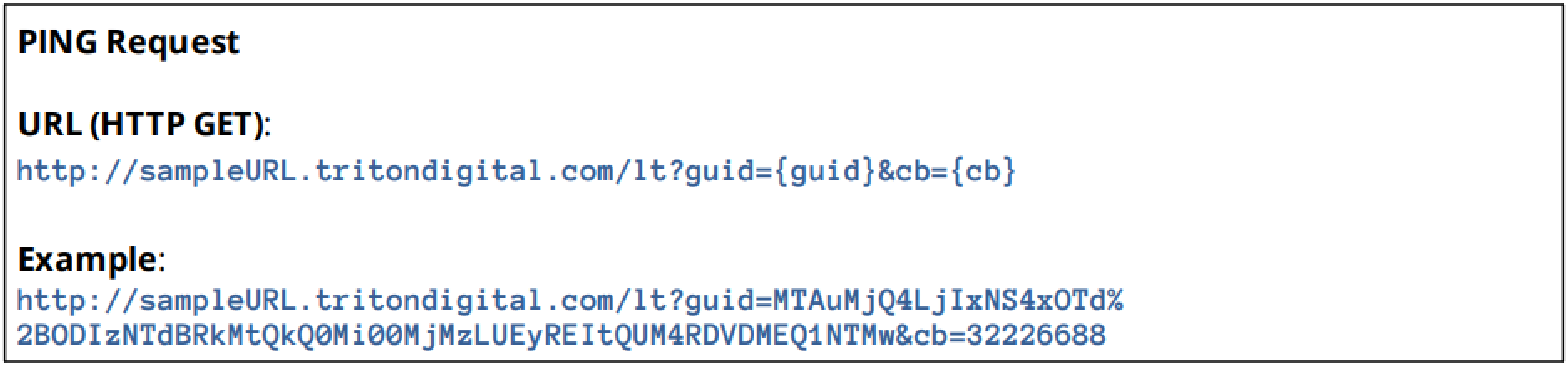
Format:

```
    {ping frequency},{guid}
```

Example:

```
    60,MTAuMjQ4LjIxNS4xOTd+ODIzNTdBRkMtQkQ0Mi00MjMzLUEyREItQUM4RDVDMEQ1NTMw
```

Listener Tracking



PING Request

PING Request

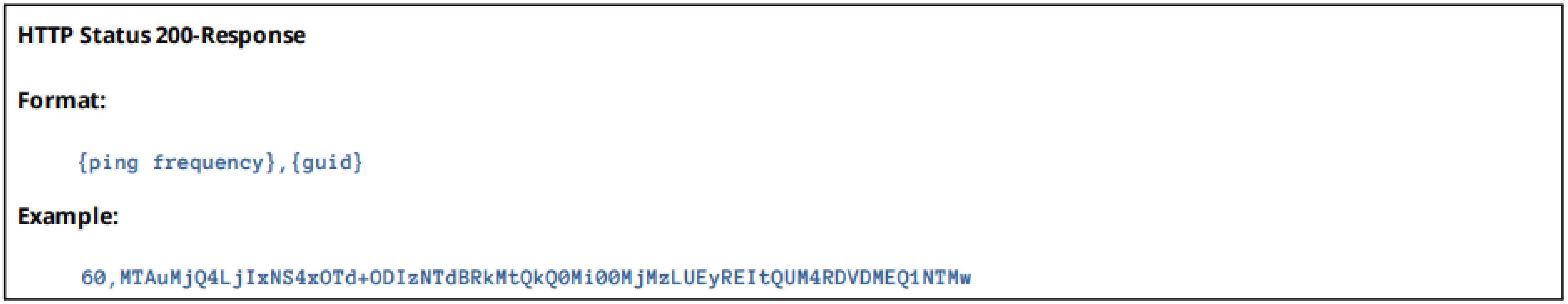
URL (HTTP GET):

```
http://sampleURL.tritondigital.com/lt?guid={guid}&cb={cb}
```

Example:

```
http://sampleURL.tritondigital.com/lt?guid=MTAuMjQ4LjIxNS4xOTd%
2BODIzNTdBRkMtQkQ0Mi00MjMzLUEyREItQUM4RDVDMEQ1NTMw&cb=32226688
```

LT Response to PING Request



HTTP Status 200-Response

Format:

```
    {ping frequency},{guid}
```

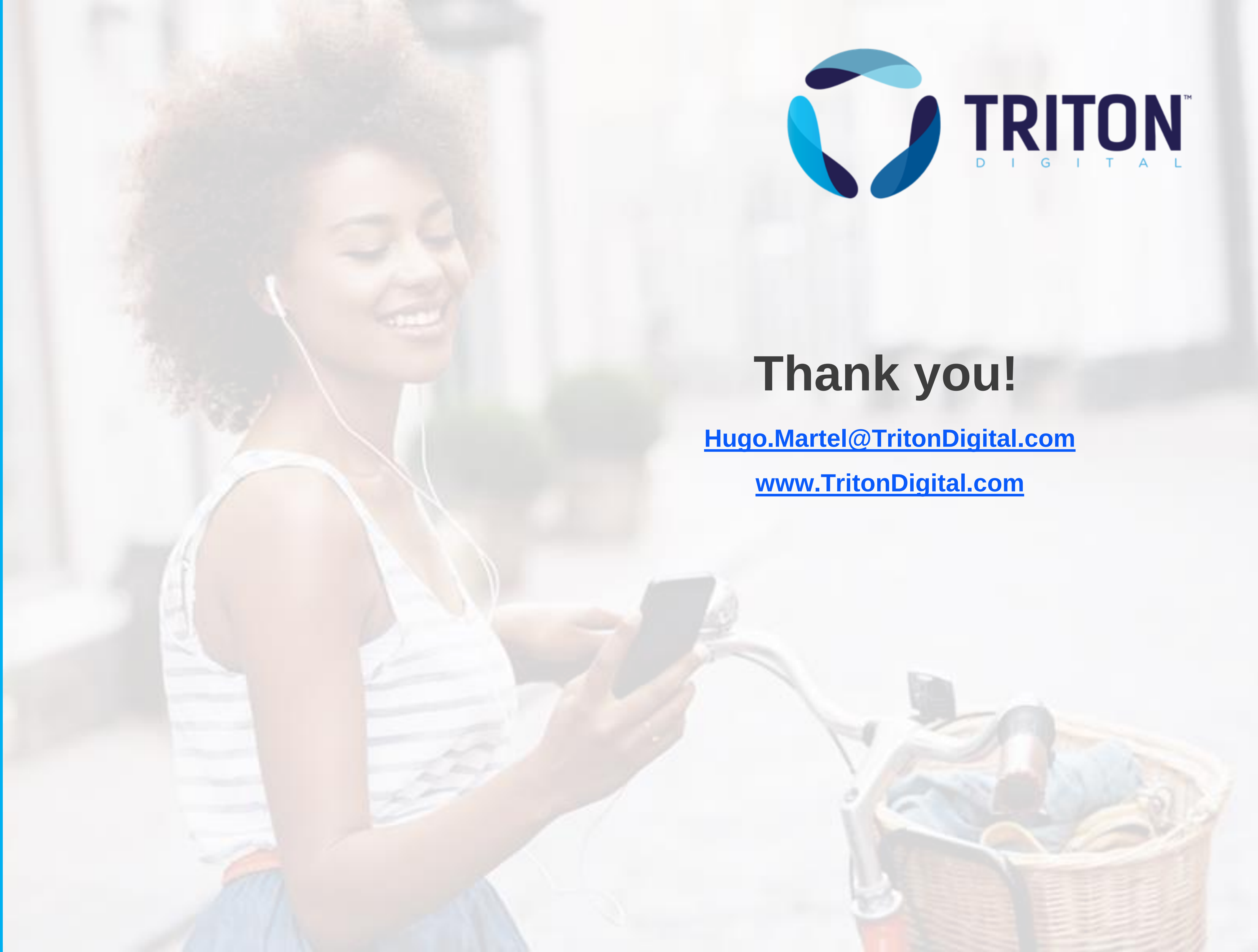
Example:

```
    60,MTAuMjQ4LjIxNS4xOTd+ODIzNTdBRkMtQkQ0Mi00MjMzLUEyREItQUM4RDVDMEQ1NTMw
```

What’s Next?

We invite you read the Triton Digital Measurement Guide, located here:

https://userguides.tritondigital.com/mea/mg/



Thank you!

Hugo.Martel@TritonDigital.com

www.TritonDigital.com