Kevin Gosse @kookiz

Christophe Nasarre

Advanced .NET debugging techniques from real world investigations

AGENDA… of an investigation

1. Identify

Gather meaningful data your can **trust**

2. Understand

Make hypothesis based on **application state** and **experience**

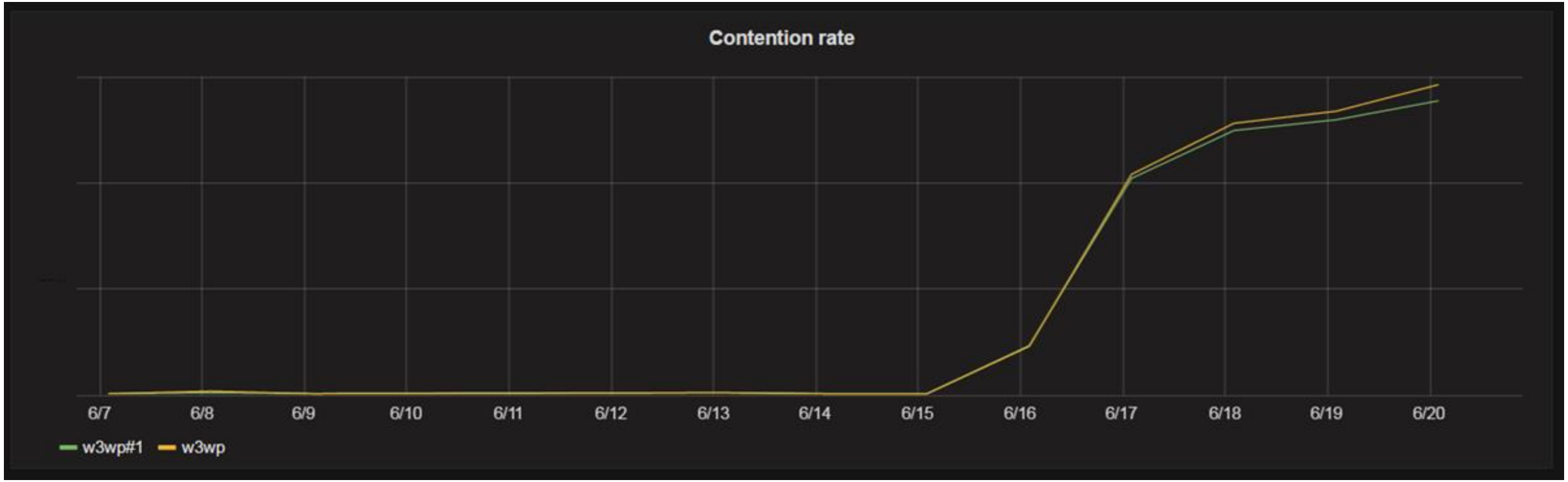
3. Verify

Validate your assertions

 Copyright © 2018 Criteo

 Copyright © 2018 Criteo

First symptom of the issue: increase in contention rate



 Copyright © 2018 Criteo

- GUESS: graph looks like a leak (but of what?)
- Looking for contention cause

 Copyright © 2018 Criteo

Look in code changes

MOAB	Repository	Title
#53746	sdk/configuration	Reorganize + small cleanups of ConfigAsCodeServiceMock
#53745	publisher/criteo-rtb	Revert "Inject CaC service in UserAgent".
#53744	audience/experience-ui	Fix postsubmit script location
#53743	adserving-backend/criteo-xrtupdatetool	Create netcore project for XRTUUpdateTool
#53743	adserving-backend/criteo-xrtupdatetool	Use consul for connection strings jira: WBSC-2781
#53742	banners/criteo-info-web	make sure uid & optout cookies are written when "static=1 redirect" happens
#53741	persistence/user-data	Connect CookieStreamer and CookieStreamWriter to Kafka Stream
#53741	bidding-strategies/external-bidder	Use the minimum bid from PublisherInfo instead of ScopedConfig.
#53741	audience/experience-api	Revert "[AXE-32] Add new STest to call AudienceCore stub"
#53741	audience/experience-api	Revert "[AXE-32] Deploy ExternalServices.Stub in sandbox"

 Copyright © 2018 Criteo

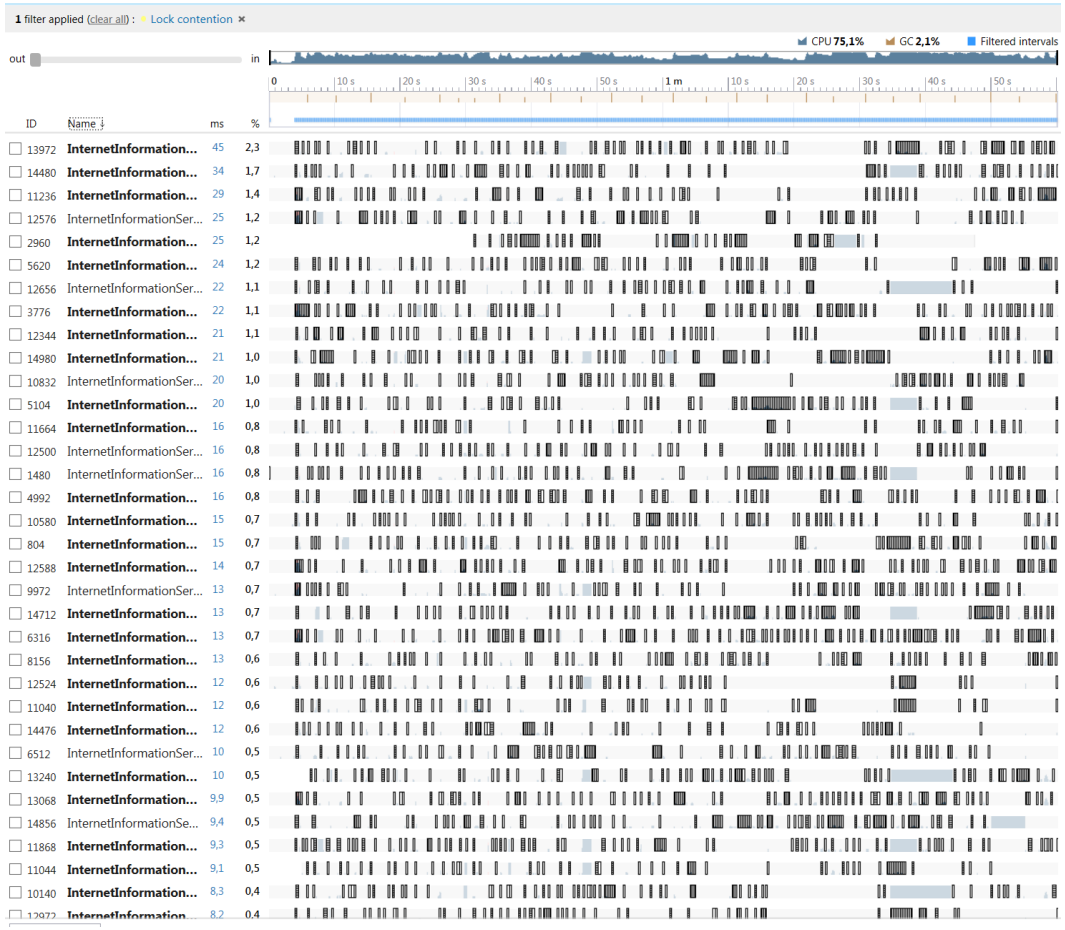
- **GUESS: graph looks like a leak (but of what?)**
- **Looking for contention cause**
  - ~~Nothing in code change~~

 Copyright © 2018 Criteo

criteo

Measure with more details: hard to beat a profiler

Comparing exact contention



 Copyright © 2018 Criteo

criteo

- **GUESS: graph looks like a leak (but of what?)**
- **Looking for contention cause**
  - ~~Nothing in code change~~
  - Sounds related to TimerQueueTimer
    - What is that?

 Copyright © 2018 Criteo

criteo

Know the code but not only your code

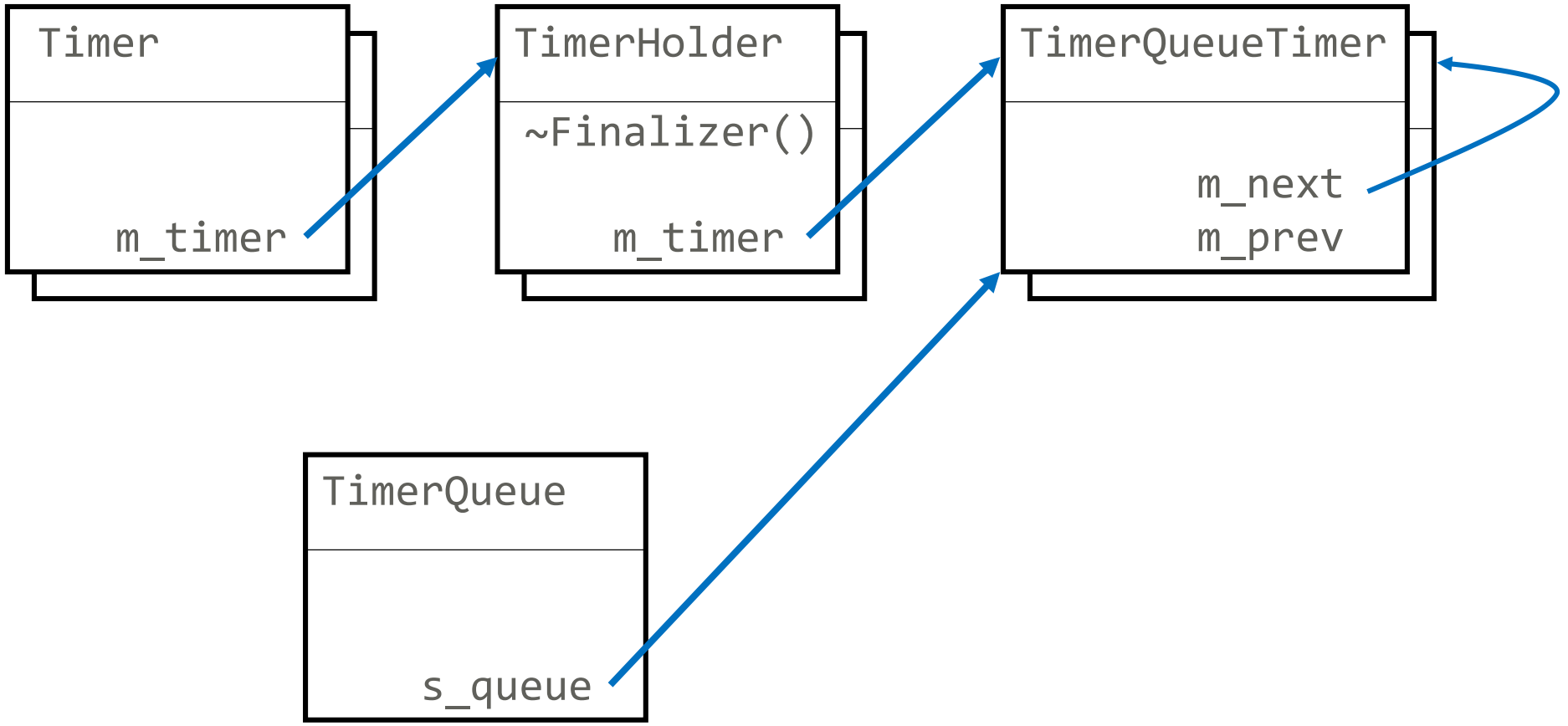
A decompiler is ALWAYS a good friend… even with https://referencesource.microsoft.com

- **Look for the source code if available on sourceof.net or a decompiler**

 Copyright © 2018 Criteo

criteo

Look into the Timer creation code



 Copyright © 2018 Criteo

criteo

- **GUESS: graph looks like a leak (but of what?)**
- **Looking for contention cause**
  - ~~Nothing in code change~~
  - Sounds related to TimerQueueTimer
    - ~~What is that?~~
    - → where do we use Timer?

 Copyright © 2018 Criteo

criteo

procdump + WinDBG + sos

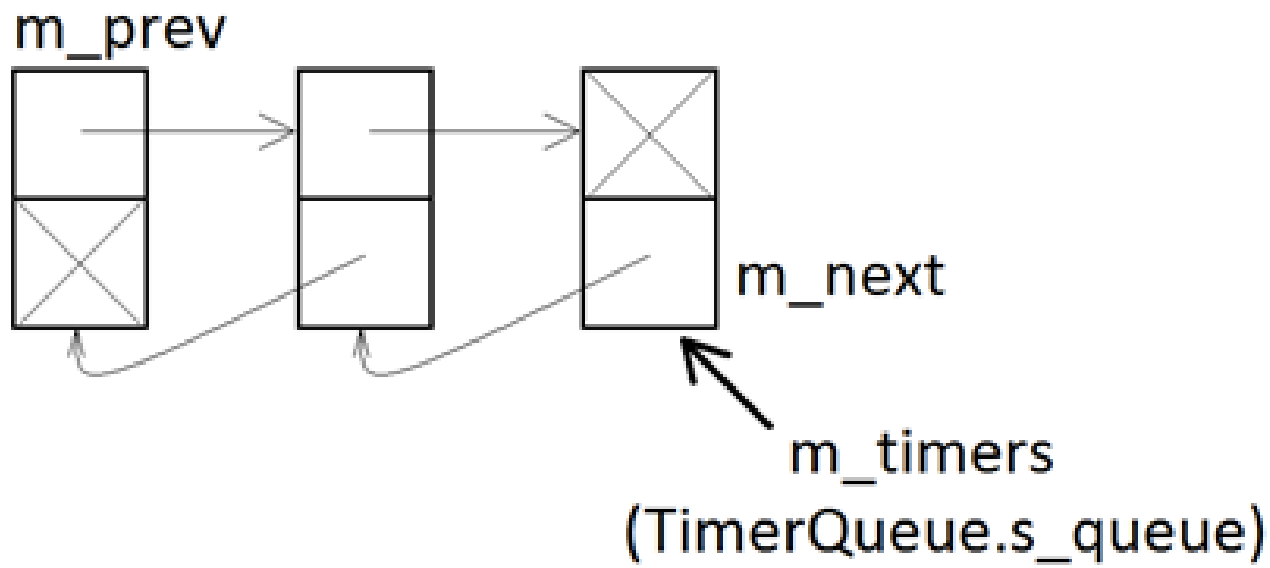
DEMO

 Copyright © 2018 Criteo

- **GUESS: graph looks like a leak (but of what?)**
- **Looking for contention cause**
  - ~~Nothing in code change~~
  - Sounds related to TimerQueueTimer
    - ~~What is that?~~
    - → where do we use 14.000+ Timer?

 Copyright © 2018 Criteo

criteo



DEMO

 Copyright © 2018 Criteo

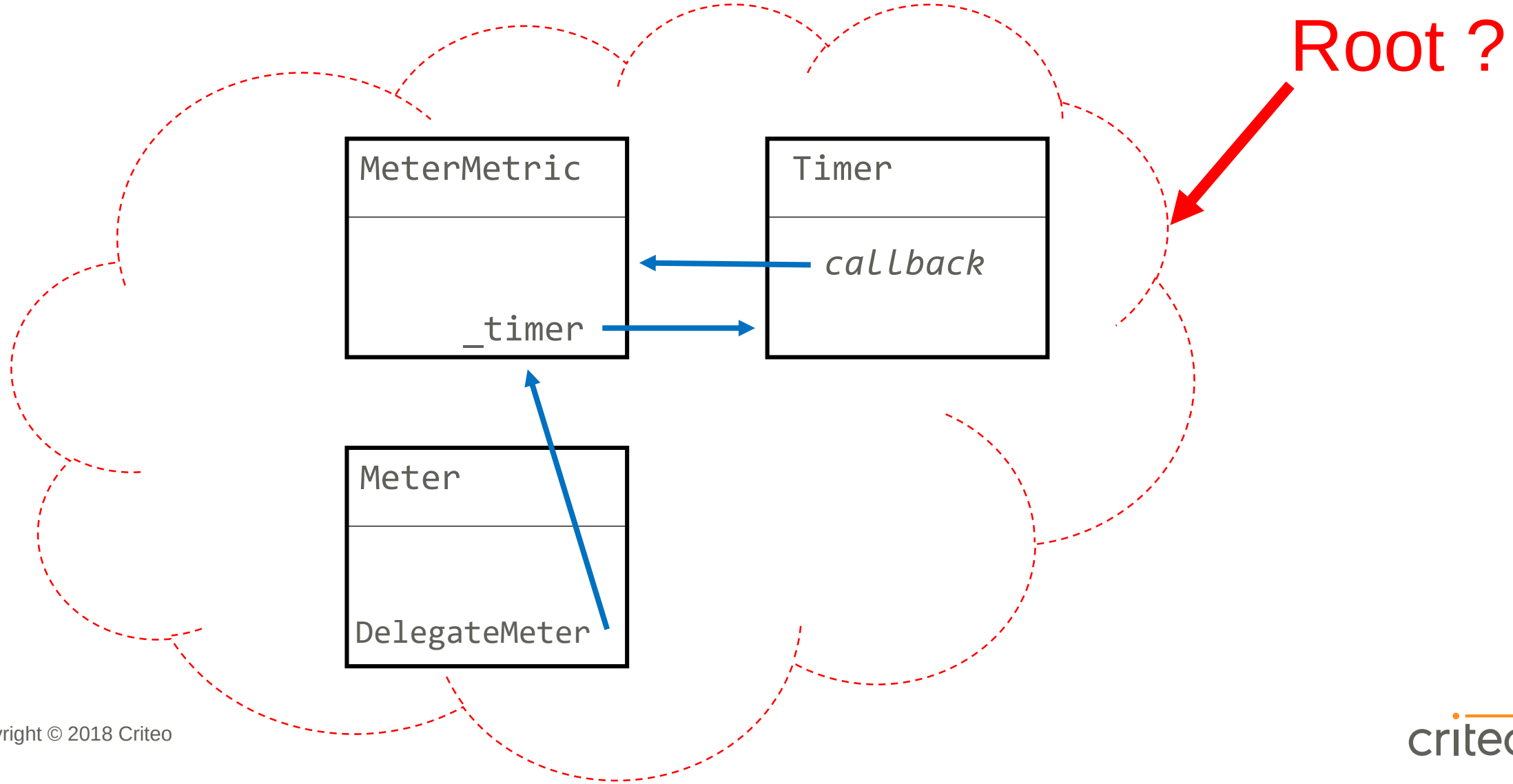
criteo

- **GUESS: graph looks like a leak (but of what?)**
- **Looking for contention cause**
  - ~~Nothing in code change~~
  - Sounds related to TimerQueueTimer
    - ~~What is that?~~
    - ~~Where do we use 14.000+ Timer?~~
  - Relations between MeterMetric and Timer?

 Copyright © 2018 Criteo

criteo

Current analysis state



 Copyright © 2018 Criteo

criteo

WinDBG + sosex.refs: who is holding us?

DEMO

 Copyright © 2018 Criteo

criteo

- **GUESS: graph looks like a leak (but of what?)**
- **Looking for contention cause**
  - ~~Nothing in code change~~
  - Sounds related to TimerQueueTimer
    - ~~What is that?~~
    - ~~Where do we use 14.000+ Timer?~~
  - Relations between MeterMetric and Timer?
    - ~~Meter~~
    - but…finalizers should have done the job?...

 Copyright © 2018 Criteo

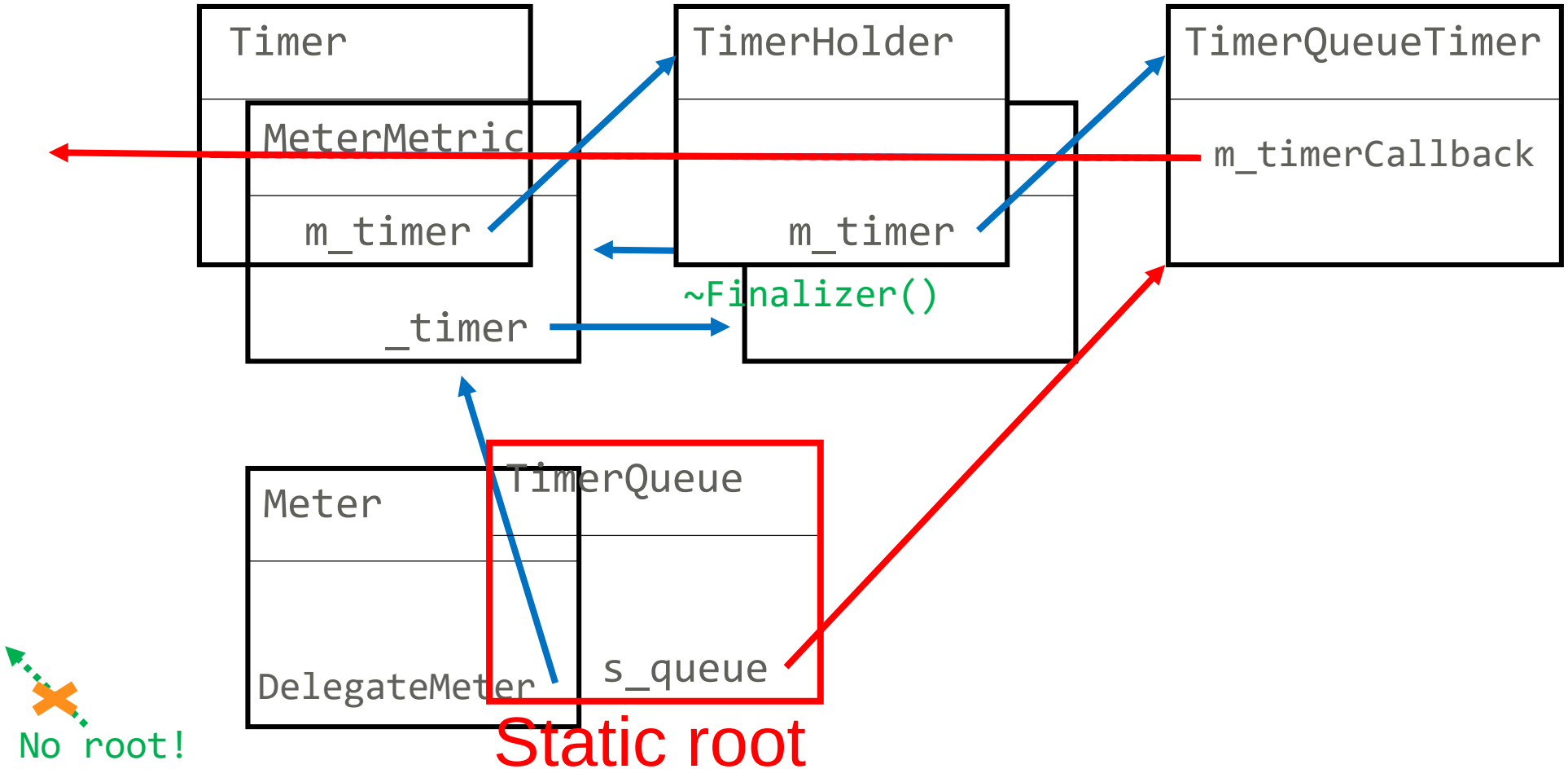
criteo

Verify what we know about finalizer

 Copyright © 2018 Criteo

Current analysis state

full reference graph please!



 Copyright © 2018 Criteo

Wrapping it up!

- **Meter does not implement IDisposable**
- **Timer is immortal if not disposed**
- **Each time a Timer ticks, a lock needs to be acquired**

→ Progressive increase of contention as the number of timers grows

 Copyright © 2018 Criteo

criteo

Mantra… Mantra… Mantra…

- **1. Identify**
- **2. Understand**
- **3. Verify**

 Copyright © 2018 Criteo

Identify | Ready for another investigation?

 Copyright © 2018 Criteo

Understand: which exceptions and why

- Nothing in the application logs
- Let’s see the exceptions
  procdump -ma ***-e 1 -f E0434352.CLR*** *<pid>*

  → ThreadAbortException
- Are we crazy enough to call Thread.Abort?
- …Catch another exception red-handed!

  CannotUnloadAppDomainException

Verify: journey from AppDomainUnload to ThreadAbortException

```
public void SomeMethod()
{
    while (someCondition)
    {
        try
        {
            // Do some stuff
            Thread.Sleep(15 * 60 * 1000);
        }
        catch (Exception ex) { ... }
    }
}
```

> “Once you eliminate the impossible, whatever remains, no matter how improbable, must be the truth”
>
> Arthur Conan Doyle

 Copyright © 2018 Criteo

Verify the behaviour of ThreadAbortException

DEMO

 Copyright © 2018 Criteo

Wrapping it up!

- **Thread.Abort is in fact « asynchronous »**
  - The AbortRequested flag is set on the Thread object
- **ThreadAbortException is thrown as soon as the thread reaches a *safe place***
- **The JIT generates code to rethrow ThreadAbortException after catch block**
  - except in 64 bit + release build with RyuJIT…

→ **Forever throwing ThreadAbortException**

 Copyright © 2018 Criteo

criteo

Thanks for your attention!

Take away:

- **Gather as much data as possible beforehand**
- **Build your own tools when needed**
- **Assume nothing, verify everything**

Additional resources:

- ClrMD (read our Criteo labs blog series)
- Contact us: k.gosse@criteo.com | c.nasarresoulier@criteo.com

criteo