While working on frontend debugging I ran into some interesting bugs.

**Chrome:**

* InspectorZWSP
* LinuxSandboxMemoryLimit
  + <https://bugs.chromium.org/p/chromium/issues/detail?id=782982>
  + <https://bugs.chromium.org/p/chromium/issues/detail?id=800348>
* IFrameRemoveHistory

**IE11:**

* WindowHistoryPushState
  + <https://web.archive.org/web/20171215235533/https://developer.microsoft.com/en-us/microsoft-edge/platform/issues/9383036/>
* CSS Precedence
* ParentContainerFocus
  + <https://support.microsoft.com/en-us/help/4088339/parent-container-can-not-get-focus-if-display-inline-block-style-used>
* MemoryLeak
* MediaQueries
  + <https://stackoverflow.com/questions/25673707/ie11-triggers-css-transition-on-page-load-when-non-applied-media-query-exists/25850649#25850649>
* MalformedFormData
* IFrameFocusClose
  + <https://developer.microsoft.com/en-us/microsoft-edge/platform/issues/872182/>

**All:**

* ParentHistoryChild

However on the Cloud team, the problems have always been simple in hindsight and more knowledge-based than *bugs*

## **Docker - Alpine Linux**

Alpine Linux is a popular choice as a base OS for containers due to its minimal design and lightweight nature. The reality is that Alpine Linux is developed for low-powered hardware and by design makes very limiting design choices.

Specifically, instead of being compiled with the popular **glibc**, Alpine is compiled (intentionally) with the alternative **musl-libc**.

There are plenty of discrepancies between glibc (common standard) and musl-libc that will result in a few days of confusion. View musl-libc's FAQ for their deviation from glibc:

<https://wiki.musl-libc.org/functional-differences-from-glibc.html>

## **Docker - Entrypoint Exec**

With exec, a child process replaces its parent such that there is one process running.

Without exec, a child process is a descendant of the parent process such that there are two processes running.

The first in this issue is that docker stop sends a SIGTERM to PID 1. In this case, the container's parent process is expected to handle docker signals and respond accordingly. If not handled intentionally, it's likely these signals will not be respected by arbitrary program X.

The second in this issue is that Kubernetes obfuscates this docker issue. When using Kubernetes to shutdown the container, the SIGTERM is sent and ignored and after *terminationGracePeriodSeconds* the SIGKILL signal is finally sent.

The result of these mistakes will not implode a system, however there will be a *terminationGracePeriodSeconds* delay between sending a stop command and successfully stopping.

## **Kubernetes - Configmap**

When using a configmap with Kubernetes what actually occurs is Kubernetes creates a `/..data/` folder and symlinks this mount to the file location.

The implication of this is that some programs do not handle symlinks and expect real files.

The most common bug involved with the discrepancy between a symlink vs a file is when watching a file or directory for changes, such as with inotify.

## **Java - WatchService**

Java's WatchService is an abstraction. On MacOS it'll return a polling watcher while on Linux it'll use inotify under the hood.

As mentioned above, itnotify has nuances with alternative file systems that require awareness. Complications can easily arise with a volume mounted in a docker container (bind mount) or when using kubernetes in the case of configmaps (symlink).

## 

## **GCP - Limitations**

* Key Size

L7 LB only supports RSA-2048 keys

<https://issuetracker.google.com/issues/35904953>

* MTU

Docker's default maximum transmission unit (MTU) is 1500 bytes. Google Cloud's MTU is 1460, essentially a smaller bandwidth tunnel leading to loss of data.

<https://cloud.google.com/compute/docs/containers#windows_known_issues>

* Distributed NFS

N2 VMs and –multi-writer limitations

* Cloud SQL internal Regional HA vs Failover HA
* GKE Ingress URL map limitation

<https://issuetracker.google.com/issues/126946582>

* false Service Account limitations
* Node Metrics collection failure

## **Elasticsearch**

* ES does not support downgrading.
* ES quorum with Kubernetes statefulsets

By default a native ES statefulset implementation will not work in a cluster configuration due to requiring a quorum.

To establish a quorum it requires a majority of nodes (or in the k8s implementation pods of the deployment). By default pods spin up in a linear fashion, therefore we need to configure the podManagementPolicy to launch and terminate all pods in parallel. Further, the default kubernetes concept of readiness is also used to determine whether a pod should communicate with local networking decisions, this also need to be adjusted as majority pods not only should spin up prior to readiness but also broadcast addresses prematurely.

### **MYSQL**

https://dev.mysql.com/doc/refman/5.7/en/identifier-case-sensitivity.html

Linux uses 0, Windows uses 1, Mac uses 2.

Knowledge is not free. You have to pay attention.

<https://www.youtube.com/watch?v=rTVOP7DNgjI>

Stay Over - Close To The Sun