Toward Pointer Leasing for Use-After-Free Prevention

Wataru Hashimoto
The University of Tokyo
hashimoto@os.ecc.u-tokyo.ac.jp

Takahiro Shinagawa
The University of Tokyo
shina@ecc.u-tokyo.ac.jp

1 Background

- **UAF (Use-After-Free)** is favorite food of attackers
  - Ranked 7th in the 2022 CWE Top 25
  - Freed object is reallocated / accessed

  ⇨ Data Manipulation / Privilege Escalation

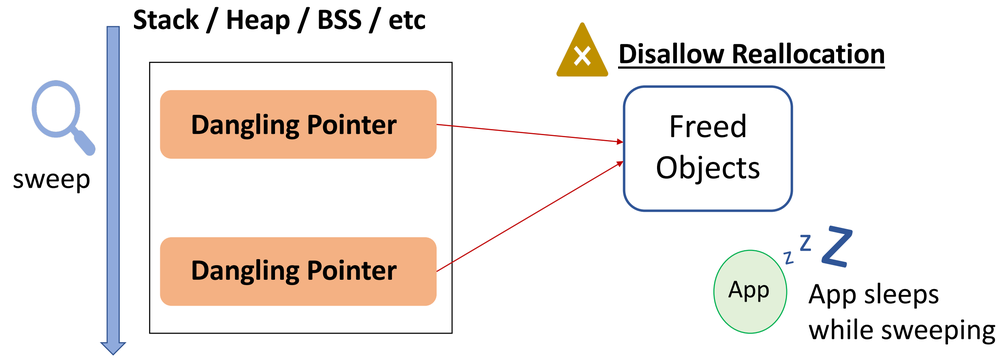
```
char *diary = malloc(0x80);
free (diary);
Credential *cred = malloc(0x80);
strcpy (diary, "UAF Data Manipulation!");
```

Reallocated

Use After Reallocation

2 Previous Work

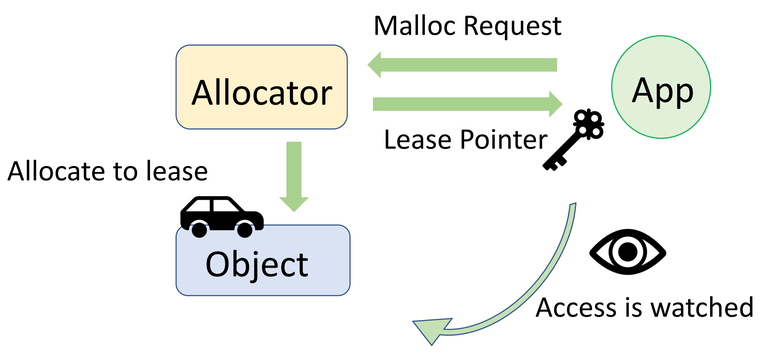
- Static analysis, One-shot allocator, HW assists ...
- GC-style: sweep memory to find dangling pointers



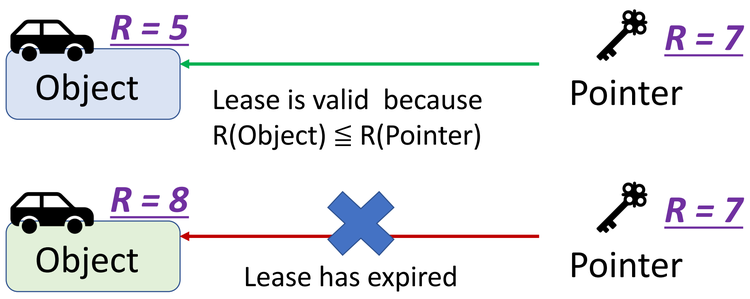
⇨ CPU/Memory overhead, downtime, probablistic

3 LeaseMalloc: "Hey, Your Lease Has Expired!"

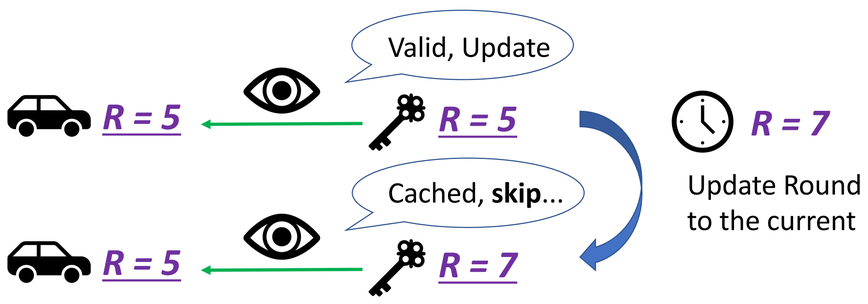
- **Concept: Time-limited "leasing" of pointers**
- Check validity of the lease on pointer accesses.



- Associate time-related expiration information (**Round**) with pointers and objects

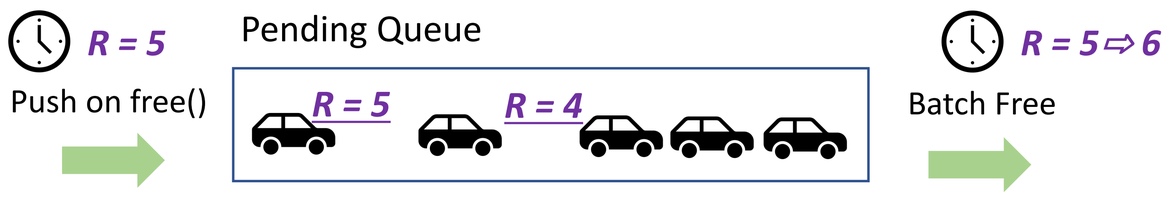


- Update Round of recently accessed pointers



- LeaseMalloc ensures lease is valid until reallocation

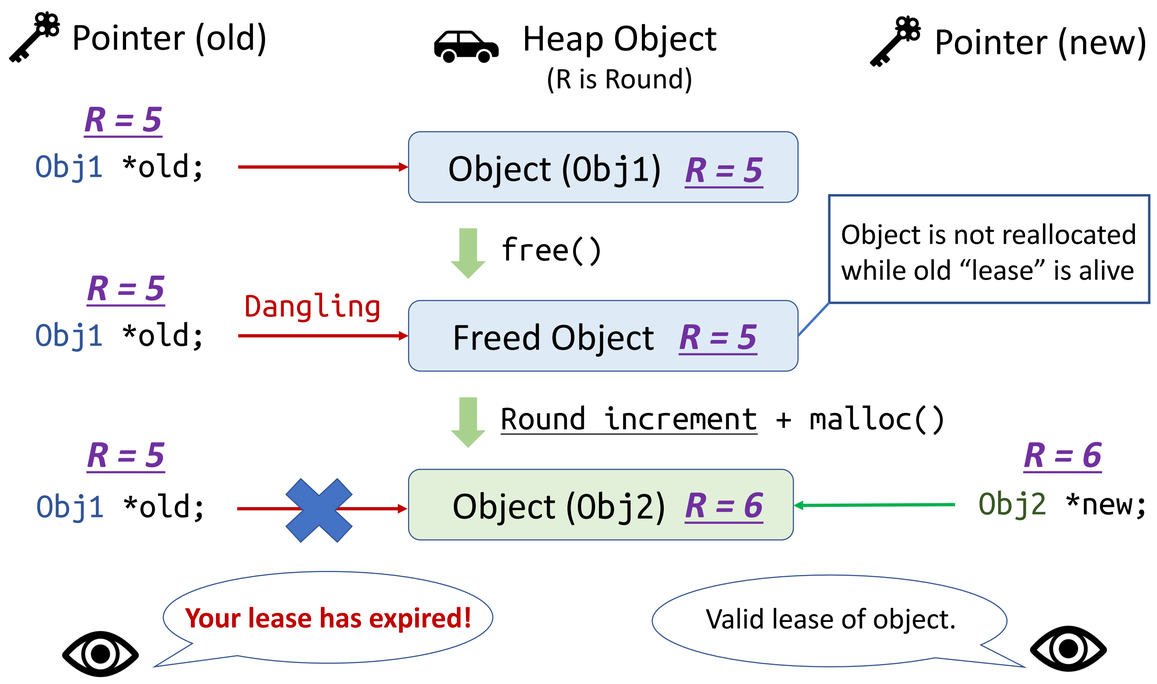
  Delay `free()` requests ⇨ Batch execute when Round is updated



How LeaseMalloc detects lease expiration?

- Watch access to new object (R=6) through old pointer (R=5)

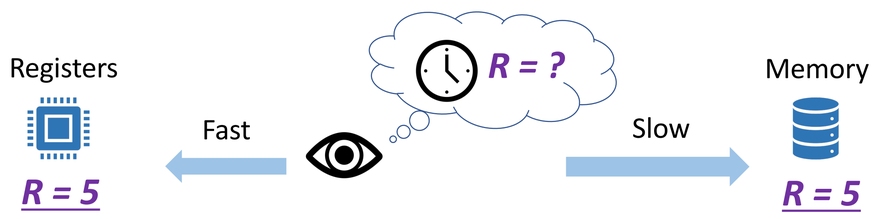
  ⇨ Detects inconsistency and report violation!



4 Optimization

- Use unused registers to store current Round

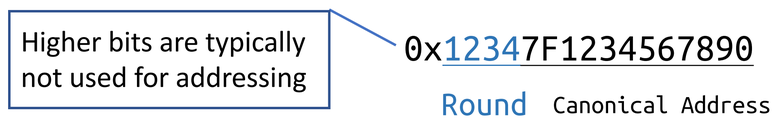
  ⇨ Validity check of lease is accelerated



- Can limit scope of instrumentation to target app, excluding stable libraries

5 Implementation

- Instrumentation with LLVM PassFramework
- JeMalloc based extension (Porting to other allocators is easy)
- Round is embedded in higher bits of pointers



6 Future Work

- Seek for effective optimizations
- Compare performance with state-of-arts