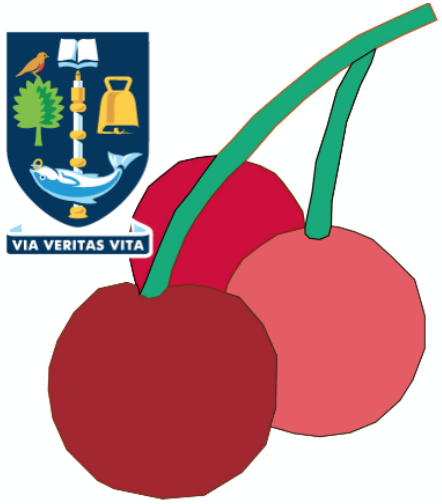

Capable VMs: Review of CHERIvoke System

Jeremy Singer

What is ‘spatial memory safety’?

```c
#include <stdio.h>

int main() {
  int x = 3;
  int *p = &x;
  long long q = (long long)p;
  q = q + 8;
  printf("%d\n", *((int *)q));
  return 0;
}
```

Compiler error

Output

Warning: cast from provenance-free integer type to pointer type will give pointer that can not be dereferenced

What is ‘temporal memory safety’?

- Avoid *use-after-free* for pointers
- Bad sequences like:
  - p $\leftarrow$ `malloc` (...)
  - `free` (p)
  - use p later on ...

CHERIvoke provides temporal mem safety

- enforced through capability mechanisms
- evaluated in a ‘real’ x86 architecture

Paper to read

- `https://doi.org/10.1145/3352460.3358288`
- CHERIvoke: Characterising Pointer Revocation using CHERI Capabilities for Temporal Memory Safety
- Xia *et al*
- MICRO-52, 2019

Things to check out

- instrumentation in `free`
- a quarantine buffer
- shadow map
- periodic sweep through memory