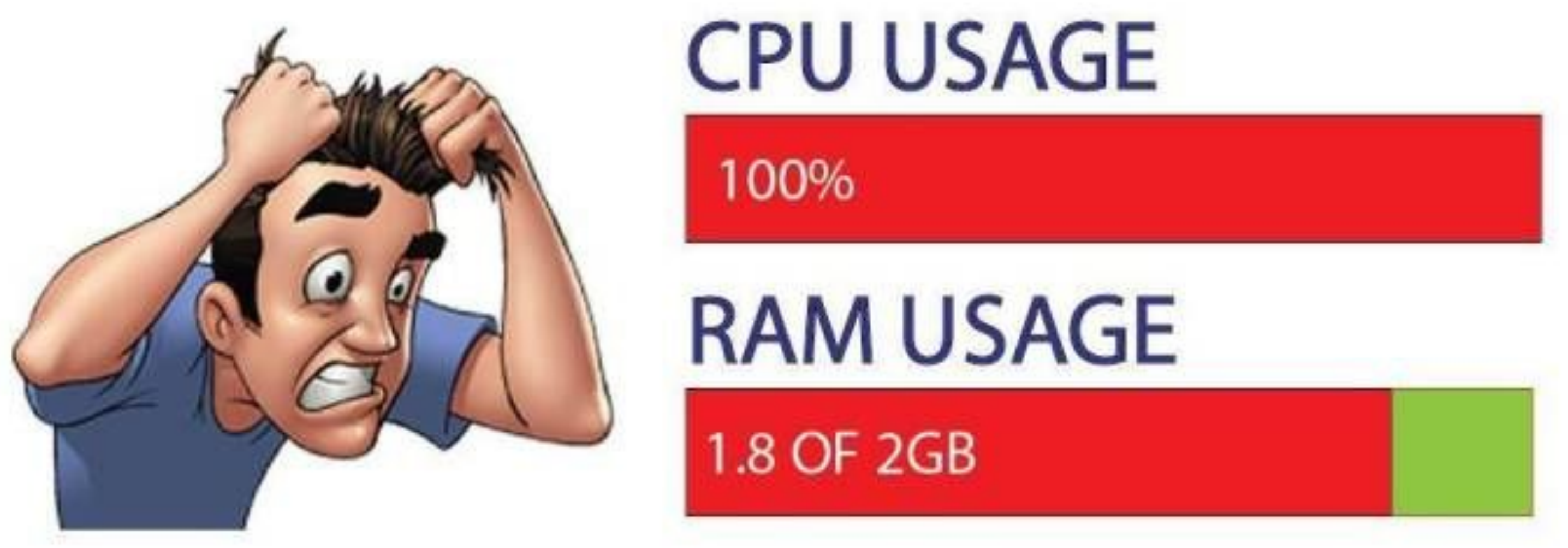
Follow-me Connections: Seamless TCP Connection Migration

Benjamin Reeves, Thomas Oliver, Jiayi Liu
{bgreeves, olivertc, liujiayi}@umich.edu

Motivation

- Cloud is shared by multi-tenants
  → high utilization by others leads to **performance degradation**
- Want to migrate applications away from overloaded hosts
- Live migration with CRIU is
  1. Quick to respond
  2. Fast and seamless

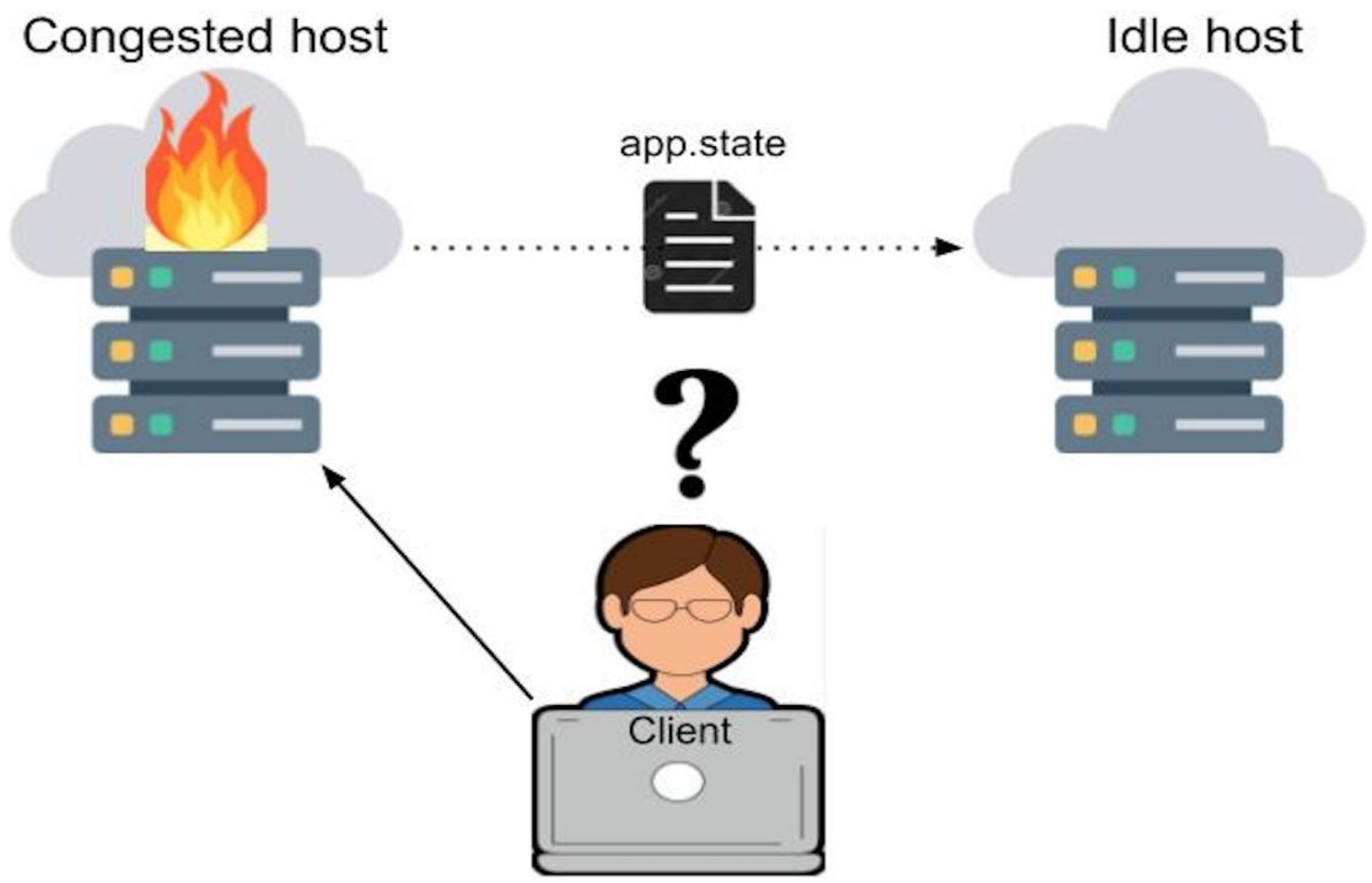


Implementation

- Change TCP stack in Linux kernel to
  - Add migration token to sockets
  - Send migration requests
  - Update kernel data structures with new host when migrate occurs
- Add support for restoring sockets on different hosts in CRIU
  - Change socket source address to make it bind to new host
  - Dump and restore migrate token

Problem

- Application should handle TCP connection migration
- CRIU can only checkpoint and restore on same host if the process has TCP connections

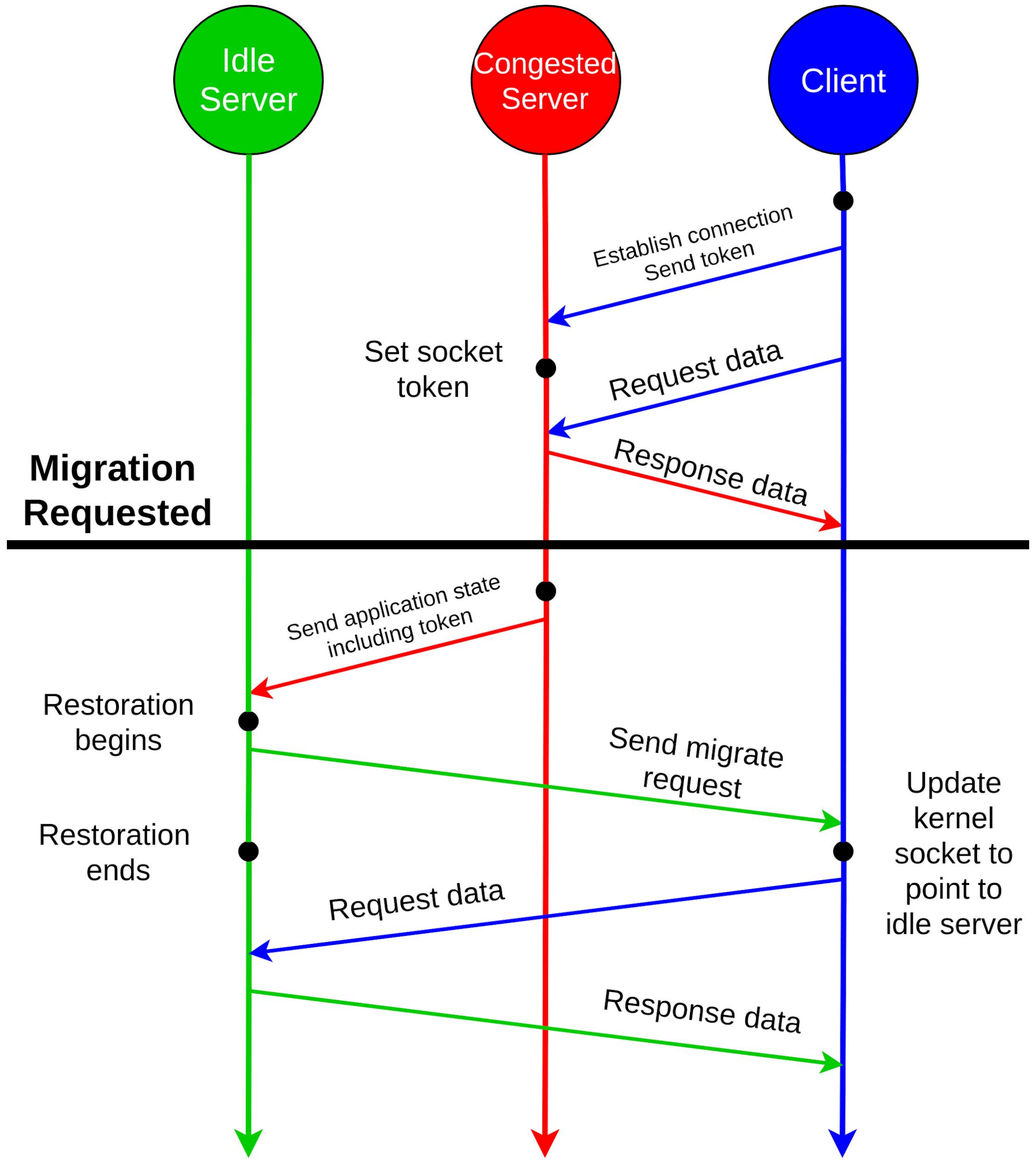


Future Work

- Automatically load-balance based on resource usage
- Handling CRIU failure conditions
- Make the migrate token cryptographically secure
- Add waiting state for client during migration
- Make migration faster using CRIU's incremental checkpointing

Design

- Initialize connection with a client-provided token
- Checkpoint application with token
- Send checkpoint to other server
- Start restoring application
- Send migration request to client with migrate token
- Finish restoring and resume connection


Idle Server
Congested Server
Client
Establish connection
Send token
Set socket token
Request data
Response data
Migration Requested
Send application state including token
Restoration begins
Send migrate request
Update kernel socket to point to idle server
Restoration ends
Request data
Response data