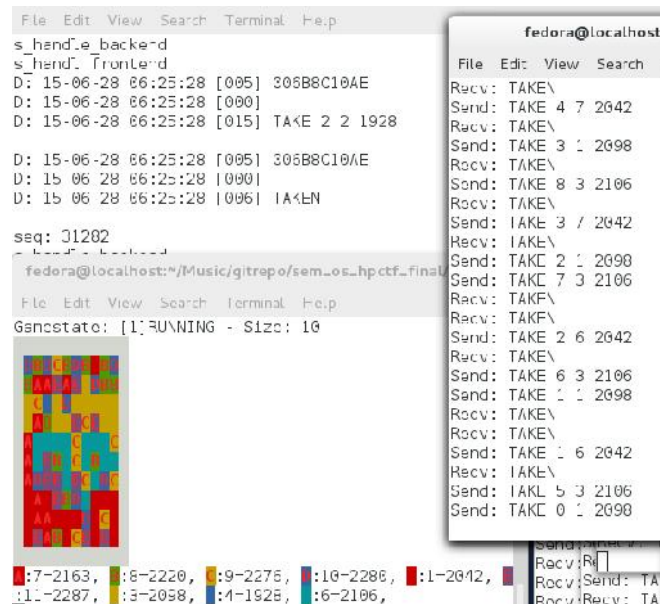


HPCTF – Hauronen Patronen's Capture the Flag using ØMQ

Hauri David

ZHAW – School of Engineering, Studiengang Informatik (28.06.2015)



The screenshot shows a terminal window with two panes. The left pane displays game logs with timestamps and messages like 'D: 15-06-28 06:25:28 [005] 305B8C10AE' and 'TAKE 2 2 1928'. The right pane shows a color-coded board state with a grid of numbers and colors (red, yellow, blue, green). Below the board, there is a legend with color-coded numbers: red for 7-2163, yellow for 6-2220, blue for 9-2275, green for 10-2286, and red for 1-2042. The bottom of the terminal shows a 'Gencstate: [1] RUNNING - Size: 10' message.

Als Basis für die Semester und Bachelor Arbeit habe ich mich intensiv mit ØMQ auseinandergesetzt. Diese Seminararbeit, bietet ein ideales Einsatzgebiet.

ØMQ¹ (also spelled ZeroMQ, 0MQ or ZMQ) is a high-performance asynchronous messaging library aimed at use in scalable distributed or concurrent applications. It provides a message queue, but unlike message-oriented middleware, a ØMQ system can run without a dedicated message broker. The library is designed to have a familiar socket-style API.

Das Spielprinzip: Mehrere Spieler loggen sich auf dem Server ein und probieren alle Felder zu erobern. Der Spieler der zuerst alle Felder erobert, gewinnt. Die Kommunikation zwischen Spieler und Server basiert auf TCP/IP und wurde mithilfe von ØMQ in Verwendung folgender Paradigmen umgesetzt:

Request Reply: Clients kommunizieren über den Port 5555 mit dem Server. Für jeden Request, wird ein Reply erwartet.

Load Balancer: Serverseitig werden die Requests an eine Backend.ipc pipe weitergegeben und von verschiedenen Workerthreads asynchron abgearbeitet.

Networkwide Key Value Map: Über den Port 5556 stellt der Server asynchrone Statusmeldungen zur Verfügung welche in einer Key Value Hash gehalten werden und abgefragt werden können.

¹ ØMQ Wikipedia: <http://en.wikipedia.org/wiki/%C3%98MQ>