



netidee

STIPENDIEN

# Improving Serverless Edge Computing for Network Bound Workloads

Zwischenbericht | Call 15 | Stipendium ID 5115

Lizenz: CC-BY

# Inhalt

|     |                                                            |   |
|-----|------------------------------------------------------------|---|
| 1   | Einleitung.....                                            | 3 |
| 2   | Status .....                                               | 3 |
| 2.1 | Meilenstein 1 - Evaluierungsszenarien definieren .....     | 3 |
| 2.2 | Meilenstein 2 - Architekturverbesserungen definieren ..... | 4 |
| 2.3 | Meilenstein 3 - Implementierung.....                       | 4 |
| 2.4 | Meilenstein 4 - Experimentelle Evaluierung.....            | 5 |
| 2.5 | Meilenstein 5 - Evaluierung durch Simulation .....         | 6 |
| 3   | Zusammenfassung Planaktualisierung.....                    | 6 |

## 1 Einleitung

Generell hat sich für meine Arbeit viel verändert. Die Ziele, Motivationen und erwarteten Ergebnisse sind zwar gleich geblieben, die Herangehensweise und Zeitlinie aber nicht. Wie bereits im letzten Planupdate beschrieben, musste ich den Zeitplan meiner Arbeit anpassen und nach hinten verschieben. Nachdem ich aber seit Juni in Bildungskarenz bin, um mich voll meiner Diplomarbeit widmen zu können, bin ich zuversichtlich, diese bald fertigstellen zu können. Ein signifikanter Anteil der Arbeit ist bereits fertiggestellt, und es wurden mehrere neue Erkenntnisse gewonnen bzw. Vermutungen bestätigt und widerlegt. Die Vorgangsweise wurde außerdem angepasst, um zu qualitativ besseren Erkenntnissen zu gelangen.

## 2 Status

### 2.1 Meilenstein 1 - Evaluierungsszenarien definieren

Hierbei ging es zunächst darum festzulegen, welche Szenarien für die Evaluierung interessant sind, wieso diese es sind, und wie eine Evaluierung passieren kann. Dieser Meilenstein war nicht besonders groß, aber dennoch wichtig, um eine nachvollziehbare Struktur und vor allem nachvollziehbare und vergleichbare Ergebnisse zu erzielen. Die Vorgangsweise war recht simpel, weil bereits fest stand welche Hardware für praktische Experimente zur Verfügung steht. Die Hardware wird in zwei Zonen unterteilt, Edge und Cloud, wobei die Clients sich in der Edge befinden. Die Netzwerkverzögerungen werden künstlich mittels moderner QoS Techniken eingefügt und basieren auf Messungen aus der echten Welt.

Auf Seite der Simulation war dies ebenfalls prinzipiell einfach, weil für einen Smart-City Aufbau, welcher eines der zentralen Szenarien für mich ist, bereits realistische Daten in Form des in Chicago gebauten „Array of Things“ existieren (Catlett et al. 2017). Die Vorgangsweise bestand hierbei daraus anhand bestehender Literatur möglichst realistische, aber dennoch umsetzbare Konfigurationen zu erzeugen.

Neben den Szenarien wurden Metriken festgelegt, die zur Evaluierung herangezogen werden, was auch in einem Blogeintrag beschrieben wurde.

Hierbei gab es keine besonderen Erfolge oder Probleme bzw. Planabweichungen. Probleme, die hier entstehen, zeigen sich leider erst später, sobald die Szenarien umgesetzt werden sollen.

## **2.2 Meilenstein 2 - Architekturverbesserungen definieren**

Die Idee hinter diesem Meilenstein war es festzulegen wodurch die erhofften Performance-Gewinne realisiert werden sollen. Die Vorgangsweise war ursprünglich so geplant, dass der source-code der entsprechenden Plattformen gelesen wird, und dann die Entscheidung fällt, was wie adaptiert wird. Diese Vorgangsweise hat sich allerdings als mäßig sinnvoll herausgestellt. Zwar haben sich so durchaus mögliche Verbesserungen finden lassen, aber diese waren stets spezifisch für die jeweilige Implementierung oder Konfiguration.

Nachdem das nicht von wesentlichem wissenschaftlichen Wert ist, habe ich mich entschieden, einen „Schritt zurück“ zu gehen, und das System abstrakter zu betrachten. Hier war die wesentliche Erkenntnis, dass zur Verbesserung der Performance zwei Dinge geschehen müssen: Load-Balancer müssen intelligenter agieren, und diese müssen effizient über das System verteilt werden. Das Ziel ist daher zu zeigen, dass diese Änderungen tatsächlich Vorteile mit sich bringen, praktisch umsetzbar sind, und zu erfassen ob bzw. wie sich diese Änderungen auf das Gesamtsystem auswirken.

Hierdurch hat sich in der Vorgangsweise der gesamten Arbeit etwas verschoben, und zwar wird weniger vom Speziellen (nämlich OpenFaaS als konkrete Implementierung) auf das Generelle geschlossen, sondern das Generelle wird betrachtet und Änderungen am Generellen werden durch das Spezielle validiert und getestet. An diesen Punkt zu gelangen war nicht einfach, aber es ist wohl die Änderung, die meine Arbeit am dramatischsten verbessert. Hierdurch ist die Arbeit präziser, fokussierter, allgemeingültiger, wissenschaftlich sauberer, und letztlich von größerem Nutzen. Obwohl dieser Punkt meine Arbeit noch etwas weiter verzögert hat, und mehr Aufwand mit sich bringt, weil auf einer fundamentalen Ebene gearbeitet wird, sehe ich dies für mich als einen besonderen Erfolg.

## **2.3 Meilenstein 3 - Implementierung**

Die ursprünglich geplante Tätigkeit war es, die geplanten Verbesserungen in die Praxis umzusetzen. Durch die eben beschriebene Änderung auf eine eher abstrakte Ebene ging es hier darum, flexibel einsetzbare und erweiterbare Komponenten zu bauen, durch die eine abstrakte Planung in Form einer konkreten Implementierung getestet werden kann.

Die erste große und wohl wichtigste Komponente hierbei war die Anpassung von traefik, einem Open-Source-load-balancer. Dieser wurde angepasst, um neben round-robin auch least-response-time, und darüber hinaus eine leicht konfigurierbare Kombination aus beliebigen Metriken zu unterstützen. Dieser Teil repräsentiert den bereits erwähnten „intelligenten load-balancer“. Für die intelligente Platzierung wurde Skippy angepasst, was seinerseits eine Reimplementierung des von Kubernetes verwendeten Schedulers ist. Diese Anpassung ist aber nach wie vor im Gange, weil sich nach Experimenten und Messungen weiterhin neue Erkenntnisse ergeben, die dann wieder eingearbeitet werden.

Ein besonderer Erfolg war hier für mich, dass ein weiteres OpenSource Projekt verwendet werden konnte, und dass ich es um Funktionen erweitert habe, die nicht nur für mich, sondern auch für andere Leute ausgesprochen nützlich sein können. Diese sind hier auffindbar:

<https://github.com/jjnp/traefik/tree/load-balancing>

Das Problem ist bei diesem Meilenstein die Abweichung vom Plan. Ursprünglich war geplant die Implementierung gänzlich abzuschließen, aber es hat sich herausgestellt, dass das Verhalten von diesen Systemen recht komplex und vor allem schwer intuitiv vorhersehbar ist. Daher muss man iterativ vorgehen (in etwa Scrum statt Wasserfall), und anstatt einer einmaligen Implementierung ständig nachbessern und anpassen.

## **2.4 Meilenstein 4 - Experimentelle Evaluierung**

Das Ziel hier war es herauszufinden, ob die durch praktische Implementierung angepassten Elemente, insbesondere der load-balancer sich wie erwartet verhalten. Die Ergebnisse sind auf gewisse Weise positiv und negativ zugleich ausgefallen. Auf der einen Seite hat sich herausgestellt, dass die Grundannahmen über das Verhalten des load-balancers stimmen. Auf der anderen Seite hat sich sehr früh gezeigt, dass selbst in isolierten Experimenten aus der neu hinzugefügten Komplexität in Sonderfällen eher unerwartetes Verhalten hervorgehen kann. Dies ist zwar eine spannende wissenschaftliche Erkenntnis, bedeutet aber mehr Aufwand, da das System in Frage komplexer ist als erwartet, und damit genauere und gründlich validierte Experimente vonnöten sind. Nähere Überlegungen und exemplarische Daten wurden im Blogeintrag „Request Routing in Edge Szenarien“ erläutert.

Die Erkenntnis, dass in gewissen Situationen unerwartete Effekte auftreten können, ist in vielerlei Hinsicht ein großer Erfolg dieses Arbeitsabschnitts. Insbesondere deshalb, weil es eine klare Motivation und Begründung dafür liefert, weshalb dieser Forschungsbereich wichtig ist und zeigt, dass Serverless-Edge-Systeme teils dramatisch andere Eigenschaften aufweisen als klassische Serverless-Systeme.

Planabweichungen gibt es insofern, als durch diese komplexeren Effekte Implementierung, praktische Evaluierung und simulierte Evaluierung gleichzeitig bzw. abwechselnd passieren, um jeweils die neuen Erkenntnisse sofort umsetzen zu können. Abgesehen von der Verzögerung hierdurch gab es eine Abweichung, weil die Geräte der TU-Wien, die zum Testen verwendet werden, für mich aufgrund von COVID-19 einige Zeit nicht nutzbar waren (kein Zugang zu den entsprechenden Räumlichkeiten). Daher wurden einige praktische Experimente (sofern möglich) auf gemieteten Cloud-Servern durchgeführt, was etwas aufwändiger war. Mittlerweile habe ich allerdings wieder Zugang zur TU, wodurch die laufenden Experimente wie erwartet durchgeführt werden können.

## **2.5 Meilenstein 5 - Evaluierung durch Simulation**

Die Haupttätigkeit hierbei ist, die Verbesserungen in größeren Simulationsszenarien zu testen als bei Experimenten mit echter Hardware möglich sind. Simulation hat sich als eine hervorragende Methode herausgestellt, und ich habe deshalb bis jetzt mehr Arbeit in Simulationen gesteckt als in praktische Evaluierungen. Das wird im Blogpost „Simulation first!“ detaillierter beschrieben. Ich konnte durch Simulationen bereits klar zeigen, dass eine Veränderung der load-balancer und eine effektive Platzierung dieser zu teils dramatischen Verbesserungen führen kann. Ich konnte auch zeigen, dass diese Verbesserungen in unterschiedlich großen Szenarien halten, wobei sie selbstverständlich unterschiedlich groß ausfallen.

Neben den vielversprechenden Ergebnissen ist ein Erfolg, dass der von mir verwendete Simulator (FaaS-Sim), der ein state-of-the-artserverless Simulator der TU ist, von mir ausgebaut werden konnte. Dadurch wurden Experimente wie meine, die starken Fokus auf das Netzwerk legen, im Simulator erst möglich. Jede Verbesserung am Simulator ist außerdem insofern ein Erfolg, als es einen Beitrag zu nachfolgender Forschung leistet.

Die bereits erwähnten generellen Verzögerungen betreffen auch die Simulation, nachdem diese abwechselnd mit praktischen Experimenten und Implementierungsarbeit ausgeführt wird.

Beispielsweise werden Performance-Daten von Experimenten mit der echten Implementierung als Basis für simulierte Werdegänge usw.

Dadurch ist auch dieser Meilenstein noch „in Progress“.

## **3 Zusammenfassung Planaktualisierung**

Die wesentlichste Anpassung am Planungsdokument ist eine erneute Verschiebung der Fertigstellung. Während bis jetzt geplant war Ende Juni sämtliche praktischen und theoretischen Evaluierungen fertiggestellt zu haben, ist nun der Plan, dass dies bis Ende Juli passiert bzw. einzelne Experimente, sofern notwendig, noch im August stattfinden. Solche verzögerten Experimente sind

allerdings kein wesentlicher Teil der Erkenntnisgewinnung mehr, sondern dienen dazu, im Rahmen der Verschriftlichung gewisse beschriebene Phänomene besser zu illustrieren bzw. auch intuitiv offensichtliche Aussagen experimentell zu untermauern.

Die beiden Meilensteine „Experimentelle Evaluierung“ und „Evaluierung durch Simulation“ haben sich in der Reihenfolge vertauscht, bzw. sind diese miteinander verschmolzen. Bis jetzt und auch aktuell entwickeln sich beide Arten der Evaluierung in Tandem. Daher passieren diese nicht nacheinander, sondern gleichzeitig, und in einem gemeinsamen Prozess. Im Plan habe ich die Reihenfolge vertauscht, und die Meilensteine zwecks Nachvollziehbarkeit aber weiterhin getrennt, auch wenn dies nicht strikt der Realität entspricht.

Die Verschriftlichung der Arbeit verschiebt sich damit ebenfalls auf August bzw. September. Daher wird entweder verspätet die Arbeit als Endbericht oder wie nach Plan ein Endbericht rein mit den Erkenntnissen an netidee übergeben.