Uma Investigação Empírica sobre Observabilidade em Sistemas 5G Nativos de Nuvem

Karlla B. Chaves Rodrigues[1], **Gustavo Zanatta Bruno**[2], **Kleber Vieira Cardoso**[1], **Sand Luz Corrêa**[1] **e Cristiano Both**[2]

[1] Universidade Federal de Goiás (UFG), Goiânia, GO – Brasil

[2]Universidade do Vale do Rio dos Sinos (UNISINOS), São Leopoldo, RS – Brasil

karllachaves@discente.ufg.br, {kleber,sand}@ufg.br,

zannatabruno@edu.unisinos.br, cbboth@unisinos.br

Abstract. *By replacing sophisticated hardware with software components running on cloud systems, 5G networks provide great flexibility. However, the softwarization of 5G networks imposes considerable challenges to network management, especially in terms of system correctness at runtime. In this context, 5G system behavior must be readily observable. This work presents an empirical investigation on observability in a 5G system composed of the core and access networks. Results show that the combined analysis of metrics and logs improves system observability and facilitates the anticipation of failure occurrences. Our study also demonstrates that, despite being helpful, the log messages generated by the deployed 5G system are poorly structured and often unintelligible.*

Resumo. *Ao migrar sofisticados recursos de hardware para componentes de software executados em sistemas de nuvem, as redes 5G proveem grande flexibilidade. No entanto, a softwarização das redes 5G impõe grandes desafios à gerência da rede, especialmente em relação à corretude do sistema em tempo de execução. Neste contexto, se torna essencial que o comportamento do sistema seja facilmente observável. Este trabalho apresenta uma investigação empírica sobre observabilidade em um sistema 5G formado pelas redes de núcleo e de acesso. Resultados mostram que a análise combinada de métricas e logs melhora a observabilidade do sistema e facilita a antecipação de ocorrências de falhas. Nossa investigação demonstra também que, apesar de úteis, as mensagens de logs geradas pelo sistema 5G implantado são pouco estruturadas e pouco inteligíveis.*

1. Introdução

A quinta geração de rede móvel celular (5G) está sendo implantada em larga escala em grandes cidades do mundo [Narayanan et al. 2021], trazendo grandes mudanças arquiteturais para as redes de telecomunicação e promovendo novos modelos de negócio para esse setor. De fato, ao adotar conceitos como computação de borda, virtualização de funções de rede, programabilidade da rede e suporte a múltiplos inquilinos, a rede 5G propõe novas formas de projetar, provisionar e gerenciar serviços fim-a-fim com qualidade de serviço (*Quality of Service* - QoS) e adaptados às necessidades de diferentes verticais (e.g., setor automotivo, agricultura, gestão municipal, governo, saúde, manufatura) [Agiwal et al. 2016].

Para atender a todos esses requisitos de forma ágil e flexível, os sistemas 5G se apoiam em uma arquitetura baseada em serviços e tecnologias nativas de computação

em nuvem [3GPP 2020a]. A junção desses dois paradigmas, por sua vez, é normalmente realizada através de uma arquitetura baseada em microsserviços [Esposito et al. 2016]. A arquitetura baseada em microsserviços apresenta diversas vantagens em relação a um projeto monolítico. Microsserviços representam pequenas unidades de implantação são fracamente acoplados e podem ser alterados, dimensionados e implantados independentemente dos demais microsserviços. Além disso, microsserviços são baseados nos recentes avanços em infraestruturas de contêineres, as quais reduzem a complexidade para implantação e elasticidade das aplicações. Entretanto, ao decompor um sistema em diversos microsserviços distribuídos, a pilha de software que o compõe pode crescer rapidamente, tornando difícil a compreensão do comportamento do sistema em tempo de execução [Balalaie et al. 2016]. Para tratar esse problema, ambientes de computação em nuvem geralmente utilizam o conceito de *observabilidade*, ou seja, a habilidade de compreender o estado interno de um sistema analisando apenas suas saídas [Russ 2019]. Essa saídas são registradas na forma de métricas, *logs* e *traces*, os quais são considerados os pilares da observabilidade.

Atualmente, grandes plataformas de computação em nuvem como Microsoft Azure [Microsoft 2021] e Google Cloud [Google 2022] utilizam soluções integradas para coleta de métricas, *log* e *traces* em suas infraestruturas para melhorar a observabilidade de seus sistemas. Além disso, nos últimos anos, diversas ferramentas usadas para implementar observabilidade foram incorporadas na pilha de software necessária para construir aplicações nativas de nuvem [CNCF 2022]. Como exemplo, podemos citar as ferramentas Prometheus[1], usada para a coleta de métricas, Fluentd[2], com foco no processamento de *logs*, e Jaeger[3], usada para a coleta e análise de *traces*. No entanto, o paradigma de arquiteturas baseadas em microsserviços é novo no contexto de redes de telecomunicações, onde tradicionalmente os serviços são disponibilizados de forma monolítica e com grande acoplamento com hardware proprietário [Contreras et al. 2020]. Com o início da implantação das redes 5G, operadores terão que lidar com diversos cenários anômalos em tempo de execução. Neste contexto, será essencial inferir de forma rápida e precisa o estado do sistema. Apesar dessa necessidade, como discutido na Seção 3, poucos trabalhos na literatura investigaram a implementação de observabilidade em sistemas 5G. Adicionalmente, os trabalhos existentes limitam-se a apenas um dos pilares da observabilidade, em geral, a coleta de métricas.

Para ajudar a preencher esta lacuna, este trabalho apresenta uma investigação empírica sobre observabilidade em um sistema 5G formado pela rede de núcleo 5G (*Core Network* – CN) e pela rede de acesso via rádio (*Radio Access Network* - RAN). Especificamente, avaliou-se a efetividade do uso combinado de dois pilares da observabilidade, métricas e *logs*, para diagnosticar anomalias em um sistema 5G. Para atingir este objetivo, um ambiente 5G utilizando duas plataformas de código aberto amplamente usadas em pesquisas em redes 5G foi reproduzido: *Free5GC* [Free5GC 2021] e *OpenAirInterface* [Alliance 2020b]. Essas plataformas foram implantadas como um conjunto de microsserviços em um *cluster* Kubernetes. Este último é o orquestrador padrão, de fato, para aplicações baseadas em contêiner mantidas pela *Cloud-Native Computing Foundation* (CNCF). Para a coleta e visualização de métricas, as ferramentas Prometheus e Grafana[4] foram utilizadas, enquanto a coleta e filtragem de *logs* foi realizada com a pilha

[1]https://prometheus.io
[2]https://www.fluentd.org
[3]https://www.jaegertracing.io
[4]https://grafana.com/

composta pelo Elasticsearch[5], Fluentd e Kibana[6] (EFK *stack*). Resultados mostram que a análise combinada de métricas e *logs* melhora a observabilidade do sistema. De fato, em um dos experimentos realizados, a análise de *logs* antecipa a ocorrência de uma falha. Resultados demonstram também que, apesar de úteis, as mensagens de *logs* geradas pelo sistema 5G implantado são pouco estruturadas e pouco inteligíveis. Resumidamente, as principais contribuições deste trabalho são: (i) o projeto e implantação de uma *testbed* que permite a reprodução de uma rede 5G formada pela rede de núcleo e rede de acesso, segundo recomendações de órgãos de padronização relevantes; (ii) a implantação de uma solução de observabilidade para a rede 5G implantada usando ferramentas nativas de nuvem e que explora informações obtidas de métricas e *logs*; e (iii) uma análise sobre a efetividade do uso combinado de métricas e *logs* para diagnosticar anomalias em sistemas 5G.

O restante deste trabalho está organizado conforme descrito a seguir. A Seção 2 apresenta fundamentos relacionados à observabilidade e à arquitetura dos sistemas 5G. A Seção 3 apresenta os trabalhos relacionados. A Seção 4 descreve a metodologia utilizada. Os resultados são discutidos na Seção 5. Finalmente, a Seção 6 apresenta as considerações finais e direções para trabalhos futuros.

2. Fundamentação

Esta seção apresenta conceitos fundamentais sobre observabilidade de sistemas. Além disso, os fundamentos da arquitetura 5G são brevemente discutidos.

2.1. Fundamentos sobre observabilidade

Observabilidade vem se tornando uma habilidade essencial para manter sistemas nativos de nuvem [CNCF 2022]. Oriundo da teoria de controle, o termo observabilidade enfatiza a necessidade de trazer maior visibilidade à compreensão do comportamento de um sistema complexo usando telemetria coletada do sistema em tempo de execução [Sridharan 2018]. Nesse contexto, diferentemente de um monitoramento caixa preta, o objetivo da observabilidade é minimizar o tempo necessário para que operadores de sistemas obtenham uma visão contextual precisa sobre a corretude e o desempenho do sistema. A telemetria extraída do sistema pode produzir três tipos de dados: métricas, *logs* e *traces*.

Métricas são atributos quantificáveis ou contáveis cujos valores podem ser coletados de duas formas [Scrocca et al. 2020]: (i) diretamente do processo em execução, que as expõe através do uso de uma biblioteca; ou (ii) a partir da infraestrutura que executa o processo. No primeiro caso, métricas são definidas e coletadas no nível da aplicação, enquanto no segundo estão associadas a níveis de utilização de recursos da infraestrutura, como CPU, memória e rede de comunicação. A coleta e persistência periódica de dados associados a métricas é usualmente referida na literatura como *monitoramento*, sendo um dos principais elementos utilizados para detecção de anomalias em um sistema computacional [Meng and Liu 2013]. O monitoramento cria séries temporais, às quais, quando analisadas, podem indicar tendências sobre o comportamento das entidades monitoradas. Em ambientes de computação em nuvem, esse conhecimento pode ser utilizado para configurar alertas ou disparar eventos de elasticidade no orquestrador. Prometheus é a ferramenta de monitoramento adotada pela CNCF, sendo também a solução de coleta de dados nativa do Kubernetes. Além do monitoramento, a visualização dos dados coletados

[5]https://www.elastic.co/

[6]https://www.elastic.co/kibana/

também desempenha um papel crítico na observabilidade de sistemas nativos de nuvem. Grafana e Kibana são ferramentas amplamente utilizadas para esse propósito.

Logs são registros textuais de operações ou erros realizados por um sistema em tempo de execução [Bhanage et al. 2021]. De maneira similar a métricas, *logs* são coletados diretamente de processos em execução. Entretanto, diferentemente de métricas, os registros de *logs* acontecem de forma não previsível, pois estão associados a ocorrências de eventos. Dependendo do formato escolhido, *logs* podem conter informações estruturadas ou desestruturadas. Adicionalmente, *logs* podem ser gerados em diferentes níveis de um sistema de computação. Para ilustrar, o orquestrador Kubernetes provê *logs* com informações sobre todo o *cluster* (e.g. *logs* de auditoria relacionados à requisições feitas ao *master node*), como também para elementos de menor granulosidade como *Pods* e nós da infraestrutura. *Logs* também podem ser coletados de contêineres em tempo de execução, quando esses escrevem dados na saída padrão. De fato, *logs* podem representar um grande volume de dados, os quais são gerados de praticamente todas as entidades de um sistema computacional. Por essa razão, em ambientes de computação em nuvem, eles são frequentemente usados para detectar anomalias e para a análise de causas de problemas. Contudo, a efetividade do uso de *logs* depende da qualidade das mensagens geradas pelos sistemas. Registros inteligíveis podem causar problemas no diagnóstico de falhas, exigir grande esforço de manutenção e degradação de desempenho [Chen and Jiang 2021]. A extração de informação a partir de *logs* é uma tarefa custosa e, em geral, exige pre-processamento e filtragem. As ferramentas Elasticsearch, Fluentd, e Kibana são frequentemente usadas em conjunto para processamento e filtragem de *logs*. Entretanto, diferentemente de métricas, a CNCF não tem um fluxo de trabalho (*pipeline*) bem definido para a coleta e filtragem de *logs*.

Enquanto métricas e *logs* proveem informações sobre serviços individuais, *traces* identificam o fluxo de requisições dentro de um sistema constituído por vários microsserviços distribuídos [Picoreti et al. 2018]. Portanto, *traces* evidenciam uma relação de causalidade numa cadeia de requisições, expondo a intrincada rede de interações típica de sistemas baseados em microsserviços. Esse conhecimento ajuda a compreender como um evento se propaga pelo sistema em tempo de execução [Esteves et al. 2021], sendo especialmente importante em ambientes distribuídos, onde não existe um relógio global e sincronizações precisas entre os nós são frequentemente impossíveis. No entanto, a geração de *traces* em sistemas distribuídos requer que o sistema observado seja instrumentado para essa finalidade. Em particular, o rastreamento e a reconstrução do fluxo de requisições exigem a propagação de metadados durante a comunicação entre os componentes do sistema (e.g., microsserviços). A ferramenta Jaeger é a proposta da CNCF para a coleta e análise de traces. Em resumo, métricas, *logs* e *traces* constituem os pilares da observabilidade. Analisadas em conjunto, essas fontes de informação podem prover uma visão fiel do comportamento de um sistema em tempo de execução [Sridharan 2018], como por exemplo os sistemas 5G.

2.2. Fundamentos sobre redes 5G

Apesar de trazerem grandes mudanças em termos de requisitos, casos de uso, paradigmas e tecnologias, a estrutura base das redes 5G permanece praticamente a mesma das gerações anteriores, sendo constituída pelas redes de acesso, de transporte e de núcleo [3GPP 2020a]. Esta estrutura está ilustrada na Figura 1.

A RAN permite a conexão de dispositivos dos usuários finais (*User Equipement* – UE) à rede da operadora através de uma ampla variedade de tecnologias de acesso.

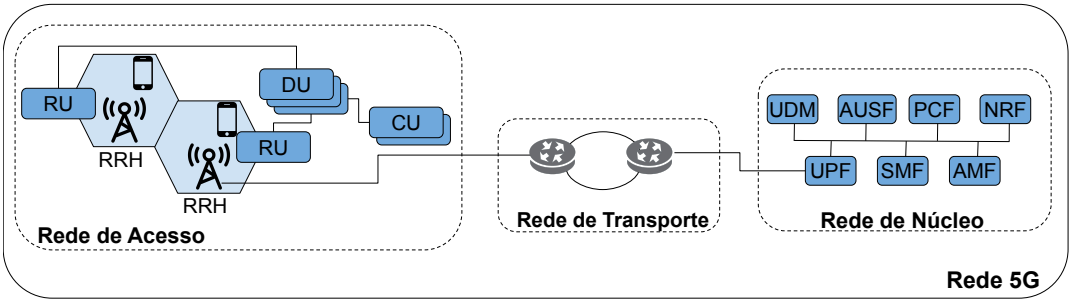

Figura 1. Arquitetura básica de uma rede 5G.

Entretanto, para atender os requisitos estabelecidos para as redes 5G, o *3rd Generation Partnership Project* (3GPP) define a *Next-Generation RAN* (NG-RAN) [3GPP 2020b]. A NG-RAN é constituída por um conjunto de estações bases, denominadas *Next-Generation Base Station* (gNBs), as quais proveem conectividade aos UEs através da tecnologia *5G New Radio* (NR). Além disso, para aproveitar os benefícios providos pela virtualização da RAN (*virtualized RAN* – vRAN) em termos de escalabilidade e centralização [Saavedra et al. 2018], uma gNB pode ser logicamente dividida em três entidades: *Radio Unit* (RU), *Distributed Unit* (DU) e *Centralized Unit* (CU). As funções do protocolo NR executadas em cada uma dessas entidades são determinadas pelo tipo de desagregação (*split*) escolhido, sendo definidos pelo 3GPP oito diferentes tipos [Larsen et al. 2019].

A rede de núcleo permite que os UEs enviem/recebam tráfego móvel de/para aplicativos hospedados na rede de dados ou na Internet. Entretanto, na 5G, a rede de núcleo foi concebida usando conceitos nativos de nuvem [Both et al. 2020]. Como consequência, a rede de núcleo segue um arquitetura baseada em serviços, com a definição de um plano de controle modular e totalmente desacoplado das funções de plano de usuário. A Figura 1 ilustra as principais funções de rede que compõem a rede de núcleo. A seguir, cada função é descrita resumidamente. A interação entre a RAN e a rede de núcleo, no plano de controle, é feito através da função de rede *Access and Mobility Management Function* (AMF). Essa função suporta o estabelecimento de conexões de sinalização criptografadas para os UEs, permitindo que esses UEs se registrem, sejam autenticados e se movam entre diferentes células de rádio na rede. A função de rede *Session Management Function* (SMF) gerencia as sessões do usuário, incluindo estabelecimento, modificação e liberação de sessões. A função de rede *Unified Data Management* (UDM) é responsável por gerenciar os dados dos usuários da rede de forma centralizada. A função de rede *Authentication Server Function* (AUSF) realiza a autenticação dos UEs usando as credenciais de acesso providas pela UDM. A função de rede *Policy Control Function* (PCF) provê controle de políticas para funcionalidade relacionadas com gerenciamento de acesso, sessões e mobilidade. A *Network Repository Function* (NRF) provê um repositório de funções de rede. Através da NRF, outras funções de rede conseguem encontrar a função de rede que oferece um determinado serviço. Por fim, a *User Plane Function* (UPF) é uma função de rede do plano de dados, sendo responsável por encaminhar e processar os dados dos UEs.

Finalmente, a rede de transporte é responsável por fornecer conectividade desde a RU (o ponto de entrada da rede para o UE) até a rede de dados, onde as aplicações estão hospedadas. Portanto, a rede de transporte envolve uma grande variedade de tecnologias, como IP/MPLS, óptico/DWDM e microondas/backhaul.

3. Trabalhos Relacionados

Vários trabalhos na literatura de redes e sistemas distribuídos propuseram mecanismos de monitoramento [Aceto et al. 2013, Lee et al. 2014], registros de *logs* [Chen and Jiang 2021] e coleta de *traces* [Mace et al. 2015, Sigelman et al. 2010, Esteves et al. 2021]. Entretanto, apenas recentemente, alguns trabalhos começaram a investigar o uso de métricas, *logs* e *traces* como forma de melhorar a observabilidade de sistemas de grande escala formados por vários microsserviços, como é o caso do sistema 5G.

Em [Scrocca et al. 2020], os autores propõem um modelo unificado de dados e processamento baseado em eventos com o objetivo de integrar informações obtidas de métricas, *logs* e *traces*. Usando o modelo proposto, os autores implementam um mecanismo para processar e analisar, em tempo quase real, as abstrações declarativas de processamento de fluxos de eventos. O mecanismo deste trabalho tem como foco a observabilidade de sistemas de orquestração de contêineres. Esse também é o foco em [Picoreti et al. 2018]. Os autores exploram o uso de métricas específicas de aplicações, combinadas com métricas de infraestrutura, para demonstrar como a observabilidade nas duas camadas (i.e., aplicação e infraestrutura) pode melhorar a orquestração de recursos em ambientes de computação em nuvem. Os autores desenvolvem um adaptador de API responsável por coletar métricas da aplicação usando a ferramenta Prometheus e convertem essas métricas para um formato que o *Horizontal Pod Autoscaler* (HPA) possa consumí-las. Esse último controla automaticamente o número de instâncias de um microsserviço em um *cluster* Kubernetes. Embora o trabalho investigue a observabilidade em diferentes níveis do sistema, a coleta de métricas é o único pilar considerado pelos autores. Os autores em [Karumuri et al. 2020] formulam a observabilidade de sistemas de grande escala como um problema de gerenciamento de grandes volumes de dados. Particularmente, os autores analisam os requisitos envolvidos no gerenciamento de cargas de trabalho geradas pela observabilidade e os desafios enfrentados nesse processo. Para realizar essa investigação, os autores utilizam como estudo de caso a solução de observabilidade presente na plataforma Slack[7], a qual provê quatro sistemas independentes (i.e., coleta de métricas, eventos, *logs* e *traces*) para implementar observabilidade.

Todos os trabalhos descritos anteriormente tem como foco a observabilidade de aplicações de computação em nuvem. De fato, a literatura atual possui apenas o trabalho de [John et al. 2017] que explora a observabilidade no contexto de redes 5G. Neste trabalho, os autores apresentam um arcabouço de monitoramento para arquiteturas de virtualização de funções de rede realizadas através de microsserviços. O arcabouço proposto estende o processo de observabilidade implementado no projeto UNIFY[8], tornando-o mais autônomo. Entretanto, o arcabouço considera apenas um pilar da observabilidade, coleta de métricas. Além disso, os autores não descrevem os softwares utilizados para emular as redes de acesso e de núcleo. De maneira similar a [John et al. 2017], o presente trabalho investiga a observabilidade em sistemas 5G. Entretanto, diferentemente de [John et al. 2017], esta investigação explora dois pilares da observabilidade. Adicionalmente, este trabalho considera um sistema 5G formado pela rede de núcleo e a rede de acesso. A Tabela 1 resume as principais características dos trabalhos analisados e ilustra como este trabalho se posiciona em relação à literatura.

[7]https://slack.com/

[8]https://www.fp7-unify.eu/

Tabela 1. Obserability work summary

Trabalho	Pilares			Sistemas 5G	
	Métricas	*Logs*	*Traces*	Core	RAN
[Scrocca et al. 2020]	✓	✓	✓		
[Picoreti et al. 2018]	✓				
[Karumuri et al. 2020]	✓	✓	✓		
[John et al. 2017]	✓			✓	✓
Este Trabalho	✓	✓		✓	✓

4. Observabilidade em Sistemas 5G

Esta seção apresenta a metodologia e a *testbed* implantada para a realização do estudo empírico sobre observabilidade em sistemas 5G. Mais especificamente, a seção destaca as opções escolhidas para a implantação da *testbed*, tanto do ponto de vista do hardware, quanto do software.

Para emular um sistema 5G em um ambiente real, foi criada uma infraestrutura de rede composta por três servidores DELL PowerEdge M610 equipados com dois processadores Intel Xeon X5660 e 192GB de RAM. Em cada servidor, uma máquina virtual (*Virtual Machine* – VM) foi criada e o sistema operacional Ubuntu 18 com kernel *low latency* foi instalado. Cada VM foi configurada com dezoito CPU(s), 24GB de RAM e um disco com capacidade de 50GB. Como *hipervisor*, o VMware ESXi 6.7 foi utilizado. Além disso, um *switch* físico, conectado com os servidores através de um *switch* virtual, foi utilizado para emular a rede de transporte. Cada *link* dessa conexão tem uma banda reservada de 10 Gbps e 1ms de latência garantida e cada VM se conecta a um *switch* virtual. Sobre a infraestrutura, foi implantado um *cluster* Kubernetes (K8S) com três nós, um deles executando o plano de controle do *master* e os outros dois desempenhando o papel de *workers*. Cada nó K8S foi implantado em uma VM da infraestrutura. Como interface de rede do contêiner (*Container Network Interface - CNI*) foi utilizado o Calico[9]. Atualmente, o K8S é a ferramenta de orquestração e gerenciamento de infraestruturas computacionais baseadas em contêineres adotada pela CNCF. Além disso, há uma tendencia acentuada do uso dessa ferramenta pela indústria de telecomunicação [Polese et al. 2022]. Esses dois motivos guiaram a escolha do K8S neste trabalho.

A Figura 2 detalha o *cluster* K8S implantado. A emulação da RAN e do UE é realizada pela ferramenta OpenAirInterface, um projeto de código aberto mantido pela OpenAirInterface Software Alliance (OSA) que implementa o padrão 3GPP em hardware de propósito geral e rádio definido por software (*Software Defined Radio - SDR*) [Alliance 2020a]. A emulação da rede de núcleo é feita pela ferramenta Free5GC, uma das primeiras implementações de código aberto de um núcleo 5G baseado em serviço e que implementa a especificação da *Realease 15* definido pelo 3GPP [Free5GC 2021]. Ambas ferramentas são amplamente utilizadas em estudos empíricos envolvendo o 5G [Tan et al. 2020, Kaltenberger et al. 2020, Morais et al. 2021]. Além disso, o OpenAirInterface opera com desagregação da RAN. Para o posicionamento e implantação das funções de rede do OpenAirInterface (i.e., RU, DU e CU) nos nós do *cluster*, a solução OPlaceRAN [Morais et al. 2021] foi utilizada. Essa solução é capaz de implantar, de forma dinâmica e automática, as funções de rádio que compõem a RAN em *Pods* do K8S, tendo como entrada as características da topologia da rede. Adicionalmente, para emular a RAN, a camada

[9]https://projectcalico.docs.tigera.io/about/about-calico

PHY é desabilitada e o UE é emulado de forma co-localizada com a RU. Como mostrado na Figura 2, de acordo com a topologia da rede usada na *testbed*, a RU foi posicionada no nó 3, a DU no nó 2, enquanto a CU foi implantada no nó 1. Posteriormente, as funções de rede do Free5GC foram implantadas como um único *Pod* executando no nó 1.

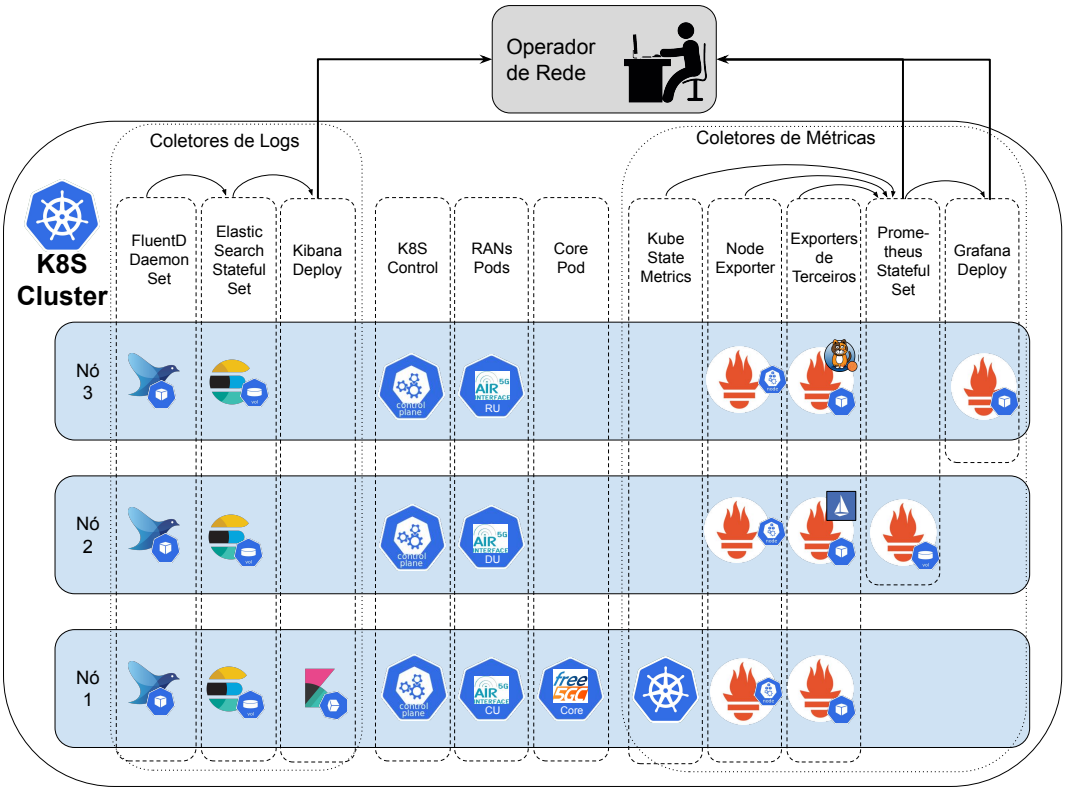

Figura 2. *Cluster* K8S utilizado nesta investigação.

Para a coleta de métricas no nível do *cluster* e dos *Pods* (e.g., CPU e memória) foi utilizada a ferramenta Prometheus, a qual obtém dados através de um elemento Node Exporter instalado em cada nó. Para a coleta de métricas relacionadas à rede, o Prometheus utiliza *Exporters* de terceiros, como por exemplo, o usado para coletar o status do Calico. A visualização das métricas coletadas é realizada através da ferramenta Grafana. Finalmente, a coleta de *logs* é obtida por uma pilha de software composta pelo Elasticsearch, Fluentd e Kibana (EFK *Stack*). Nessa pilha, o Elasticsearch é o mecanismo que permite consulta e análise dos *logs*, sendo comumente usado para indexar e consultar grandes volumes de dados. A ferramenta Kibana é o *front-end* de visualização de dados. O Fluentd, por sua vez, é a ferramenta responsável pela coleta dos *logs* gerados pelos *Pods*, a filtragem dos dados coletados e o envio para o Elasticsearch. Além disso, as métricas são coletadas a cada 30 segundos e armazenadas em um dispositivo de armazenamento no mesmo data center.

5. Avaliação Experimental

Nesta seção, os dois cenários de testes desenhados com o propósito de avaliar a observabilidade do sistema 5G descrito na Seção 4 são descritos. A Subseção 5.1 descreve os resultados obtidos para um cenário de teste de conectividade, enquanto a Subseção 5.2 apresenta os resultados para um cenário de teste de desempenho. Além disso, uma discussão sobre a qualidade dos *logs* extraídos do sistema é apresentada na Subseção 5.3.

5.1. Teste de conectividade

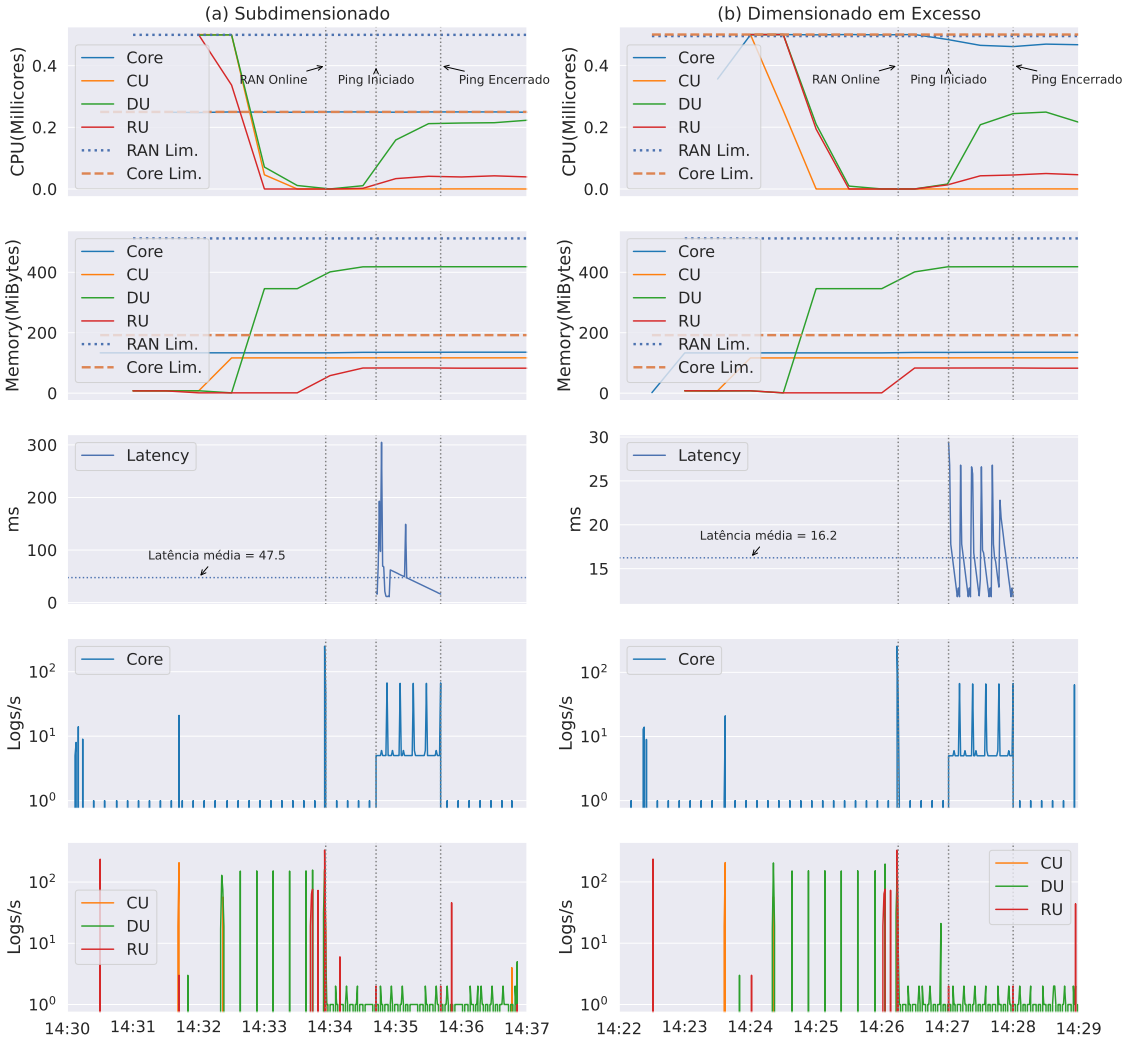

Figura 3. Resultados da execução do cenário de teste de conectividade. A primeira coluna (a) representa resultados para o teste com subdimensionamento de recurso, enquanto a segunda coluna (b) ilustra os resultados do teste com dimensionado de recurso em excesso.

No primeiro cenário de teste, o sistema 5G em estudo é submetido a uma falha de provisionamento de recursos durante um teste de conectividade entre um UE e a rede de dados. O objetivo deste teste é verificar se o uso de métricas e/ou *logs* extraídas do sistema é capaz de detectar a falha injetada. Para atingir esse objetivo, os seguintes passos são executados. Primeiramente, a quantidade de CPU provisionada para o *Pod* que executa os serviços da rede de núcleo é subdimensionada, de forma que esse *Pod* receba apenas a metade (0.25 Millicores) do recurso necessário para seu funcionamento normal (0.5 Millicores). A memória é provisionada com recursos suficientes para a operação de todos os *Pods*. Em seguida, os *Pods* da RAN e da rede de núcleo são iniciados (E1). Após a inicialização de todo o sistema, a ferramenta *ping* é utilizada para injetar sondas na sessão de dados estabelecida entre o UE e a rede de dados, usando o UE como origem das sondas (E2). Finalmente, a geração de sondas é encerrada (E3). O consumo de CPU e memória dos *Pods* correspondentes aos serviços da RAN e da rede de núcleo, bem como

a latência média das sondas enviadas, são as métricas analisadas neste cenário. Os *logs* extraídos dos *Pods* executando esses serviços são analisados em relação ao número de mensagens escritas por unidade de tempo. A Figura 3(a) apresenta os resultados do teste com subdimensionamento de recursos, enquanto a Figura 3(b) ilustra os resultados de um teste similar, porém onde os recursos são provisionados em excesso. Nas figuras, o eixo X representa o tempo ao longo dos testes. As linhas pontilhadas verticais em cada gráfico representam os instantes de tempo em que os eventos E1, E2 e E3 ocorrem em ambos os testes. A primeira e a segunda linha da Figura 3 representam, respectivamente, o consumo de CPU e da memória dos *Pods* observados ao longo do teste. Os rótulos "RAN Lim" e "Core Lim" nestas linhas representam os valores máximo de recurso provisionados para esses recursos. A terceira linha representa a latência observada para as sondas enviadas ao longo dos testes, bem como o valor médio obtido. A quarta e quinta linha representam a taxa de escrita nos *logs* dos serviços da rede de núcleo e da RAN, respectivamente. Observa-se que o consumo de CPU no *Pod* dos serviços da rede de núcleo se mantém no limite máximo durante todo o teste com subdimensionamento. Observa-se também que a latência média das sondas enviadas pelo *ping* nesse teste é três vezes maior que o teste com provisionamento em excesso. Essas observações, por sua vez, são suficientes para identificar e diagnosticar o problema de subdimensionamento de recursos nos serviços da rede de núcleo. Entretanto, como mostrado nas figuras, o comportamento dos *logs* ao longo dos dois testes é muito similar e não contribui para a detecção da anomalia. Conclui-se, portanto, que, neste cenário, a coleta e análise de métricas é suficiente para detectar a anomalia injetada.

5.2. Teste de desempenho

No segundo cenário de teste, o sistema 5G implantado é submetido a uma falha de provisionamento de recursos semelhante ao do cenário anterior, porém durante um teste de desempenho entre um UE e a rede de dados. Novamente, o objetivo desse teste é verificar se o uso de métricas e/ou *logs* extraídas do sistema é capaz de detectar a falha injetada. Para atingir esse objetivo, os seguintes passos são executados. Primeiramente, a quantidade de CPU provisionada para o *Pod* que executa os serviços da rede de núcleo é subdimensionada, de forma que esse *Pod* receba apenas uma fração (1 Millicores) do recurso necessário para seu funcionamento normal (4 Millicores). A memória é provisionada com recursos suficientes para a operação de todos os *Pods*. Em seguida, os *Pods* da RAN e da rede de núcleo são iniciados (E1). Após a inicialização de todo o sistema, a ferramenta *Iperf* é utilizada para gerar tráfego entre o UE e a rede de dados, usando o UE como origem dos dados (E2). No entanto, observou-se que após o primeiro minuto da inicialização da geração do tráfego, a sessão de dados estabelecida entre o UE e a rede de dados é encerrada de forma anormal (E3). Em seguida, a geração de tráfego é encerrada (E4). O consumo de CPU e memória dos *Pods* correspondentes aos serviços da RAN e da rede de núcleo, bem como a vazão obtida entre UE e rede de dados são as métricas analisadas neste cenário. Os *logs* extraídos dos *Pods* executando esses serviços são analisados em relação ao número de mensagens escritas por unidade de tempo. A Figura 4(a) apresenta os resultados do teste com subdimensionamento de recursos, enquanto a Figura 4(b) ilustra os resultados de um teste similar, porém onde os recursos são provisionados em excesso. Nas figuras, o eixo X representa o tempo ao longo dos testes. As linhas pontilhadas verticais em cada gráfico representam os instantes de tempo em que os eventos E1, E2, E3 e E4 ocorrem em ambos os testes. Entretanto, como o teste com recursos em excesso não apresenta erros, o evento E3 (anormalidade) não é observado para esse experimento. A primeira e a segunda linha da Figura 4 representam, respectivamente, o consumo de CPU

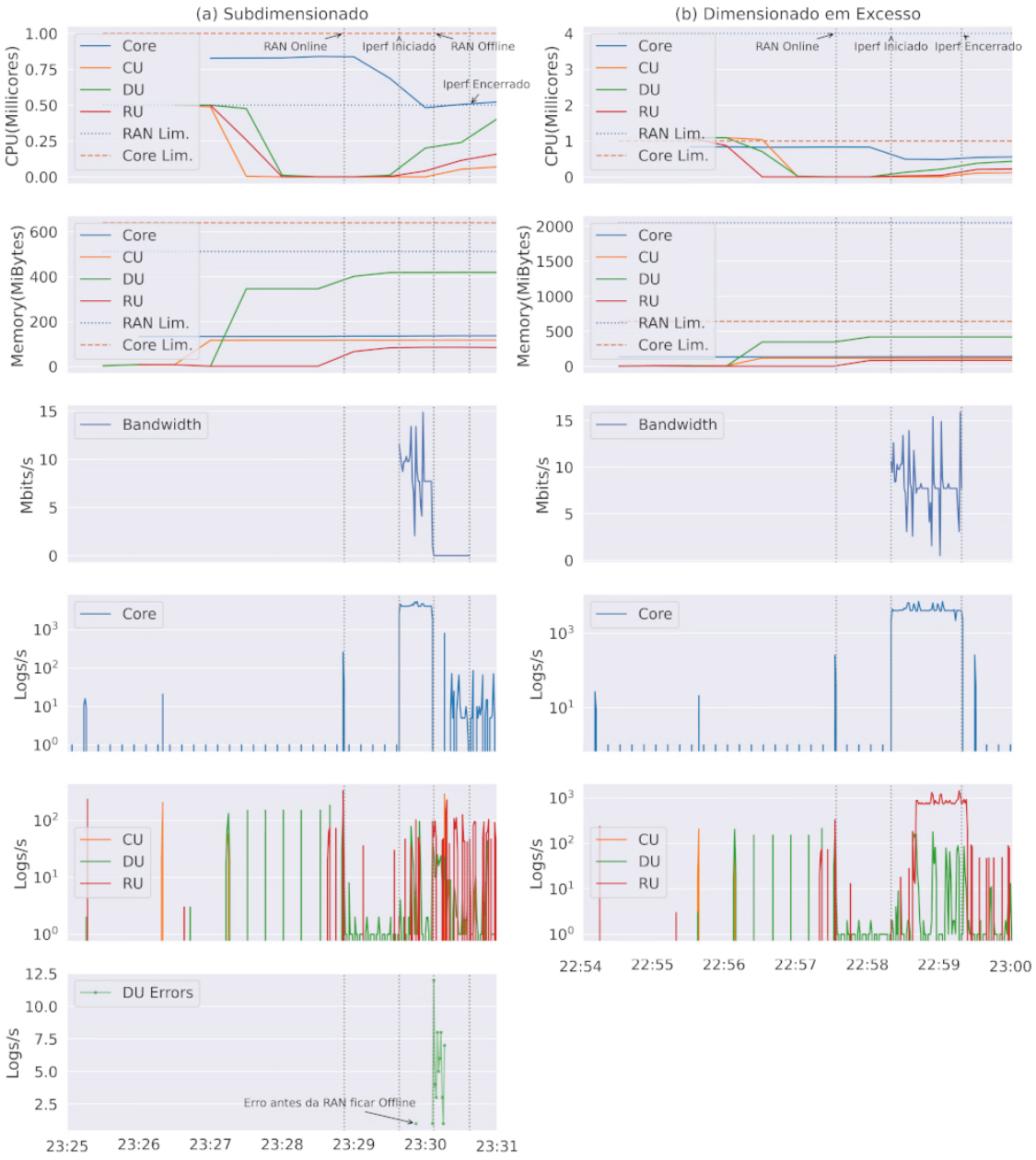

Figura 4. Resultados da execução do cenário de teste de desempenho. A primeira coluna (a) representa resultados para o teste com subdimensionamento de recurso, enquanto a segunda coluna (b) ilustra os resultados do teste com dimensionado de recurso em excesso.

e da memória dos *Pods* observados ao longo do teste. Os rótulos ”RAN Lim” e ”Core Lim” nessas linhas representam os valores máximos provisionados para esses recursos. A terceira linha representa a vazão observada ao longo dos testes a partir do momento em que a geração de tráfego foi iniciada. A quarta e a quinta linha representam a taxa de escrita nos *logs* dos serviços da rede de núcleo e da RAN, respectivamente. Finalmente, a sexta linha representa os momentos em que o *Log* da DU emite uma mensagem anômala, a qual não é observada no teste com provisionamento de recursos em excesso. Observa-se que o consumo de CPU nos *Pods* dos serviços da RAN aumenta em ambos os testes, a

partir do início da geração do tráfego. Além disso, observa-se que, após a sessão de dados do UE se encerrar de forma anormal no teste com subdimensionamento, a vazão cai para zero. Antes desse momento, porém, o *log* da DU emite uma mensagem anormal, como mostrado no última linha da Figura 4(a). Essa mensagem é emitida repetidamente antes da vazão zerar. Portanto, neste cenário, a coleta e análise de *logs* é capaz de detectar a anomalia antes dela ser capturada pelas métricas. Conclui-se que o uso combinado de métricas e *logs* aumenta as chances de antecipação de problemas em um sistema 5G.

5.3. Qualidade dos *Logs*

Neste trabalho também analisou-se a qualidade das mensagens registradas nos *logs* do Free5GC e do OpenAirInterface. Em geral, para serem úteis, é importante que as mensagens de *logs* sejam acompanhadas de uma estampa de tempo, expressem o nível de gravidade da informação (e.g., INFO, WARN, ERROR) e o serviço/processo onde foram geradas. É importante também que as mensagens sejam inteligíveis. A Figura 5 mostra o painel de visualização do Kibana com algumas mensagens coletadas do *log* da RU. É possível notar que as mensagens são pouco estruturadas e pouco inteligíveis. Infelizmente, essa constatação foi observada nos *logs* das demais unidades da RAN e dos serviços da rede de núcleo. Portanto, apesar dos *logs* dos serviços terem ajudado a antecipar uma anormalidade, suas mensagens não são claras e requerem um esforço não negligenciável para compreendê-las.

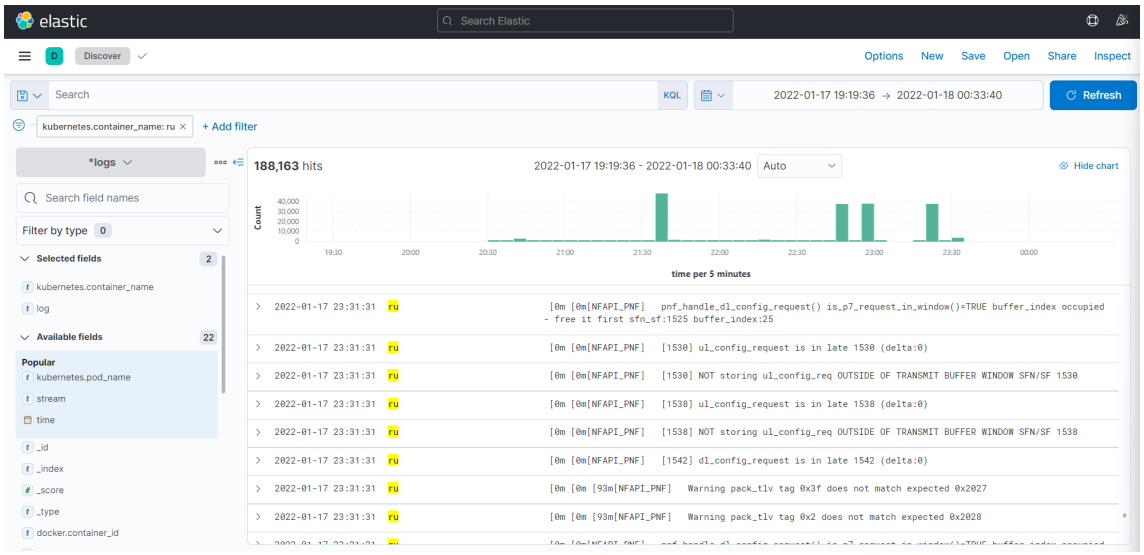

Figura 5. Mensagens de *log* vistas através da *interface* do Kibana.

6. Conclusão e Trabalhos Futuros

Este trabalho apresentou uma investigação empírica sobre observabilidade em sistemas 5G. Em particular, foi mostrado como a análise combinada de métricas e *logs* pode melhorar a observabilidade de sistemas 5G e ajudar a antecipar a ocorrência de falhas. Em geral, observabilidade é uma área de pesquisa emergente em computação em nuvem e requer mais investigação da comunidade de redes, especialmente no uso de outros pilares além do monitoramento tradicional. Como trabalhos futuros, pretende-se integrar aprendizagem de máquina à investigação realizada e agregar análise de *traces* para abranger, de forma completa, todos os pilares da observabilidade no contexto das redes 5G.

Agradecimentos

Os autores agradecem à Rede Nacional de Ensino e Pesquisa (RNP) pelo apoio financeiro via Programa de Monitoramento (PMon) 2021 e à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) por meio dos Projetos Slicing Future Internet Infrastructures (SFI2), concessão 2018/23097-3, e SAMURAI: núcleo 5G inteligente e integração de múltiplas redes de acesso, concessão 20/05127-2.

Referências

3GPP (2020a). 5G; System architecture for the 5G System (5GS) (3GPP TS 23.501 version 16.6.0 Release 16). Technical report, 3GPP.

3GPP (2020b). Study on New Radio Access Technology; Radio Access Architecture and Interfaces (Release 14). Technical report, 3GPP.

Aceto, G. et al. (2013). Cloud monitoring: A survey. *Computer Networks*, 57(9):2093–2115.

Agiwal, M., Roy, A., and Saxena, N. (2016). Next Generation 5G Wireless Networks: A Comprehensive Survey. *IEEE Communications Surveys Tutorials*, 18(3):1617–1655.

Alliance, O. S. (2020a). OpenAirInterface — 5G software alliance for democratising wireless innovation. Disponível em:https://openairinterface.org/. Acessado em 01/02/2022.

Alliance, O. S. (2020b). Openairinterface 5G Wireless Implementation. Disponível em: https://gitlab.eurecom.fr/oai/openairinterface5g/. Acessado em 01/02/2022.

Balalaie, A., Heydarnoori, A., and Jamshidi, P. (2016). Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture. *IEEE Software*, 33(3):42–52.

Bhanage, D. A., Pawar, A. V., and Kotecha, K. (2021). IT Infrastructure Anomaly Detection and Failure Handling: A Systematic Literature Review Focusing on Datasets, Log Preprocessing, Machine & Deep Learning Approaches and Automated Tool. *IEEE Access*, 9:156392–156421.

Both, C. B. et al. (2020). Soft5G+: explorando a softwarização nas redes 5G. In *Minicursos do XXXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, chapter 3, pages 91–139. Sociedade Brasileira de Computação.

Chen, B. and Jiang, Z. M. J. (2021). A Survey of Software Log Instrumentation. *ACM Comput. Surv.*, 54(4).

CNCF (2022). Cloud native glossary. Disponível em: https://glossary.cncf.io/observability/. Acessado em 03/02/2022.

Contreras, L. M. et al. (2020). Computing at the Edge: But, what Edge? In *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–9.

Esposito, C., Castiglione, A., and Choo, K.-K. R. (2016). Challenges in Delivering Software in the Cloud as Microservices. *IEEE Cloud Computing*, 3(5):10–14.

Esteves, T., Neves, F., Oliveira, R., and Paulo, J. a. (2021). CAT: Content-Aware Tracing and Analysis for Distributed Systems. In *Proceedings of the 22nd International Middleware Conference*, page 223–235. Association for Computing Machinery.

Free5GC (2021). Open source 5G core network base on 3GPP R15. Disponível em: https://github.com/free5gc/free5gc. Acessado: 01/02/2022.

Google (2022). Google Cloud's operations suite (formerly Stackdriver). Disponível em: https://cloud.google.com/products/operations. Acessado em 03/02/2022.

John, W. et al. (2017). Meeting the Observability Shallenges for VNFs in 5G Systems. In *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 1127–1130.

Kaltenberger, F. et al. (2020). OpenAirInterface: Democratizing innovation in the 5G Era. *Computer Networks*, 176:107284.

Karumuri, S. et al. (2020). Towards Observability Data Management at Scale. In *SIGMOD Record*, volume 49.

Larsen, L. M. P., Checko, A., and Christiansen, H. L. (2019). A Survey of the Functional Splits Proposed for 5G Mobile Crosshaul Networks. *IEEE Communications Surveys Tutorials*, 21(1):146–172.

Lee, S., Levanti, K., and Kim, H. S. (2014). Network monitoring: Present and future. *Computer Networks*, 65:84–98.

Mace, J., Roelke, R., and Fonseca, R. (2015). Pivot Tracing: Dynamic Causal Monitoring for Distributed Systems. page 378–393. Association for Computing Machinery.

Meng, S. and Liu, L. (2013). Enhanced Monitoring-as-a-Service for Effective Cloud Management. *IEEE Transactions on Computers*, 62(9):1705–1720.

Microsoft (2021). Cloud monitoring guide: Observability. Disponível em: https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/manage/monitor/observability. Acessado em 03/02/2022.

Morais, F. Z. et al. (2021). OPlaceRAN – a Placement Orchestrator for Virtualized Next-Generation of Radio Access Network.

Narayanan, A. et al. (2021). A Variegated Look at 5G in the Wild: Performance, Power, and QoE Implications. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, page 610–625, New York, NY, USA. Association for Computing Machinery.

Picoreti, R. et al. (2018). Multilevel Observability in Cloud Orchestration. In *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing*, pages 776–784.

Polese, M. et al. (2022). Understanding O-RAN: Architecture, Interfaces, Algorithms, Security, and Research Challenges.

Russ, M. (2019). *Chaos Engineering Observability*. O'Reilly Media, Inc.

Saavedra, A. et al. (2018). WizHaul: On the Centralization Degree of Cloud RAN Next Generation Fronthaul. *IEEE Transactions on Mobile Computing*, 17(10):2452–2466.

Scrocca, M. et al. (2020). The Kaiju Project: Enabling Event-Driven Observability. *Proceedings of the 14th ACM International Conference on Distributed and Event-Based Systems*, pages 84–91.

Sigelman, B. H. et al. (2010). Dapper, a Large-Scale Distributed Systems Tracing Infrastructure. Technical report, Google, Inc.

Sridharan, C. (2018). *Distributed Systems Observability*. O'Reilly Media, Inc.

Tan, T.-J. et al. (2020). *A Reliable Intelligent Routing Mechanism in 5G Core Networks*. Association for Computing Machinery.