THE PERSPECTIVES OF USING FREEBSD IN CLUSTER ARCHITECTURES

Maria-Elena Mihailescu[1], Darius Mihai[2], Mihai Carabas[3], Nicolae Tapus[4]

Managing a cluster infrastructure is a difficult task: system administrators must choose the most suitable hardware and software solutions, while making trade-offs and checking various configurations. When choosing the operating system that will run on the cluster's servers, system administrators must consider system's security and stability and resource allocation, isolation and orchestration methods.

This paper analyses the requirements a cluster-centered operating system must meet and shows that FreeBSD is a suitable candidate for running in cluster architectures. Then, a hybrid architecture for a general-purpose cluster is proposed. Afterwards, the current status of deploying the proposed configuration in National University of Science and Technology Politehnica Bucharest's cluster infrastructure and the issues we encountered are presented.

Keywords: FreeBSD, cluster, OpenStack, operating system requirements, heterogeneous architecture

1. Introduction

In a cluster architecture, multiple physical systems are configured to share their resources. For a good system management, system administrators decide on the operating systems that better fit their configuration, budget, security and performance requirements. Moreover, for a better end-user access, resource management and monitoring, system administrators may choose to implement a private cloud infrastructure using specific software such as OpenStack, Apache CloudStack, OpenNebula or VMWare vSphere.

[1]PhD student, Faculty of Automatic Control and Computer Science, National University of Science and Technology POLITEHNICA Bucharest, Romania, e-mail: maria.mihailescu@upb.ro

[2]PhD student, Faculty of Automatic Control and Computer Science, National University of Science and Technology POLITEHNICA Bucharest, Romania, e-mail: darius.mihai@upb.ro

[3]Professor, Faculty of Automatic Control and Computer Science, National University of Science and Technology POLITEHNICA Bucharest, Romania, e-mail: mihai.carabas@upb.ro

[4]Professor, Faculty of Automatic Control and Computer Science, National University of Science and Technology POLITEHNICA Bucharest, Romania, e-mail: nicolae.tapus@upb.ro

There are multiple server-optimized operating systems that one can choose for their cluster environment. Even though some are Windows-centered, the majority are part of the UNIX family. Nevertheless, the UNIX family has multiple sub-families: RedHat-based, Debian-based or BSD-based operating systems, the Linux-based operating systems being the preferred ones.

BSD-based operating systems differ from the Linux-based ones by their development philosophy. Linux-based distributions are built over the Linux kernel. In contrast, the BSD distributions (derived from 4.4BSD) come as a complete solution, the kernel and userspace applications being developed and shipped together.

National University of Science and Tehnology Politehnica Bucharest (UNSTPB) has a general-purpose private cloud that is used for many use-cases. It provides students and researchers access to resources for laboratories or research activities, it runs various web and e-learning applications or provides means for High Performance Computing (HPC). However, at the moment this paper is written, it runs only on Linux-based operating systems.

There is no major performance comparison between FreeBSD and Linux, both having their advantages and disadvantages. Moreover, various FreeBSD features were researched in UNSTPB: bhyve for ARM platforms [23, 21], save-restore for bhyve [20, 26, 22], migration for bhyve [18, 17], libvdsk support in FreeBSD [19, 25] or OpenStack and FreeBSD related projects [24]. Being a general-purpose cluster and given our expertise by developing features for FreeBSD, some of the research projects that are developed in UNSTPB may benefit of a FreeBSD environment.

This paper presents the perspectives of using FreeBSD components in cluster architectures as they were tested in the National University of Science and Tehnology Politehnica Bucharest. We explored various methods of using FreeBSD, from using it as a cloud image or a compute node in our private cloud infrastructure (OpenStack) to a bare-metal operating system on server nodes to benefit of using the bhyve hypervisor. Nevertheless, some of the components could not be integrated in our infrastructure, thus, we consider implementing the necessary changes.

This paper is structured as follows: the second section presents the requirements for an operating system to be suitable for cluster infrastructures and describes FreeBSD's features and particularities. The third section presents various FreeBSD components developed in UNSTPB and proposes a heterogeneous architecture that integrates FreeBSD in the UNSTPB's cluster. The proposed architecture would benefit both end-users and system administrators. Section four presents the current status of architecture deployment while the final section shows this paper's conclusions.

2. Considerations related to cluster management

Managing a cluster infrastructure can become a difficult task for system administrators [3, 5, 8, 14]. They must find the most suitable architecture, middleware and tools to manage the given hardware resources while respecting their requirements and constraints. Some examples of complex cluster, grid or cloud architectures [1] that must be configured and fine-tuned for serving their purposes are the following: edge-computing [5], HPC architectures [8] or application-centered solutions [14].

2.1. Requirements for the server-centred operating systems

Based on the literature, we extracted a set of aspects that must be considered when managing a cluster architecture:

- **(on-demand) resource allocation** - resource management [3] defines how the cluster is used and how the resources are allocated. However, it is not a trivial task since there are multiple allocation algorithms [11], each having its advantages and disadvantages.
- **resource isolation** - given that the cluster will run the user's application, each job must be isolated from the others (i.e., a user's activity must not be affected by others, and their data must not be visible to others). The isolation is usually implemented by using virtualization [5, 6, 10, 14], either by using native virtualization (type-1 or type-2 hypervisors) or containers (lightweight virtualization).
- **security** - while part of the security constraints are resolved by resource isolation, other aspects must be considered [8]: network hardening, firewalls, VPNs and bastion hosts, authorized access.
- **easy resource checkpoint and migration** - for backup, load-balancing or hardware/software operations, the virtualized environments (virtual machines) must be, when needed, checkpointed or migrated on other available hosts [6, 10, 18] with minor downtime.
- **storage and network performance** - considering each cluster's scope, various design choices and fine-tuning configurations must be done to achieved the desired performance. An extensive such configuration is presented in [8] where authors show their expertise in setting up efficient storage (using NFS, Lustre, LVM and RAID6) and network (dedicated network connection between clusters) configurations.
- **high-availability and redundancy** - critical infrastructures must implement various mechanisms to ensure data [8] and running redundancy [3].
- **software stability** - all-level software stack choices must consider software stability over time [8, 6] so that little to no changes are needed over time.

- **configuration management** - while configuring many nodes, system administrators must ensure uniform setup across nodes, thus a configuration management tool (Puppet, Ansible) is required [8].
- **resource orchestration** - while considering each particular aspect of the infrastructure, system administrators also need an orchestration method to overview and better manage their cluster [5, 8, 14].

The previously described aspects are some generic ones. Each system administrator must decide what their desired use-cases are and must choose what better suits their infrastructure (e.g., generic vs. specialised cluster, high-performance or low-power centred).

2.2. FreeBSD as server-centered operating system

FreeBSD[1] is one of the most used BSD-based distributions. It has numerous features that can be used in server environments: isolation using bhyve (native virtualization) or jails (containerization tool), sandboxing using CAPSICUM, network virtualization, native ZFS support.

While Unix-based operating systems dominate the server market, system administrators tend to search for the most suitable, best performance one. There is no concluding study that states that one operating system is better than the other. Instead, each operating system has its advantages and disadvantages, and it is the system administrator's task to choose the one that has better performance for their dedicated use-cases.

The literature presents various comparisons between FreeBSD and other operating systems [2, 4, 6, 12, 7, 10, 16] on multiple research directions: sandboxing mechanisms (FreeBSD, OpenBSD and Linux) [2], resource isolation mechanisms (containers in FreeBSD and Linux [6, 7, 10]), schedulers [12], non-volatile memory (NVRAM) [16] or applications (DNS64 implementations for FreeBSD and Linux) [15]. These studies show that, depending on the usage, there are cases when FreeBSD is recommended (e.g., jails have better network performance compared to Docker [7], FreeBSD components seem to be more stable [6]) and others where Linux is recommended (e.g., bind9 has better performance for DNS64 on Linux than on FreeBSD [15]).

Furthermore, the literature states that FreeBSD has the following particularities:

- software stability and coherence - for Capsicum [2], containers [6], ULE scheduler when used for a long period[12], bind9 [15].
- security - Capsicum support [2, 13], bhyve hypervisor uses Capsicum [2], jails usage [10], distributed denial of service (DDoS) response [4].
- monitoring features [9].

[1]The FreeBSD Project, Online: https://www.freebsd.org/, last accessed 6th of September 2023

2.3. Discussion regarding the usage of FreeBSD in cluster infrastructures

Considering the literature review, there is no perfect operating system for a server. In some cases, Linux has a better performance, while in others FreeBSD does. It is always the system administrator's task to study the market and choose the most suitable operating system for their cluster needs.

Taking into account that FreeBSD has tools for resource allocation and isolation (bhyve and jails), advanced security mechanisms (Capsicum), is stable, has good storage (ZFS) and network management features, we consider that FreeBSD is a suitable candidate for usage in a cluster environment. Furthermore, FreeBSD development teams from UNSTPB may benefit from this since they can shift from using local hardware resources (e.g., personal computers) to cloud ones. Moreover, UNSTPB cluster team may conduct research activities of heterogeneous clusters.

3. FreeBSD components in Cluster Architecture

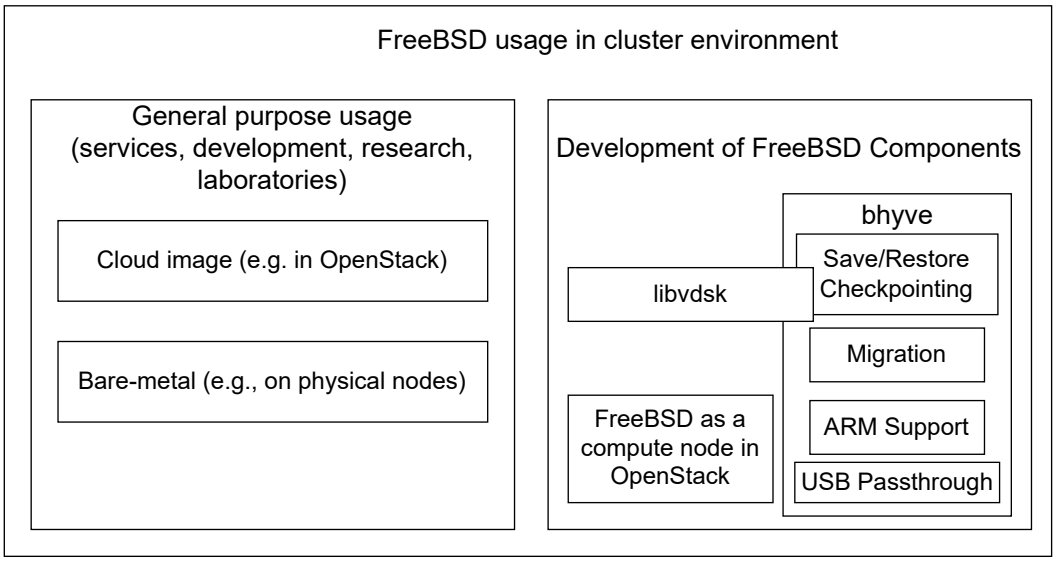

Fig. 1. FreeBSD components that are used, developed or considered for improvements in UNSTPB

UNSTPB's private-cloud infrastructure is orchestrated using OpenStack. Thus, our research explores the FreeBSD components that may be used in cluster architectures. Figure 1 shows the FreeBSD components considered by the researchers from UNSTPB: from system administration to end-user centered ones. Most of the components presented in Figure 1 were partially, under or considered to be developed in UNSTPB:

- bhyve save/restore and checkpoint features [18, 20, 26, 22] - students from UNSTPB worked on implementing the save/restore functionality that was

added into upstream in 2019[2] (at the moment this paper is written, the code must be manually compiled using `WITH_BHYVE_SNAPSHOT` to activate this feature). Moreover, various improvement projects have been started since then: support for multiple same-type devices, CAPSICUM support for save/restore, checkpoint using ZFS, checkpoint using libvdsk.

- bhyve migration features - a migration feature is a necessity in cluster environments. Thus, progress towards implementing warm and live migration was started in UNSTPB [18, 17].
- bhyve on ARM - UNSTPB is also exploring porting bhyve on ARM [23, 21].
- FreeBSD as a compute node in OpenStack - a project that aimed to use FreeBSD as a computed node in OpenStack started in 2020 [24] when we managed to run a FreeBSD compute node and create a virtual machine without network. However, disabling the checks made by the `oslo-privsep` module was required for this to work. It is worth mentioning that parallel progress towards OpenStack support in FreeBSD is also implemented by the FreeBSD Foundation[3].
- libvdsk for bhyve - bhyve is currently using only raw disk files. `libvdsk` aims to bring support for various file formats in bhyve.

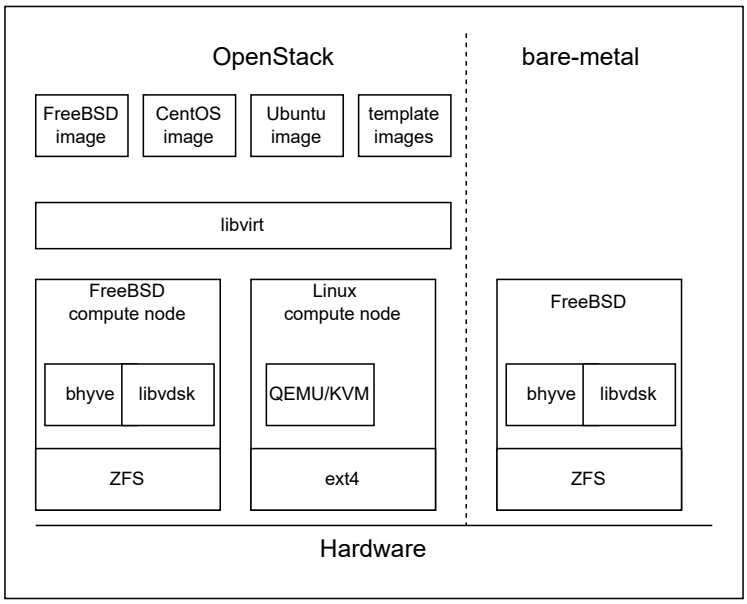

Fig. 2. Perspectives of integrating FreeBSD components in UNSTPB's cluster

[2]FreeBSD Code, online: https://github.com/freebsd/freebsd-src, last accessed 6th of September 2023

[3]The FreeBSD Foundation, OpenStack on FreeBSD, online: https://www.freebsd.org/status/report-2022-10-2022-12/openstack-on-freebsd/, last accessed: 6th of September 2023

Adding the previously mentioned features to FreeBSD, we can obtain high-availability, redundancy and resource migration by using bhyve's save/restore, checkpoint and migration features. Moreover, by using OpenStack and FreeBSD compute nodes, we can obtain resource orchestration and allocation. Considering these properties, as well as the ones we presented in section 2.**2**, we conclude that FreeBSD meets all the requirements we have presented in 2.**1**.

It is worth mentioning that having a cloud infrastructure to easily test bhyve changes would help developers since changes in the code-base means that the code must be recompiled (for each change, the kernel and/or userspace must be recompiled and installed; also, compiling such large code-base needs many resources). Moreover, modifying the kernel structures may lead to kernel panics and disk image corruption (which may further lead to code loss and environment regeneration). Thus, a snapshotting and reverting mechanism is needed. Furthermore, having the bhyve development setup in the cloud, users are not bound to physical computers and developers can work remotely.

4. **Current Status**

The scope of this study is to present the perspectives of using FreeBSD in cluster infrastructures. Thus, we will not present the status of the development projects[4]. However, without these features being added into upstream, we cannot fully integrate FreeBSD in our cluster environment. Thus, we only tested parts of the proposed cluster architecture in a testing environment. This section presents the current results we have obtained.

4.1. **FreeBSD and bhyve on bare-metal and in nested environments**

Considering the many bhyve development projects from UNSTPB, our end goal is to check if the FreeBSD cloud images can run bhyve in our server architecture. However, bhyve does not implement nested virtualization yet, and developers must use a FreeBSD virtual machine in a Linux environment in OpenStack for testing changes in the bhyve code base. Thus, we also considered Linux servers.

We considered for testing two servers with Intel(R) Xeon(R) CPU E5-2660 v4 @ 2.00GHz (14 cores, hyperthreading enabled) with 380GB of RAM. At the time of these tests were executed, we used FreeBSD 13.0 amd64 with bhyve for one of the servers and various Linux distributions with kvm for the other one. On the FreeBSD server we tested bhyve, while on the Linux server, we started a FreeBSD virtual machine, and then, tested bhyve.

Table 1 summarizes the results we have obtained while testing FreeBSD using the before mentioned scenario. While bare-metal FreeBSD and bhyve worked fine, on this server, we could not manage to use bhyve in KVM, even

[4]FreeBSD Projects in UNSTPB, online: https://github.com/FreeBSD-UPB/freebsd-src/wiki, last accessed: 6th of September 2023

though we tried various KVM configurations (disable `x2lapic`, tested both `host-model` and `host-passthrough` configurations in libvirt).

Table 1

FreeBSD and bhyve tests on bare-metal and nested virtualization

	FreeBSD bare-metal	**CentOS 8 (host)**		**Ubuntu (host)**
		FreeBSD12 (guest)	**FreeBSD 13 (guest)**	**FreeBSD13 (guest)**
bhyveload	ok	VM is blocked (no interraction)	kernel panic in bhyve guest	kernel panic in bhyve guest
uefi	ok	VM is blocked (no interraction)	VM is blocked (no interraction)	VM is blocked (no interraction)

4.2. FreeBSD in OpenStack

As cloud-init support in FreeBSD is still under development[5], we could not properly test FreeBSD images in UNSTPB OpenStack infrastructure. However, adding a private FreeBSD qcow2 image with a predefined user and password worked as intended. Nevertheless, due to the nested virtualization issues presented in section 4.**1**, we cannot use bhyve under KVM.

As presented in section 3, we are working on setting FreeBSD as a compute node in OpenStack. Further research showed that the errors we have encountered in [24] may be linked to the `oslo-privsep` module. The privilege separation is implemented in OpenStack using Linux capabilities. Since FreeBSD does not have capabilities, we disabled the security checks made by `oslo-privsep` [24]. However, this is not a secure implementation and may have generated the networking issues presented in [24]. Thus, we tried porting Linux capabilities to FreeBSD's CAPSICUM. Nevertheless, this was more complicated than anticipated due to the incompatibilities between the two security paradigms. This work is still in progress.

5. Conclusions

This paper presents the perspectives of using FreeBSD in a cluster infrastructure. FreeBSD has many advantages (stability, security enhancements, virtualization tools, efficient networking and storage solutions). Various improvements such as save/restore, checkpointing or virtual machine migration can improve the usability of FreeBSD in a cluster infrastructure.

[5]The FreeBSD Foundation, online: https://freebsdfoundation.org/project/freebsd-as-a-tier-i-cloud-init-platform/, last accessed 6th of September 2023

We proposed an architecture (Figure 2) that uses various FreeBSD components to create a heterogeneous cluster architecture. Nevertheless, until various features are not fully implemented and merged into upstream (cloud-init support, libvdsk, migration), we cannot fully test this architecture.

As further work, we plan to solve the bhyve in KVM nested virtualization issues and continue our integration plans.

REFERENCES

[1] *Al Etawi and Namer Ali*, A comparison between cluster, grid, and cloud computing, International Journal of Computer Applications., **179**(2018), No. 32, 37–42.

[2] *J. Anderson*, A comparison of unix sandboxing techniques., FreeBSD Journal (2017).

[3] *L.A. Barroso and U. Hölzle, Urs and P. Ranganathan*, The datacenter as a computer: Designing warehouse-scale machines., Springer Nature (2019).

[4] *E. Pettersson*, Comparison of System Performance During DDoS Attacks in Modern Operating Systems., (2017).

[5] *C. Pahl and B. Lee*, Containers and clusters for edge cloud architectures–a technology review., 2015 3rd international conference on future internet of things and cloud, IEEE (2015), 379–386.

[6] *F.X. Puig and J.J. Villalobos and I. Rodero and M. Parashar*, Exploring the potential of freebsd virtualization in containerized environments., Proceedings of the10th International Conference on Utility and Cloud Computing (2017), 191–192.

[7] *C. Ryding and R. Johansson*, Jails vs Docker: A performance comparison of different container technologies. (2020).

[8] *S. Varrette and P. Bouvry and H. Cartiaux and F. Georgatos*, Management of an academic HPC cluster: The UL experience., 2014 International Conference on High Performance Computing & Simulation (HPCS), IEEE (2014), 959–967.

[9] *A. Fengler*, Monitoring FreeBSD Systems What to (Not) Monitor., AsiaBSDCon 2019, pag 47.

[10] *Y. Takagawa and K. Matsubara*, Yet another container migration on FreeBSD., AsiaBSDCon 2019 Proceedings, 97–102 .

[11] *S.H.H. Madni and M.S.A. Latiff and Y. Coulibaly and S.M. Abdulhamid*, Recent advancements in resource allocation techniques for cloud computing environment: a systematic review., Journal Cluster Computing, Springer **20**(2017), 2489–2533.

[12] *J. Bouron and S. Chevalley and B. Lepers and W. Zwaenepoel and R. Gouicem and J.Lawall and G. Muller and J. Sopena*, The Battle of the Schedulers:{FreeBSD}{ULE} vs. Linux {CFS}., 2018 USENIX Annual Technical Conference (USENIX ATC 18) (2018), 85–96.

[13] *J. Anderson and S. Godfrey and R. NM. Watson*, Towards oblivious sandboxing with Capsicum., FreeBSD Journal (2017).

[14] *G. Tang and K. Yu and K. Veeraraghavan and J. Kaldor and S. Michelson and T. Kooburat and A. Anbudurai and M. Clark and K. Gogia and L. Cheng and others*, Twine: A unified cluster management system for shared infrastructure., 14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20) (2020), 787–803.

[15] *G. Lencse and S. Répás*, Performance analysis and comparison of different DNS64 implementations for Linux, OpenBSD and FreeBSD., 2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA), IEEE (2013), 877–884.

[16] *J. Rabenstein and D. Nguyen and O. Giersch and C. Eichler and T. Hönig and J. Nolte and W. Schröder-Preikschat*, On the Performance of NVRAM-based Operating Systems: A Case Study with Linux and FreeBSD., (2013).

[17] *M.E. Mihailescu and D. Mihai and M. Carabas and N. Tapus, Nicolae*, Improving and testing live migration for bhyve., 2022 21st RoEduNet Conference: Networking in Education and Research (RoEduNet), IEEE (2022), 1–5.

[18] *M.E. Mihailescu and M. Carabas*, FreeBSD-Live Migration feature for bhyve., AsiaBSDCon 2019.

[19] *S. Weisz and M. Carabas*, FreeBSD Virtualization-Improving block I/O compatibility in bhyve., AsiaBSDCon 2019 .

[20] *E.B. Postolache and D. Mihai and M.E. Mihailescu and S. Weisz and M. Barbulescu and M. Carabas and N. Tapus*, Title., 2020 19th RoEduNet Conference: Networking in Education and Research (RoEduNet), IEEE (2020), 1–5.

[21] *A.C. Martin and D. Mihai and M.E. Mihailescu and M. Carabas and N. Tapus*, Symmetric Multiprocessor Support for bhyve on arm64., 2022 21st RoEduNet Conference: Networking in Education and Research (RoEduNet), IEEE (2022), 1–4.

[22] *I. Mihalache and M.E. Mihăilescu and D. Mihai and M. Carabas and N. Ţăpuş*, bhyve-checkpoint functionality based on zfs., 2022 IEEE 18th International Conference on Intelligent Computer Communication and Processing (ICCP), IEEE (2022), 259–262.

[23] *D. Mihai and M.E. Mihailescu and M. Carabas and N. Ţăpuș*, Booting a Linux Kernel Under Bhyve on ARMv7., Advances in Information and Communication: Proceedings of the 2021 Future of Information and Communication Conference (FICC), Springer, **1**(2021), 93–101 .

[24] *A. Mitran and M.E. Mihăilescu and D. Mihai and S. Weisz and M. Carabas and N. Tăpus*, FreeBSD as a compute node in OpenStack., 2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP), IEEE (2020), 547–554.

[25] *V.M. Babalau and S. Weisz and D. Mihai and M.E. Mihailescu and M. Carabas and N. Tapus*, libvdsk-Adding support for the VMDK images., 2020 International Conference on Computing, Networking, Telecommunications & Engineering Sciences Applications (CoNTESA), IEEE (2020),45–50.

[26] *I. Mihalache and M.E. Mihailescu and D. Mihai and M. Carabas and N. Tapus*, bhyve-JSON format and capsicum support for the snapshot feature., 2021 IEEE 17th International Conference on Intelligent Computer Communication and Processing (ICCP), IEEE (2021), 25–29.