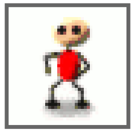
FreeBSD Enterprise Storage

Sławomir Wojciech Wojtczak



vermaden@interia.pl
vermaden.wordpress.com
twitter.com/vermaden
bsd.network/@vermaden

https://is.gd/bsdstg

What is Enterprise Storage?

What is Enterprise Storage?

The **`wikipedia.org/wiki/enterprise_storage`** page tells nothing about *enterprise*.

What is Enterprise Storage?

The **`wikipedia.org/wiki/enterprise_storage`** page tells nothing about *enterprise*.

Actually just redirects to **`wikipedia.org/wiki/data_storage`** page.

What is Enterprise Storage?

The **`wikipedia.org/wiki/enterprise_storage`** page tells nothing about *enterprise*.

Actually just redirects to **`wikipedia.org/wiki/data_storage`** page.

The other **`wikipedia.org/wiki/computer_data_storage`** page also does the same.

What is Enterprise Storage?

The **`wikipedia.org/wiki/enterprise_storage`** page tells nothing about *enterprise*.

Actually just redirects to **`wikipedia.org/wiki/data_storage`** page.

The other **`wikipedia.org/wiki/computer_data_storage`** page also does the same.

The **`wikipedia.org/wiki/enterprise`** is just meta page with links.

Common Charasteristics of Enterprise Storage

Common Charasteristics of Enterprise Storage

- Category that includes services/products designed for **large organizations**.

Common Charasteristics of Enterprise Storage

- Category that includes services/products designed for **large organizations.**

- Can handle **large volumes of data** and **large numbers of simultanous users.**

Common Charasteristics of Enterprise Storage

- Category that includes services/products designed for **large organizations**.
- Can handle **large volumes of data** and **large numbers of simultanous users**.
- Involves **centralized storage** repositories such as SAN or NAS devices.

Common Charasteristics of Enterprise Storage

- Category that includes services/products designed for **large organizations.**
- Can handle **large volumes of data** and **large numbers of simultanous users.**
- Involves **centralized storage** repositories such as SAN or NAS devices.
- Requires more time and **experience/expertise** to set up and operate.

Common Charasteristics of Enterprise Storage

- Category that includes services/products designed for **large organizations**.
- Can handle **large volumes of data** and **large numbers of simultanous users**.
- Involves **centralized storage** repositories such as SAN or NAS devices.
- Requires more time and **experience/expertise** to set up and operate.
- Generally **costs more** than consumer or small business storage devices.

Common Charasteristics of Enterprise Storage

- Category that includes services/products designed for **large organizations.**
- Can handle **large volumes of data** and **large numbers of simultanous users.**
- Involves **centralized storage** repositories such as SAN or NAS devices.
- Requires more time and **experience/expertise** to set up and operate.
- Generally **costs more** than consumer or small business storage devices.
- Generally offers **higher reliability/availability/scalability**.

EnterpriCe or EnterpriSe?

EnterpriCe or EnterpriSe?

DuckDuckGo does not provide search results count :(

Google search for *enterprice* word gives ~ **1 500 000** results.

Google search for *enterprise* word gives ~ **1 000 000 000** results (**1000** times more).

EnterpriCe or EnterpriSe?

DuckDuckGo does not provide search results count :(

Google search for *enterprice* word gives ~ **1 500 000** results.

Google search for *enterprise* word gives ~ **1 000 000 000** results (**1000** times more).

- Most dictionaries for *enterprice* word sends you to *enterprise* term.

EnterpriCe or EnterpriSe?

DuckDuckGo does not provide search results count :(

Google search for *enterprice* word gives ~ **1 500 000** results.

Google search for *enterprise* word gives ~ **1 000 000 000** results (**1000** times more).

- Most dictionaries for *enterprice* word sends you to *enterprise* term.

- Given the **PRICE** of many *enterprise* solutions it could be *enterPRICE* ...

EnterpriCe or EnterpriSe?

DuckDuckGo does not provide search results count :(

Google search for *enterprice* word gives ~ **1 500 000** results.

Google search for *enterprise* word gives ~ **1 000 000 000** results (**1000** times more).

- Most dictionaries for *enterprice* word sends you to *enterprise* term.
- Given the **PRICE** of many *enterprise* solutions it could be *enterPRICE* ...
- ... or *enterpri$e* as well :)

EnterpriCe or EnterpriSe?

DuckDuckGo does not provide search results count :(

Google search for *enterprice* word gives ~ **1 500 000** results.

Google search for *enterprise* word gives ~ **1 000 000 000** results (**1000** times more).

- Most dictionaries for *enterprice* word sends you to *enterprise* term.
- Given the **PRICE** of many *enterprise* solutions it could be *enterPRICE* …
- … or *enterpri$e* as well :)
- When in doubt just use **S** version - *Enterprise.*

Internal Solutions - Filesystems

Internal Solutions - Filesystems

- **UFS** - classic/mature/traditional small memory footprint UNIX filesytem.

Internal Solutions - Filesystems

- **UFS** - classic/mature/traditional small memory footprint UNIX filesytem.
  - UFS with **Soft Updates** (SU) allows **Snapshots** and **`dump(8)`**/**`restore(8)`** features.

Internal Solutions - Filesystems

- **UFS** - classic/mature/traditional small memory footprint UNIX filesytem.
  - UFS with **Soft Updates** (SU) allows **Snapshots** and **`dump(8)`**/**`restore(8)`** features.
  - UFS with **Journaled Soft Updates** (SU+J) with ultra fast **`fsck(8)`** process.

Internal Solutions - Filesystems

- **UFS** - classic/mature/traditional small memory footprint UNIX filesytem.
  - UFS with **Soft Updates** (SU) allows **Snapshots** and **`dump(8)`**/**`restore(8)`** features.
  - UFS with **Journaled Soft Updates** (SU+J) with ultra fast **`fsck(8)`** process.

- **ZFS** - modern pooled UNIX storage.

Internal Solutions - Filesystems

- **UFS** - classic/mature/traditional small memory footprint UNIX filesytem.
    - UFS with **Soft Updates** (SU) allows **Snapshots** and **`dump(8)`**/**`restore(8)`** features.
    - UFS with **Journaled Soft Updates** (SU+J) with ultra fast **`fsck(8)`** process.

- **ZFS** - modern pooled UNIX storage.
    - **Stable ZFS** - based on FreeBSD Base System ZFS implementaiton.

Internal Solutions - Filesystems

- **UFS** - classic/mature/traditional small memory footprint UNIX filesytem.
  - UFS with **Soft Updates** (SU) allows **Snapshots** and **`dump(8)`**/**`restore(8)`** features.
  - UFS with **Journaled Soft Updates** (SU+J) with ultra fast **`fsck(8)`** process.

- **ZFS** - modern pooled UNIX storage.
  - **Stable ZFS** - based on FreeBSD Base System ZFS implementaiton.
  - **Latest ZFS** - based on ZoL/ZoF OpenZFS repository (use FreeBSD Ports).

Internal Solutions - Filesystems

- **UFS** - classic/mature/traditional small memory footprint UNIX filesytem.
    - UFS with **Soft Updates** (SU) allows **Snapshots** and **`dump(8)`**/**`restore(8)`** features.
    - UFS with **Journaled Soft Updates** (SU+J) with ultra fast **`fsck(8)`** process.

- **ZFS** - modern pooled UNIX storage.
    - **Stable ZFS** - based on FreeBSD Base System ZFS implementaiton.
    - **Latest ZFS** - based on ZoL/ZoF OpenZFS repository (use FreeBSD Ports).
    - Differences - **`http://open-zfs.org/wiki/Feature_Flags`** - detailed information.

Internal Solutions - Filesystems

- **UFS** - classic/mature/traditional small memory footprint UNIX filesytem.
  - UFS with **Soft Updates** (SU) allows **Snapshots** and **`dump(8)`**/**`restore(8)`** features.
  - UFS with **Journaled Soft Updates** (SU+J) with ultra fast **`fsck(8)`** process.

- **ZFS** - modern pooled UNIX storage.
  - **Stable ZFS** - based on FreeBSD Base System ZFS implementaiton.
  - **Latest ZFS** - based on ZoL/ZoF OpenZFS repository (use FreeBSD Ports).
  - Differences - **`http://open-zfs.org/wiki/Feature_Flags`** - detailed information.

- **FAT/EXT2** - FreeBSD maintains BSD licensed FAT/EXT2 filesystem implementations.

Internal Solutions - Frameworks

Internal Solutions - Frameworks

- **GEOM** - FreeBSD's Modular Disk Transformation Framework.

Internal Solutions - Frameworks

- **GEOM** - FreeBSD's Modular Disk Transformation Framework.
  - Access/control of classes through use of providers/devices in `/dev` directory.

Internal Solutions - Frameworks

- **GEOM** - FreeBSD's Modular Disk Transformation Framework.
  - Access/control of classes through use of providers/devices in `/dev` directory.
  - Provides various storage related features and utilites:

Internal Solutions - Frameworks

- **GEOM** - FreeBSD's Modular Disk Transformation Framework.
    - Access/control of classes through use of providers/devices in `/dev` directory.
    - Provides various storage related features and utilites:
        - Software RAID0/RAID1/RAID10/RAID3/RAID5 configurations.

Internal Solutions - Frameworks

- **GEOM** - FreeBSD's Modular Disk Transformation Framework.
  - Access/control of classes through use of providers/devices in `/dev` directory.
  - Provides various storage related features and utilites:
    - Software RAID0/RAID1/RAID10/RAID3/RAID5 configurations.
    - Transparent encryption of underlying devices with GELI/GDBE (like LUKS).

Internal Solutions - Frameworks

- **GEOM** - FreeBSD's Modular Disk Transformation Framework.
  - Access/control of classes through use of providers/devices in `/dev` directory.
  - Provides various storage related features and utilites:
    - Software RAID0/RAID1/RAID10/RAID3/RAID5 configurations.
    - Transparent encryption of underlying devices with GELI/GDBE (like LUKS).
    - Transparent filesytem journaling for ANY filesystem with GJOURNAL.

Internal Solutions - Frameworks

- **GEOM** - FreeBSD's Modular Disk Transformation Framework.
  - Access/control of classes through use of providers/devices in `/dev` directory.
  - Provides various storage related features and utilites:
    - Software **RAID0**/**RAID1**/**RAID10**/**RAID3**/**RAID5** configurations.
    - Transparent encryption of underlying devices with **GELI**/**GDBE** (like LUKS).
    - Transparent filesytem journaling for ANY filesystem with **GJOURNAL**.
    - Export block device over network with **GEOM GATE** devices (like NFS for block).

Internal Solutions - Frameworks

- **GEOM** - FreeBSD's Modular Disk Transformation Framework.
  - Access/control of classes through use of providers/devices in `/dev` directory.
  - Provides various storage related features and utilites:
    - Software **RAID0**/**RAID1**/**RAID10**/**RAID3**/**RAID5** configurations.
    - Transparent encryption of underlying devices with **GELI**/**GDBE** (like LUKS).
    - Transparent filesytem journaling for ANY filesystem with **GJOURNAL**.
    - Export block device over network with **GEOM GATE** devices (like NFS for block).

- **FUSE** - BSD licensed FUSE filesystem implementation with 7.04 - 7.23 protocol support.

Internal Solutions - Frameworks

- **GEOM** - FreeBSD's Modular Disk Transformation Framework.
  - Access/control of classes through use of providers/devices in `/dev` directory.
  - Provides various storage related features and utilites:
    - Software **RAID0**/**RAID1**/**RAID10**/**RAID3**/**RAID5** configurations.
    - Transparent encryption of underlying devices with **GELI**/**GDBE** (like LUKS).
    - Transparent filesytem journaling for ANY filesystem with **GJOURNAL**.
    - Export block device over network with **GEOM GATE** devices (like NFS for block).

- **FUSE** - BSD licensed FUSE filesystem implementation with 7.04 - 7.23 protocol support.
  - Details - **`https://freebsd.org/news/status/report-2019-04-2019-06.html#FUSE`**

Internal Solutions - Frameworks

- **GEOM** - FreeBSD's Modular Disk Transformation Framework.
  - Access/control of classes through use of providers/devices in `/dev` directory.
  - Provides various storage related features and utilites:
    - Software **RAID0**/**RAID1**/**RAID10**/**RAID3**/**RAID5** configurations.
    - Transparent encryption of underlying devices with **GELI**/**GDBE** (like LUKS).
    - Transparent filesytem journaling for ANY filesystem with **GJOURNAL**.
    - Export block device over network with **GEOM GATE** devices (like NFS for block).

- **FUSE** - BSD licensed FUSE filesystem implementation with 7.04 - 7.23 protocol support.
  - Details - **`https://freebsd.org/news/status/report-2019-04-2019-06.html#FUSE`**
  - Classic - NTFS3G/exFAT/EXT2/EXT3/EXT4/XFS/HFS/MTP/BTRFS/LinuxLVM/...

Internal Solutions - Frameworks

- **GEOM** - FreeBSD's Modular Disk Transformation Framework.
  - Access/control of classes through use of providers/devices in `/dev` directory.
  - Provides various storage related features and utilites:
    - Software **RAID0**/**RAID1**/**RAID10**/**RAID3**/**RAID5** configurations.
    - Transparent encryption of underlying devices with **GELI**/**GDBE** (like LUKS).
    - Transparent filesytem journaling for ANY filesystem with **GJOURNAL**.
    - Export block device over network with **GEOM GATE** devices (like NFS for block).

- **FUSE** - BSD licensed FUSE filesystem implementation with 7.04 - 7.23 protocol support.
  - Details - **`https://freebsd.org/news/status/report-2019-04-2019-06.html#FUSE`**
  - Classic - NTFS3G/exFAT/EXT2/EXT3/EXT4/XFS/HFS/MTP/BTRFS/LinuxLVM/...
  - Virtual - s3fs/gphotofs/mp3fs/rar2fs/sqlfs/sshfs/unionfs/wikipediafs/...

Internal Solutions - Availability

Internal Solutions - Availability

- **HAST** - Highly Available Storage (like DRBD).

Internal Solutions - Availability

- **HAST** - Highly Available Storage (like DRBD).
  - Transparent storage across several machines connected over TCP/IP network.

Internal Solutions - Availability

- **HAST** - Highly Available Storage (like DRBD).
  - Transparent storage across several machines connected over TCP/IP network.
  - Can be understood as network based RAID1 (mirror).

Internal Solutions - Availability

- **HAST** - Highly Available Storage (like DRBD).
  - Transparent storage across several machines connected over TCP/IP network.
  - Can be understood as network based RAID1 (mirror).
  - With FreeBSD's CARP and **`devd(8)`** allows highly available storage cluster.

Internal Solutions - Availability

- **HAST** - Highly Available Storage (like DRBD).
  - Transparent storage across several machines connected over TCP/IP network.
  - Can be understood as network based RAID1 (mirror).
  - With FreeBSD's CARP and **`devd(8)`** allows highly available storage cluster.

- **CARP** - Common Address Redundancy Protocol.

Internal Solutions - Availability

- **HAST** - Highly Available Storage (like DRBD).
  - Transparent storage across several machines connected over TCP/IP network.
  - Can be understood as network based RAID1 (mirror).
  - With FreeBSD's CARP and **`devd(8)`** allows highly available storage cluster.

- **CARP** - Common Address Redundancy Protocol.
  - Allows multiple hosts to share the same IP address and Virtual Host ID (VHID).

Internal Solutions - Availability

- **HAST** - Highly Available Storage (like DRBD).
  - Transparent storage across several machines connected over TCP/IP network.
  - Can be understood as network based RAID1 (mirror).
  - With FreeBSD's CARP and **`devd(8)`** allows highly available storage cluster.

- **CARP** - Common Address Redundancy Protocol.
  - Allows multiple hosts to share the same IP address and Virtual Host ID (VHID).
  - Provides high availability for one or more services.

Internal Solutions - Availability

- **HAST** - Highly Available Storage (like DRBD).
  - Transparent storage across several machines connected over TCP/IP network.
  - Can be understood as network based RAID1 (mirror).
  - With FreeBSD's CARP and **devd(8)** allows highly available storage cluster.

- **CARP** - Common Address Redundancy Protocol.
  - Allows multiple hosts to share the same IP address and Virtual Host ID (VHID).
  - Provides high availability for one or more services.
  - Provides floating shared highly available IP address.

UFS with Soft Updates (SU)

UFS with Soft Updates (SU)

- Mature **classic filesystem** with very **small memory footprint.**

UFS with Soft Updates (SU)

- Mature **classic filesystem** with very **small memory footprint.**
- Supports **TRIM** natively which allows efficient data deletion on SSDs.

UFS with Soft Updates (SU)

- Mature **classic filesystem** with very **small memory footprint.**
- Supports **TRIM** natively which allows efficient data deletion on SSDs.
- Supports **read only snaphots** (not available in SUJ mode).

UFS with Soft Updates (SU)

- Mature **classic filesystem** with very **small memory footprint.**
- Supports **TRIM** natively which allows efficient data deletion on SSDs.
- Supports **read only snaphots** (not available in SUJ mode).
- Neither Journaling (SUJ) nor Soft Updates (SU) guarantees no data will be lost.

UFS with Soft Updates (SU)

- Mature **classic filesystem** with very **small memory footprint.**
- Supports **TRIM** natively which allows efficient data deletion on SSDs.
- Supports **read only snaphots** (not available in SUJ mode).
- Neither Journaling (SUJ) nor Soft Updates (SU) guarantees no data will be lost.
- They (SUJ/SU) make sure that **filesystem metadata will remain consistent.**

UFS with Soft Updates (SU)

- Mature **classic filesystem** with very **small memory footprint.**
- Supports **TRIM** natively which allows efficient data deletion on SSDs.
- Supports **read only snaphots** (not available in SUJ mode).
- Neither Journaling (SUJ) nor Soft Updates (SU) guarantees no data will be lost.
- They (SUJ/SU) make sure that **filesystem metadata will remain consistent.**
- Advantage of SU/SUJ is that filesystem can be **mounted immediately after crash.**

UFS with Soft Updates (SU)

- Mature **classic filesystem** with very **small memory footprint.**
- Supports **TRIM** natively which allows efficient data deletion on SSDs.
- Supports **read only snaphots** (not available in SUJ mode).
- Neither Journaling (SUJ) nor Soft Updates (SU) guarantees no data will be lost.
- They (SUJ/SU) make sure that **filesystem metadata will remain consistent.**
- Advantage of SU/SUJ is that filesystem can be **mounted immediately after crash.**
- UFS (with SU/SUJ) requires traditional `fsck(8)` in background to make it clean.

UFS with Journaled Soft Updates (SUJ)

UFS with Journaled Soft Updates (SUJ)

- Primary purpose was to **eliminate long filesystem check times** with `fsck(8)`.

UFS with Journaled Soft Updates (SUJ)

- Primary purpose was to **eliminate long filesystem check times** with `fsck(8)`.

- SUJ journal (`.sujournal`) logs **only two inconsistencies** possible in SU:

UFS with Journaled Soft Updates (SUJ)

- Primary purpose was to **eliminate long filesystem check times** with `fsck(8)`.

- SUJ journal (`.sujournal`) logs **only two inconsistencies** possible in SU:
  - Allocated but unreferenced blocks.

UFS with Journaled Soft Updates (SUJ)

- Primary purpose was to **eliminate long filesystem check times** with `fsck(8)`.

- SUJ journal (`.sujournal`) logs **only two inconsistencies** possible in SU:
    - Allocated but unreferenced blocks.
    - Incorrectly high link counts (including unreferenced inodes).

UFS with Journaled Soft Updates (SUJ)

- Primary purpose was to **eliminate long filesystem check times** with `fsck(8)`.

- SUJ journal (`.sujournal`) logs **only two inconsistencies** possible in SU:
  - Allocated but unreferenced blocks.
  - Incorrectly high link counts (including unreferenced inodes).

- **200 GB** data on disk takes **1 second** with SUJ under `fsck(8)`.

UFS with Journaled Soft Updates (SUJ)

- Primary purpose was to **eliminate long filesystem check times** with `fsck(8)`.

- SUJ journal (`.sujournal`) logs **only two inconsistencies** possible in SU:
  - Allocated but unreferenced blocks.
  - Incorrectly high link counts (including unreferenced inodes).

- **200 GB** data on disk takes **1 second** with SUJ under `fsck(8)`.
  - Same disk with SU only (w/o Journaling) takes **27 minutes** (1800 times more).

UFS with Journaled Soft Updates (SUJ)

- Primary purpose was to **eliminate long filesystem check times** with `fsck(8)`.

- SUJ journal (`.sujournal`) logs **only two inconsistencies** possible in SU:
  - Allocated but unreferenced blocks.
  - Incorrectly high link counts (including unreferenced inodes).

- **200 GB** data on disk takes **1 second** with SUJ under `fsck(8)`.
  - Same disk with SU only (w/o Journaling) takes **27 minutes** (1800 times more).

- **10 TB** data on disk takes **1 minute** with SUJ under `fsck(8)`.

UFS with Journaled Soft Updates (SUJ)

- Primary purpose was to **eliminate long filesystem check times** with `fsck(8)`.

- SUJ journal (`.sujournal`) logs **only two inconsistencies** possible in SU:
  - Allocated but unreferenced blocks.
  - Incorrectly high link counts (including unreferenced inodes).

- **200 GB** data on disk takes **1 second** with SUJ under `fsck(8)`.
  - Same disk with SU only (w/o Journaling) takes **27 minutes** (1800 times more).

- **10 TB** data on disk takes **1 minute** with SUJ under `fsck(8)`.
  - Same disk with SU only (w/o Journaling) took approximately **10 hours**.

Why UFS in 2020?

Why UFS in 2020?

- UFS has both **fragments** and **blocks**:

Why UFS in 2020?

- UFS has both **fragments** and **blocks**:
    - Files smaller than blocksize can be stored in individual fragments.

Why UFS in 2020?

- UFS has both **fragments** and **blocks**:
    - Files smaller than blocksize can be stored in individual fragments.
    - Can create filesystem optimized for small files (1 KB) with (8 KB) blocksize.

Why UFS in 2020?

- UFS has both **fragments** and **blocks**:
  - Files smaller than blocksize can be stored in individual fragments.
  - Can create filesystem optimized for small files (1 KB) with (8 KB) blocksize.
- UFS can be **grown online** with **`growfs(8)`** at boot time or anytime at system work.

Why UFS in 2020?

- UFS has both **fragments** and **blocks**:
  - Files smaller than blocksize can be stored in individual fragments.
  - Can create filesystem optimized for small files (1 KB) with (8 KB) blocksize.
- UFS can be **grown online** with `growfs(8)` at boot time or anytime at system work.
- Soft Updates is great for **apps/databases with their own log** (like PostgreSQL):

Why UFS in 2020?

- UFS has both **fragments** and **blocks**:
  - Files smaller than blocksize can be stored in individual fragments.
  - Can create filesystem optimized for small files (1 KB) with (8 KB) blocksize.
- UFS can be **grown online** with **`growfs(8)`** at boot time or anytime at system work.
- Soft Updates is great for **apps/databases with their own log** (like PostgreSQL):
  - Using Journaling would log everything twice - SU **passes data through**.

Why UFS in 2020?

- UFS has both **fragments** and **blocks**:
  - Files smaller than blocksize can be stored in individual fragments.
  - Can create filesystem optimized for small files (1 KB) with (8 KB) blocksize.
- UFS can be **grown online** with **`growfs(8)`** at boot time or anytime at system work.
- Soft Updates is great for **apps/databases with their own log** (like PostgreSQL):
  - Using Journaling would log everything twice - SU **passes data through**.
- Soft Updates has interesting property regarding **short lived (temporary) files**:

Why UFS in 2020?

- UFS has both **fragments** and **blocks**:
  - Files smaller than blocksize can be stored in individual fragments.
  - Can create filesystem optimized for small files (1 KB) with (8 KB) blocksize.
- UFS can be **grown online** with **`growfs(8)`** at boot time or anytime at system work.
- Soft Updates is great for **apps/databases with their own log** (like PostgreSQL):
  - Using Journaling would log everything twice - SU **passes data through**.
- Soft Updates has interesting property regarding **short lived (temporary) files**:
  - Create file + write data to it + delete it (in short time span).

Why UFS in 2020?

- UFS has both **fragments** and **blocks**:
  - Files smaller than blocksize can be stored in individual fragments.
  - Can create filesystem optimized for small files (1 KB) with (8 KB) blocksize.
- UFS can be **grown online** with **`growfs(8)`** at boot time or anytime at system work.
- Soft Updates is great for **apps/databases with their own log** (like PostgreSQL):
  - Using Journaling would log everything twice - SU **passes data through**.
- Soft Updates has interesting property regarding **short lived (temporary) files**:
  - Create file + write data to it + delete it (in short time span).
  - Neither data nor metadata from this file will ever touch filesystem.

Why UFS in 2020?

- UFS has both **fragments** and **blocks**:
  - Files smaller than blocksize can be stored in individual fragments.
  - Can create filesystem optimized for small files (1 KB) with (8 KB) blocksize.
- UFS can be **grown online** with `growfs(8)` at boot time or anytime at system work.
- Soft Updates is great for **apps/databases with their own log** (like PostgreSQL):
  - Using Journaling would log everything twice - SU **passes data through**.
- Soft Updates has interesting property regarding **short lived (temporary) files**:
  - Create file + write data to it + delete it (in short time span).
  - Neither data nor metadata from this file will ever touch filesystem.
- Netflix uses UFS in 2020 for their content storage on FreeBSD.

ZFS - Zettabyte File System

ZFS - Zettabyte File System

- Modern pooled storage.

ZFS - Zettabyte File System

- Modern pooled storage.
- Always consistent on-disk state - no `fsck(8)` needed.

ZFS - Zettabyte File System

- Modern pooled storage.
- Always consistent on-disk state - no `fsck(8)` needed.
- **Snapshots** (read only) and **clones** (read write).

ZFS - Zettabyte File System

- Modern pooled storage.
- Always consistent on-disk state - no `fsck(8)` needed.
- **Snapshots** (read only) and **clones** (read write).
- Provides end-to-end data integrity with **checksums**.

ZFS - Zettabyte File System

- Modern pooled storage.
- Always consistent on-disk state - no `fsck(8)` needed.
- **Snapshots** (read only) and **clones** (read write).
- Provides end-to-end data integrity with **checksums**.
- Have **self-healing** features.

ZFS - Zettabyte File System

- Modern pooled storage.
- Always consistent on-disk state - no `fsck(8)` needed.
- **Snapshots** (read only) and **clones** (read write).
- Provides end-to-end data integrity with **checksums**.
- Have **self-healing** features.
- Have built-in **redundancy**.

ZFS - Zettabyte File System

- Modern pooled storage.
- Always consistent on-disk state - no `fsck(8)` needed.
- **Snapshots** (read only) and **clones** (read write).
- Provides end-to-end data integrity with **checksums**.
- Have **self-healing** features.
- Have built-in **redundancy**.
- Scalable design and **dynamic striping**.

ZFS - Zettabyte File System

- Modern pooled storage.
- Always consistent on-disk state - no `fsck(8)` needed.
- **Snapshots** (read only) and **clones** (read write).
- Provides end-to-end data integrity with **checksums**.
- Have **self-healing** features.
- Have built-in **redundancy**.
- Scalable design and **dynamic striping**.
- Variable blocksize.

ZFS - Zettabyte File System

- Modern pooled storage.
- Always consistent on-disk state - no `fsck(8)` needed.
- **Snapshots** (read only) and **clones** (read write).
- Provides end-to-end data integrity with **checksums**.
- Have **self-healing** features.
- Have built-in **redundancy**.
- Scalable design and **dynamic striping**.
- Variable blocksize.
- Builtin **replication**/**compression**/**encryption**/**deduplication**.

ZFS - Zettabyte File System

- Modern pooled storage.
- Always consistent on-disk state - no `fsck(8)` needed.
- **Snapshots** (read only) and **clones** (read write).
- Provides end-to-end data integrity with **checksums**.
- Have **self-healing** features.
- Have built-in **redundancy**.
- Scalable design and **dynamic striping**.
- Variable blocksize.
- Builtin **replication**/**compression**/**encryption**/**deduplication**.
- Possible to add **read cache** as L2ARC (2nd Level of *Adaptive Replacement Cache*).

ZFS - Zettabyte File System

- Modern pooled storage.
- Always consistent on-disk state - no `fsck(8)` needed.
- **Snapshots** (read only) and **clones** (read write).
- Provides end-to-end data integrity with **checksums**.
- Have **self-healing** features.
- Have built-in **redundancy**.
- Scalable design and **dynamic striping**.
- Variable blocksize.
- Builtin **replication**/**compression**/**encryption**/**deduplication**.
- Possible to add **read cache** as L2ARC (2nd Level of *Adaptive Replacement Cache*).
- Possible to add **write cache** as ZIL (*ZFS Intent Log*).

ZFS - Zettabyte File System

- Modern pooled storage.
- Always consistent on-disk state - no `fsck(8)` needed.
- **Snapshots** (read only) and **clones** (read write).
- Provides end-to-end data integrity with **checksums**.
- Have **self-healing** features.
- Have built-in **redundancy**.
- Scalable design and **dynamic striping**.
- Variable blocksize.
- Builtin **replication**/**compression**/**encryption**/**deduplication**.
- Possible to add **read cache** as L2ARC (2nd Level of *Adaptive Replacement Cache*).
- Possible to add **write cache** as ZIL (*ZFS Intent Log*).
- Simple administration - two simple `zfs(8)` and `zpool(8)` commands.

ZFS - Common Myths

ZFS - Common Myths

- Myth #1 - **lots of RAM is needed.**

ZFS - Common Myths

- Myth #1 - **lots of RAM is needed.**
  - RAM is **only cache** for ZFS (called ARC) and its size can be tuned down to even 10MB for example.

ZFS - Common Myths

- Myth #1 - **lots of RAM is needed.**
  - RAM is **only cache** for ZFS (called ARC) and its size can be tuned down to even 10MB for example.
  - Use **`vfs.zfs.arc_min`** and **`vfs.zfs.arc_max`** in **`/boot/loader.conf`** file.

ZFS - Common Myths

- Myth #1 - **lots of RAM is needed.**
  - RAM is **only cache** for ZFS (called ARC) and its size can be tuned down to even 10MB for example.
  - Use **`vfs.zfs.arc_min`** and **`vfs.zfs.arc_max`** in **`/boot/loader.conf`** file.
  - Use **`kern.maxvnodes`** in **`/etc/sysctl.conf`** file if needed to limit for sure.

ZFS - Common Myths

- Myth #1 - **lots of RAM is needed**.
  - RAM is **only cache** for ZFS (called ARC) and its size can be tuned down to even 10MB for example.
  - Use **`vfs.zfs.arc_min`** and **`vfs.zfs.arc_max`** in **`/boot/loader.conf`** file.
  - Use **`kern.maxvnodes`** in **`/etc/sysctl.conf`** file if needed to limit for sure.
  - I have used 2TB ZFS mirror with 512RAM and it was rock stable for several years.

ZFS - Common Myths

- Myth #1 - **lots of RAM is needed.**
    - RAM is **only cache** for ZFS (called ARC) and its size can be tuned down to even 10MB for example.
    - Use **`vfs.zfs.arc_min`** and **`vfs.zfs.arc_max`** in **`/boot/loader.conf`** file.
    - Use **`kern.maxvnodes`** in **`/etc/sysctl.conf`** file if needed to limit for sure.
    - I have used 2TB ZFS mirror with 512RAM and it was rock stable for several years.
- Myth #2 - **ECC RAM must be used.**

ZFS - Common Myths

- Myth #1 - **lots of RAM is needed.**
  - RAM is **only cache** for ZFS (called ARC) and its size can be tuned down to even 10MB for example.
  - Use **`vfs.zfs.arc_min`** and **`vfs.zfs.arc_max`** in **`/boot/loader.conf`** file.
  - Use **`kern.maxvnodes`** in **`/etc/sysctl.conf`** file if needed to limit for sure.
  - I have used 2TB ZFS mirror with 512RAM and it was rock stable for several years.
- Myth #2 - **ECC RAM must be used.**
  - All filesystems benefit from ECC RAM and ZFS is no different here.

ZFS - Common Myths

- Myth #1 - **lots of RAM is needed.**
  - RAM is **only cache** for ZFS (called ARC) and its size can be tuned down to even 10MB for example.
  - Use **`vfs.zfs.arc_min`** and **`vfs.zfs.arc_max`** in **`/boot/loader.conf`** file.
  - Use **`kern.maxvnodes`** in **`/etc/sysctl.conf`** file if needed to limit for sure.
  - I have used 2TB ZFS mirror with 512RAM and it was rock stable for several years.
- Myth #2 - **ECC RAM must be used.**
  - All filesystems benefit from ECC RAM and ZFS is no different here.
  - ZFS without ECC RAM is safer then other filesystems with ECC RAM (checksums).

ZFS - Common Myths

- Myth #1 - **lots of RAM is needed.**
  - RAM is **only cache** for ZFS (called ARC) and its size can be tuned down to even 10MB for example.
  - Use **`vfs.zfs.arc_min`** and **`vfs.zfs.arc_max`** in **`/boot/loader.conf`** file.
  - Use **`kern.maxvnodes`** in **`/etc/sysctl.conf`** file if needed to limit for sure.
  - I have used 2TB ZFS mirror with 512RAM and it was rock stable for several years.
- Myth #2 - **ECC RAM must be used.**
  - All filesystems benefit from ECC RAM and ZFS is no different here.
  - ZFS without ECC RAM is safer then other filesystems with ECC RAM (checksums).
- Myth #3 - **bad for laptop/desktop.**

ZFS - Common Myths

- Myth #1 - **lots of RAM is needed.**
  - RAM is **only cache** for ZFS (called ARC) and its size can be tuned down to even 10MB for example.
  - Use **`vfs.zfs.arc_min`** and **`vfs.zfs.arc_max`** in **`/boot/loader.conf`** file.
  - Use **`kern.maxvnodes`** in **`/etc/sysctl.conf`** file if needed to limit for sure.
  - I have used 2TB ZFS mirror with 512RAM and it was rock stable for several years.
- Myth #2 - **ECC RAM must be used.**
  - All filesystems benefit from ECC RAM and ZFS is no different here.
  - ZFS without ECC RAM is safer then other filesystems with ECC RAM (checksums).
- Myth #3 - **bad for laptop/desktop.**
  - Single disk devices still benetif from **snapshots**/**clones**/**checksums**/**compression**/**deduplication**.

ZFS - Common Myths

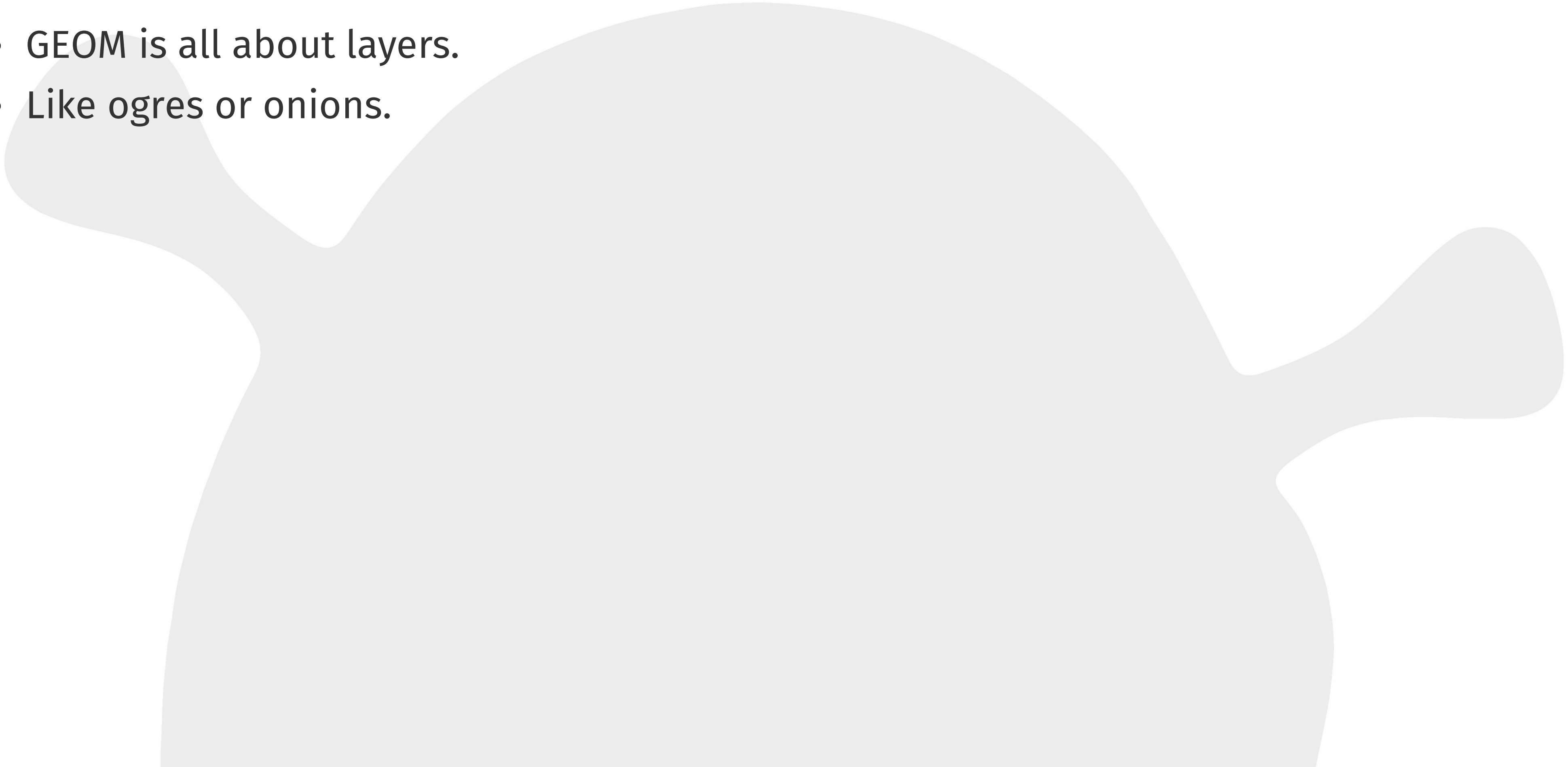
- Myth #1 - **lots of RAM is needed.**
  - RAM is **only cache** for ZFS (called ARC) and its size can be tuned down to even 10MB for example.
  - Use **`vfs.zfs.arc_min`** and **`vfs.zfs.arc_max`** in **`/boot/loader.conf`** file.
  - Use **`kern.maxvnodes`** in **`/etc/sysctl.conf`** file if needed to limit for sure.
  - I have used 2TB ZFS mirror with 512RAM and it was rock stable for several years.
- Myth #2 - **ECC RAM must be used.**
  - All filesystems benefit from ECC RAM and ZFS is no different here.
  - ZFS without ECC RAM is safer then other filesystems with ECC RAM (checksums).
- Myth #3 - **bad for laptop/desktop.**
  - Single disk devices still benetif from **snapshots**/**clones**/**checksums**/**compression**/**deduplication**.
  - ZFS allows **bulletproof upgrades** with **ZFS Boot Environments** - **`https://is.gd/BECTL`** - more here.

GEOM Idea

GEOM Idea

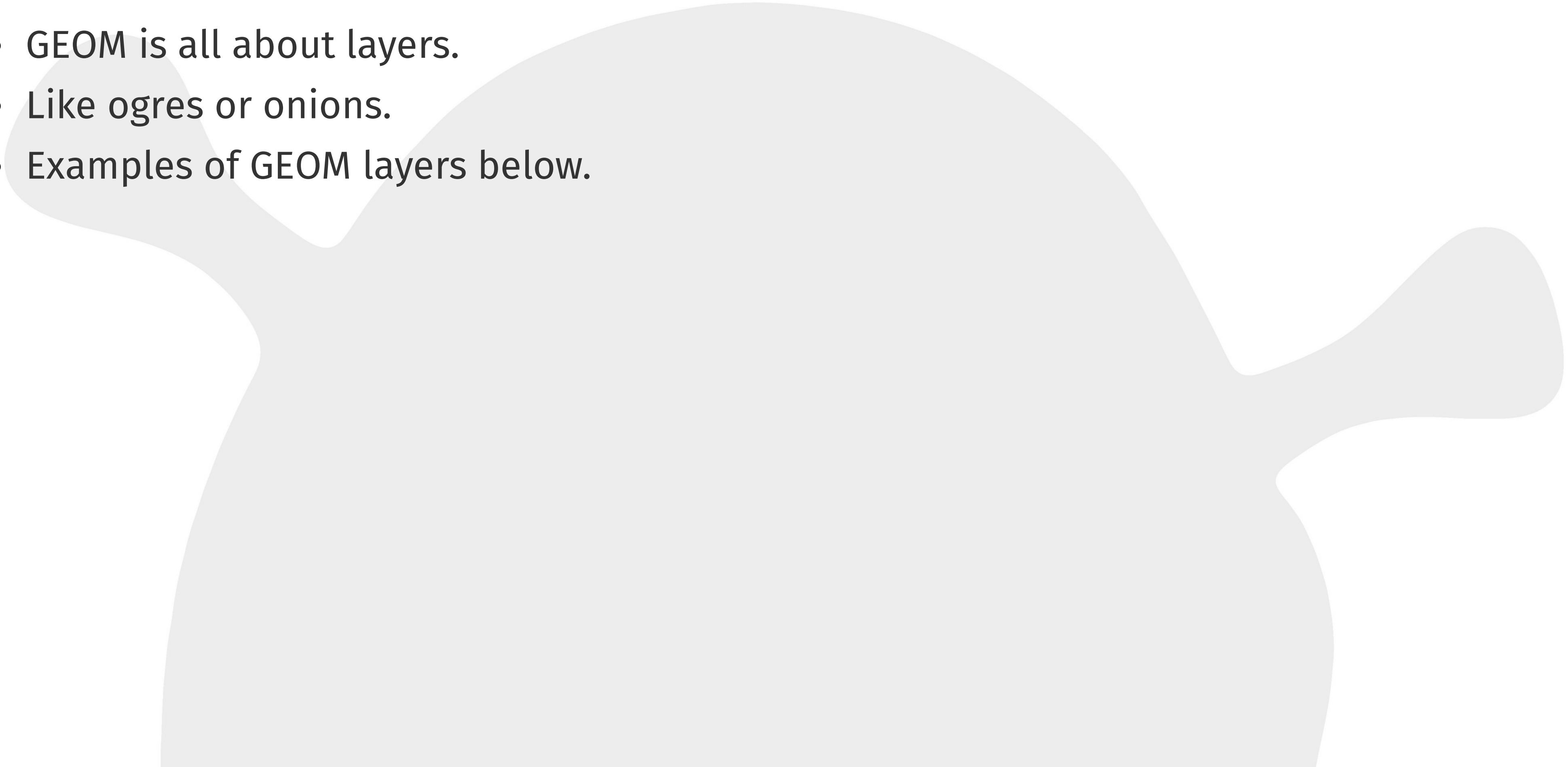
- GEOM is all about layers.

GEOM Idea



- GEOM is all about layers.
- Like ogres or onions.

GEOM Idea

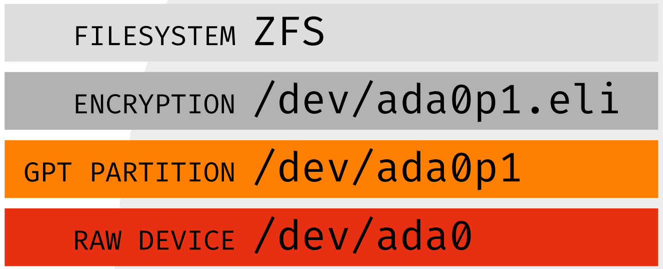


- GEOM is all about layers.
- Like ogres or onions.
- Examples of GEOM layers below.

GEOM Idea

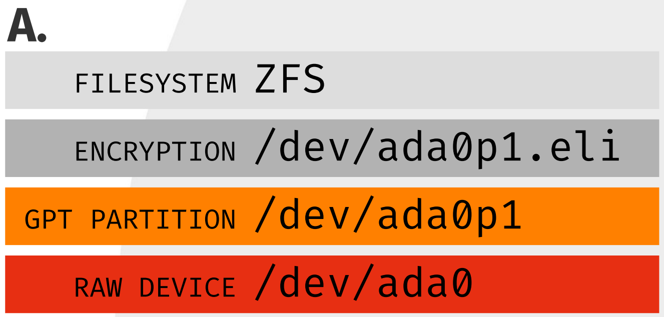
- GEOM is all about layers.
- Like ogres or onions.
- Examples of GEOM layers below.
- **A.** ZFS on GELI (encryption) on GPT (p1) partition.

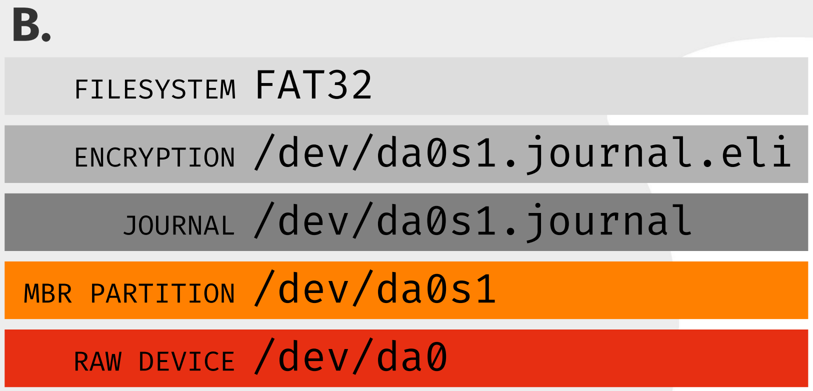
A.



GEOM Idea

- GEOM is all about layers.
- Like ogres or onions.
- Examples of GEOM layers below.
- **A.** ZFS on GELI (encryption) on GPT (p1) partition.
- **B.** FAT32 on GELI on GJOURNAL (journaling) on MBR (s1) partition/slice.





GEOM Classes/Providers (1/2)

• **CACHE**	`/sbin/gcache`	Optional read cache for GEOM RAID3 **`graid3(8)`** class.
• **CONCAT**	`/sbin/gconcat`	Concat multiple devices into one virtual device.
• **DBE**	`/sbin/gbde`	GEOM based disk encryption (older).
• **ELI**	`/sbin/geli`	Block device disk encryption (modern).
• **GATE**	`/sbin/ggate*`	Export block device over network (like NFS for block).
• **JOURNAL**	`/sbin/gjournal`	Generic block device level journal provider.
• **LABEL**	`/sbin/glabel`	Manual and automatic labelization provider.
• **MIRROR**	`/sbin/gmirror`	Mirror (RAID1) provider.
• **MOUNTVER**	`/sbin/gmountver`	Queues I/O requests and waits for provider.
• **MULTIPATH**	`/sbin/gmultipath`	Device multipath configuration provider.

GEOM Classes/Providers (2/2)

• **NOP**	`/sbin/gnop`	Provider to example emulate different blocksize.
• **PART**	`/sbin/gpart`	Partition (BSD/MBR/GPT/...) GEOM device providers.
• **RAID**	`/sbin/graid`	Software RAID management (Intel/JMicron/SiI/Promise/...).
• **RAID3**	`/sbin/graid3`	RAID3 provider.
• **RAID5**	`sysutils/graid5`	RAID5 provider (available from FreeBSD Ports).
• **SCHED**	`/sbin/gsched`	Change scheduling policy of requests going to provider.
• **SHSEC**	`/sbin/gshsec`	Setup shared secret between given providers.
• **STRIPE**	`/sbin/gstripe`	Stripe (RAID0) provider (RAID10 with **`gmirror(8)`** provider).
• **VIRSTOR**	`/sbin/gvirstor`	Like *Virtual Memory* allows overcommit for block devices.
• **VINUM**	`/sbin/gvinum`	RAID 0/1/10/5 provider (older VxVM style volume manager).

GEOM Examples (1/2)

```
# geom disk list // 12 TB Toshiba 7200RPM

Geom name: da0
Providers:
1. Name: da0
   Mediasize: 12000138625024 (11T)
   Sectorsize: 512
   Stripesize: 4096
   Stripeoffset: 0
   Mode: r1w1e2
   descr: ATA TOSHIBA MG07ACA1
   lunid: 50000398e8c9d3d5
   ident: 98G0A10CF95G
   rotationrate: 7200
   fwsectors: 63
   fwheads: 255
```

```
# geom disk list // 4 TB Samsung SSD

Geom name: ada0
Providers:
1. Name: ada0
   Mediasize: 4000787030016 (3.6T)
   Sectorsize: 512
   Mode: r1w1e2
   descr: Samsung SSD 860 QVO 4TB
   lunid: 5002538e40f16748
   ident: S4CXNF0M404495P
   rotationrate: 0
   fwsectors: 63
   fwheads: 16
```

GEOM Examples (2/2)

```
# gpart show da0
⇒          40  23437770672  da90  GPT  (11T)
           40  23435673600     1  freebsd-zfs  (11T)
  23435673640      2097072        - free -  (1.0G)
```

```
# gpart show ada0
⇒         40  1953525088  ada1  GPT  (932G)
          40      409600     1  efi  (200M)
      409640        1024     2  freebsd-boot  (512K)
      410664         984        - free -  (492K)
      411648  1953112064     3  freebsd-zfs  (931G)
  1953523712        1416        - free -  (708K)
```

```
# geli status
      Name  Status  Components
ada1p3.eli  ACTIVE  ada1p3
ada0p1.eli  ACTIVE  ada0p1
 da0p1.eli  ACTIVE  da0p1
```

```
# glabel status
        Name  Status  Components
gpt/efiboot0     N/A  ada1p1
gpt/gptboot0     N/A  ada1p2
```

Internal Solutions - Summary

Internal Solutions - Summary

UFS

Internal Solutions - Summary

UFS

ZFS

Internal Solutions - Summary

UFS

ZFS

FreeBSD Ecosystem

GEOM/FUSE/HAST/CARP/UFS/ZFS/...

External Solutions - Distributed Filesystems

External Solutions - Distributed Filesystems

- **Ceph** - distributed storage object/block/filesystem with performance/reliability.
  - **https://ceph.io/** - FreeBSD Ports - `net/ceph14`

External Solutions - Distributed Filesystems

- **Ceph** - distributed storage object/block/filesystem with performance/reliability.
    - **https://ceph.io/** - FreeBSD Ports - `net/ceph14`
- **GlusterFS** - distributed filesystem scales to petabytes for thousands of clients.
    - **https://gluster.org/** - FreeBSD Ports - `net/glusterfs`

External Solutions - Distributed Filesystems

- **Ceph** - distributed storage object/block/filesystem with performance/reliability.
  - **https://ceph.io/** - FreeBSD Ports - `net/ceph14`
- **GlusterFS** - distributed filesystem scales to petabytes for thousands of clients.
  - **https://gluster.org/** - FreeBSD Ports - `net/glusterfs`
- **LeoFS** - highly scalable fault-tolerant distributed filesystem.
  - **http://leo-project.net/** - FreeBSD Ports - `databases/leofs`

External Solutions - Distributed Filesystems

- **Ceph** - distributed storage object/block/filesystem with performance/reliability.
  - **https://ceph.io/** - FreeBSD Ports - `net/ceph14`
- **GlusterFS** - distributed filesystem scales to petabytes for thousands of clients.
  - **https://gluster.org/** - FreeBSD Ports - `net/glusterfs`
- **LeoFS** - highly scalable fault-tolerant distributed filesystem.
  - **http://leo-project.net/** - FreeBSD Ports - `databases/leofs`
- **LizardFS** - highly reliable/scalable/efficient distributed filesystem.
  - **https://lizardfs.org/** - FreeBSD Ports - `sysutils/lizardfs`

External Solutions - Distributed Filesystems

- **Ceph** - distributed storage object/block/filesystem with performance/reliability.
  - **https://ceph.io/** - FreeBSD Ports - `net/ceph14`
- **GlusterFS** - distributed filesystem scales to petabytes for thousands of clients.
  - **https://gluster.org/** - FreeBSD Ports - `net/glusterfs`
- **LeoFS** - highly scalable fault-tolerant distributed filesystem.
  - **http://leo-project.net/** - FreeBSD Ports - `databases/leofs`
- **LizardFS** - highly reliable/scalable/efficient distributed filesystem.
  - **https://lizardfs.org/** - FreeBSD Ports - `sysutils/lizardfs`
- **Minio** - Amazon S3 compatible distributed object storage server.
  - **https://minio.io/** - FreeBSD Ports - `www/minio`

External Solutions - Software Services

External Solutions - Software Services

- **Syncthing** - encrypted file sync tool to replace cloud services with something open.
  - **https://syncthing.net/** - FreeBSD Ports - `net/syncthing`

External Solutions - Software Services

- **Syncthing** - encrypted file sync tool to replace cloud services with something open.
  - **https://syncthing.net/** - FreeBSD Ports - `net/syncthing`
- **Nextcloud** - personal cloud which runs on your own server (also check *OwnCloud*).
  - **https://nextcloud.com/** - FreeBSD Ports - `www/nextcloud`

External Solutions - Software Services

- **Syncthing** - encrypted file sync tool to replace cloud services with something open.
  - **https://syncthing.net/** - FreeBSD Ports - `net/syncthing`

- **Nextcloud** - personal cloud which runs on your own server (also check *OwnCloud*).
  - **https://nextcloud.com/** - FreeBSD Ports - `www/nextcloud`

- **Seafile** - file hosting software system.
  - **https://seafile.com/** - FreeBSD Ports - `www/seafile-server`

External Solutions - Software Services

- **Syncthing** - encrypted file sync tool to replace cloud services with something open.
  - **https://syncthing.net/** - FreeBSD Ports - `net/syncthing`
- **Nextcloud** - personal cloud which runs on your own server (also check *OwnCloud*).
  - **https://nextcloud.com/** - FreeBSD Ports - `www/nextcloud`
- **Seafile** - file hosting software system.
  - **https://seafile.com/** - FreeBSD Ports - `www/seafile-server`
- **Ganesha** - NFS file server that runs in userspace mode.
  - **https://nfs-ganesha.github.io/** - FreeBSD Ports - `net/nfs-ganesha` + `net/nfs-ganesha-kmod`

External Solutions - Software Services

- **Syncthing** - encrypted file sync tool to replace cloud services with something open.
  - **https://syncthing.net/** - FreeBSD Ports - `net/syncthing`
- **Nextcloud** - personal cloud which runs on your own server (also check *OwnCloud*).
  - **https://nextcloud.com/** - FreeBSD Ports - `www/nextcloud`
- **Seafile** - file hosting software system.
  - **https://seafile.com/** - FreeBSD Ports - `www/seafile-server`
- **Ganesha** - NFS file server that runs in userspace mode.
  - **https://nfs-ganesha.github.io/** - FreeBSD Ports - `net/nfs-ganesha` + `net/nfs-ganesha-kmod`
- **Samba** - free SMB/CIFS and AD/DC server and client.
  - **https://samba.org/** - FreeBSD Ports - `net/samba410`

External Solutions - Availability

External Solutions - Availability

- **Corosync** - communication system for implementing HA within applications.
  - **https://corosync.github.io/corosync/** - FreeBSD Ports - `sysutils/corosync`

External Solutions - Availability

- **Corosync** - communication system for implementing HA within applications.
  - **https://corosync.github.io/corosync/** - FreeBSD Ports - `sysutils/corosync`
- **Pacemaker** - high availability cluster resource manager.
  - **https://wiki.clusterlabs.org/wiki/Pacemaker** - FreeBSD Ports - `sysutils/pacemaker2`

External Solutions - Availability

- **Corosync** - communication system for implementing HA within applications.
  - **https://corosync.github.io/corosync/** - FreeBSD Ports - `sysutils/corosync`
- **Pacemaker** - high availability cluster resource manager.
  - **https://wiki.clusterlabs.org/wiki/Pacemaker** - FreeBSD Ports - `sysutils/pacemaker2`
- **Heartbeat** - highly portable subsystem for high availability clustering.
  - **http://linux-ha.org/** - FreeBSD Ports - `sysutils/heartbeat`

External Solutions - Availability

- **Corosync** - communication system for implementing HA within applications.
  - **https://corosync.github.io/corosync/** - FreeBSD Ports - `sysutils/corosync`
- **Pacemaker** - high availability cluster resource manager.
  - **https://wiki.clusterlabs.org/wiki/Pacemaker** - FreeBSD Ports - `sysutils/pacemaker2`
- **Heartbeat** - highly portable subsystem for high availability clustering.
  - **http://linux-ha.org/** - FreeBSD Ports - `sysutils/heartbeat`
- **FreeBSD Services Control** - monitoring and automatic restarting for services.
  - **https://github.com/bsdtrhodes/freebsd-fscd/** - FreeBSD Ports - `sysutils/fsc`

External Solutions - Availability

- **Corosync** - communication system for implementing HA within applications.
  - **https://corosync.github.io/corosync/** - FreeBSD Ports - `sysutils/corosync`
- **Pacemaker** - high availability cluster resource manager.
  - **https://wiki.clusterlabs.org/wiki/Pacemaker** - FreeBSD Ports - `sysutils/pacemaker2`
- **Heartbeat** - highly portable subsystem for high availability clustering.
  - **http://linux-ha.org/** - FreeBSD Ports - `sysutils/heartbeat`
- **FreeBSD Services Control** - monitoring and automatic restarting for services.
  - **https://github.com/bsdtrhodes/freebsd-fscd/** - FreeBSD Ports - `sysutils/fsc`
- **Daemontools** - utilities for controlling and automatic restarting of processes.
  - **http://cr.yp.to/daemontools.html** - FreeBSD Ports - `sysutils/daemontools`

External Solutions - Listing

The **sysutils/lsblk** port provides similar to Linux block storage list tool on FreeBSD.

```
# lsblk
DEVICE          MAJ:MIN SIZE TYPE                                        LABEL MOUNT
da0               0:79  3.6T GPT                                             - -
  da0p1           0:92  3.6T dragonfly-hammer                                - -
  da0p1.eli       2:160 3.6T zfs                                             - -
ada1              0:99  932G GPT                                             - -
  ada1p1          0:101 200M efi                                  gpt/efiboot0 -
  ada1p2          0:102 512K freebsd-boot                         gpt/gptboot0 -
  <FREE>          -:-   492K -                                               - -
  ada1p3          0:103 931G freebsd-zfs                              gpt/zfs0 <ZFS>
  ada1p3.eli      0:106 931G zfs                                             - -
  <FREE>          -:-   708K -                                               - -
```

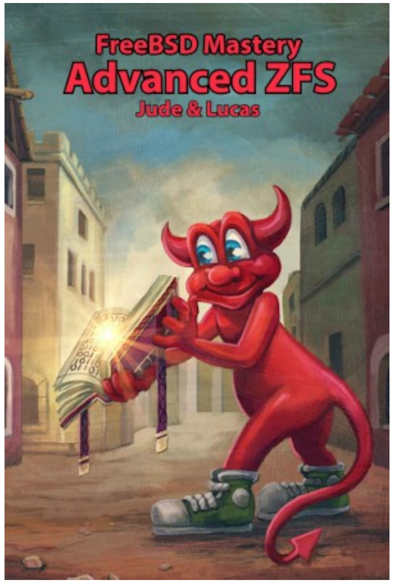
Commercial FreeBSD Storage Appliances

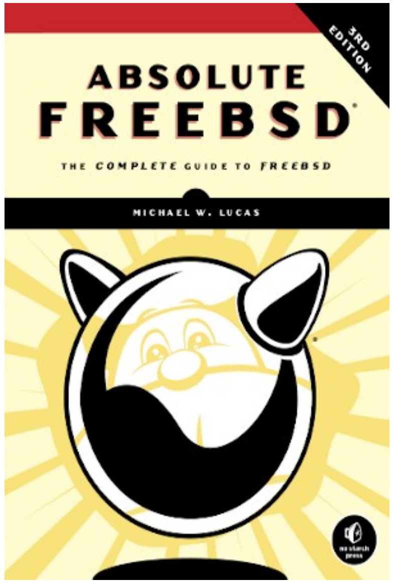
- **Spectra Verde Array** - https://spectralogic.com/
- **SGI ArcFiniti MAID Disk Arrays** - https://sgi.com/
- **QNAP Enterprise Storage (QES)** - https://qnap.com/qes/
- **Panasas ActiveStor Solutions** - https://panasas.com/
- **Netflix Open Connect Appliance** - https://netflix.com/
- **NetApp ONTAP Storage** - https://netapp.com/
- **Dell EMC Isilon OneFS Clustered Scale-Out Storage** - https://dellemc.com/
- **Dell Compellent Enterprise Storage** - https://dellemc.com/
- **Great Lakes SAN** - https://glsan.com/homeport/
- **RawDR** - https://rawdr.org/
- **iXsystems TrueNAS** - https://ixsystems.com/

Free/Open FreeBSD Storage Appliances

- **iXsystems FreeNAS** - https://freenas.org/
- **XigmaNAS** (NAS4Free) - https://xigmanas.com/
- **ZFSguru** - http://zfsguru.com/

Books on FreeBSD Storage





All written by **Michael W. Lucas** accompanied by **Allan Jude** for ZFS filesystem.

What Linux Has to Offer?

What Linux Has to Offer?

- EXT3 is/was **very limited** (even in its times) with only 2 TB file size limit.

What Linux Has to Offer?

- EXT3 is/was **very limited** (even in its times) with only 2 TB file size limit.
- EXT4 has similar (little larger) 16 TB file size limit.

What Linux Has to Offer?

- EXT3 is/was **very limited** (even in its times) with only 2 TB file size limit.
- EXT4 has similar (little larger) 16 TB file size limit.
- EXT4 **almost killed KDE** (almost lost their repositories) because of bugs in EXT4.

What Linux Has to Offer?

- EXT3 is/was **very limited** (even in its times) with only 2 TB file size limit.
- EXT4 has similar (little larger) 16 TB file size limit.
- EXT4 **almost killed KDE** (almost lost their repositories) because of bugs in EXT4.
  - More Here: **KDE Almost Lost All of Their Git Repositories - Phoronix**
    `https://www.phoronix.com/scan.php?page=news_item&px=MTMzNTc`

What Linux Has to Offer?

- EXT3 is/was **very limited** (even in its times) with only 2 TB file size limit.
- EXT4 has similar (little larger) 16 TB file size limit.
- EXT4 **almost killed KDE** (almost lost their repositories) because of bugs in EXT4.
  - More Here: **KDE Almost Lost All of Their Git Repositories - Phoronix**
    `https://www.phoronix.com/scan.php?page=news_item&px=MTMzNTc`
- XFS has **only metadata checksums** but **not for data**. Reasonable file size limits.

What Linux Has to Offer?

- EXT3 is/was **very limited** (even in its times) with only 2 TB file size limit.
- EXT4 has similar (little larger) 16 TB file size limit.
- EXT4 **almost killed KDE** (almost lost their repositories) because of bugs in EXT4.
  - More Here: **KDE Almost Lost All of Their Git Repositories - Phoronix**
    `https://www.phoronix.com/scan.php?page=news_item&px=MTMzNTc`
- XFS has **only metadata checksums** but **not for data**. Reasonable file size limits.
- BTRFS works in RAID0/RAID1 mode but **complete system rollcack is not possible**.

What Linux Has to Offer?

- EXT3 is/was **very limited** (even in its times) with only 2 TB file size limit.
- EXT4 has similar (little larger) 16 TB file size limit.
- EXT4 **almost killed KDE** (almost lost their repositories) because of bugs in EXT4.
  - More Here: **KDE Almost Lost All of Their Git Repositories - Phoronix** `https://www.phoronix.com/scan.php?page=news_item&px=MTMzNTc`
- XFS has **only metadata checksums** but **not for data**. Reasonable file size limits.
- BTRFS works in RAID0/RAID1 mode but **complete system rollcack is not possible**.
  - BTRFS warnings available here: **`https://wiki.debian.org/Btrfs#Warnings`**

What Linux Has to Offer?

- EXT3 is/was **very limited** (even in its times) with only 2 TB file size limit.
- EXT4 has similar (little larger) 16 TB file size limit.
- EXT4 **almost killed KDE** (almost lost their repositories) because of bugs in EXT4.
  - More Here: **KDE Almost Lost All of Their Git Repositories - Phoronix**
    `https://www.phoronix.com/scan.php?page=news_item&px=MTMzNTc`
- XFS has **only metadata checksums** but **not for data**. Reasonable file size limits.
- BTRFS works in RAID0/RAID1 mode but **complete system rollcack is not possible**.
  - BTRFS warnings available here: **`https://wiki.debian.org/Btrfs#Warnings`**
- STRATIS uses XFS over LVM and `device-mapper` to imitate pools like in ZFS.

What Linux Has to Offer?

- EXT3 is/was **very limited** (even in its times) with only 2 TB file size limit.
- EXT4 has similar (little larger) 16 TB file size limit.
- EXT4 **almost killed KDE** (almost lost their repositories) because of bugs in EXT4.
  - More Here: **KDE Almost Lost All of Their Git Repositories - Phoronix**
    `https://www.phoronix.com/scan.php?page=news_item&px=MTMzNTc`
- XFS has **only metadata checksums** but **not for data**. Reasonable file size limits.
- BTRFS works in RAID0/RAID1 mode but **complete system rollcack is not possible**.
  - BTRFS warnings available here: **`https://wiki.debian.org/Btrfs#Warnings`**
- STRATIS uses XFS over LVM and `device-mapper` to imitate pools like in ZFS.
  - Plan to achieve checksums for data somewhere in the future.

What Linux Has to Offer?

- EXT3 is/was **very limited** (even in its times) with only 2 TB file size limit.
- EXT4 has similar (little larger) 16 TB file size limit.
- EXT4 **almost killed KDE** (almost lost their repositories) because of bugs in EXT4.
  - More Here: **KDE Almost Lost All of Their Git Repositories - Phoronix**
    `https://www.phoronix.com/scan.php?page=news_item&px=MTMzNTc`
- XFS has **only metadata checksums** but **not for data**. Reasonable file size limits.
- BTRFS works in RAID0/RAID1 mode but **complete system rollcack is not possible**.
  - BTRFS warnings available here: **`https://wiki.debian.org/Btrfs#Warnings`**
- STRATIS uses XFS over LVM and `device-mapper` to imitate pools like in ZFS.
  - Plan to achieve checksums for data somewhere in the future.
- ZFS almost *"first class citizen"* in **Ubuntu** but **ZFS Boot Enviroments still not available**.

What Linux Has to Offer?

What Linux Has to Offer?

- STRATIS is created and developed by Red Hat.

What Linux Has to Offer?

- STRATIS is created and developed by Red Hat.
  - After huge and undisputed success of `systemd` STRATIS will thrive for sure.

What Linux Has to Offer?

- STRATIS is created and developed by Red Hat.
  - After huge and undisputed success of `systemd` STRATIS will thrive for sure.
  - Design and future ideas described in **Stratis Software Design** document.

    `https://stratis-storage.github.io/StratisSoftwareDesign.pdf`

What Linux Has to Offer?

- STRATIS is created and developed by Red Hat.
  - After huge and undisputed success of `systemd` STRATIS will thrive for sure.
  - Design and future ideas described in **Stratis Software Design** document.
    `https://stratis-storage.github.io/StratisSoftwareDesign.pdf`
- **Red Hat Enterprise Linux** is generally considered most business oriented Linux.

What Linux Has to Offer?

- STRATIS is created and developed by Red Hat.
  - After huge and undisputed success of `systemd` STRATIS will thrive for sure.
  - Design and future ideas described in **Stratis Software Design** document.

    `https://stratis-storage.github.io/StratisSoftwareDesign.pdf`
- **Red Hat Enterprise Linux** is generally considered most business oriented Linux.
  - Does not have supported filesystem that provides data consistency/checksums.

What Linux Has to Offer?

- STRATIS is created and developed by Red Hat.
  - After huge and undisputed success of `systemd` STRATIS will thrive for sure.
  - Design and future ideas described in **Stratis Software Design** document.

    `https://stratis-storage.github.io/StratisSoftwareDesign.pdf`

- **Red Hat Enterprise Linux** is generally considered most business oriented Linux.
  - Does not have supported filesystem that provides data consistency/checksums.
  - With RHEL8 its possible to detect bit rot using `dm-integrity` kernel code.

What Linux Has to Offer?

- STRATIS is created and developed by Red Hat.
  - After huge and undisputed success of `systemd` STRATIS will thrive for sure.
  - Design and future ideas described in **Stratis Software Design** document.

    `https://stratis-storage.github.io/StratisSoftwareDesign.pdf`

- **Red Hat Enterprise Linux** is generally considered most business oriented Linux.
  - Does not have supported filesystem that provides data consistency/checksums.
  - With RHEL8 its possible to detect bit rot using `dm-integrity` kernel code.
    - More here: **What is Bit Rot and How Can I Detect It on RHEL?**

      `https://redhat.com/en/blog/what-bit-rot-and-how-can-i-detect-it-rhel`

What Linux Has to Offer?

- STRATIS is created and developed by Red Hat.
  - After huge and undisputed success of `systemd` STRATIS will thrive for sure.
  - Design and future ideas described in **Stratis Software Design** document.
    `https://stratis-storage.github.io/StratisSoftwareDesign.pdf`

- **Red Hat Enterprise Linux** is generally considered most business oriented Linux.
  - Does not have supported filesystem that provides data consistency/checksums.
  - With RHEL8 its possible to detect bit rot using `dm-integrity` kernel code.
    - More here: **What is Bit Rot and How Can I Detect It on RHEL?**
      `https://redhat.com/en/blog/what-bit-rot-and-how-can-i-detect-it-rhel`
  - Not possible with RHEL7 or RHEL6 versions of Red Hat Enterprise Linux.

Example Implementation of FreeBSD Storage

Example Implementation of FreeBSD Storage

- Inspitations?

Example Implementation of FreeBSD Storage

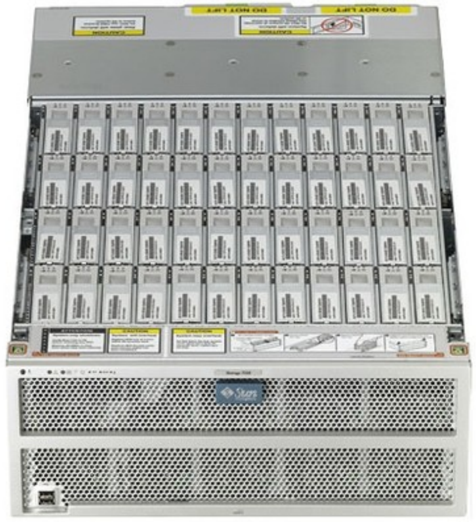
- Inspitations?
  - **Sun Storage 7210**
    - `https://docs.oracle.com/cd/E19360-01/pdf/821-1388.pdf` (Page 37)

Example Implementation of FreeBSD Storage

- Inspitations?
  - **Sun Storage 7210**
    - `https://docs.oracle.com/cd/E19360-01/pdf/821-1388.pdf` (Page 37)
  - **Sun Fire X4500/X4540**
    - `https://docs.oracle.com/cd/E19469-01/819-4359-19/CH3-maint.html`

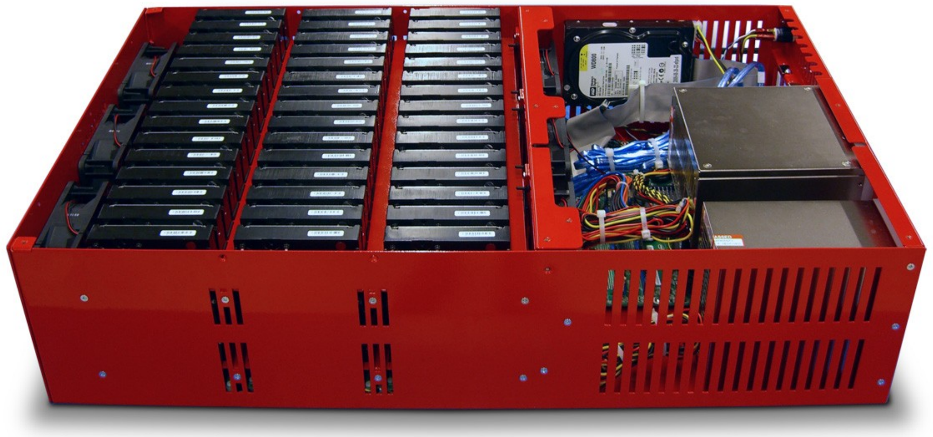
Example Implementation of FreeBSD Storage

- Inspitations?
  - **Sun Storage 7210**
    - `https://docs.oracle.com/cd/E19360-01/pdf/821-1388.pdf` (Page 37)
  - **Sun Fire X4500/X4540**
    - `https://docs.oracle.com/cd/E19469-01/819-4359-19/CH3-maint.html`



Example Implementation of FreeBSD Storage

- Inspitations?
  - **Backblaze Storage Pod**
    - `https://www.backblaze.com/b2/storage-pod.html`



Idea Taken to the Extreme

Thunder SX FA100-B7118 (100 Bays)

Supermicro 6048R-E1CR90L (90 Bays)

Zstor GS41100 (100 Bays)

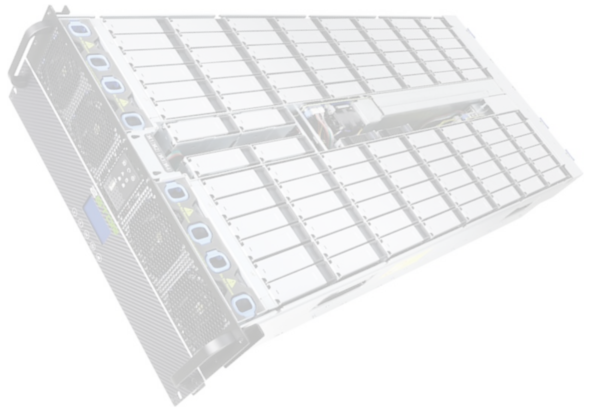
Inspur NF5486M5 (104 Bays)

Idea Taken to the Extreme

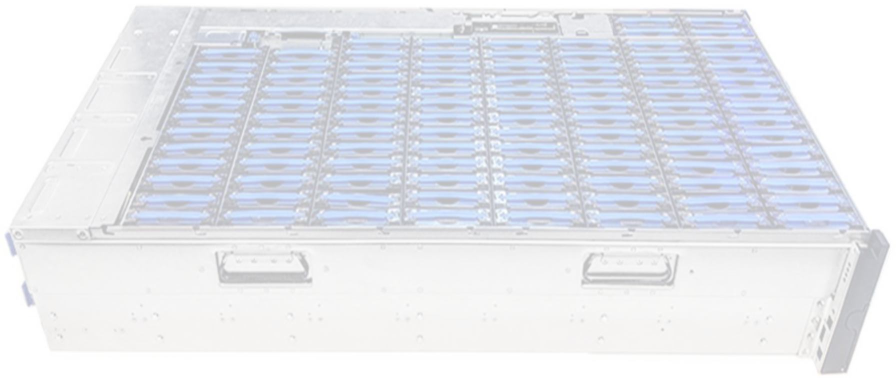
Thunder SX FA100-B7118 (100 Bays)

Supermicro 6048R-E1CR90L (90 Bays)

Zstor GS41100 (100 Bays)



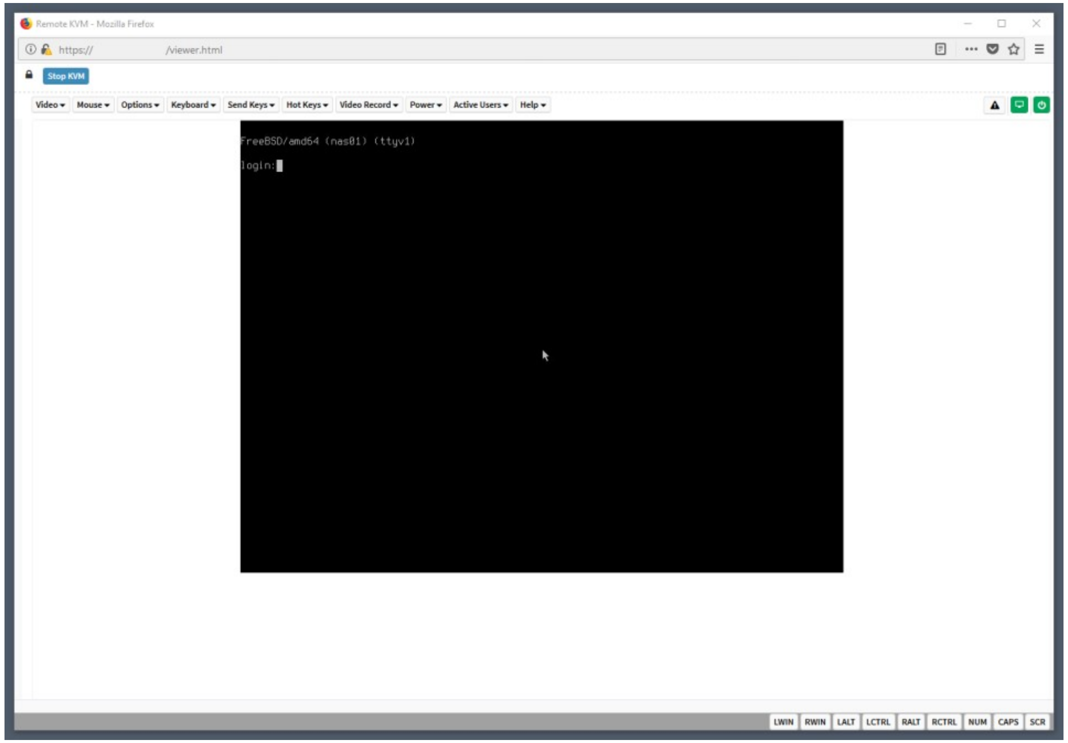
Inspur NF5486M5 (104 Bays)



Idea Taken to the Extreme

Thunder SX FA100-B7118 (Management)

- Provides HTML5 based plugin free **Remote Control**.



```
FreeBSD/amd64 (nas01) (ttyv1)

login:
```

Idea Taken to the Extreme

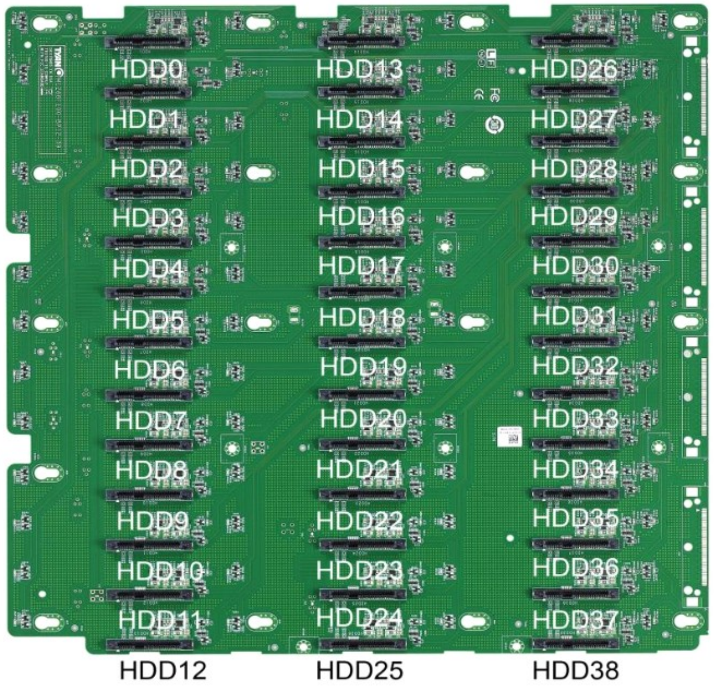
Thunder SX FA100-B7118 (Hardware)

- 2 x 10-Core Intel Xeon Silver 4114 CPU @ 2.20GHz (**20 Cores Total**)
- 4 x 32 GB RAM DDR4 (**128 GB Total**)
- 2 x Intel SSD DC S3500 240 GB (**System**)
- 90 x Toshiba HDD MN07ACA12TE 12 TB (**Data**)
- 2 x Broadcom SAS3008 Controller
- 2 x Intel X710 DA-2 10GE Card (**4 x 10GE Total**)
- 2 x Power Supply
- 8 x Free Disks Slots

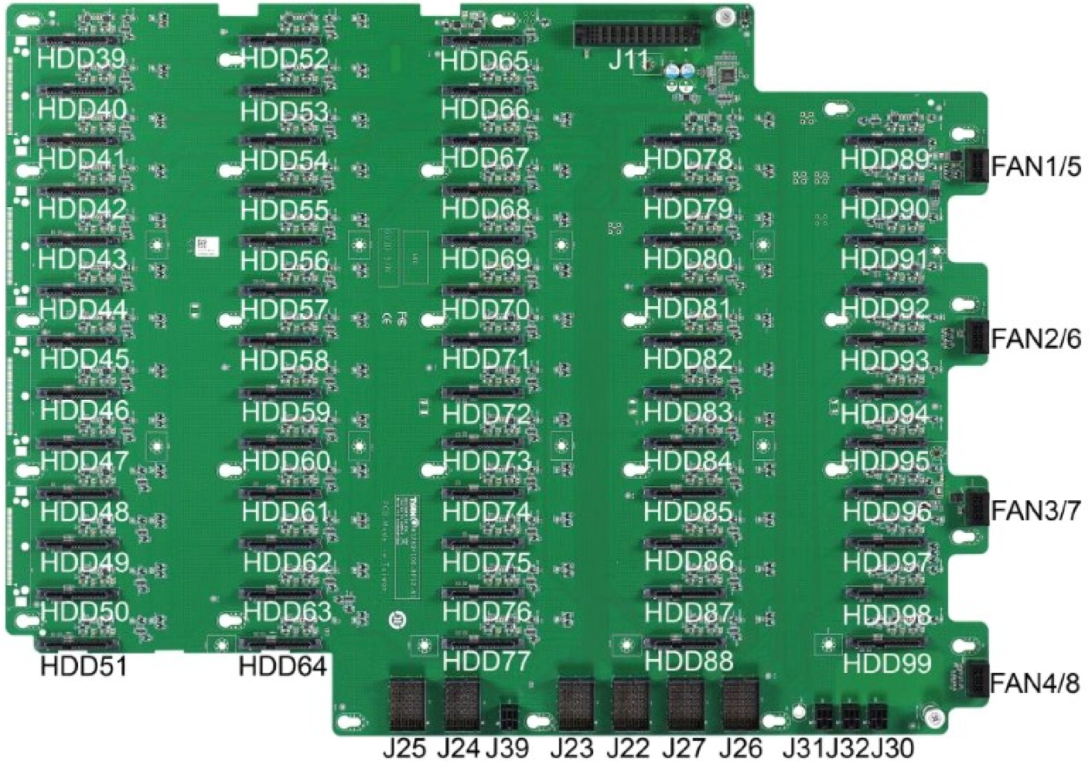
Idea Taken to the Extreme

Thunder SX FA100-B7118 (Disks Split Between Controllers)

M1288F100-BP12-39 (39 Disks)



M1289F100-BP12-61 (61 Disks)



Idea Taken to the Extreme

Thunder SX FA100-B7118 (ZFS Configuration)

- ZFS Pool - System - **RAID1** (ZFS Mirror) - **One SSD Disk Per Controller**

Idea Taken to the Extreme

Thunder SX FA100-B7118 (ZFS Configuration)

- ZFS Pool - System - **RAID1** (ZFS Mirror) - **One SSD Disk Per Controller**
- ZFS Pool - Data - **RAID60** (ZFS Striped RAIDZ2) - **36:48 Data Ratio** - **2:4 Spare Ratio**

```
DISKS  CONTENT
   12  raidz2-0
   12  raidz2-1
   12  raidz2-2
   12  raidz2-3
   12  raidz2-4
   12  raidz2-5
   12  raidz2-6
    6  spares
   90  TOTAL
```

Idea Taken to the Extreme

Thunder SX FA100-B7118 (ZFS Data Pool Status)

```
# zpool status
  pool: nas02
 state: ONLINE
  scan: scrub repaired 0 in 0 days 00:00:05 with 0 errors on Fri May 31 10:26:29 2019
config:

        NAME          STATE     READ WRITE CKSUM
        nas02         ONLINE       0     0     0
          raidz2-0    ONLINE       0     0     0
            da0p1     ONLINE       0     0     0
            da1p1     ONLINE       0     0     0
            da2p1     ONLINE       0     0     0
            da3p1     ONLINE       0     0     0
            da4p1     ONLINE       0     0     0
            da5p1     ONLINE       0     0     0
            da6p1     ONLINE       0     0     0
            da7p1     ONLINE       0     0     0
            da8p1     ONLINE       0     0     0
            da9p1     ONLINE       0     0     0
            da10p1    ONLINE       0     0     0
            da12p1    ONLINE       0     0     0
          raidz2-1    ONLINE       0     0     0
            ( ... )
```

Idea Taken to the Extreme

Thunder SX FA100-B7118 (ZFS Data Pool Status)

```
          ( ... )
          da71p1  ONLINE       0     0     0
          da72p1  ONLINE       0     0     0
          da73p1  ONLINE       0     0     0
          da74p1  ONLINE       0     0     0
        spares
          da36p1    AVAIL
          da37p1    AVAIL
          da85p1    AVAIL
          da86p1    AVAIL
          da87p1    AVAIL
          da88p1    AVAIL

errors: No known data errors

# zpool list nas02
NAME    SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG    CAP  DEDUP  HEALTH  ALTROOT
nas02   915T  1.42M   915T        -         -     0%     0%  1.00x  ONLINE  -

# zfs list nas02
NAME    USED  AVAIL  REFER  MOUNTPOINT
nas02    88K   675T   201K  none
```

Idea Taken to the Extreme

Thunder SX FA100-B7118 (Storage Performance)

- FreeBSD's builtin `diskinfo(8)` tool.

```
# diskinfo -ctv /dev/zvol/nas02/iscsi/test
( ... )
Transfer rates:
        outside:       102400 kbytes in   0.036938 sec =  2772213 kbytes/sec
        middle:        102400 kbytes in   0.043076 sec =  2377194 kbytes/sec
        inside:        102400 kbytes in   0.034260 sec =  2988908 kbytes/sec
```

Idea Taken to the Extreme

Thunder SX FA100-B7118 (Storage Performance)

- FreeBSD's builtin `diskinfo(8)` tool.

```
# diskinfo -ctv /dev/zvol/nas02/iscsi/test
( ... )
Transfer rates:
        outside:       102400 kbytes in   0.036938 sec =  2772213 kbytes/sec
        middle:        102400 kbytes in   0.043076 sec =  2377194 kbytes/sec
        inside:        102400 kbytes in   0.034260 sec =  2988908 kbytes/sec
```

- Eight concurrent `dd(8)` processes.

```
# dd if=/dev/zero of=FILE${X} bs=128m status=progress
174214610944 bytes (174 GB, 162 GiB) transferred 385.042s, 452 MB/s
1302+0 records in
1301+0 records out
174617264128 bytes transferred in 385.379296 secs (453104943 bytes/sec)
```

Idea Taken to the Extreme

Thunder SX FA100-B7118 (Storage Performance)

- FreeBSD's builtin `diskinfo(8)` tool.

```
# diskinfo -ctv /dev/zvol/nas02/iscsi/test
( ... )
Transfer rates:
        outside:       102400 kbytes in   0.036938 sec =   2772213 kbytes/sec
        middle:        102400 kbytes in   0.043076 sec =   2377194 kbytes/sec
        inside:        102400 kbytes in   0.034260 sec =   2988908 kbytes/sec
```

- Eight concurrent `dd(8)` processes.

```
# dd if=/dev/zero of=FILE${X} bs=128m status=progress
174214610944 bytes (174 GB, 162 GiB) transferred 385.042s, 452 MB/s
1302+0 records in
1301+0 records out
174617264128 bytes transferred in 385.379296 secs (453104943 bytes/sec)
```

- About **3 GB/s** of sustained disk subsystem performance.

Idea Taken to the Extreme

Thunder SX FA100-B7118 (FreeBSD Network Configuration)

```
# head -5 /etc/rc.conf
  defaultrouter="10.20.30.254"
  ifconfig_ixl0="up"
  ifconfig_ixl1="up"
  cloned_interfaces="lagg0"
  ifconfig_lagg0="laggproto lacp laggport ixl0 laggport ixl1 10.20.30.2/24 up"

# ifconfig lagg0
lagg0: flags=8843 metric 0 mtu 1500
        options=e507bb
        ether a0:42:3f:a0:42:3f
        inet 10.20.30.2 netmask 0xffffff00 broadcast 10.20.30.255
        laggproto lacp lagghash l2,l3,l4
        laggport: ixl0 flags=1c
        laggport: ixl1 flags=1c
        groups: lagg
        media: Ethernet autoselect
        status: active
        nd6 options=29
```

Idea Taken to the Extreme

Thunder SX FA100-B7118 (Network Performance)

- Test performed with `iperf3(1)` from two **Windows Server 2016** machines.

Idea Taken to the Extreme

Thunder SX FA100-B7118 (Network Performance)

- Test performed with `iperf3(1)` from two **Windows Server 2016** machines.
  - Unfortunatelly with **1500 MTU** (no **Jumbo Frames** for more performance).

Idea Taken to the Extreme

Thunder SX FA100-B7118 (Network Performance)

- Test performed with `iperf3(1)` from two **Windows Server 2016** machines.
  - Unfortunatelly with **1500 MTU** (no **Jumbo Frames** for more performance).
  - The `iperf3(1)` server started on the **FreeBSD** machine.

```
# iperf3 -s
```

Idea Taken to the Extreme

Thunder SX FA100-B7118 (Network Performance)

- Test performed with `iperf3(1)` from two **Windows Server 2016** machines.
  - Unfortunatelly with **1500 MTU** (no **Jumbo Frames** for more performance).
  - The `iperf3(1)` server started on the **FreeBSD** machine.

    ```
    # iperf3 -s
    ```

  - Two `iperf3(1)` clients started on the **Windows Server 2016** machine.

Idea Taken to the Extreme

Thunder SX FA100-B7118 (Network Performance)

- Test performed with `iperf3(1)` from two **Windows Server 2016** machines.
  - Unfortunatelly with **1500 MTU** (no **Jumbo Frames** for more performance).
  - The `iperf3(1)` server started on the **FreeBSD** machine.

```
# iperf3 -s
```

  - Two `iperf3(1)` clients started on the **Windows Server 2016** machine.
  - Output below from one of the **Windows Server 2016** machines.

```
# C:\iperf-3.1.3-win64>iperf3.exe -c nas02 -P 8
( ... )
[SUM]   0.00-10.00  sec  10.8 GBytes  9.26 Gbits/sec                  receiver
( ... )
```

Idea Taken to the Extreme

Thunder SX FA100-B7118 (Network Performance)

- Test performed with `iperf3(1)` from two **Windows Server 2016** machines.
    - Unfortunatelly with **1500 MTU** (no **Jumbo Frames** for more performance).
    - The `iperf3(1)` server started on the **FreeBSD** machine.

```
# iperf3 -s
```

    - Two `iperf3(1)` clients started on the **Windows Server 2016** machine.
    - Output below from one of the **Windows Server 2016** machines.

```
# C:\iperf-3.1.3-win64>iperf3.exe -c nas02 -P 8
( ... )
[SUM]   0.00-10.00  sec  10.8 GBytes  9.26 Gbits/sec                  receiver
( ... )
```

- Each **Windows Server 2016** machine had only one 10GE interface.

Idea Taken to the Extreme

Thunder SX FA100-B7118 (Network Performance)

- Test performed with `iperf3(1)` from two **Windows Server 2016** machines.
  - Unfortunatelly with **1500 MTU** (no **Jumbo Frames** for more performance).
  - The `iperf3(1)` server started on the **FreeBSD** machine.

```
# iperf3 -s
```

  - Two `iperf3(1)` clients started on the **Windows Server 2016** machine.
  - Output below from one of the **Windows Server 2016** machines.

```
# C:\iperf-3.1.3-win64>iperf3.exe -c nas02 -P 8
( ... )
[SUM]   0.00-10.00  sec  10.8 GBytes  9.26 Gbits/sec                  receiver
( ... )
```

- Each **Windows Server 2016** machine had only one 10GE interface.
- The **FreeBSD** machine had two 10GE interfaces configured in LACP mode.

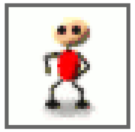
Idea Taken to the Extreme

Thunder SX FA100-B7118 (More Tests and Details)

- More details on dedicated blog post on **`https://vermaden.wordpress.com`** page.
  - **FreeBSD Enterprise 1 PB Storage**
  - `https://vermaden.wordpress.com/2019/06/19/freebsd-enterprise-1-pb-storage/`

Questions?

Sławomir Wojciech Wojtczak

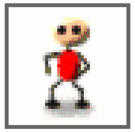


vermaden@interia.pl
vermaden.wordpress.com
twitter.com/vermaden
bsd.network/@vermaden

https://is.gd/bsdstg

Thank You!

Sławomir Wojciech Wojtczak



vermaden@interia.pl
vermaden.wordpress.com
twitter.com/vermaden
bsd.network/@vermaden

https://is.gd/bsdstg