1. What is EfficientNodes?

EfficientNodes is an application for the **audit-proof archiving** of data. These can consist of structured and unstructured information (files and their metadata).

Because EfficientNodes creates any number of copies of the data, it is also a very powerful data **backup** and **restore** function.

2. How does EfficientNodes work?

Each **node** works with **data sources** and **data targets**.

These can be:

- Filesystems (SMB, NFS)
- Cloud Storage
- Applications

Each node can work parallel with any number of data targets. This is called a multi-cloud and multi-tier capability of EfficientNodes.
These nodes can be combined into service chains using very efficient and simple mechanisms.
Any number of power chains can be operated in parallel.

This can create the following archive chain, for example:

N 0	=	Data Source (Filesystem)
N 1	=	Copy 1 (Filesystem)
N 2.1	=	Copy 21 (Cloud Storage 1)
N 2.2	=	Copy 22 (Cloud Storage 2)

```
                              =>    N 2.1
N 0     =>     N 1
                              =>    N 2.2
```

In principle, the data is only **copied once** to the respective nodes. Changes in the original (N 0) are recognized and corresponding **versions** are created **in the archive**.

Thanks to the **integrated versioning** of EfficientNodes, there is no longer any need to constantly back up the data, as is usual in data backup (e.g. full backups weekly, monthly and annual backups).

Another advantage is a significantly lower system load by **avoiding full backups** and a corresponding to **lower space requirement** in the archive.

Furthermore, files and their metadata can be **combined into transparent data containers** (size, quantity, logical criteria - e.g. customer numbers) through automated processing.

3. EfficientNodes fulfills the 3-2-1 rule

The **3-2-1** rule states:

3 copies of the data
2 of them on different storage systems
1 of them at another location (offsite, offline)

- EfficientNodes is operated according to best practice with at least 3 copies
- EfficientNodes creates its own copies of the data, regardless of the technology of the respective storage system (technical separation - no synchronous mirroring of a manufacturer)
- The EfficientNodes CloudArchive (HTTPS) creates an offsite copy of the data (no access via shared files such as SMB or NFS - virus risk - manipulation)

These measures result in a very high level of security within EfficientNodes.

4. Metadata processing - XML command files

In addition to unstructured information (files), EfficientNodes also processes structured information.
This can be pure right-click information (file - right-click - properties), or metadata from applications (such as invoice, insurance, production numbers or document classes).

This turns unstructured information in the file system into structured information in EfficientNodes.

The metadata can be accepted via the integrated communication server (Orchestra® from Soffico) or via direct interfaces with the applications
(Application **software integration**).

Technical metadata is accepted by EfficientNodes' own integrations (**JetDetect**) with manufacturers (Microsoft Windows Server, NetApp FAS Fpolicy, Hitachi HNAS RESTFull API, ...).

With **JetDetect HW integration**, the information (new files, changes to files ...) is processed almost in "real time".

EfficientNodes can of course, as is usual with most programs for data backup, obtain the respective information (XML files) through its own services (**XML generator**).
This of course leads to a greater load on the system (“Treewalk” in the file system) and takes all the longer.

These methods (**SW, HW integration, XML generator**) can also be used in **parallel**.

5. Audit security, legal security and blockchain

EfficientNodes creates **signatures** (# SHA-256) of the newly added objects right at the beginning of the processing chain. Like the other metadata, signatures are passed on within the processing chain (3-2-1 rule).

Verifications can be carried out on objects in individual nodes or on the entire processing chain.

5.1 Verification - Validation

A new **signature** is calculated for the object to be examined (file or data container) and this is **compared with the existing signature** (s). If the signatures are the same, it is proven that the object has not been changed or damaged by other influences.

This results in a **very high degree of revision security**, since in the event of manipulation, archive objects and their signatures have to be penetrated in each node.

5.2 Maximum security with blockchain

An even **higher level of security** is achieved through additional to store the signatures in the **blockchain**. EfficientNodes stores the receipts (seals) sent back by the blockchain provider in the corresponding nodes and data containers.

The signatures are **stored in different blockchains** and cannot be changed there.

A check of the data with the help of the signatures and the seal is possible at any time within the EfficientNodes Management Client (HTTP).
As described above, a new signature is calculated and this is then compared with the one stored in the blockchain (s). If the **values are identical, the data are valid**.

5.3 Real-time protection of archives - ArchiveWatch

With ArchiveWatch, EfficientNodes is able to **monitor the archives in "real time"**.

This is done via the JetDetect HW integration with the respective manufacturers.
Unauthorized changes or deletions to the data in the archives are displayed immediately and, if required, appropriate actions are initiated.

Actions can be, for example:

- Critical warning messages watchdog
- Email notifications
- Shutdown of archives (take offline)
- …

5.4 Signatures versus WORM versus GDPR (General Data Protection Regulation)

A WORM system (write once read many) is nowadays mostly a hard disk storage system from one manufacturer. When the data is saved on these systems, a retention period is set during which the data can no longer be changed.

The disadvantage here is that the protection against changes only exists as long as the data is on the respective system. If the system is replaced, often after a few years, the data has to be laboriously migrated. In doing so, they **often lose all protection**.

In contrast, the **signatures in EfficientNodes always offer protection**, as they are carried on every node or on newly created nodes.
No data migration is necessary within EfficientNodes - new nodes are simply created as copies and others are removed or shut down.
WORM systems are just as susceptible to **technical failures or logical errors** (firmware) as other storage systems.

This is why a second system from the same manufacturer is usually used and the data is then replicated.
However, this violates rule 2 of the 3-2-1 rule (2 copies on different systems - technical separation). A separate **backup for these systems follows as a requirement**.

Another problem with **WORM is dealing with the GDPR**.
If data from e.g. customers are to be deleted at their request, this is not possible retrospectively on WORM systems.

6. Data container

EfficientNodes usually creates archives as the first copy, in which the data is stored in the file system, the metadata separately in XML files.

The **advantage** of this is that if data is lost in the data source (N0), the applications can immediately take over the archive file system (N1) ("remapping"). The data in the archive (N1) are saved as a 1: 1 copy of the original data.

The **disadvantage** with this is that in the further processes (3-2-1 rule) a lot of, mostly very small files and their metadata have to be processed.

The **solution are data containers** in which the **data, metadata, signatures and transactions** are stored according to logical criteria.

All relevant information is logically stored together in one object in these data containers. As a result, the highest level of data integrity is achieved.

In addition, the data containers are provided with their **own signature**, optionally of course with a **blockchain seal**. This increases the security (individual signatures of the files + container signatures) and verifiability.
If the signature or seal of the **container is valid**, the individual objects within the container are also valid.
If the signature or the seal of the container is not valid, the individual objects within the container can be examined.

In addition, this leads to a **minimization of the costs** for blockchain seals - e.g. 1 seal instead of 1000 seals for each individual file within the container.

The data containers can also be **encrypted (AES-256)** on request.

When transporting the data container (LAN or WAN), there are **improvements in speed in the range of 1: 1000** - with a significantly lower system load.

The data containers can of course be stored much **more efficiently** and also more securely in the **CloudArchive**.

When **exporting** from the data container, the **original source file system is created again** - no data break with loss of the original data paths.

7. CloudArchive

The EfficientNodes **CloudArchive** can be operated in your **own data center** (data center), in an **external data center** by or for one or more customers.
It can be installed as a CloudArchive **Appliance**, as a **VM**, or in one or more containers under **Linux** (Windows).

The **CloudArchive provides HTTPS ports** with which other nodes communicate.
For example, a node has a file system as its data source and the CloudArchive as its data destination.
In the same way, a node can have the CloudArchive as its data source and a file system or another CloudArchive as the data destination.

The advantage here is that **no shares via SMB or NFS are used,** but an HTTPS gateway. The data is therefore not visible in the network, as is the case with file shares, and is therefore offline. If the CloudArchive is in a different location, one can speak of offsite and offline storage of the data (see 3-2-1 rule: 1 copy offsite, offline).

This is particularly **important** when, as is usually the case today, **tape backups are no longer used**. The CloudArchive also serves as protection against viruses and ransomware.

Due to the HTTPS ports and the encrypted transmission of the data container, **no VPN tunnel** with its own encryption is required.
By **transmitting relatively large data containers,** WAN and LAN connections are used very effectively (latency times, TCP headers, loss due to encryption).

In a **classic ObjectStorage**, the files and metadata are stored in a **proprietary, manufacturer-specific format**. The interface for this is usually an **Amazon S3** (Simple Storage Service) compatible.

In the EfficientNodes **CloudArchive,** the **data is stored transparently** in the same way as in an FS node or container node, **protected by an HTTPS gateway**.

Additional nodes can be installed behind a CloudArchive:

N 0	=	Data Source
N 1	=	FS Node
N 2	=	Container Node
N 3	=	CloudArchive
N 4	=	Container Node
N 5	=	FS Node

N 0	=>	N 1	=>	N 2	=>	N3	=>	N4	=>	N6
Source		FS Node		Container Node		**CloudArchive**		Container Node		FS Node

Appendix 1: Overview of information

Attestation carried out by Rödl & Partner in the 2nd half of 2020

TÜV Rheinland certificate certified service process - archiving process

Patent granted by the German Patent and Trademark Office - procedure for securing data (Patent No. 10 2014 108 417)

Here are a few links to our software solution EfficientNodes

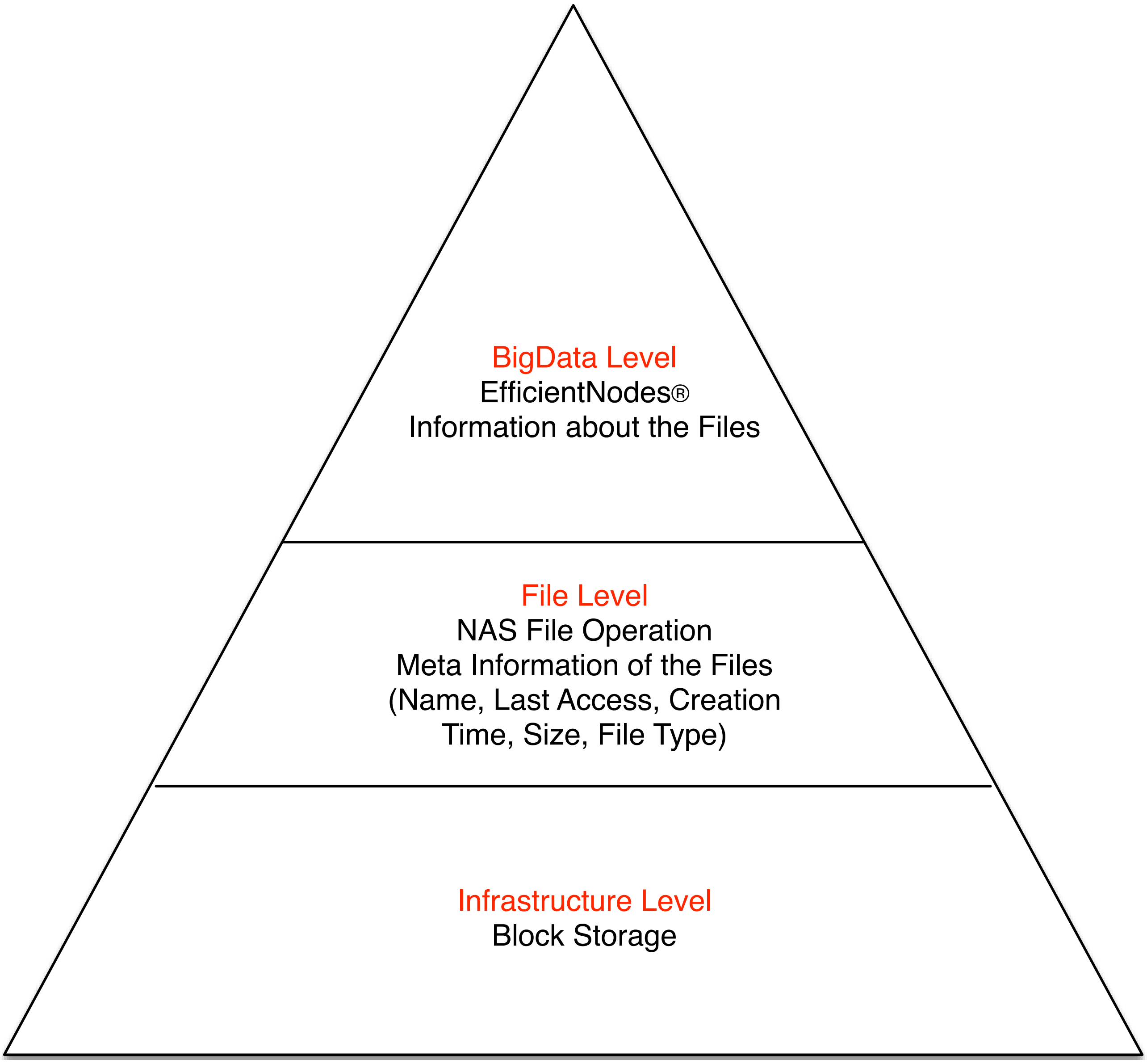
EfficientNodes Movie:
https://www.youtube.com/watch?v=P0u38KAfqEM

EfficientNodes Movie App Cloud:
https://www.youtube.com/watch?v=14S42gr6XI4

EfficientNodes Software with Blockchain:
https://www.youtube.com/watch?v=iK1-VCYJfhc

EfficientNodes Movie Disaster Recovery:
https://www.youtube.com/watch?v=nzCLstzFyGU

Appendix 2 pyramid:



BigData Level
EfficientNodes®
Information about the Files

File Level
NAS File Operation
Meta Information of the Files
(Name, Last Access, Creation Time, Size, File Type)

Infrastructure Level
Block Storage