EFFICIENTNODES

1. Was ist EfficientNodes?

EfficientNodes ist eine Anwendung zum **revisionssicheren Archivieren** von Daten. Diese können aus strukturierten und unstrukturierten Informationen bestehen (Dateien und deren Metadaten).

Dadurch dass EfficientNodes beliebig viele Kopien der Daten erzeugt, ist es gleichzeitig auch eine sehr leistungsfähige **Datensicherung und Wiederherstellung**.

2. Wie funktioniert EfficientNodes?

Jeder **Node** von EfficientNodes arbeitet mit **Datenquellen und Datenzielen**.

Diese können sein:

- Dateisysteme (SMB, NFS)
- Cloud Speicher
- Anwendungen

Dabei kann jeder **Node** mit beliebig vielen Datenzielen parallel arbeiten. Deshalb spricht man von einer **Multi-Cloud** und **Multi-Tier** Fähigkeit von EfficientNodes.
Diese Nodes können durch sehr effiziente und dabei einfache Mechanismen zu **Leistungsketten** zusammen gefügt werden. Es können **beliebig viele Leistungsketten** parallel betrieben werden.

Damit kann z.B. folgende Archiv Kette entstehen:

N 0 = Datenquelle (Dateisystem)
N 1 = Kopie 1 (Dateisystem)
N 2.1 = Kopie 21 (Cloud Speicher 1)
N 2.2 = Kopie 22 (Cloud Speicher 2)

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

Die Daten werden grundsätzlich nur **einmal** auf die jeweiligen Nodes **kopiert**. Änderungen im Original (N 0) werden erkannt und entsprechende **Versionen im Archiv** erzeugt.

Durch die integrierte **Versionierung** von EfficientNodes besteht keine Notwendigkeit mehr die Daten, wie in der Datensicherung üblich, ständig neu zu Sichern (z.B. Vollsicherungen wöchentlich, Monats- und Jahressicherungen).

Ein weiterer Vorteil ist eine deutlich geringere Systembelastung durch **Vermeidung von Vollsicherungen** und ein entsprechend **geringer Platzbedarf** im Archiv.

Des weiteren können durch automatisierte Verarbeitungsprozesse Dateien und deren Metadaten zu transparenten **Datencontainern** (Größe, Menge, logische Kriterien - z.B. Kundennummern) zusammen gefaßt werden.

albin.brandl@efficientnodes.de

3. EfficientNodes erfüllt die 3-2-1 Regel

Die **3-2-1** Regel besagt:

3 Kopien der Daten
2 davon auf unterschiedlichen Speichersystemen
1 davon auf einem anderen Standort (offsite, offline)

- EfficientNodes wird nach best practice mit mindestens 3 Kopien betrieben
- EfficientNodes schafft, unabhängig von der Technik des jeweiligen Speichersystems, eigene Kopien der Daten (technische Trennung - keine synchrone Spiegelung eines Herstellers)
- Durch das CloudArchive (HTTPS) von EfficientNodes wird eine Offsite Kopie der Daten erzeugt (kein Zugriff über Dateifreigaben wie SMB oder NFS - Virengefahr - Manipulation)

Durch diese Maßnahmen entsteht ein sehr hoher Sicherheitsstand innerhalb EfficientNodes.

4. Verarbeitung von Metadaten - XML Befehlsdateien

EfficientNodes verarbeitet neben unstrukturierten Informationen (Dateien) auch strukturierte Informationen.
Diese können reine Rechtsklick Informationen (Datei - Rechtsklick - Eigenschaften) sein, oder Metadaten von Anwendungen (wie Rechnungs- ,Versicherten-, Fertigungsnummern oder Dokumentklassen).

Damit werden aus unstrukturierten Informationen im Dateisystem strukturierte Informationen in EfficientNodes.

Die MetaDaten können über den integrierten Kommunikationsserver (Orchestra® von Soffico) angenommen werden, oder über direkte Schnittstellen mit den Anwendungen (Applikation **SW-Integration**).

Technische Metadaten werden über eigene Integrationen (**JetDetect**) von EfficientNodes mit Herstellern (Microsoft Windows Server, NetApp FAS Fpolicy, Hitachi HNAS RESTFull API, …) angenommen.
Bei **JetDetect HW-Integration** werden die Informationen (neue Dateien, Änderungen an Dateien…) nahezu in „Echtzeit“ verarbeitet.

EfficientNodes kann natürlich, wie bei den meisten Programmen zur Datensicherung üblich, die jeweilen Informationen (XML Dateien) durch eigene Services (**XML-Generator**) gewinnen.
Dies führt natürlich zu einer größeren Belastung des Systems („Treewalk“ im Dateisystem) und dauert umso länger.

Diese Methoden (**SW-, HW-Integration, XML Generator**) können auch **parallel** eingesetzt werden.

albin.brandl@efficientnodes.de

5. Revisionssicherheit, Rechtssicherheit und Blockchain

EfficientNodes erstellt gleich zu Beginn der Verarbeitungskette **Signaturen** (# SHA-256) von den neu aufgenommenen Objekten. Signaturen werden, wie die anderen Metadaten auch, innerhalb der Verarbeitungskette (3-2-1 Regel) weiter vererbt.

Verifikationen können auf Objekte in einzelnen Nodes oder auf die ganze Verarbeitungskette erfolgen.

5.1 Verifikationen - Validierung

Es wird eine neue **Signatur** von dem zu untersuchendem Objekt (Datei oder Datencontainer) errechnet und diese mit der oder den **vorhandenen Signaturen verglichen**. Sind die Signaturen gleich, ist bewiesen, dass das Objekt nicht verändert oder durch andere Einflüsse beschädigt wurde.

Damit entsteht ein **sehr hohes Maß an Revisionssicherheit**, da im Falle einer Manipulation Archivobjekte und deren Signaturen in jedem Node penetriert werden müssen.

5.2 Höchste Sicherheit mit Blockchain

Eine **noch höheres Maß an Sicherheit** wird durch einer zusätzlichen Speicherung der Signaturen in der **Blockchain** erreicht. EfficientNodes speichert die darauf vom Blockchain Provider (CryptoWerk®) zurück gesendeten Quittungen (Seals) in den entsprechenden Nodes und Datencontainern.

Die Signaturen werden in **verschiedenen Blockchain gespeichert** und können dort auch nicht verändert werden.

Eine Überprüfung der Daten mit Hilfe der Signaturen und der Seal ist jederzeit innerhalb vom EfficientNodes Management Client (HTTP) möglich.
Es wird hierbei wie weiter oben beschrieben, eine neue Signatur errechnet, und dies dann mit der in der oder den Blockchain(s) gespeicherten verglichen. Sind die Werte **identisch**, sind die **Daten valide**.

5.3 Echtzeit Schutz der Archive - ArchiveWatch

Durch ArchiveWatch ist EfficientNodes in der Lage, die Archive in „Echtzeit“ zu überwachen.
Dies erfolgt über die JetDetect HW-Integration mit den jeweiligen Herstellern.
Dabei werden unerlaubte Änderungen oder Löschungen an den Daten in den Archive sofort angezeigt und auf Wunsch entsprechende Aktionen eingeleitet.

Aktionen können beispielsweise sein:

- Kritische Warnmeldungen Watchdog
- Email Benachrichtigungen
- Abschaltung von Archive (Offline nehmen)
- …

albin.brandl@efficientnodes.de

5.4 Signaturen versus WORM versus DSGVO

Ein WORM System (write once read many) ist heutzutage meist ein Festplatten Speichersystem eines Herstellers. Bei der Speicherung auf diesen Systemen wird eine Aufbewahrungszeit der Daten festgelegt, in der die Daten nicht mehr verändert werden können.

Nachteilig dabei ist, dass der Schutz vor Veränderung nur solange existiert, solange die Daten auf dem jeweiligen System liegen. Wird das System ersetzt, oft nach wenigen Jahren, müssen die Daten aufwendig migriert werden. Dabei **verlieren sie oftmals jeglichen Schutz**.

Dem gegenüber bieten die **Signaturen in EfficientNodes immer Schutz**, da sie auf jedem Node oder auch neu erzeugtem Node mitgeführt werden.
Es ist innerhalb von EfficientNodes keine Migration der Daten notwendig - es werden einfach neue Nodes als Kopien erzeugt und andere entfernt oder still gelegt.
WORM Systeme sind genauso anfällig vor **technischen Ausfällen oder logischen Fehlern** (Firmware) wie andere Speichersysteme.

Deshalb wird dann meist noch ein zweites System vom gleichem Hersteller verwendet, und die Daten dann repliziert.
Dies verletzt jedoch die Regel 2 von der 3-2-1 Regel (2 Kopien auf unterschiedlichen Systemen - technische Trennung). Daraus folgt als **Anforderung ein eigenes Backup** für diese Systeme.

Eine weitere Problematik von **WORM** ist der **Umgang mit der DSGVO**.
Sollen Daten von z.B. Kunden auf deren Aufforderung gelöscht werden, ist dies auf WORM Systemen im Nachhinein nicht möglich.

albin.brandl@efficientnodes.de

6. Datencontainer

EfficientNodes erzeugt meist als erste Kopie Archive, in denen die Daten im Dateisystem gespeichert werden, die Metadaten davon getrennt in XML Dateien.

Vorteil damit ist, dass bei Verlust von Daten in der Datenquelle (N0) die Anwendungen das Archiv Dateisystem (N1) sofort übernehmen können („remapping“). Die Daten im Archiv (N1) werden als eine 1:1 Kopie der Originaldaten gespeichert.

Nachteil damit ist, dass bei den weiteren Prozessen (3-2-1 Regel) sehr viele, meist sehr kleine Dateien und deren Metadaten verarbeitet werden müssen.

Die **Lösung sind Datencontainer**, in denen nach logischen Kriterien, die **Daten, Metadaten, Signaturen und Transaktionen** gespeichert werden.

In diesen Datencontainern sind somit in einem Objekt **sämtliche relevanten Informationen** logisch zusammen gespeichert. Damit einher gehend wird eine **höchste Integrität der Daten** erreicht.

Zusätzlich werden die Datencontainer mit **einer eigenen Signatur** versehen, optional natürlich auch mit **einer Blockchain Seal**. Dies erhöht die **Sicherheit** (Einzelsignaturen der Dateien + Container Signaturen) und **Überprüfbarkeit**.
Ist die Signatur oder das Seal des **Containers valide**, sind auch die einzelnen Objekte innerhalb des Containers valide.
Ist die Signatur oder das Seal des **Containers nicht valide**, können die einzelnen Objekte innerhalb des Containers untersucht werden.

Außerdem führt dies zu einer **Minimierung der Kosten für Blockchain Seal** - z.B. 1 Seal statt 1000 Seal für jede einzelne Datei innerhalb des Containers.

Die Datencontainer können nach Anforderung auch **verschlüsselt (AES-256)** werden.

Beim Transport der Datencontainer (LAN oder WAN) ergeben sich **Verbesserungen der Geschwindigkeit im Bereich 1:1000** - bei deutlich geringerer Systembelastung.

Die Datencontainer können damit natürlich viel **effizienter und auch sicherer im CloudArchive** gespeichert werden.

Beim **Export** aus den Datencontainern entsteht wieder das **originale Quell Dateisystem** - kein Datenbruch mit Verlust der Original Datenpfade.

albin.brandl@efficientnodes.de

7. CloudArchive

Das EfficientNodes **CloudArchive** kann im **eigenen Rechenzentrum** (RZ), in einem **externen RZ** vom oder für einem oder mehrere Kunden betrieben werden.
Es kann als CloudArchive **Appliance**, als **VM**, oder in einem oder mehreren **Containern** unter **Linux (Windows) i**nstalliert werden.

Das **CloudArchive stellt HTTPS Ports** bereit, mit denen andere Nodes kommunizieren. Beispielsweise hat ein Node als Datenquelle ein Dateisystem und als Datenziel das CloudArchive. Genauso kann ein Node als Datenquelle das CloudArchive haben, und als Datenziel ein Dateisystem oder ein anderes CloudArchive.

Vorteil dabei ist, dass **keine Freigaben über SMB oder NFS** benutzt werden, sondern ein HTTPS Gateway. Die Daten sind somit nicht, wie bei Dateifreigaben im Netzwerk sichtbar, somit offline. Ist das CloudArchive in einem anderem Standort, kann man von offsite und offline Speicherung der Daten sprechen (siehe 3-2-1 Regel: 1 Kopie offsite, offline).

Dies ist insbesondere **wichtig**, wenn wie heute meist üblich, **keine Bandsicherung** mehr verwendet wird. Das CloudArchive dient auch als Schutz vor Viren und Ransomware.

Durch die HTTPS Ports, und durch die verschlüsselte Übertragung der Datencontainer bedingt, ist auch **kein VPN Tunnel** mit eigener Verschlüsselung erforderlich.
Durch die **Übertragung relativ großer Datencontainer**, werden WAN und auch LAN Verbindungen sehr effektiv benutzt (Latenz Zeiten, TCP Header, Verlust durch Verschlüsselung).

In einem **klassischem ObjectStorage** werden die Dateien und Metadaten in einem **proprietären, Hersteller spezifischem Format** abgelegt. Die Schnittstelle hierfür ist in der Regel eine **Amazon S3** (Simple Storage Service) kompatibel.

Im EfficientNodes **CloudArchive** werden die **Daten transparent**, in gleicher Art und Weise wie in einem FS Node oder Container Nodes, **geschützt durch ein HTTPS Gateway**, gespeichert.

Hinter einem CloudArchive können weitere Nodes installiert werden:

N 0	=	Datenquelle
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
Quelle		FS Node		Container Node		**CloudArchive**		Container Node		FS Node

albin.brandl@efficientnodes.de