WEB|codeX
INTERNETSERVICES
Schönfließer Straße 4
10439 Berlin

WEB|codeX
INTERNETSERVICES

Ablaufdiagramme

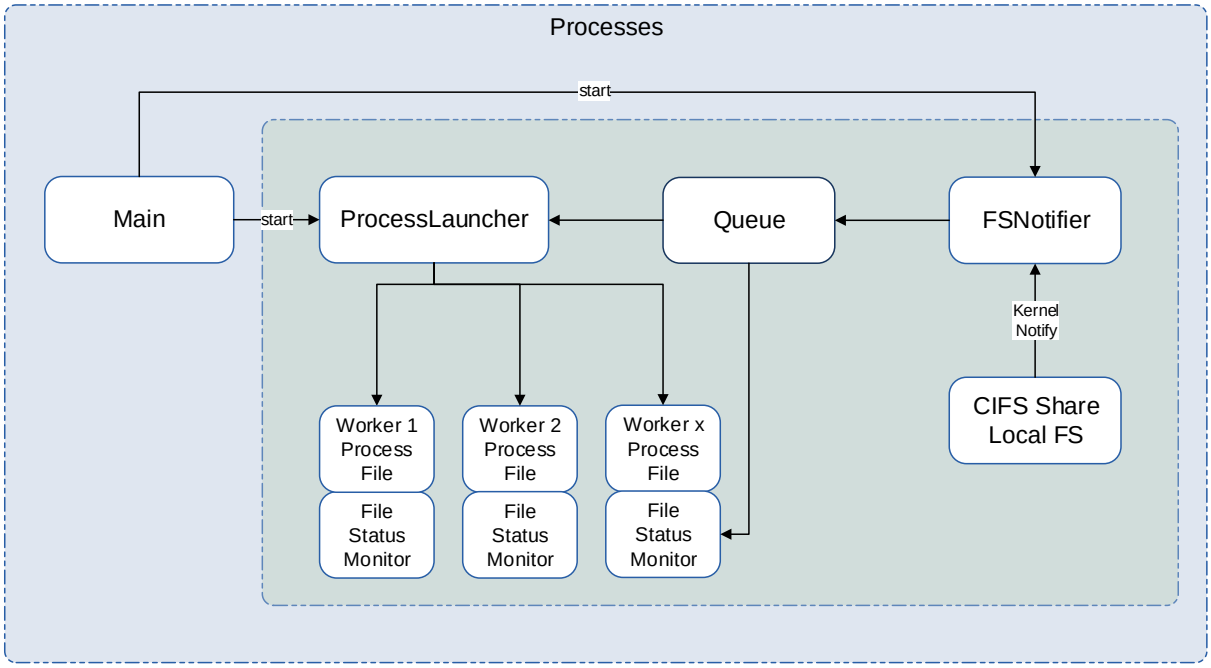
Cloud Communicator - BKP Push, BKP Pull

Das folgende Dokument beschreibt die logischen Abläufe der Programmteile „Backup Push" und „Backup Pull".

1. Backup Push

Nachfolgende werden die Abläufe der Komponente „Backup Push" der Cloud Communication Appliance beschrieben:

1.1 Übersicht



1.2 Ablauf Programmteil Main

Nr		Command
1		Start Program
2		Set Signal Handlers for SIGINT, Keyboard Interrupt
3		Start File System Notifier Thread (Process)
4		Start Process Launcher Thread (Process)
5		Loop until Signal for Shutdown SIGINT received

1.3 Ablauf Programmteil ProcessLauncher

Nr	Command
1	Init Queue Processor Object
2	Init ThreadsRun and ThreadsTime Dictionary
3	Loop 1 Start
4	Get Queue Notify Msgs (Non Blocking) and call Queprocessor.process when Msg exists Get back all Active Threads
5	Loop 2 Start
6	Loop on ThreadsRun Dictionary
7	IF: Thread not in ThreadsRun
8	Add to Dictionary and set ThreadsTime = 0
9	ELSE: Count ThreadsTime +1
10	Loop 2 End
11	Loop 3 Start
12	Loop on ThreadsRun
13	IF: Thread Timeout Reached
14	Terminate Thread and remove Thread from Dictionaries
15	Loop 3 End
16	Sleep (1)
17	Loop 1 End

1.4 Ablauf Programmteil QueueProcessor

Nr	Command
1	IF: Message received (Type, Filename, Timestamp) and Filename not in existing Object Dictionary
2	Start a new File Status Monitor Thread
3	
4	IF: Filename in Object Queue Dictionary
5	send Message to FileStatusMonitor
6	
7	Return Thread Object

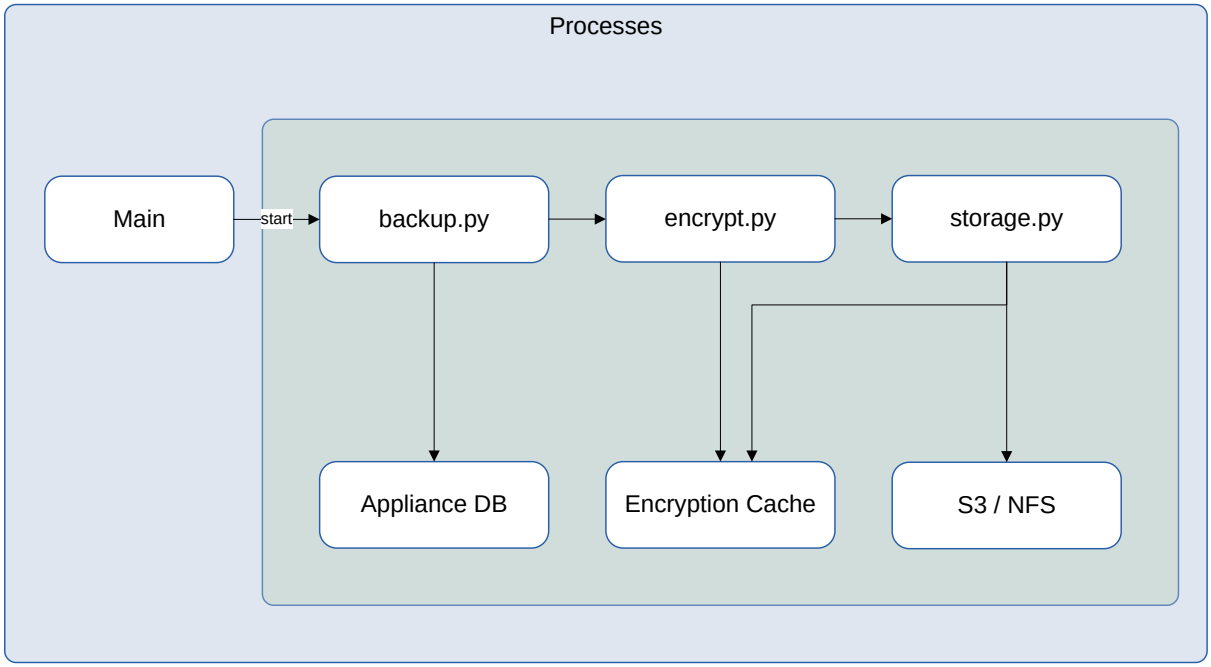
1.5 Ablauf Programmteil FileStatusMonitor

Nr	Command
1	Global Timeout for Loop1 set: 30 Process Launcher Iterations
2	Terminate / Kill Thread
3	
4	Loop 1 Start
5	While FileStatus = 0
6	Wait for FILE_CREATE and CLOSE_WRITE is set Set Timeout 30 Seconds
7	Loop 1 End
8	
9	Set Timeout for Loop2 (No IN_DELETE Msg received in 30 Seconds):
10	Encrypt File and transfer
11	
12	Loop 2 Start
13	IF: IN_DELETE Message received: Indication for File Deletion
14	Do NOT transfer file, exit(0)
15	Loop 2 End

2. Backup Pull

Nachfolgende werden die Abläufe der Komponente „Backup Pull" der Cloud Communication Appliance beschrieben:

2.1 Übersicht



2.2 Ablauf Programmteil Main

Nr		Command
1		Select all active Shares from Appliance DB
2		Loop 1 Start
3		Loop on Shares
4		Call Backup Class with
5		folder_path, smb_id, passphrase
6		Loop 1 End
7		Exit 0

2.3 Ablauf Programmteil InitBackup

Nr	Command
1	Reset Statistics
2	Insert new Transfer into bkp."transfer" Appliance DB Table => get back Inserted TransferID Primary Key Value
3	Calculate File Stats Count Files Count Links Count Dirs ByteSizes
4	Insert into Appliance DB bkp."transfer" Table
5	Init Encryption Object with passphrase Private Key
6	
7	Check for Full / Incremental Backup
8	IF: Full Backup
9	call FullBackup Method
10	
11	IF: IncrementalBackup
12	Check if a Full Backup for the given share exists
13	IF no:
14	Do a FullBackup (call FullBackup Method)
15	IF yes:
16	call IncrementalBackup Method
17	
18	For both Backup Methods:
19	Update BKP Status, write Status to Appliance bkp."transfer" DB Table

2.3 Ablauf Programmteil FullBackup

Nr	Command
1	Get Method Parameter: Path, InsertDBID (Path on first Call: Root of CIFS Share)
2	
3	Loop 1 Start
4	Read in File/Dir List of given Path
5	Get File Statistics, Permissions
6	IF: File Type = "file"
7	Generate SHA512 SUM of File
8	Insert File/Dir into bkp."transfer_file" Appliance DB Table
9	=> Get back FileInsertID
10	
11	IF: FileType = "file":
12	Start DB Transaction
13	Insert File into bkp."remote_file" Appliance DB Table
14	Encrypt File
15	IF: No Error:
16	Commit Transaction
17	IF: Error:
18	Rollback Transaction
19	
20	Update Result Code, Result Message in bkp."transfer_file" Appliance DB Table
21	
22	IF: FileType = "directory"
23	Recursive Method Call with actual Path, FileInsertID

2.4 Ablauf Programmteil IncrementalBackup

Nr	Command
1	Get Method Parameter: Path, InsertDBID (Path on first Call: Root of CIFS Share)
2	
3	Loop 1 Start
4	Read in File/Dir List of given Path
5	Get File Statistics, Permissions
6	IF: File Type = "file"
7	Generate SHA512 SUM of File
8	Insert File/Dir into bkp."transfer_file" Appliance DB Table
9	=> Get back FileInsertID
10	
11	IF: FileType = "file":
12	Start DB Transaction
13	Check if Remote File already exists in bkp."remote_file" Table (SHA512 Compare of File SUMs Filesystem, remote_file DB Column)
14	Insert File into bkp."remote_file" Appliance DB Table (when file exists: remote_file_id will be set)
15	Encrypt File
16	IF: No Error:
17	Commit Transaction
18	IF: Error:
19	Rollback Transaction
20	
21	Update Result Code, Result Message in bkp."transfer_file" Appliance DB Table
22	
23	IF: FileType = "directory"
24	Recursive Method Call with actual Path, FileInsertID

2.5 Ablauf Programmteil GenerateKey

Nr	Command
1	Open File for writing with given Key Parameter
2	Generate Random ASCII Values and write to File

2.6 Ablauf Programmteil EncryptKey

Nr	Command
1	Open Temporary Files tmpPassphrase, tmpEncryptedKey
2	Write Temporary Passphrase File with unencrypted Passphrase
3	Encrypt Key with openSSL (enc_keys dir) with Temporary Passphrase File and write out to tmpEncryptedKey File
4	Move Temporary Encrypted Key to Encryption Dir

2.7 Ablauf Programmteil DecryptKey

Nr	Command
1	Open Temporary Files tmpPassphrase, tmpDecryptedKey
2	Write Temporary Passphrase File with unencrypted Passphrase in
3	Decrypt Key with openSSL (enc_keys dir) with Temporary Passphrase File and write out to tmpDecryptedKey File
4	Return Temporary Decrypted Key Filename

2.8 Ablauf Programmteil EncryptData

Nr	Command
1	Decrypt Key File with given Passphrase (and get temporary File Name back)
2	Encrypt given Filename with openSSL and temporary Decrypted KeyFileName
3	Call Storage Class and Transfer File

2.9 Ablauf Programmteil DecryptData

Nr	Command
1	Decrypt Key File with given Passphrase (and get temporary File Name back)
2	Decrypt given Filename with openSSL and temporary Decrypted KeyFileName