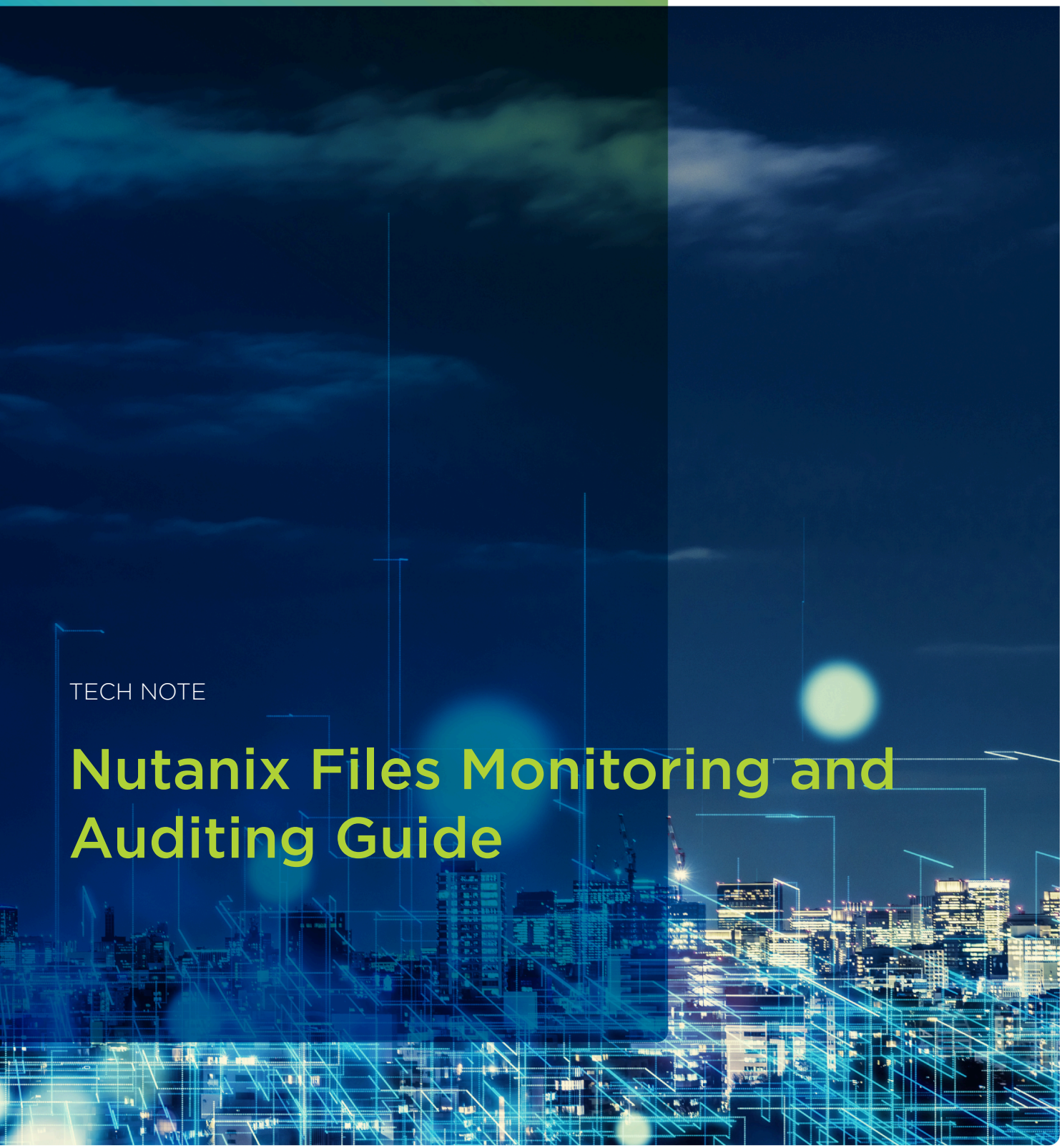
v1.3 | September 2022 | TN-2116



TECH NOTE

Nutanix Files Monitoring and Auditing Guide

NUTANIX
YOUR ENTERPRISE CLOUD

Copyright

Copyright 2022 Nutanix, Inc.

Nutanix, Inc.
1740 Technology Drive, Suite 150
San Jose, CA 95110

All rights reserved. This product is protected by U.S. and international copyright and intellectual property laws. Nutanix and the Nutanix logo are registered trademarks of Nutanix, Inc. in the United States and/or other jurisdictions. All other brand and product names mentioned herein are for identification purposes only and may be trademarks of their respective holders.

Contents

1. Executive Summary

Nutanix Files is a software-defined, scale-out file storage solution that provides a repository for unstructured data, such as home directories, user profiles, departmental shares, application logs, backups, and archives. Flexible and responsive to workload requirements, Files is a fully integrated, core component of Nutanix software.

You can deploy Nutanix Files on an existing cluster or a standalone cluster. Unlike standalone NAS appliances, Files consolidates VM and file storage, eliminating the need to create an infrastructure silo. Administrators can manage Files with Nutanix Prism, just like VM services, which unifies and simplifies management. Integration with Active Directory enables support for quotas and access-based enumeration, as well as self-service restores with the Windows previous version feature. Nutanix Files also supports file server cloning, which lets you back up Files on site, run antivirus scans, and perform machine learning without affecting production.

Nutanix Files can operate as a dedicated cluster or be collocated on a cluster running user VMs. Nutanix supports Files with both VMware ESXi and AHV. Files provides native high availability and uses AOS storage for intracluster data resilience and intercluster asynchronous disaster recovery. AOS storage also offers data efficiency techniques such as erasure coding (EC-X) and deduplication.

Customers often ask us for a way to monitor and search file activity on Nutanix Files so that they can take corrective actions as necessary---for example, when a user has renamed a directory at the top level and the organization's other users can no longer find the path to their files. To meet this need, Nutanix has been working with partners to develop rich functionality such as DFS-R capabilities (providing global read-write functionality across multiple sites), file activity monitoring with analytics, auditing, and so on to integrate with existing file server vendors by registering for notifications from clients about file operations.

© 2022 Nutanix, Inc. All rights reserved

2. Introduction

Purpose

This document covers the current list of Files-related APIs and describes the on-wire format of file notifications that you can configure with Files to empower others to develop solutions on top of Files.

Document Version History

Version Number	Published	Notes
1.0	July 2019	Original publication.
1.1	July 2020	Refreshed content and updated Nutanix overview.
1.2	July 2021	Updated the Notification Policies APIs section.
1.3	September 2022	Refreshed content and updated sample API responses throughout.

Terminology and Concepts

- SMB: Server Message Block protocol.
- NFS: Network File System protocol.
- Google Protobuf: Method for storing and interchanging structured information.
- Prism: UI for managing and monitoring Nutanix clusters.
- Notification policy: Configuration that allows you to select client activity that matches given parameters.

© 2022 Nutanix, Inc. All rights reserved

- Notification engine: Module that transmits notifications to partners.

Requirements

- Ability to enable and disable file notifications at a share level.
- Support for both NFS and SMB protocols.
- Ability to filter specific file operations.
- Ability to send notifications to multiple partner servers.
- Ability to filter activity based on file extensions.
- Ability to filter activity based on user.

Limitations

- No synchronous file notification delivery.
- All configurations must occur at the share level.
- No support for monitoring alternate data streams (ADS).
- No support for sparse or offline attributes.

Workflow

1. In Nutanix Prism, select the file server you need to configure file notifications for. In Prism, create a user that the partner notification server can use to make REST calls directly to Files.
2. With the user created in the step above, partner software can register with the Files instance by specifying the partner server IP, the port, and a description (for example, the name of the partner software).
3. Once registered, the partner server can make a discover REST API call to Files, which returns the list of targets (file shares).
4. The partner software can then make a REST call to Files to create a policy that selects the desired file notifications.

© 2022 Nutanix, Inc. All rights reserved

3. APIs

Partner server APIs allow us to configure and get information about file servers, shares, partner servers, and notification policies. The web client uses HTTPS requests (based on SSL authentication) to communicate with the Nutanix Files file server. The HTTP server runs with its unique self-signed SSL certificate. Authentication is complete when the Files server verifies the identity of the web client through user credentials.

All URLs referenced in this document have the following base, in which <port> is the port number for the Nutanix Aplos service that handles REST calls on the file server:

```
https://<hostname>:<port>/api/nutanix/v3
```

Note: Use 9440 as the port number the Aplos service listens on.

The APIs can return the following responses:

- 200 for Success.
- 400 for Bad Request Error.
- 404 for URL Not Found.
- 500 for Internal Server Error.

The following sections present some of the APIs you can use to register and configure notification policies.

File Server APIs

The Get File Servers API (GET /file_servers) retrieves the Nutanix Files file server information. Use this API to fetch the universally unique identifier (UUID) and the name of the Files file server you're going to manage.

Table: File Servers API Fields

© 2022 Nutanix, Inc. All rights reserved

Field Name	Type	Description	Required or Optional
name	String	Name of the file server.	Optional
uuid	String	Unique identifier for the file server.	Optional

Sample transcript:

```
{
  "metadata": {
    "flags": [...],
    "links": [...],
    "totalAvailableResults": 0,
    "messages": [...],
    "extraInfo": [...]
  },
  "data": {
    "tenantId": "string",
    "extId": "string",
    "links": [...],
    "name": "string",
    "description": "string",
    "memoryGib": 12,
    "vcpus": 4,
    "cvmIpAddresses": [...],
    "containerExtId": "string",
    "clusterExtId": "string",
    "sizeInGib": 0,
    "version": "string",
    "rebalanceNeeded": true,
    "dnsDomainName": "string",
    "compressionEnabled": true,
    "fileBlockingExtensions": [...],
    "vms": [...],
    "dnsServers": [...],
    "ntpServers": [...],
    "internalNetworks": [...],
    "externalNetworks": [...],
    "nvmsCount": 1,
    "externalPrimaryNetworkExtId": "string"
  }
}
```

Mount Target (Share) APIs

The List Mount Targets API (POST /mount_targets/list) retrieves information about all the shares the Nutanix Files file server hosts so the partner server can manage them.

© 2022 Nutanix, Inc. All rights reserved

The Get Mount Targets API (GET /mount_targets/{uuid}) retrieves information about the specified share UUID.

Table: Mount Targets API Fields

Field Name	Type	Description	Required or Optional
name	String	Name of the mount target.	Required
protocol_type_list	List of strings	List of allowed protocols for this share. Examples: ["SMB"] for SMB-only access share, ["NFS"] for NFS-only access export, ["SMB", "NFS"] for both SMB and NFS access.	Optional
uuid	String	Unique identifier for the mount target.	Optional

Sample transcript:


```
{
  "metadata": {
    "flags": [...],
    "links": [...],
    "totalAvailableResults": 0,
    "messages": [...],
    "extraInfo": [...]
  },
  "data": [
    {
      "tenantId": "string",
      "extId": "string",
      "links": [...],
      "name": "string",
      "description": "string",
      "fileServerExtId": "string",
      "maxSizeGib": 0,
      "type": "$UNKNOWN",
      "path": "string",
      "parentMountTargetExtId": "string",
      "enableCompression": true,
      "isEnableCompression": true,
      "fileBlockingExtensions": [...],
      "protocol": "$UNKNOWN",
      "secondaryProtocol": [...],
      "enablePreviousVersion": false,
```


© 2022 Nutanix, Inc. All rights reserved

```
        "isEnablePreviousVersion": false,
        "smbProperties": {...},
        "nfsProperties": {...},
        "multiProtocolProperties": {...},
        "blockedClients": {...},
        "statusType": "$UNKNOWN",
        "state": "$UNKNOWN",
        "longnameEnabled": false,
        "isLongnameEnabled": false,
        "wormSpec": {...},
        "workloadType": "$UNKNOWN"
      }
    ]
}
```


Partner Server APIs

The partner server APIs register the partner server to receive notifications for specified file operations. We return the UUID to uniquely identify this server.

The Post Partner Servers operation (POST /partner_servers) submits a request to create a partner server based on the input parameters.

The List Partner Servers API (POST /partner_servers/list) lists all partner servers registered on the Nutanix Files file server. We return the connection status to verify whether the registration resulted in a successful connection to the server.

The Delete Partner Servers API (DELETE /partner_servers/{uuid}) removes the partner servers registered on the Nutanix Files file server. This API prevents the partner server from receiving further notifications.

The Get Partner Servers API (GET /partner_servers/{uuid}) retrieves the partner server information registered on the Nutanix Files file server for the given UUID.

The Put Partner Servers operation (PUT /partner_servers/{uuid}) submits a request to update a partner server based on the input parameters.

Table: Partner Servers API Fields

Field Name	Type	Description	Required or Optional
name	String	Name of the partner server.	Required
description	String	Briefly describes the partner server.	Optional

© 2022 Nutanix, Inc. All rights reserved

Field Name	Type	Description	Required or Optional
usage_type	String	["NOTIFICATION"]. In use cases not described in this document, the usage type may vary.	Required
vendor_name	String	["syslog"] for external syslog support. ["peer"] for proto-based support. ["kafka"] or ["varonis"] for Kafka or Varonis pipeline implementation.	Required
server_type	String	["PRIMARY"]. In a future release, Nutanix plans to support active-passive HA for partner server implementation.	Required
uuid	String	Unique identifier for the partner server.	Optional
health_status	String	Indicates the health status of the partner server.	Optional
ip	String	The IP address of the partner server.	Required (one of the four)
ipv4	String	The IP address of the partner server in IPv4 format.	Required (one of the four)
ipv6	String	The IP address of the partner server in IPv6 format.	Required (one of the four)

© 2022 Nutanix, Inc. All rights reserved

Field Name	Type	Description	Required or Optional
port	Integer	The port the partner server needs to send notifications to.	Required (one of the four)
partner_custom_properties	String	Files allows you to configure some custom properties based on third-party vendor. (Using this field isn't common.)	Optional

Sample request transcript:

```
{
  "spec": {
    "name": "string",
    "resources": {
      "usage_type": "string",
      "vendor_name": "string",
      "server_info": {}
    },
    "description": "string"
  },
  "api_version": "string",
  "metadata": {
    "last_update_time": "2017-08-29T04:24:23.512Z",
    "kind": "partner_server",
    "uuid": "string",
    "creation_time": "2017-08-29T04:24:23.512Z",
    "spec_version": 0,
    "owner_reference": {
      "kind": "user",
      "name": "string",
      "uuid": "string"
    },
    "categories": {},
    "name": "string"
  }
}
```

Sample response:

```
{
     "status": {
         "state": "string",
         "message_list": [
         {
             "message": "string",
             "reason": "string",
             "details": {}
         }
```

© 2022 Nutanix, Inc. All rights reserved

```
            ],
            "name": "string",
            "resources": {
                  "usage_type": "string",
                  "vendor_name": "string",
                  "server_list": [
                  {
                        "health_status": "string",
                        "server_type": "string",
                        "address": {
                              "ip": "string",
                              "ipv6": "string",
                              "port": 0,
                              "fqdn": "string"
                        }
                  }
                  ]
            },
            "description": "string"
      },
      "spec": {
            "name": "string",
            "resources": {
                  "usage_type": "string",
                  "vendor_name": "string",
                  "server_list": [
                     {}
                  ]
            },
            "description": "string"
      },
      "api_version": "string",
      "metadata": {
            "last_update_time": "2017-07-19T23:03:15.510Z",
            "kind": "partner_server",
            "uuid": "string",
            "creation_time": "2017-07-19T23:03:15.510Z",
            "spec_version": 0,
            "owner_reference": {
                  "kind": "user",
                  "name": "string",
                  "uuid": "string"
            },
            "categories": {},
            "name": "string"
      }
}
```


Notification Policies APIs

The Post Notification Policies API (POST /notification_policies) creates the notification policy to be applied for the given policy server.

The List Notification Policies API (POST /notification_policies/list) lists the notification policies that exist on the Nutanix Files file server.

© 2022 Nutanix, Inc. All rights reserved

The Delete Notification Policies API (DELETE /notification_policies/{uuid}) deletes the notification policy with the given UUID on the Nutanix Files file server.

The Get Notification Policies API (GET /notification_policies/{uuid}) retrieves the notification policy of the given UUID on the Nutanix Files file server.

The Put Notification Policies API (PUT /notification_policies/{uuid}) updates the notification policy of the given UUID on the Nutanix Files file server.

Table: Notification Policies API Fields

Field Name	Type	Description	Required or Optional
name	String	Name of the notification policy.	Required
description	String	Briefly describes the notification policy.	Optional
mount_target_list	List of strings	List of mount target UUIDs you need to enable notifications for.	Optional
all_mount_targets	Boolean	Determines whether the policy takes effect for all shares.	Optional
black_list: user	List of strings	SID, UID, or username.	Optional
black_list: client_ip, ipv6, ip_range, ip	List of strings	Single IPv6 address, range of IPv4 addresses (with subnet mask), or single IPv4 address. You can block multiple client IPs.	Optional
file_operation_list	List of strings	See the following list of file operations.	Required

© 2022 Nutanix, Inc. All rights reserved

Field Name	Type	Description	Required or Optional
partner_server_reference_list	List of strings	List of partner server UUIDs that should receive the events. With multiple partner servers, events are load balanced.	Required
protocol_type_list	List of strings	List of protocols that need to have notifications enabled. For example, for a share using both SMB and NFS access, we could specify SMB, NFS, or both.	Required
uuid	String	Unique identifier for the notification policy.	Optional
is_secure	Boolean	Set to false. Nutanix does not currently support secure messages.	Optional

The `file_operation_list` object can contain the following strings:

- "FILE_CREATE" to receive file create operations.
- "FILE_DELETE" to receive file delete operations.
- "FILE_READ" to receive file read operations.
- "FILE_WRITE" to receive file write operations.
- "FILE_OPEN" to receive file open operations with write access.
- "FILE_CLOSE" to receive file close operations (corresponding to the open files).

© 2022 Nutanix, Inc. All rights reserved

- "DIRECTORY_CREATE" to receive the directory create operation.
- "DIRECTORY_DELETE" to receive the directory delete operation.
- "SETATTR" to receive set attribute operations.
- "RENAME" to receive the rename operation.
- "SYMLINK_CREATE" to receive symlink operations (for NFS only).
- "LINK_CREATE" to receive hard link operations (for NFS only).
- "SECURITY" to receive DACL changes (for SMB only).

Sample request transcript:

```
{
  "spec": {
    "name": "string",
    "resources": {},
    "description": "string"
  },
  "api_version": "string",
  "metadata": {
    "last_update_time": "2017-08-29T04:24:23.505Z",
    "kind": "notification_policy",
    "uuid": "string",
    "creation_time": "2017-08-29T04:24:23.505Z",
    "spec_version": 0,
    "owner_reference": {
      "kind": "user",
      "name": "string",
      "uuid": "string"
    },
    "categories": {},
    "name": "string"
  }
}
```

Sample response:

```
{
  "status": {
    "state": "string",
    "message_list": [
      {
        "message": "string",
        "reason": "string",
        "details": {}
      }
    ],
    "name": "string",
    "resources": {
      "protocol_type_list": [
        "string"
      ],
      "file_extension_list": [
```

© 2022 Nutanix, Inc. All rights reserved

```
        {
          "file_type": "string"
        }
      ],
      "mount_target_reference_list": [
        {
          "kind": "mount_target",
          "name": "string",
          "uuid": "string"
        }
      ],
      "partner_server_reference_list": [
        {
          "kind": "partner_server",
          "name": "string",
          "uuid": "string"
        }
      ],
      "file_operation_list": [
        "string"
      ],
      "is_secure": false,
      “all_mount_targets”: true,
      "file_extension_list_mode": "string"
    },
    "description": "string"
  },
  "spec": {
    "name": "string",
    "resources": {},
    "description": "string"
  },
  "api_version": "string",
  "metadata": {
    "last_update_time": "2017-08-29T04:24:24.134Z",
    "kind": "notification_policy",
    "uuid": "string",
    "creation_time": "2017-08-29T04:24:24.134Z",
    "spec_version": 0,
    "owner_reference": {
      "kind": "user",
      "name": "string",
      "uuid": "string"
    },
    "categories": {},
    "name": "string"
  }
}
```


© 2022 Nutanix, Inc. All rights reserved

4. Message Format

The following sections propose a format for sending information describing the notification events to the policy server. Like XML, Google Protobuf messages are easily extensible and compatible both forward and backward. Each notification contains a fixed-length header followed by the variable-length payload that describes the file event.

Notification Fields

```
message NotificationEvent {

   // File Access Event.
   optional AccessType opcode = 1;
   // unc_path.
  optional string unc_path = 2;

   // renamed path.
   optional string rename_path = 3;

   // path is file or folder.
   optional PathType path_type = 4;

   // user sid
   optional string sid = 5;

   // user uid
   optional uint64 uid = 6;

   // user gid
   optional uint64 gid = 7;

   // user name.
   optional string username = 8;

   // domain information
   optional string domain = 9;

   // time-stamp at which the event happend.
   optional uint64 timestamp_msec = 10;

   // client ip address.
   optional string ip_address = 11;

   // share access mask.
   optional uint32 share_access = 12;

   // access mask.
   optional uint32 access_mask = 13;
```

© 2022 Nutanix, Inc. All rights reserved

```
    // system error.
    optional int32 sys_errno = 14;

    // create options.
    optional uint32 create_options = 15;

    // file open handle.
    optional int64 file_handle = 16;

    // system error.
    optional int32 sys_errno = 17;

    // rpc_id
    optional int64 message_id = 18;

    // link path
    optional string link_path = 19;

    // previous owner_name
    optional string prev_owner_name = 20;

    // new owner_name
    optional string new_owner_name = 21;

    // previous owner sid
    optional string prev_owner_sid = 22;

    // new owner sid
    optional string new_owner_sid = 23;
}

enum PathType {
    // undefined path.
    kUnknownPath = 0;

    // path is folder.
    kDirectoryPath = 1;

    // path is file.
    kFilePath = 2;
}

// Description of File Access Types
enum AccessType {
    /* file open with write access */
    kFileOpen = 0x1;

    /* file close after modification */
    kFileClose = 0x2;

    /* file is read first time */
    kFileRead = 0x4;

    /* file is written first time */
    kFileWrite = 0x8;

    /* file is created */
    kFileCreate = 0x10;

    /* file is deleted */
    kFileDelete = 0x20;
```

© 2022 Nutanix, Inc. All rights reserved

```
    /* file or directory is renamed */
    kRename = 0x40;

    /* directory is created */
    kDirectoryCreate = 0x80;

    /* directory is deleted */
    kDirectoryDelete = 0x100;

    /* security info change */
    kSecurityInfo = 0x200;

    /* setattr info changes */
    kSetAttr = 0x400;

    /* symlink info changes */
    kSymlink = 0x800;

    /* link (hard) changes */
    kLink = 0x1000;

    /* health event */
    kHealth = 0x10000;

    /* No Op */
    kNoop = 0xFFFFF;
}
```

© 2022 Nutanix, Inc. All rights reserved

5. Additional API Configuration Options

Performance

Enabling file notifications has some performance impact on Nutanix Files client latencies. For optimal performance, run partner server software as a virtualized solution in the AOS cluster where the file server VMs (FSVMs) are deployed and make them part of the same subnet.

Nutanix Files sends file notifications asynchronously to the notification engine module from our protocol layer and queues them in memory or on disk to be sent asynchronously to the partner servers.

When the partner server can't keep up with notification frequency and the queue is completely full, you can throttle incoming notifications using the enqueue operation. Each notification is enqueued in the order that file events occur and receives an rpc_id as part of the RPC header. The multithreaded notification engine module dequeues these events and sends them simultaneously to the notification server. Because the server receives the notification events at the same time, the system must use the rpc_id field to reserialize them on the partner server implementation.

When the enqueue request encounters long wait times, Nutanix Files alerts the file server administrator to take corrective action.

Reliability

Because notification events are stored either in memory or on disk, notification engine or Files node failures result in the loss of events that haven't been sent to the partner server. However, during intermittent connection issues with the partner server, Nutanix Files keeps file events in memory or on disk up to the queue length. Files then retries sending them when the connection resumes, starting from the last persisting read_index.

© 2022 Nutanix, Inc. All rights reserved

Debuggability

A health check runs periodically to assess connectivity with the partner server. If there are connection issues, the system alerts the file server administrator to debug any issues.

As an alternative method for determining connection status, the partner server can issue a GET partner server API request. When connection issues occur, the partner server software can then alert the administrator that events may be lost.

© 2022 Nutanix, Inc. All rights reserved

6. Appendix

References

1. Nutanix Files
2. Nutanix Files: Sizing Guide
3. Nutanix Files Migration Guide

© 2022 Nutanix, Inc. All rights reserved

About Nutanix

Nutanix is a global leader in cloud software and a pioneer in hyperconverged infrastructure solutions, making clouds invisible and freeing customers to focus on their business outcomes. Organizations around the world use Nutanix software to leverage a single platform to manage any app at any location for their hybrid multicloud environments. Learn more at www.nutanix.com or follow us on Twitter @nutanix.

© 2022 Nutanix, Inc. All rights reserved