Fact Sheet (updated 23 August 2016)

WHY THE AURISTOR FILE SYSTEM IS RIGHT FOR YOU

When it comes to unstructured data management organizations are facing broad challenges. End user work flows require increasing access to information from an ever increasing range of devices and geographic locations. At the same time organizations must protect their intellectual property, the personal identifying information of their community members, and other restricted data from both active attacks and unintentional exposure. Data classification and compartmentalization is the industry best practice but the deployment of competing overlapping systems with differing access capabilities and security properties leads to data duplication, increased complexity and additional challenges to policy enforcement.

Thirty years ago researchers from IBM and Carnegie Mellon University's Andrew Project had the foresight to build the foundation of a solution, AFS©[1] (or the /afs file namespace). /afs, the world's first cloud computing platform, was so far ahead of its time there were few organizations that could benefit from it. Unlike the World Wide Web that followed, the /afs vision took a security first approach to network communications combining single sign-on, wire privacy, strong mutual authentication, fine grained access control and audit capabilities to permit organizations to protect their information on open networks. /afs combined security with location independence and replication providing high availability to data stored on file storage servers distributed either for redundancy or geographic locality. By relying upon the standard file system programming interface, application use of /afs is portable to all computing form factors and programming languages.

The AuriStor File System is the next generation /afs file namespace, fulfilling the full potential of the AFS vision and meeting the tomorrow's challenges today. AuriStor combines the best ideas of the Andrew File System with today's security best practices to produce a secure high-performance storage solution that will satisfy the requirements of the next decade. The AuriStor File System permits organizations to deploy one file system namespace that can safely and securely cross all institutional boundaries: multiple internal data centers, the demilitarized zone, and hosted cloud services.

---

[1] AFS is a registered mark of IBM

AURISTOR®

255 West 94th Street #6B
New York, New York
10025-6985 USA
+1-212-769-9018
sales@auristor.com
https://www.auristor.com/filesystem/

Organizations with /afs deployments can migrate to AuriStorFS easily without loss of data, without flag days and at a reasonable cost.

AURISTOR AND THE /AFS VISION

AuriStor is a secure, high performance, scalable storage solution that fulfills the /afs vision:

- Location transparency:
  - one uniform global file system name space
  - data can be stored in a single data center or distributed across any mix of private and public data centers including Amazon Web Services and Microsoft Azure
- User Mobility:
  - location independent data storage accessible from any networked device
- Security:
  - Federated Mutual Authentication:
    - GSS-API and Microsoft SSPI: Kerberos v5
    - Combined identity authentication
  - Wire Privacy
    - AES-256 encryption
  - Multi-layer authorization model
    - Per-object access control lists (user modifiable)
    - Per-volume access control lists (system administrator modifiable)
    - Multi-factor access control entries grant separate permissions to
      - Anonymous
      - User
      - Machine
      - Anonymous@Machine
      - User@Machine
  - File Servers enforce policies
    - Wire privacy requirements
    - Integrity protection requirements
    - Authentication requirements
    - Volume hosting restrictions prevent serving data without matching
      - Volume Security Policies
      - Volume Labeling
  - Volume Policies specify administrator defined data classifications
    - Security Policies
    - Labeling
    - Access Control Lists
  - Hierarchical group management
  - Structured audit data stream
    - Records every remote procedure call
    - Authentication name sequence

        - Incoming network endpoint (address and port)
        - OpCode, Parameters and Result
- High Availability:
    - maintenance without data access outages
    - data can be migrated and replicated without data access outages
- Integrity:
    - end users can trust the system with the one and only copy of their data
- Heterogeneity:
    - Broad client platform support including Windows, Linux, OSX, iOS, Solaris, and others
    - Support for a heterogeneous mix of server platforms
        - Red Hat Enterprise Linux, Debian Linux, Fedora Core Linux, Ubuntu Linux, Solaris, OSX
        - Back-end storage devices including DASD, SAN, NAS and Cloud
        - Processor architectures: X64, X86, Sparc, PPC, Power Series 7/8
- Self service capabilities:
    - End users can restore from online backups
    - End user defined authorization groups and access control policies
- Atomic Publishing:
    - Perfect for publishing web sites, software deployment and system configuration
- Real time Collaboration:
    - Distributed file locking permits shared data access
- Scalability:
    - Ready for multi-producer / multi-consumer workflows involving tens of thousands of nodes
    - Each cell can store up to 2^{159} file streams of up to 16 exabytes.
- Distributed Administration
    - Secure remote administration of all data and services
- 100% backward compatibility with IBM® AFS® and OpenAFS™[2] clients and servers

AuriStor extends the proven /afs vision into the 21st century.

THE RELATIONSHIP BETWEEN AURISTOR FILE SYSTEM AND AFS

AuriStorFS inherits the best features and capabilities of the /afs model and addresses its most significant weaknesses and limitations including performance, stability, reliability, and privacy issues.

[2] OpenAFS is a registered mark of IBM

AURISTOR
YOUR DATA · YOUR FILE SYSTEM

The following table provides a baseline feature comparison.

Feature	AuriStor 1.0	OpenAFS 1.6
Year 2038 Safe	Yes	No
Timestamp Granularity	100ns (UNIX Epoch)	1s (UNIX Epoch)
Rx Throughput	> 8.2 gbit/second[3]	< 2.4 gbits/second
Rx Window Size	255 packets / 354KB	32 packets / 44KB
Volume IDs per Cell	2^{64}	2^{31}
Object IDs per Volume	2^{95} directories and 2^{95} files	2^{30} directories and 2^{30} files
Objects per Volume	2^{90} directories or files	2^{26} directories or files
Objects per Directory[4]	Up to 2,029,072	up to 64,447
Maximum Distributed DB Size	16 exabytes (2^{64} bytes)	2 gigabytes (2^{31} bytes)
Access Control Lists (ACL)	Per Object	Per Directory
Access Control Entries (ACE) per ACL	Unlimited	20
Directory ACL Inheritance	Yes	No
Volume Access Control Policies	Yes	No
Mandatory Locking	Yes	No
GSS Authentication (RxGK)	Yes	No
AES-256/SHA-1 Wire Privacy	Yes	No
AES-256/SHA-2 Wire Privacy[5]	Awaiting IETF Standardization	No
Mandatory Security Levels[6]	Yes	No
Cache Poisoning Attack Protection[7]	YFS-RxGK: Yes; RxKad: No	No
Combined Identity Authentication	Yes (user@machine)	No
Perfect Forward Secrecy	YFS-RxGK: Yes; RxKad: No	No
Default Volume Quota	20 GB	5000 KB
Maximum Assignable Quota	16 zettabytes (2^{74} bytes)	2 terabytes (2^{41} bytes)
Maximum Reported Volume Size	16 zettabytes (2^{74} bytes)	2 terabytes (2^{41} bytes)
Maximum Volume Size	16 zettabytes (2^{74} bytes)	16 zettabytes (2^{74} bytes)
Maximum Partition Size	16 zettabytes (2^{74} bytes)	16 zettabytes (2^{74} bytes)
Servers run as “root”	No	Yes
POSIX O_DIRECT support	Yes	No
iOS Support	Yes	No
IBM AFS 3.6 client support	Yes (AFS protocol only)	Yes
OpenAFS 1.x client support	Yes (AFS protocol only)	Yes
AuriStor 1.0 client support	Yes	Yes (AFS protocol only)
IBM AFS 3.6 DB server support	Yes (AFS protocol only)	Yes

[3] Per Rx Listener thread: on Red Hat Enterprise Linux 7 (or above) AuriStorFS can support one Rx Listener thread per CPU Core permitting saturation of multiple bonded 10gbit network interfaces.
[4] File names longer than 16 bytes require more than one directory entry.
[5] https://datatracker.ietf.org/doc/draft-ietf-kitten-aes-cts-hmac-sha2/
[6] A Security Level is defined as a Rx Security class (rxkad or rxgk) combined with cryptographic requirements for data privacy and integrity protection. Security Levels are enforced by the File Server.
[7] AFS Clients are susceptible to cache poisoning attacks because the AFS token session key used for authenticating the file server is visible to the end user. It is therefore possible for the end user to spoof the file server to the AFS cache manager without detection.

Feature	AuriStor 1.0	OpenAFS 1.6
OpenAFS 1.x server support	Yes (AFS protocol only)	Yes
AuriStor 1.0 server support	Yes	Yes (AFS protocol only)
DB Servers support AuriStor file servers	Yes	Yes (AFS protocol only)
DB Servers support AFS file servers	Yes	Yes
Thread safe libraries	Yes	No
File Lock State Callback Notification	Yes	No
Atomic Create File in Locked State	Yes	No
Valid AFS3 Volume Status Info Replies	Yes	No
Server Thread Limits	Up to OS capability	256
Dynamic Thread Pools	Yes	No
File Server Meltdowns[8]	No	Yes
IPv6 capable	Yes	No
Microsoft Direct Access compatible	Yes	No
Kerberos Profile based configuration	Yes	No

Upgrading from OpenAFS to AuriStor can be done incrementally. AuriStor clients and servers integrate two file system protocols: AuriStor and AFS3. The AFS3 protocol stack is backward compatible with IBM AFS 3.6 and all OpenAFS releases. The AuriStor protocol stack provides enhanced functionality, performance optimizations and future extensibility. AuriStor caching algorithms are more efficient and reduce overall network traffic.

Although many capabilities of prior /afs implementations have exceeded those of other file systems such as NFS3, NFS4, CIFS, and Lustre, they were held back by an inability to extend the protocol capabilities, decades of accumulated technical debt, and a lack of investment.

As a result, many use cases were deemed off limits. For example, mail delivery services[9] or any grid compute jobs that required simultaneous read/write access from hundreds of compute nodes.

AURISTOR MULTI LAYER SECURITY

One of the AuriStor File System's strengths is that all network communication provides for:

1. Federated mutual authentication
2. Wire privacy and integrity protection
3. Access control
4. Audit logging

---

[8] OpenAFS File Servers are limited to processing slightly more than one remote procedure call at a time regardless of the number of configured worker threads. A simple test using two UNIX client machines can demonstrate the adverse side effects. On client A copy a file that is large enough to require several minutes to complete to a directory. On client B perform "ls –l" (directory listing with status information) of the directory to which the file is being copied. The "ls –l" will not complete the directory listing until client A's copy completes. When the number of clients accessing the target directory is larger than the number of worker threads the file server is unable to respond to any client request until the copy completes. This is one example of a file server meltdown scenario.

[9] http://wiki.openafs.org/AdminFAQ/#mail

AURISTOR
YOUR DATA · YOUR FILE SYSTEM

AuriStorFS enhances the security of the /afs name space by introducing

1. a new wire privacy and authentication framework
2. authentication and authorization for managed devices
3. combined identity authentication
    a. user@machine
4. multi-factor authorization
    a. anonymous
    b. user
    c. machine
    d. anonymous@machine
    e. user@machine
5. volume based wire security policies
6. volume based maximum access lists
7. volume based labeling
8. file server enforced wire security policies
9. file server labeling
10. need to know metadata exposure

These capabilities are combined to improve end user experience by reducing firewall restrictions and virtual private network dependencies while at the same time improving security at multiple layers. Data classification and security requirements are expressed in the file namespace as metadata and enforced by the key management service and the file servers. It is no longer necessary to harden the outer walls and dig tunnels to secure access while leaving the data vulnerable to internal threats. AuriStor provides the capabilities necessary to enforce data access policies while simultaneously permitting end users the self service capabilities they require to accomplish their day to day tasks

THE YFS-RXGK SECURITY CLASS

AFS and AuriStorFS use the Rx remote procedure call protocol to exchange data between clients and servers, and between servers. With Rx, authentication and wire privacy is provided by a negotiable security class. The RxKAD security class developed at Carnegie Mellon University in the 1980s, used to this day by AFS, provides weak security guarantees. It relies upon raw Kerberos service tickets and the non-standard “fcrypt” encryption algorithm, a dumbed down version of the deprecated U.S. Data Encryption Standard (DES) algorithm.

Whereas AuriStor incorporates a flexible Rx security class that is built from Internet Engineering Task Force (IETF) standard components:

- GSS-API Version 2, Update 1 authentication framework (RFC 2743)
- GSS-API Pseudo Random Function extension (RFC 4401)
- Kerberos Version 5 GSS-API Mechanism: Version 2 (RFC 4121)
- Kerberos v5 Cryptographic framework (RFC 3961)
- Advanced Encryption Standard (AES) Encryption for Kerberos 5 (RFC 3962)

The security class is named YFS-RxGK for "Your File System[10]'s Rx GSS-API/Kerberos Crypto". The initial deployment of YFS-RxGK in AuriStor 1.0 supports only Kerberos v5 authentication but as a GSS-API based mechanism it can be extended to support other GSS-API mechanisms that implement the PRF extension. The Kerberos Cryptographic framework currently supports AES256-SHA1, AES128-SHA1, 3DES-CBC-CRC, and RC4-HMAC. When the IETF standardizes new encryption types, such as AES-CTS-HMAC-SHA2[11], those algorithms will become available within future releases of the AuriStor File System.

YFS-RxGK is implements "combined identity authentication". When a user communicates with a network service from a keyed machine, the YFS-RxGK authentication asserts both the user's identity and the machine's identity. This combined identity is logged to the audit stream and can be used to grant permissions.

AUTHORIZATION IMPROVEMENTS

The AuriStor Protection Service manages the assignment of AuriStor user, machine and network identifiers, the assignment of group identifiers and manages hierarchical group memberships. These identifiers are used to construct access control entries which can be applied to file system objects (directories, files, symlinks, mount points) and volumes. AuriStor supports all of the fine-grained access control flexibility provided by the AFS administration model. Users are permitted to define their own groups and group memberships. Users can assign those groups to access control lists within volumes they own or hold administrative permissions.

ACCESS CONTROL LISTS

AuriStor supports all of the fine-grained access control flexibility provided by the AFS volume administration model. Users are permitted to define their own groups and group memberships. Users can assign those groups to access control lists within volumes they own or are granted administrative privileges. End users (and administrators) often make mistakes and grant the groups *system:anyuser* or *system:authuser* access to directories and volumes which they shouldn't.

IT Security staff often complain that AFS is insecure because this self service capability has resulted in unwanted data exposures. AuriStor addresses this concern with the Volume Maximum Access List. This ACL can be set only by a system administrator and applies to the entire volume. Any access to an object in the volume must pass both the object's access check as well as the volume's access check.

AuriStor ACLs can be applied to files, symlinks and mount points in addition to directories. Cross-directory hard links are supported.

---

[10] Your File System, Inc. is the former name of AuriStor, Inc.
[11] "AES Encryption with HMAC-SHA2 for Kerberos 5", https://datatracker.ietf.org/doc/draft-ietf-kitten-aes-ctshmac-sha2/, will be available as soon as standardization is complete. A compliant implementation was contracted by AuriStor, Inc. and contributed to Heimdal Kerberos, https://github.com/heimdal/heimdal/tree/lukeh/aes-cts-hmac-sha2.

When the YFS-RxGK security class is used, AuriStor authenticates the combined identity of the user and the client machine's cache manager (aka the machine). In most file systems an access control entry (ACE) provides a set of permissions for a single authenticated entity, a user, a machine, or a network interface. AuriStor classifies Protection Service Entities as one of *user*, *machine*, or *network*. AuriStor uses these classifications to evaluate a complex ACE[12] which grants a set of permissions to a protection entity set

For example, here is an access control list that provides the user John the ability to *read* and *lookup* from all machines but when John is accessing the file system from *host.your-cell-name.com* he is also granted the *insert*, *delete*, *write*, *lock*, and *admin* rights.

John	rl
Jonn, host.your-cell-name.com	rlkidwa

This approach will be extended in the future to permit IP Security network identities such that not only must the user access from a particular device but the IP Security protected address of the connection must be viewed as coming from a particular address or subnet.

PROTECTION ENTRY MANAGEMENT

The AuriStor Protection Service enhances the management of Protection entries. It is a good security practice that organizations do not reuse authorization identifiers because doing so risks the accidental grant of unwanted data access to new users of the entity identifier. At the same time organizations wish to be able to remove access to their data. AuriStor supports both of these goals by adding a "disabled" attribute that can be applied to entities in the protection database. This ensures that the entity name and identifier cannot be reused while guaranteeing that the entity cannot obtain better than "anonymous" privileges to the file system data.

In order to support AuriStor cells (those running AuriStor database services) with OpenAFS file servers, the AuriStor Protection entities also support an "afs only" attribute. An entity with the "afs only" attribute applied will be treated as "anonymous" by AuriStor File Servers.

SECURITY POLICIES

In AFS, the use of authentication, integrity protection and encryption is left up to the client configuration. As a result there is little that can be done to hide user data from eavesdroppers. AuriStor provides a flexible solution. First, AuriStor servers can be configured to only accept Rx connections established using explicit security criteria. Second, when RxGK is used to secure a file server connection the file server specific token obtained by the client states the acceptable security policy. Finally, volumes can be marked with a security policy indicating what the Rx connection properties are permitted when accessing data in the volume. Volumes can only be attached by or copied to file servers with security policies at least as strong as that required by the volume policy.

FILE SERVER LABELS

[12] Complex ACEs will be present in AuriStor 1.1.

Security policies are special case of file server labeling. AuriStor permits file servers to be assigned arbitrary label categories and values. This will permit organizations to define labels that can be assigned to file servers and volumes that will restrict the file servers a volume can be moved or replicated to.

TOKEN MANAGEMENT AND PERFECT FORWARD SECRECY

AuriStor RxGK authentication token management is more sophisticated than AFS. AFS permits one token per cell which contains a 56-bit session key used to authenticate a user entity to all servers within the cell. This session key is known to the user entity and the cache manager which opens the door to server spoofing and cache poisoning attacks. AuriStor implements managed machine identities and keyed cache managers (clients). AuriStor RxGK tokens store combined identities representing a user entity on a specific managed or unmanaged cache manager. The session keys are not known to the user entity which prevents the possibility of server spoofing. Finally, each file server contacted by a cache manager requires its own RxGK token and each new Rx connection has a unique session key. This prevents a compromised host from being used to compromise connections to other servers within the same cell.

MACHINE IDENTIFICATION

Managed machine identities are first class IDs in the AuriStor Protection Service. This permits secure machine groups to be defined and assigned to ACLs that do not rely upon IP addresses. The immediate gain is the ability to obtain RxGK security benefits for connections established by managed devices.

NEED TO KNOW ACCESS TO METADATA

AFS and AuriStor cells maintain and communicate a large quantity of metadata that contain identifiers that can leak information to passive listeners and active attackers. The metadata includes names of users, groups, and volumes; group memberships; volume access patterns; file identifiers and service statistics. AuriStor restricts the availability of this metadata to system administrators and an explicit list of non-administrator authorized users.

SERVER MANAGEMENT

Each AuriStor Service process running on a server is managed by a separate service, the BOS Overseer Service. The AuriStor BOS service installed on each machine has its own authentication identity and access control independent of other machines. The AuriStor BOS service does not run as “root”.

The AuriStor BOS service is responsible for starting, stopping and restarting AuriStor File System and Database Services as required. The AuriStor File System and Database services do not execute as “root”. AuriStor File System authentication is independent from that used for individual BOS service instances.

AuriStor BOS also permits remote management of AuriStor configuration, access to log files, and configuration of approved administrators for the machine.

AURISTOR
YOUR DATA · YOUR FILE SYSTEM

AURISTOR FILE SYSTEM PERFORMANCE IMPROVEMENTS

Many organizations deploy large numbers of AFS file servers attached to many small file partitions. Increasing the number of partitions is used to reduce the time necessary to restart a server hosting large numbers of volumes. Large file server pools reduce the risk that resource contention on one set of volumes, directories, files and callbacks will have a negative impact on the ability to service requests if the file server's limited thread pool is exhausted. This best practice when combined with off the shelf server configurations such as the Dell R720 result in a significant number of processor cores and I/O bandwidth unused. This deployment strategy has become best practice out of necessity.

AuriStor Servers are designed to utilize the full capabilities of the server hardware. Best practice is to deploy fewer physical servers with more processor cores, more network bandwidth and fewer larger partitions. The number of file servers within a cell should be determined based upon the required geographic deployment of the servers, the number of required file server security policies, and the volume replication strategy for disaster recovery. It is anticipated that most organizations with six or more file servers per cell can significantly reduce the number of file servers deployed using AuriStor.

RX NETWORK TRANSPORT IMPROVEMENTS

All AFS clients and servers communicate using a UDP-based half-duplex RPC protocol called "Rx". An Rx implementation's performance is limited by the number of incoming packets the Rx Listener is capable of processing. Lock contention, processor cache-line invalidation, and packet state management overhead can severely impact the maximum throughput of any Rx implementation.

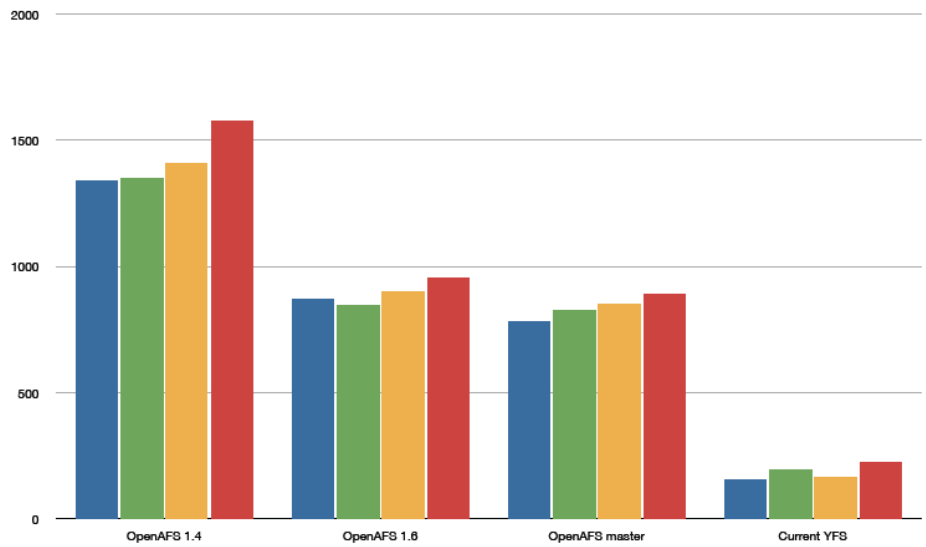
The AuriStor Rx implementation is designed to minimize lock contention between the listener thread which processes all inbound network packets and the application threads that consume them. Lockless data structures are used wherever possible and aligned to avoid cross-core cache line invalidation. The packet loss recovery algorithms have been significantly improved to permit the use of larger window sizes without suffering from performance degradation.

The AuriStor Rx implementation transmits half as many acknowledgement packets as OpenAFS Rx. This reduces the work load on the peer's rx listener thread and the time spent contending locks. The result is more time available for processing data packets.

AuriStor Rx is fully compatible with IBM AFS and OpenAFS. When AuriStor Rx is used by both endpoints sustained rates exceeding 8.2 Gbits/second per listener thread can be achieved. Significant improvements can be achieved simply by deploying AuriStor Rx on the endpoint transmitting bulk data.

When AuriStor Rx is deployed on Red Hat Enterprise Linux 7.x or Debian Linux 8, more than one Rx Listener thread can be run in parallel. In these configurations AuriStor Rx performance is bound by the number of processor cores and the available network bandwidth.

AURISTOR
YOUR DATA · YOUR FILE SYSTEM



Time (in ms) to copy 60Mbytes of data over RX

AURISTOR FILE SERVER PERFORMANCE IMPROVEMENTS

The AuriStor file server has been designed to avoid lock contention and improve scale via the use of dynamic thread pools. AuriStor servers are capable of processing thousands of RPCs at a time limited by the OS and hardware capabilities. A single AuriStor file server properly configured can service the load of more than sixty OpenAFS 1.6 file servers. AuriStor server threads contend for resources less frequently and hold them exclusively for short time periods. The result is significantly reduced risk of cascading call waits leading to meltdowns. AuriStor server deployments can consist of fewer physical servers and thereby save organizations money that would otherwise be spent on electricity, server acquisition, and maintenance.

Most importantly, work flows involving multiple producers and consumers sharing a common set of the directories or files are now safe to execute in the /afs name space.

AURISTOR DATABASE SERVER PERFORMANCE IMPROVEMENTS

AuriStor and AFS rely upon a number of distributed databases to store Location information about file servers and volumes and Protection information about users and groups. The distributed transaction processing protocol is UBIK. The AuriStor UBIK implementation is built upon the same Rx RPC stack and thread pools as the File Servers. Each database server instance can process thousands of requests in parallel with lower request/response latency.

Improvements in the UBIK configuration management and the quorum establishment algorithms permit the use of multiple IP addresses per server and the use of database quorums that are split across private and public networks.

Management of UBIK databases no longer must be performed from a system that has connectivity to the primary address for each service. AuriStor UBIK permits servers to be ranked in arbitrary order regardless of the IPv4 addresses assigned to the machine.

IPV6 SUPPORT

As of 30 June 2014 the U.S. Federal Government requires support for IPv6 as part of all software purchases. In particular, all public network services must support clients that exist on IPv6 only networks. AuriStor satisfies this requirement with the following functionality matrix.

RPC Interface	IPv4	IPv6	IPv6 Only
UBIK VOTE	Yes	No	No
UBIK DISK	Yes	No	No
VL	Yes	Yes	No
PT	Yes	Yes	No
VOL	Yes	Yes	No
RXAFS (File)	Yes	Yes	No
RXAFSCB (Cache Manager)	Yes	Yes	Yes
BOZO	Yes	Yes	Yes

All administrative tools (bos, vos, pts, udebug, cmdebug, etc.) are IPv6 enabled. AuriStor implements IETF RFC 6555 “Happy Eyeballs: Success with Dual Stack Hosts” to ensure that network calls are promptly completed regardless of which protocols can be used to reach the required host.

The AuriStor support for IPv6 permits AuriStor clients to communicate with IPv4 only AFS servers across Microsoft Direct Access gateways. The Microsoft Direct Access domain lookup policy must include the in-addr.arpa sub domains for internally deployed IPv4 addresses.

Each AuriStor server must be assigned a unique IPv4 address. AuriStor servers cannot operate on IPv6 only networks. This restriction will be removed in a future release. However, servers on IPv6 only networks will not be accessible to IBM AFS and OpenAFS clients.

IPV6 PERFORMANCE IMPLICATIONS

The behavior of IPv6 networks differs considerably from IPv4 networks.

- Packet fragmentation is the responsibility of the sender and not the routers.
- UDP packet responses must be manually sent on the same interface and address on which the incoming packet was received.
- ICMP6 response processing must be implemented by the sender to detect UDP packet delivery failures and the underlying reason.
- The minimum packet size is 1280 octets (increased from 560 octets.)
- Most IPv6 nodes are multi-homed supporting
    - Link local addresses
    - Permanent addresses

- Temporary addresses
- Teredo Tunneling addresses

These changes alter the assumptions upon which the AFS Rx protocol stack were designed. For example, the AFS Rx packet size growth algorithms are not IPv6 compatible. The algorithm will result in repeated retry delays and packet fragmentation for all calls sending or receiving more than 1444 octets of data.

The AuriStor Rx implementation has been designed with these differences in mind. In many circumstances it performs better as a result.

UBIK REPLICATED DATABASE IMPROVEMENTS

The distributed services such as the Location and Protection Services rely upon the Ubik replication protocol developed by Mike Kazar. "Ubik's goals[13] were quite simple:

- "Handle small numbers of write transactions (averaging perhaps one update per second over the running of a server.)
- "Handle moderate to high numbers of read transactions between 10 and 100 transactions per second), on multiple servers (for load sharing and availability purposes).
- "Provide both read and update database access in the event of a moderate number of server failures.
- "Handle network partitions reasonably. Network partitions are a reasonably common failure mode in *the CMU* environment.
- "Provide transactional serializability and failure-atomicity guarantees.
- "Provide a simple programming abstraction (a replicated, transactional disk file)."

These goals were appropriate for the 1980s. The resulting implementation does not perform well under increasing load. The voting algorithm could be adversely impacted when 100s of application service requests per second are received. Initial startup and network partition recovery could require up to 7.5 minutes before write transaction processing can be resumed. In addition, there are serious privacy and data integrity concerns.

The AuriStor Ubik implementation addresses all of these issues:

- The election algorithm has been modified to reduce the average quorum establishment to 8 seconds and the maximum recovery period to 34 seconds.
- Vote validation ensures that only votes from database servers sharing the identical configuration are counted when electing a coordinator. This prevents database forking during network partitioning events.
- All communication is integrity protected and encrypted to ensure the privacy of the election and the database contents.
- RPC timeouts for server-to-server and client-to-server connections prevent an unresponsive server from deadlocking other cell participants.
- Short circuit elections
- Ubik databases can be comfortably replicated across 40 database servers.

---

[13] 1989 IEEE, "Ubik: Replicated Servers MadeEasy", Michael Leon Kazar

- The AuriStor Rx implementation increases the parallelism of each server instance and decreases the per request latency.

The end results are AFS cells that perform well at scale whether in a single data center or distributed across geographical locations.

FILE NAMESPACE IMPROVEMENTS

The AuriStor File System implements many small but important functional improvements:

- Mandatory locking. Windows applications expect mandatory locking to be enforced by the file server. AuriStor file servers do.
- Per file ACLs, ACL inheritance and cross directory hard links.
- Volume Status info updated with each file server RPC avoids significant numbers of RPCs from clients and permits further efficiencies in client caching.

CONFIGURATION IMPROVEMENTS

- A new CellServDB format supports a broad range of endpoint types such as IPv6; and permits arbitrary port numbers, priorities and weights for load balancing.
- Other configuration files such as ThisCell, TheseCells, CellAlias, NetRetrict, NetInfo, krb.conf, krb.excl, etc., and all command line parameters are consolidated into a flexible configuration profile.
- Log rotation is compatible with “logrotate”

CLIENT IMPROVEMENTS

The AuriStor distribution includes brand new installation packages for:

- Microsoft Windows
- OSX
- Red Hat Enterprise Linux
- Debian
- Fedora

Not only are the installers digitally signed but so are the executables, shared libraries and kernel modules. On OSX and Windows, digital signatures are necessary for integration with built-in firewall services. The Windows installer is an all-in-one package providing all necessary 64-bit and 32-bit components including Heimdal Kerberos/GSS assemblies.

MICROSOFT WINDOWS

- Supported platforms
    - Windows 7 and above
    - Windows Server 2008 R2 and above

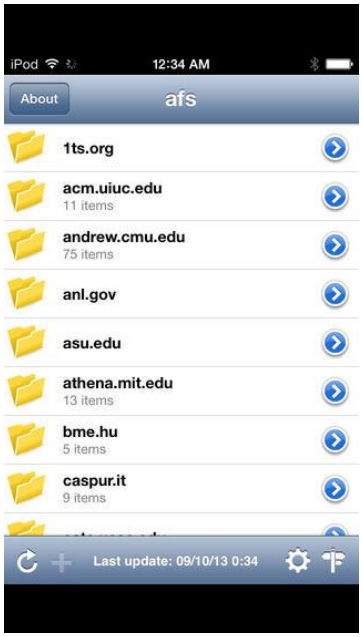
  - Windows 10 and Server 2016 certification (pending)
  - Older platforms can be supported under contract
- Hardened UNC Access (MS15-011 / MS15-014) provided for
  - Configuration distribution
  - Binary execution
  - Roaming profiles
- ICMP Filter driver to enhance Rx performance
  - Faster detection of down servers
  - Path MTU discovery
  - Retransmission of fragmented packets
- Microsoft Direct Access compatible
- Domain joined machines authenticate machine to AuriStor Services automatically

APPLE OSX

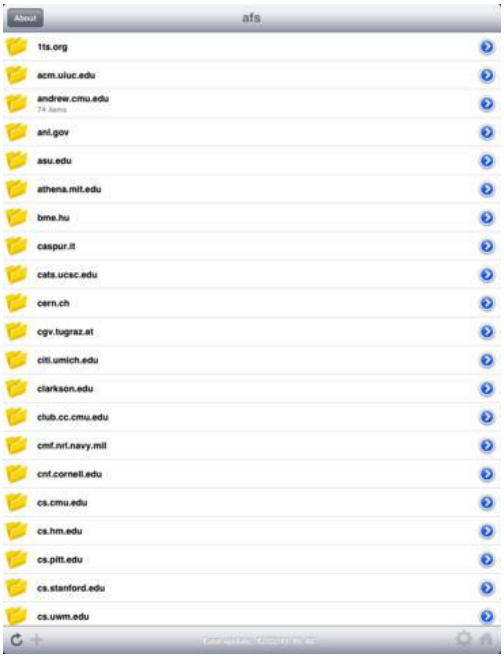
- Supported platforms
  - Mavericks, Yosemite, El Capitan, and Sierra (beta)
- Digitally signed installers
- Digitally signed kernel module

APPLE IOS

Distributed via iTunes, iYFS is a native AFS file browser for iPhone, iPod, and iPad.



iPhone and iPod

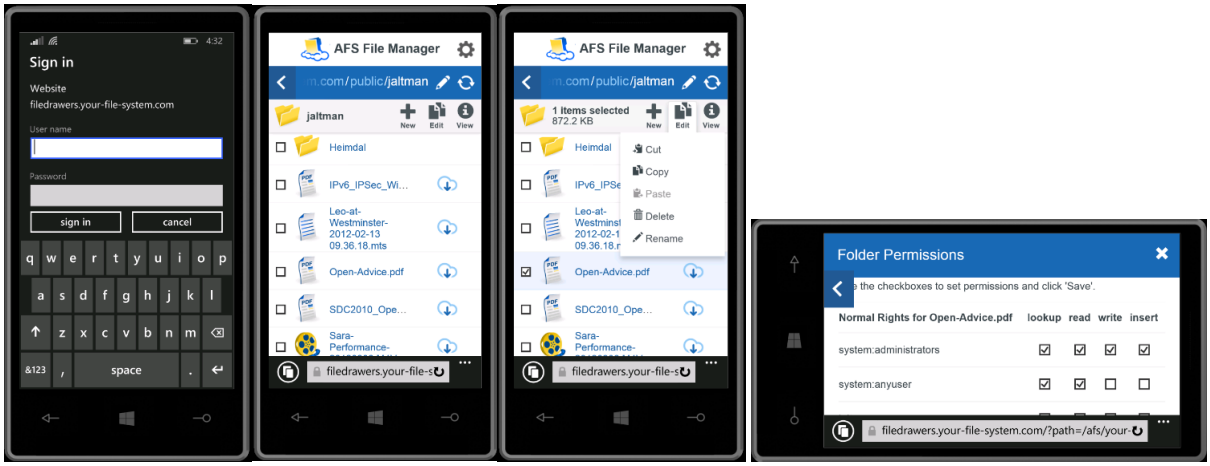


iPad

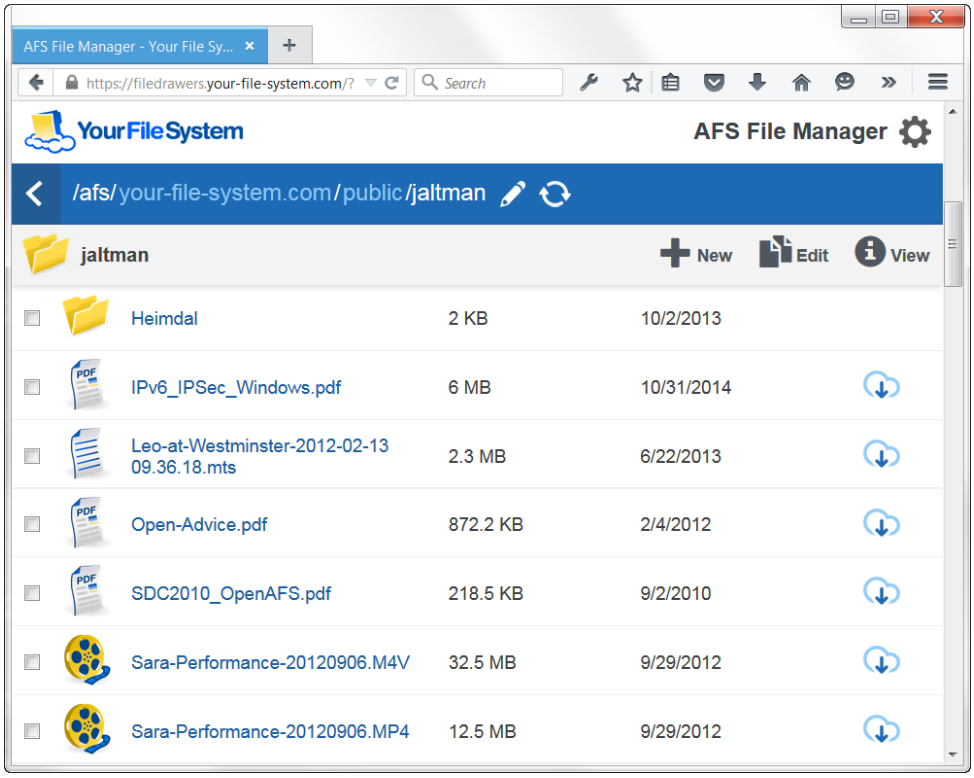
AURISTOR
YOUR DATA · YOUR FILE SYSTEM

AFS-WEB INTERFACE

The AFS-Web interface is an Apache hosted web service that provides access to the /afs file namespace. Derived from the University of Michigan's Filedrawers project, AFS-Web is compatible with all desktop and mobile web browsers. AFS-Web is the perfect companion for desktops and mobile devices for which no native client is available.



Windows Phone 8.1



Mozilla Firefox on Microsoft Windows

AURISTOR
YOUR DATA · YOUR FILE SYSTEM

SOURCE CODE QUALITY

Reliable, robust, secure software products require audited and maintainable source code. Whereas OpenAFS is the product of thirty years of collected technical debt, AuriStor, Inc. has paid off the debt and produced clean maintainable software modules that are easily tested and audited. The AuriStorFS code base is audited and tested on a daily basis using tools such as *Coverity®*, *clang*, *valgrind*, and *stack*. Approximately 1000 vulnerabilities have been identified and corrected.

AuriStorFS testing is performed on functional units, libraries, rpc services and whole cells involving in some cases hundreds of database and file servers. As of this writing there are more than 4000 individual tests.

Additional software quality improvements include:

- Sensitive data is securely overwritten before memory is released
- Guards have been added to prevent random stack data from inclusion in network and inter-process messages
- Hundreds of thousands of lines of unused or otherwise unmaintained source code have been removed
    - Conditional features
    - Unsupported platforms (Linux 2.4, HPUX, DUX, SGI, XP, WIN95, etc)
    - Unused functions
    - Unreachable statements
- Removal of shared data objects from libraries
- Removal of pre-IBM 3.6 RPCs and supporting code

MIXED AURISTORFS AND OPENAFS DEPLOYMENTS

AuriStorFS was designed to permit in-place migrations from OpenAFS without a flag day. However, there are restrictions and limitations depending on the mix of OpenAFS and AuriStor clients and servers within the cell.

AURISTOR CLIENT WITH AFS VOLUME LOCATION SERVER

The AuriStor client when communicating with an AFS Volume Location Server is restricted to AFS capabilities.

- Enhanced RxGK authentication and AES-256/SHA-1 security are not possible within the cell.
- File Servers must listen on port 7000 and IPv6 cannot be supported

AURISTOR CLIENT WITH AFS FILE SERVER

The AuriStor client when communicating with an AFS File Server is restricted to AFS capabilities.

- Rx performance is better than an AFS client but is degraded
- RxKAD authentication and FCRYPT wire privacy MUST be used even if the cell supports RxGK
- Mandatory Security Levels are not enforced
- Timestamps, Sizes, Volume IDs, and File IDs are restricted.
- Mandatory lock requests are ignored

- Additional round trips are required for volume status information, locks after create, and other RPCs.
- Quota and volume size information may be falsified.

AURISTOR FILE SERVER AND AFS VOLUME LOCATION SERVICE

The AuriStor File Server can be deployed in an OpenAFS cell but is restricted to AFS capabilities.

- RxGK authentication cannot be enabled.
- File Server must listen on port 7000.
- Volume ID name space restricted to AFS limits.
- Only AFS compatible capabilities can be advertised.
- Security policy requirements cannot be enforced.
- IPv6 addresses cannot be registered with the Volume Location service.

AFS FILE SERVER AND AURISTOR VOLUME LOCATION SERVICE

The AFS file server can be deployed in an AuriStor cell with the following restrictions.

- File server cannot enforce or advertise mandatory security level requirements.
- File server must listen on port 7000.
- RxGK is unavailable.
- File server cannot host volumes with IDs larger than 2^{32}-1.
- File server cannot be a replica site for volumes stored on an AuriStor file server.

AFS AND AURISTOR VOLUME SERVER INTEROPERABILITY

AFS and AuriStor file servers can be deployed within a single cell with the following restrictions.

- Volumes cannot be replicated from an AuriStor server to an AFS server.
- Volumes cannot be moved from an AuriStor server to an AFS server if there would be data loss
    - ACL data
    - Volume attributes
    - Directories with more entries than the AFS limits
    - Extended attributes and alternate data streams
- All data transfers from an AuriStor volume server to an AFS volume server are protected with FCRYPT and not AES-256/SHA-1.
- All data transfers from an AFS volume server to an AuriStor volume server are unprotected.
- Rx performance is improved over AFS to AFS but degraded compared to AuriStor to AuriStor.

AFS CLIENT AND AURISTOR VOLUME LOCATION AND FILE SERVERS

AFS clients are restricted to the capabilities of the AFS protocol and the RxKAD security class. AFS clients can continue to interoperate within an AuriStor cell provided that the AuriStor servers are not configured to require

the use of RxGK, keyed cache managers, combined tokens, or AES-256/SHA-1 wire privacy. In addition, AFS clients experience the following restrictions:

- Volumes with IDs larger than 2^{32}-1 are inaccessible
- Directories that contains objects with per-object ACLs are inaccessible
- Directories containing more entries than can be represented in AFS are inaccessible
- Extended attributes and alternate data streams are hidden. Deletion of the object will delete the extended attributes and alternate data streams without notice
- Mandatory locks cannot be requested but will be enforced when issued to other clients
- Read-write volume replicas are unavailable
- Volume quotas and sizes will be faked if the actual sizes are larger than 2^{31} KBs
- No ability to enable directory ACL inheritance
- Other restrictions as required to enforce security policies and prevent data corruption

AURISTOR AVAILABILITY

AuriStor is being licensed to organizations in a pre-1.0 state that can accept limited availability of documentation.

AuriStor 1.0 will be generally available when documentation is complete.

AURISTOR FILE SYSTEM SERVER PLATFORMS

- Red Hat Enterprise Linux 7.x
  - Intel/AMD x64
  - IBM Power Series 7 and 8
  - ARM
- Red Hat Enterprise Linux 6.x
  - Intel/AMD x64
- Red Hat Enterprise Linux 5.x
  - Intel/AMD x64
- Red Hat Fedora 21+
- Debian 8
- Ubuntu 15
- Apple OSX 10.9, 10.10, and 10.11
- Solaris 11.x
- Windows Storage Server 2012 R2 (experimental)

AURISTOR
YOUR DATA · YOUR FILE SYSTEM

WHY AURISTOR FILE SYSTEM IS A COMMERCIAL PRODUCT

AuriStorFS is a commercial product as a response to the *free rider problem*[14] that significantly underfunded the development of the open source OpenAFS. As *a global public good*[15], OpenAFS suffers from non-excludability and non-rivalry. The problems faced by open source developers are similar to those of a street musician that attempts to cover all living expenses and musical training solely from donations received from passersby. Although a handful of people will throw in a quarter or a small number of dollars, the vast majority of the public enjoys the music without contributing.

The core OpenAFS developers, like those of OpenSSL, Heimdal and many other open source projects, are unable to take care of their families and cover the costs of new development. File systems similar to security projects require developers with specialized knowledge. As a cross-platform file system the required expertise spans across the inner workings of all supported operating systems, network protocol design, security protocol design and deployment, package management, and more.

From 2005 to 2012 OpenAFS development was subsidized by the OpenAFS gatekeepers and the personal sacrifices of other key developers. Instead of relying on the goodwill of the community to cover on-going expenses, AuriStorFS is the result of up-front investment which must be recouped from end user organizations that wish to benefit from it.

LICENSING TERMS

The base price is US$21,000 per cell per year which includes a perpetual use license that provides for:

- Up to 4 database service instances (3 production and 1 test/maintenance)
- Up to 4 file service instances (3 production and 1 test/maintenance)
- Up to 1000 User or Machine IDs included
    - Disabled and “AFS Only” attributed entities do not count towards the license limit
- Unlimited AuriStor client software for all supported platforms
- One year of unlimited 9-5 M-F EST e-mail and web support with a 4 hour response time
- Permission to use any version of AuriStorFS released within one year after purchase
- Security updates for each licensed version for at least two years from date of release

Additional file or database service instances within a cell start at $2500 each and decrease to $1000 each based upon the number.

Single server cells are licensed at a base price of US$6,500.

[14] https://en.wikipedia.org/wiki/Free_rider_problem
[15] https://en.wikipedia.org/wiki/Global_public_good

Additional User or Machine IDs are priced according to the following table:

Number of Combined User and Machine IDs		Cost
1	1,000	$0
1,001	2,500	$1,375
2,501	5,000	$3,500
5,001	10,000	$5,000
10,001	25,000	$10,500
25,001	50,000	$20,000
50,001	100,000	$30,000
100,001	200,000	$40,000
200,001	300,000	$50,000
300,001	400,000	$60,000

There is no additional cost for client software. Digitally signed installers will be available for organization to distribute to its community without restrictions. Organizations are encouraged to distribute client installers customized with cell specific configuration.

Licenses provide for perpetual use of the most recent version of AuriStor for which the organization is entitled regardless of whether or not the organization renews the license in subsequent years. Audits of the numbers of server instances and User/Machine IDs will be performed periodically. Feature updates are only available to customers with an active product support license. Security updates will be distributed for at least two years from the release of each AuriStorFS version. Feature releases are planned on a four-month cadence or three times per year.

At the end of each year the customer can opt to:

- Pay nothing and continue to use the most recent licensed version of AuriStor indefinitely under the terms of the prior license without support, or
- Purchase a new perpetual use license providing access to new software updates and technical support at the published rates.

AURISTOR
YOUR DATA · YOUR FILE SYSTEM

SITE LICENSE OPTIONS

Organizations that wish to deploy unlimited numbers of AuriStorFS cells for internal use can make use of one of the following site license options:

Cell Limit	Per Cell DB Server Limit	Per Cell File Server Limit	Per Cell Authorization Entity Limit	Per Cell Client Limit	Annual Price
Unlimited	4	4	10,000	Unlimited	US$500,000
Unlimited	8	10	100,000	Unlimited	US$750,000
Unlimited	12	25	250,000	Unlimited	US$1,000,000
Unlimited	16	50	500,000	Unlimited	US$1,500,000
Unlimited	20	75	1,000,000	Unlimited	US$2,500,000
Unlimited	40 (max)	252 (max)	Unlimited	Unlimited	US$5,000,000

AURISTOR FILE SYSTEM SOURCE CODE AVAILABILITY

AuriStorFS is a closed source proprietary product that incorporates a large number of source and documentation files licensed under the IBM Public License 1.0 (IPL10). Under the terms of IPL10, AuriStor, Inc. will upon request from an authorized entity provide source code for the modified and unmodified IPL10 licensed files. This file set is insufficient to generate AuriStorFS executables, libraries and packaging.

A source code license for the complete source tree is available to all AuriStorFS Enterprise Licensees for a one-time cost of US$50,000 per organization. The organization must agree to restrict access to the source code and not redistribute it to third parties. Locally derived patches cannot be exchanged with other AuriStorFS source code licensees without the permission of AuriStor, Inc. Locally derived patches can be contributed to AuriStor, Inc. as a work-for-hire for possible inclusion in future releases.

AURISTOR TRAINING OPTIONS

AuriStor, Inc. offers on-site and Internet hosted training on AuriStorFS in-general and migration from AFS at a cost of US$6000 per class. Each class may be attended by up to 12 students.

THE AURISTOR ROAD MAP

Major releases of AuriStor are planned to be issued once every four months. They will include continuing improvements to performance, scalability and usability for both clients and servers.

The AuriStor 1.0 product contains a core set of functionality which will be the basis for integrating additional capabilities in the coming months. The following set of features is intended for delivery in the year following the 1.0 release. The order of delivery is subject to change based upon customer prioritization.

- Extended attributes and Alternate Data Streams

- Read/write replication
- Active/passive File Server failover
    - Migrate partitions between servers
- Hierarchical UBIK
    - Improved scalability across multiple geographic locations
    - Isolation domains for improved survivability
- Protection service callbacks for File Servers
    - User and Group changes will be immediately reflected in File Server authorization decisions
- New directory format
    - Larger directory sizes
    - Alternate data streams
    - Extended attributes
    - Object type hints
- Extended callback protocol
    - Directory updates (inserts, deletes, renames) pushed to caches
    - Invalidation of file data range instead of whole file
    - Volume management notifications (offline, move, release)
    - Server side byte range locking capabilities
- Monitoring Service
- File System Favorites
    - Windows Explorer Shell
    - OSX Finder
    - iYFS
    - WebAFS

The following capabilities are planned for an as yet unscheduled release:

- Native Android and Windows Phone clients
- Encryption at rest
    - Encrypted vice partitions
    - End-to-end encryption utilizing an independent key server and per object key management
- Time Machine style snapshot capability
- File Server integrated malware scanning
- File Server integrated personal information scanning
- Additional security features
    - Fine grained delegation of administrative volume operations
        - Backup, Release, Volume ACL, Volume Security Policy
    - Public key encryption of server and client debugging and statistics data
- Indexing Service

CONTACT INFORMATION

For additional information please contact:

AuriStor, Inc.
sales@auristor.com
https://www.auristor.com/filesystem/
+1 212 769-9018 United States
+44 (0) 20 3239 1966 United Kingdom