

CIFS/clustered ONTAP/Off-box AV/Architecture Spec

NetApp Confidential

< CIFS | clustered ONTAP/Off-box AV

<i>Clustered ONTAP Off-box AV Architecture Spec version 1.1</i>	<i>15 March 2013</i>	<i>Approved</i>	<i>dl-cmode-offbox- vscan</i>
---	--------------------------	-----------------	-----------------------------------

All architecture specifications must be linked into the PAG's tree-of-specs, and must be approved by an owner of a higher-level node in that tree

Contents

- 1 Overview
- 2 Related Documents
- 3 Terminology
- 4 Requirements
- 5 Dependencies
 - 5.1 Others who are counting on this architecture:
 - 5.2 This architecture needs the following from others:
- 6 Architectural Block Diagram
- 7 Subsystem and Interface Descriptions
 - 7.1 M-host and {N,D}-blade
 - 7.2 N-blade and D-blade
 - 7.3 N-blade and SMB clients
 - 7.4 N-blade and vscanners
 - 7.4.1 N-blade and avshim
 - 7.4.2 N-blade and vscan engine
 - 7.5 avshim and vscan engine
- 8 Flow Diagrams
 - 8.1 Basic request/response flow
 - 8.2 Control flows initiated by vscanner
- 9 Cluster Considerations
- 10 Security
 - 10.1 Avshim
 - 10.2 Vscan engine
 - 10.3 Intra-vscanner security
- 11 Vserver and multi-tenancy considerations
 - 11.1 ONTAP
 - 11.2 Vscanner
 - 11.3 UI, vserver health
- 12 Modularity considerations
 - 12.1 Off-box components

- 12.1 Off-box components
 - 12.2 N-blade components
 - 12.2.1 Connection Manager
 - 12.2.2 Dispatcher
- 13 Use of architectural interfaces
- 14 Resource Requirements
- 15 Feature Interaction Dependencies and Impacts
 - 15.1 FlexCache interactions
 - 15.2 MetroCluster / Vserver DR interactions
 - 15.3 Snaplock interactions
 - 15.4 HA
 - 15.5 NDU impact
- 16 Test Strategies
- 17 Reliability
- 18 Availability and Non-Disruptive Operations (NDO)
 - 18.1 7-mode NDU and Cluster NDU (Rolling Upgrades)
 - 18.2 Unplanned Component Failures
 - 18.3 HA Considerations
 - 18.4 Maintenance Operations
- 19 Supportability
- 20 Scalability
- 21 Open Issues
- 22 Revision History
- 23 Approvals

1 Overview

NetApp customers, and especially our enterprise customers, require on-access anti-virus protection. "Plan A" for clustered ONTAP was to provide this functionality using onboard AV scanning, utilizing the M-host to run a user-space application which makes calls into libraries provided by our AV partners: McAfee, Sophos, Symantec, and Trend. Due to a number of factors, some technical, some business, the onboard solution was not satisfactory, and the prospects for a quick solution were not good.

As a consequence, we have chosen to implement an off-box AV solution which leverages the existing 7-mode vscan technology. This document assumes some familiarity with the 7-mode vscan implementation, so as a refresher, here is a brief overview:

- 7-mode ONTAP has a "vscan" command which administrators use to enable, monitor, and configure one or more external Windows boxes (*vscanners*) which are running AV-partner-provided software (called in this document a *vscan engine*) which is capable of receiving requests from the filer to scan files, and then of sending responses to the filer, indicating the clean/infected state of the scanned file. This facility is available in 7-mode for on-access scanning of CIFS protocol requests only.
- The filer has a special CIFS share, `ontap_admin$`, which allows privileged access to administrative clients. Requests on this share can access any path on a vfiler, bypass any locks, and ignore security on files. This is required so that the vscan engine can perform its functions.
- A Windows server running a vscan engine registers with the filer as a vscanner using MSRPC (Microsoft's version of DCE/RPC).
- When 7-mode ONTAP detects that we need to scan a file, a request is sent using MSRPC, with the filer

in a client role, to one of the attached vscanners. That notification request returns immediately after the request has been sent.

- The vscan engine opens the file it has been asked to scan, performs the scan, and if infected, performs any cleanup/quarantine operations. It then sends a completion notification request to the filer, again over MSRPC.
- The 7-mode CIFS code then re-drives the request to WAFL, this time with the scan status in the message. WAFL marks the scan status appropriately in a bit in the in-core inode.
- The scanned status bit is not persistently stored. And any time one of the vscanners updates its scan pattern database, a message goes to WAFL to traverse the entire list of in-core inodes, clearing the scanned bit.

With appropriate modifications, this mechanism can be used to perform scanning for clustered ONTAP.

- It is scalable — an arbitrary number of vscanners can serve an arbitrary number of vservers.
- It provides high performance with greatly reduced impact to ONTAP resources, compared to onboard AV.
 - It has essentially zero M-host memory footprint, and essentially zero M-host CPU requirements.
- It is robust — multiple vscanners provide redundancy.
- It is flexible — different vendors can be used on the same cluster, and if desired, a pool of vscanners can provide scanning for one or more entire clusters.
- It provides 100% compatibility for our AV vendor partners, trivially allowing them to provide the latest versions of vscan engines and update mechanisms for their software.
- It decouples AV vendor partners' development schedules from those of NetApp.

When it was evident that we would have to provide an alternative to the onboard AV feature, we examined a number of approaches, including ICAP and WCCP. But for a file server, where it is not always required to read the entire file for scanning, the RPC-based approach we chose avoids a general issue with ICAP and WCCP: those protocols conflate the notification with the actual sending of the data to be scanned. That, combined with the fact that we already have a lot of mind-share around our existing 7-mode vscan technology, makes the approach documented here a better choice. If AppHost were available, we would have certainly considered using that, but it is too far in the future to be a viable alternative. We also considered, and prototyped, an AV implementation based on the clustered ONTAP fpolicy feature, but we found that the performance was not acceptable, and further, some of the business problems were not addressed by that approach.

The modified 7-mode vscan approach that we chose attempts to minimize changes required of our AV partners, while at the same time minimizing the impact to other areas of ONTAP. Almost all of the work is in the N-blade, and nothing we are doing precludes bringing back onboard AV if/when we can solve the issues that caused us to pull it from the product.

Another goal of this work is to improve the robustness and manageability of off-box AV, compared to 7-mode AV. This is one of the big pain points for existing customers who use 7-mode AV, and we aim to make the management of this feature much easier, and to provide much more information on the state of AV for storage administrators.

There are also two serious weaknesses with security in the 7-mode vscan implementation:

- We require the vscanner to open up a named pipe for anonymous access
 - Note that unlike 7-mode vscan, there is no requirement with this design for ONTAP to establish an outbound connection with the vscan engine. This will avoid a host of problems that are often seen when 7-mode vscan is configured, such as those described in kb 2011317.

- Any client with an administrative login, from any IP address, is allowed to register as a vscanner

The new design does not suffer from either of these issues.

The implementation involves no on-disk changes, and only minor changes to SpinNp. And since the required SpinNp changes are already checked into SierraNevada.0, we have considerable flexibility in release vehicles for this functionality.

2 Related Documents

- Engineering Requirements Response Document

3 Terminology

vscanner

A system running Windows which has a vscan engine and avshim running on it.

vscan engine

AV partner-provided software which is installed on a vscanner to perform anti-virus scans.

avshim

NetApp-provided software running on a vscanner which looks like a 7-mode filer to the vscan engine, but interfaces with vservers on a cluster to receive scan requests and send scan result notifications.

4 Requirements

The primary requirement for this feature is to provide on-access anti-virus protection that is comparable in both scaling and performance impact to the existing 7-mode vscan feature. We will also provide some important enhancements to the manageability and supportability for clustered ONTAP, as will be described further in this document. Please see the ERRD for a complete discussion of the requirements.

5 Dependencies

One of the desirable properties of the design we have chosen is that it has few interactions with other engineering groups. Almost all new work is in the N-blade, and the largest impact is to the CIFS protocol code, which this is a part of. There are of course some changes to the D-blade, and there are UI changes, but we will require only reviews from other groups.

The new avshim component will be developed by the ONTAP CIFS team, with consulting assistance from other groups within NetApp that produce software intended to run in a Windows environment. The biggest dependency is for help with the installation and packaging, and this has already been arranged for.

We will be doing some performance testing within CIFS engineering, but we will also benefit greatly from consulting with the Performance Engineering team, and again, we have already engaged a team there to assist with performance testing.

Outside of engineering, the support organization is most impacted by this, as they will be interacting with it on a regular basis. Field personnel have been, and will continue to be, a valuable resource for feedback on this feature.

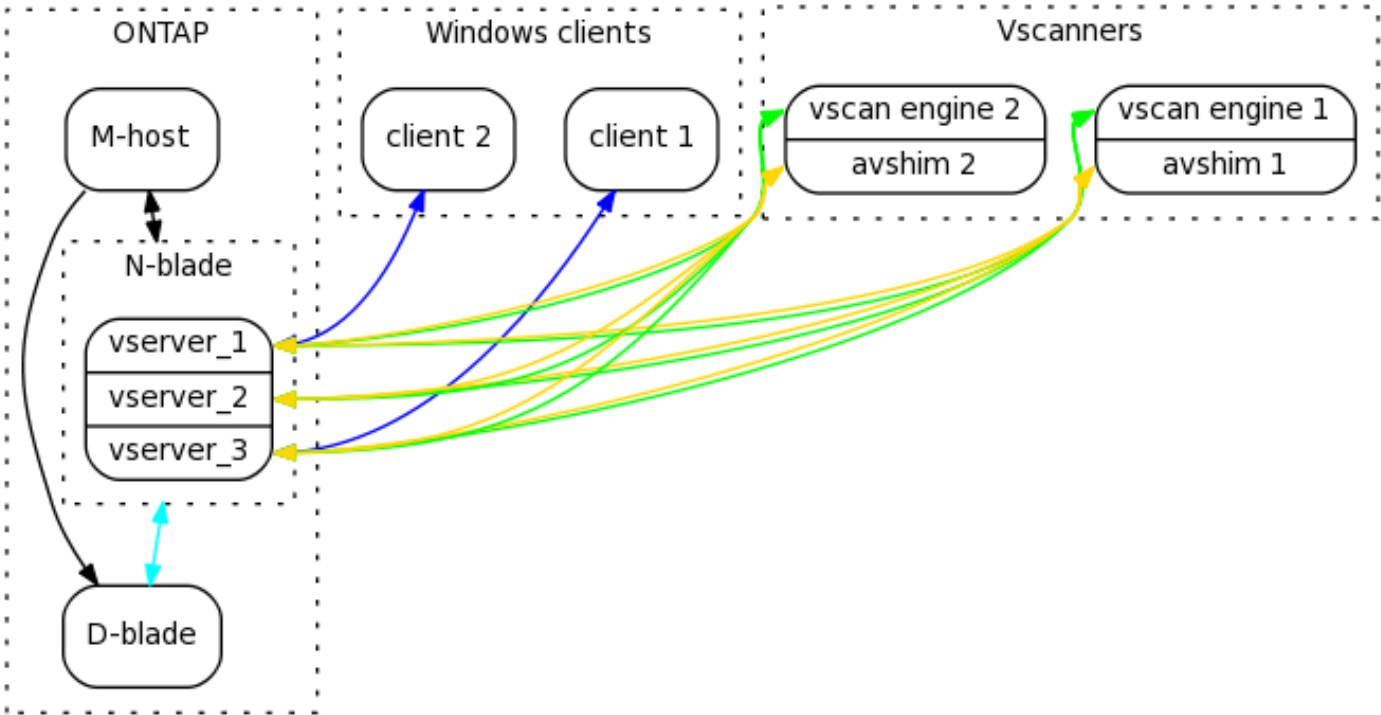
5.1 Others who are counting on this architecture:

None.

5.2 This architecture needs the following from others:

Item	Description of Dependency or Issue	Affected Group	Contact	Tracking ID
1	Assistance with packaging and installation of Windows avshim service	MVA	Chandan Manjunatha, Kevin Mackey	678515
2	Assistance with performance testing and characterization	Performance Engineering	Nayla Nassif	N/A

6 Architectural Block Diagram



Legend:

The **ONTAP** box comprises the standard M-host, N-blade, and D-blade components of Data ONTAP. They are not necessarily on a the same node, and multiple nodes are supported.

The **Windows clients** box represents normal SMB1, SMB2, or SMB3 Windows clients, connecting to CIFS shares over vsriver data LIFs.

The **Vscanners** box comprises one or more vscanners, which are Windows boxes, each of which has both the NetApp-supplied *avshim* component, and the AV-partner-supplied *vscan engine*. Those two components communicate with each other on the same vscanner using a slightly modified version of the 7-mode vscan MSRPCs, over an in-memory loopback connection. There is no pre-set limit to the number of vscanners that can

be configured. One or more vscanners can be configured to service only a single vservers, or they may service multiple vservers, or even all the vservers on a cluster, as long as the accounts used for connecting to all vservers are valid. In fact, there is no requirement for a vscanner to service only vservers on a single cluster, as long as all the vservers on all the clusters have compatible security information and are configured for those vscanners.

The **blue** lines show TCP connections between Windows SMB clients and vservers LIFs. These are normal client connections.

The **gold** lines show TCP connections between the avshim component on a vscanner, and the vservers. At least one connection will exist between each avshim component and each vservers which has LIFs on a node, for each node in the cluster. These are SMB connections which are authenticated as a special user account which is allowed to access the `vscan` named pipe which is used for the notification protocol.

The **green** lines show TCP connections between the vscan engine component on a vscanner, and the vservers. At least one connection will exist between each vscan component and each vservers which has LIFs on a node, for each node in the cluster. If multiple data LIFs are available for a given vservers on a node, the avshim may use a round-robin among those LIFs when creating scan requests. These are SMB connections authenticated using a specially configured administrative user account which is allowed to access the `ontap_admin$` share, granting supercred and superlock privileges to the vscan engine.

The **cyan** line represents the normal SpinNp connection between the N-blade and the D-blade.

The **black** lines show intra-node communications which use KSMF to retrieve and send configuration information from the M-host. Both N-blade and D-blade require various pieces of this configuration information, and status information can also flow up to the M-host from the N-blade, via KSMF. In this design, it is never necessary for an N-blade or D-blade to contact the M-host of another node, as the configuration information is replicated cluster-wide.

7 Subsystem and Interface Descriptions

7.1 M-host and {N,D}-blade

As described in the block diagram, configuration information flows between the M-host and N-blade and D-blade, using KSMF. The actual size of this data will be limited, comprising a small number of per-vservers tables with policy and scanning rules, along with a list of allowed vscanners and privileged user accounts.

The D-blade requires a table used to map version numbers of attached scanners to the version numbers that are recorded after a file is scanned. The details of how this is done are described in the design spec. Here, we just note that this information is updated only when a vscan engine registers or updates its virus pattern database, so the volume of changes here is small.

Additionally, the N-blade updates the M-host with information about the attached vscanners and their anti-virus database versions, as well as conditions on each vscanner that are relevant to the storage administrator.

7.2 N-blade and D-blade

The SpinNp protocol was modified slightly in SN.0 to accommodate the needs of off-box AV. File operations that may require scanning (open, read, close, rename) have some additional fields which are used to convey whether a scan is required, and after a scan, what the scan results were. There is also a new SpinNp status which indicates that the D-blade has determined that a given request has triggered a scan.

Unlike the onboard AV design, once the D-blade determines that a file must be scanned, the N-blade never sends the request to the D-blade until it has a valid scan status. And the offbox AV checks on the D-blade all run in any waffinity context. All of this reduces the amount of overhead greatly, compared with onboard AV.

7.3 N-blade and SMB clients

This is the normal SMB1, SMB2, or SMB3 traffic that we handle today. There are no additional limitations or requirements on those clients in order to support offbox AV. Both IPv4 and IPv6 will be supported, as usual.

7.4 N-blade and vscanners

The N-blade has two different connections to the vscanners: one for the notification protocol, and one for the vscan engine to access the files to be scanned. Both of these connections use standard, authenticated SMB1 or SMB2 sessions, initiated by the vscanner.

7.4.1 N-blade and avshim

The avshim component is loaded as a service module on a Windows box which is to be used for anti-virus scanning. In a manner described later, the avshim obtains the list of all vservers LIFs that this vscanner is supposed to service, and it establishes an SMB connection with a new named pipe, called `vscan`, on at least one data LIF on each node of the cluster that has a data LIF for that vservers. The N-blade uses this named pipe connection to send notifications and receive scan results. This is a normal SMB connection which is authenticated using normal SMB authentication mechanisms, typically Kerberos. Unlike 7-mode, there is no requirement to modify the security in any way on the vscanner in order to support ONTAP, which eliminates the number one source of problems when installing 7-mode vscanners.

The data which is sent over this pipe is a new notification protocol which is adapted and extended from the existing 7-mode MSRPC calls. This new protocol is currently used only internally, but because we might wish to expose it to our AV partners in the future, we will be providing a complete specification of the protocol as part of this work.

The protocol specifies the wire format of the requests and responses for commands to

- Request a scan of a file from a vscanner and report the results of a scan
- Notify a vservers on a node that the vscan engine has updated its scan database version
- Request status of the avshim and vscan engine
- Report status of a vservers on a node to the avshim
- Register/unregister an avshim

Flow control on this interface is handled by the avshim. When the avshim connects to the N-blade, it opens the `vscan` named pipe, then hangs one or more reads on that pipe. When the N-blade is ready to send a request, say to scan a file, it responds to the read, sending the request itself as the data to be read. If there is not a read pending on the named pipe, the N-blade will queue the scan request for later sending. Ultimately, the original client will receive back-pressure if scan requests take too long to keep up with the incoming request rate.

Responses are written by the avshim on the N-blade's `vscan` named pipe, and the handler for named pipe writes vectors the results to the N-blade code that is waiting for the completions. If these write requests are not handled speedily, they will queue in the CIFS and networking code, same as any other SMB request.

7.4.2 N-blade and vscan engine

When a request to scan is triggered by the D-blade, a special status is returned. This in turn triggers the N-blade to send a scan notification to the avshim over the `vscan` named pipe. This request is forwarded within the vscanner Windows box to the vscan engine using the 7-mode MSRPC vscan API over an in-memory loopback connection. At that point, the vscan engine must open the file for scanning. The path that is handed to the vscan engine is set to use a special share, `ontap_admin$`, which is available only to offbox vscan and `fpolicy`. Part of the offbox vscan configuration is a per-vserver list of IP addresses that are allowed to access the `ontap_admin$` share. There is also a per-vserver list of privileged user accounts that are allowed to access this share.

The extra security around use of this share is due to the fact that superlock and superuser privileges are granted: requests to this share are allowed to access any path in the vserver where the request arrived, regardless of file locks or file permissions.

In addition to granting unlimited access to the vserver, paths may optionally have a special first component which is the encoded file handle of the root of the volume which has the final component of the path. This allows us to forgo re-parsing the entire path each time the vscan engine opens the file, which will typically be several times per scan. We must ensure that vserver boundaries are enforced for this encoded filehandle.

Flow control for these requests is identical to that of normal CIFS requests, though we may want to consider increasing the priority of AV-related work in the future, if analysis shows significant benefit. But that is out of scope for the first release of this project.

7.5 avshim and vscan engine

Both of these components are running on a Windows server that is configured as a vscanner. On startup, the avshim registers as a local-host-only server for the 7-mode NTAPVSCP MSRPC API. That allows a properly configured vscan engine to register with the avshim, just as it would have done had the vscan engine been configured to serve a 7-mode filer. We have introduced a new scan type in order to reduce the possibility of misconfiguration. If the vscan engine does not register with the correct scan type we will return an error.

If the vscan engine successfully registers with the avshim using the new scan type, it indicates that the avshim is then free to send requests over the existing 7-mode NTAPVSRQ MSRPC API which the vscan engine exposes. At that point, the avshim should have already established a named pipe connection on the N-blade's `vscan` pipe with each vserver that it is to service. When a scan request comes in from an N-blade, the avshim will then pass the request to the vscan engine on the NTAPVSRQ API and await the response. When the scan completes, the vscan engine calls the NTAPVSCP API to report the results of the scan, and those results are then passed on to the appropriate N-blade using its `vscan` named pipe.

Flow control between the avshim and the vscan engine is governed by the number of threads that are available for the MSRPC communications. Requests from the avshim to the vscan engine are basically just reformatted versions of the incoming requests from the N-blade, and they are sent from a thread which handles the named pipe reads for a given vserver LIF. If the vscan engine does not have a thread available to handle the request, the calling avshim thread will hang, which will prevent that thread from issuing a new read on the N-blade's `vscan` named pipe. The number of threads available for the vscan engine is a configurable parameter that the AV partners expose if the default is not correct.

Completions from the vscan engine are handled similarly. We will also provide a parameter which allows an administrator or support person to modify the number of threads available to receive scan completion requests.

8 Flow Diagrams

8.1 Basic request/response flow

The following diagram shows the way requests are handled as they move from the client to the N-blade, the D-blade, possibly to the vscanner (avshim and vscan engine), and back to the N-blade and the original client. A note on terminology in the chart: "alt" means multiple alternative flows follow, with the alternatives labeled inside square brackets. Because of a bug in the wiki, the chart may not be rendered. If that happens, try re-loading the page.

```
Client -> +N-blade : file open request
note right of N-blade : determine whether file-op\nshould be scanned,\nusing configured policies.
N-blade -> D-blade : SpinNp-Open, including\noff-box vscan parameters.
note right of D-blade : D-blade checks persisted\nscan information

alt file known to be clean
D-blade -> N-blade : Normal file-open
N-blade -> Client : Normal file-open response
else file known to be infected
D-blade -> N-blade : Access-denied response
N-blade -> Client: Access-denied response
else scan required on file
D-blade -> N-blade : SpinNp-error: vscan-required\n(with file-cookie)
N-blade -> +avshim : Scan-notification\n(including file-path, file-handle)
avshim -> +vscan\nengine: (MSRPC-Req)\nscan-request

vscan\nengine -> +N-blade : File-Read request\n(privileged)
N-blade -> +D-blade : SpinNp-Read request\n(super-locker, super-user)
D-blade -> -N-blade : Read-response
N-blade -> -vscan\nengine : File-Read response
note left of vscan\nengine : Requests continue until file is\nscanned, quarantined, etc.

vscan\nengine -> -avshim : (MSRPC-Req)\nscan-completion
avshim -> -N-blade : Scan-Response\n(including vendor, version)

alt scan-status: clean
N-blade -> +D-blade : Redrive SpinNp-Open request\n(with vendor-version,\nfile-cookie, scan-status)
note right of D-blade : Handles scan results persistence,\nperforms original operation.
D-blade -> -N-blade : SpinNp-Open response
else scan-status: infected
N-blade -> +D-blade : Redrive SpinNp-Open request\n(with vendor-version,\nfile-cookie, scan-status)
note right of D-blade : Handles scan results persistence.
D-blade -> -N-blade : Access-denied response
end
end

N-blade -> -Client : appropriately translated\nD-blade response
```

8.2 Control flows initiated by vscanner

This diagram shows the N-blade sending an update to the M-host over KSMF when a new vscanner registers with the N-blade. It also shows the vscan engine later sending a virus database version update notice.

```
vscan\nengine -> avshim : MSRPC\nregister with avshim
avshim -> N-blade : notification protocol\nvscan engine registered
alt user account and IP address are valid according to the config
N-blade -> M-host : KSMF\nupdate - new vscanner
note right of M-host : M-host replicates this\nto cluster
else invalid user account or IP address
N-blade -> avshim : notification protocol\nvscan named pipe does not exist\naccess denied
avshim -> vscan\nengine : MSRPC\nregistration failed
end

vscan\nengine -> avshim : MSRPC\nvirus database file update
avshim -> N-blade : notification protocol\nvirus version update
N-blade -> M-host : KSMF\nupdate - new version
note right of M-host : M-host makes available\nto D-blade for persistence\nmapping
```

9 Cluster Considerations

The point of this project is to provide 7-mode-like off-box anti-virus to clustered ONTAP, and naturally this feature will be conceptually familiar to existing 7-mode customers. If the AV partners so choose, it will be possible for them to provide a single version of their vscan engine which works with both 7-mode and clustered ONTAP. That works because the MSRPC API is virtually identical, with the primary distinction being that the loopback connection is the only one that will work with the avshim. There is nothing preventing AV partners from supporting simultaneous access to both 7-mode and clustered ONTAP from the same vscanner, though whether they choose to allow this configuration is up to them.

The actual UI used to administer off-box AV in clustered ONTAP is completely different from 7-mode, but all 7-mode functionality will be available, as well as significant extensions to allow better control over which entities that are allowed to be vscanners, superior diagnostic and other supportability features, and more flexible control over vscan policies.

All configuration and status information is replicated cluster-wide. There is no node-scoped configuration.

10 Security

The need to avoid rogue clients masquerading as vscanners, and the need to allow the vscan engine to have unfettered access to all paths on a vserver, means that we must pay special attention to security for the vscanner. We will consider the avshim and the vscan engine separately, then the communications between those components.

10.1 Avshim

The avshim component is responsible for the notification protocol connection between the vscanner and the N-blade. This is a standard SMB connection, initiated by the avshim, which logs into the Active Directory Windows user account that is configured when the avshim is installed. Immediately after authenticating, the avshim attempts to open the `vscan` named pipe on the N-blade. Any attempt to open this named pipe will fail unless the incoming IP address and logged-in account are in the lists of accounts and IP addresses that are maintained as part of the per-vserver off-box vscan configuration. Administrators should take steps to prevent source IP spoofing, as this would defeat one of the N-blade checks. Ingress filtering (http://en.wikipedia.org/wiki/Ingress_filtering) is one obvious choice.

If this off-box vscan configuration is dynamically modified to remove the IP address or login account of an avshim connection that is active, we will tear down that connection as part of the actions that are taken when the updates are received via KSMF on the N-blade. This behavior makes it harder for a rogue client to masquerade as a vscanner, and it also makes it easier to do things like switch off a group of vscanners for maintenance, or to cut over to a group of vscanners that are used as secondary vscanners, or in a disaster recovery situation, etc.

10.2 Vscan engine

The vscan engine connects to the N-blade over a normal SMB connection, typically authenticating using Kerberos and Active Directory. The N-blade to use is specified in the UNC path that the avshim passes in, which may be any of the active data LIFs for a vservers. For the first implementation we will always pass a path that names the node where the original CIFS request arrived. In future releases, we may choose to favor the node where the D-blade is local.

The vscan engine must be allowed access to the `ontap_admin$` share, which means that it will be able to access any file on a vservers, regardless of locks or file security. To make sure that only vscan engines are permitted this access, we maintain a list of IP addresses, and another list of user login accounts, which are maintained by a vservers (or optionally a cserver) administrator. No client is permitted to connect to the `ontap_admin$` share unless its IP address and login account are in those lists. Again, administrators should be encouraged to take steps against source IP spoofing. Using special accounts which are only used for AV can improve security as well.

The path that the avshim hands to the vscan engine has the form

```
\\<server>\ontap_admin$\<encoded-file-handle>\<path-from-vservers-root>
```

Paths in the `ontap_admin$` share are allowed to specify an encoded file handle. This path format allows an optimization where the storage controller does not have to repeatedly deal with possible multiple junction crossings, symlinks, etc., when parsing the path. The rest of the path will be relative to the vservers root, and is not actually used when opening the file from the privileged share. The path would only be used by the vscan engine for logging, or if it needs to examine the path for some other reason related to scanning. If a request comes into the `ontap_admin$` share which does not have the encoded file handle, it will be parsed as usual. This might be the case if the vscan engine is working in a quarantine path, for example.

The file handle encoding is constructed to avoid collisions with normal paths. The details of the encoding are documented in the design spec. *We must take care to make sure that the volume to which the path refers actually belongs to the vservers which owns the LIF on which this request arrived.*

For additional protection, the vscanner can be configured to require SMB signing, and if desired, accesses to this share can be audited, though the performance impact of these options may not make them feasible to use on a heavily loaded cluster.

10.3 Intra-vscanner security

The avshim and the vscan engine communicate over a local, in-memory MSRPC connection. It would be possible to require the two components to use MSRPC security in order to send requests to each other, and if a customer were foolish enough to set up a vscanner on a non-dedicated machine which could be accessed by non-administrator users, it would be a good idea to require such security. However, to avoid an unnecessary

performance hit, we will make the explicit requirement here that any machine to be used as a vscanner will **not** have non-administrative users running on it, and that the **only** applications running there will be related to anti-virus. The wisdom of this ought to be self-evident, but we will still want to document this carefully, just to be sure.

To prevent some other machine from pretending to be a vscan engine, the avshim will only accept requests from a local source. Further, only one instance of the avshim can run on a single vscanner, and the avshim will accept only one registration from a vscan engine. We will work with our AV partners to make sure that the vscan engine has similar checks and precautions.

11 Vserver and multi-tenancy considerations

11.1 ONTAP

Off-box AV is intrinsically a vservers-scoped feature. The configuration in ONTAP always has a vservers qualifier, and the vscanner components always connect to vservers LIFs for the notification protocol and scan operations. In all cases, the vservers administrator has final say over the configuration of the off-box AV feature. This means that there is no provision for cserver administrators to disable off-box AV on a vservers, except when operating in a vservers admin role.

For convenience in a situation where a cluster has multiple vservers that are all in the same security domain, it is possible to set up a cluster-wide set of default scan policies and allowed IP addresses and login accounts. These can always be overridden by a vservers administrator, but for simplicity, this cserver config makes it easy to set up a single configuration that will apply to all vservers.

11.2 Vscanner

The configuration for the avshim includes a list of vservers and/or cserver management IP addresses which the avshim uses to get the list of data LIFs. This is done by issuing a "net-interface-get-iter" zapi query to get the list of data LIFs on all nodes, for just one vservers, in the case of vservers management IP addresses, or for an entire cluster, in the case of cserver management IP addresses.

This design supports only a single login account for all vservers that are handled by a given vscanner, so if true multi-tenancy is required, it will be necessary to set up separate vscanners for each security domain. We don't anticipate that this will be a severe problem, since even if we supported multiple security domains, most customers in a multi-tenancy setting would probably not want to mingle their data on a common vscanner, and in any case, each tenant network would likely be in a separate IP space.

If a vservers is migrated, the off-box AV configuration should migrate with it, though customers would likely have a different set of vscanners at the destination. The *ruleset* feature described later would facilitate doing this, by providing a mechanism for conditional enabling of a set of configurations (policies, allowed vscanners and login accounts) to be used when in the destination environment.

11.3 UI, vservers health

The vservers stop and start commands should impact clients identically, whether or not off-box vscan is enabled.

The overall health of a vservers can be greatly affected if the off-box vscan feature is not functioning properly. We will expose per-vservers counters and EMS messages which should definitely be incorporated into any vservers health monitoring packages. *I don't know what's involved with making that happen.*

12 Modularity considerations

12.1 Off-box components

All functions that are exposed to vscanners are over standard SMB connections. As mentioned above, the notification protocol is an extension of the MSRPC protocol that is used for 7-mode vscan engines, and it will be documented separately. The intra-vscanner communications are almost identical to the 7-mode vscan MSRPCs, which are documented in an SDK we provide to our AV partners. In this case, the avshim and vscan engine are actually separate executables, and the RPC API defines the interface between those modules.

12.2 N-blade components

There are two new components in the N-blade that comprise a sizeable fraction of the new work for the off-box vscan feature. These are the **Connection Manager**, and the **Dispatcher**. These components are suitable for use with any version of SMB, and should work with little or no modification for NFS traffic, as well as on-demand scanning, if/when those are implemented.

12.2.1 Connection Manager

This component is responsible for tracking the current configuration settings for each vservers. Accessing this information will normally only be necessary when the configuration changes, or when a new vscanner registers. The C++ class that implements this functionality exposes methods for querying and setting this information in a thread-safe manner, as it will be accessed and updated from different threads.

The Connection Manager also tracks the dynamic avshim connection state, as vscanners connect and disconnect from a vservers. This code runs in a specific N-blade context (module_id_RPC), so that these structures can be accessed without locks by the other primary N-blade component, Dispatcher.

12.2.2 Dispatcher

The Dispatcher handles routing of ops that are identified by the protocol code as needing scanning. Its methods run in a specific N-blade context (module_id_RPC), where the Connection Manager keeps the list of available vscanners. Dispatcher constructs a vscan request, which includes all the information needed to create the notification protocol message, and it sends this vscan request to the network context of the selected named pipe in order to perform the scan. The pending scan is placed in a per-network context list of pending requests. When the scan completes, the response will always return to the same context, so the request can be removed from proper pending list and the original message can be extracted from the scan request and sent to the appropriate network context to continue the request, this time with the scan status.

13 Use of architectural interfaces

- KSMF — used to communicate configuration information between blades and the M-host.
- SpinNp — modified to carry information needed for vscan operations. p4 change in SN.0
- ZAPI — used by the avshim to get the list of vservers data LIFs to connect to.

14 Resource Requirements

The resource requirements of this feature are modest, mostly consisting of data structures used to shepherd a request through the scan process. Here are the resources that will see an increased consumption when off-box AV is enabled:

- Each request that triggers a scan will require the allocation of one vscan structure which has all the state information needed to track the scan request and re-drive the original request. This structure is less than 2K, so if a node managed to get 1000 pending requests, it might consume ~2MB of memory for these structures.
- Notification traffic with the avshim is modest: typically on the order of 1K per request and perhaps 200 bytes per response. We have seen no more than 20% saturation of a single 1GB network connection during a maximum stress FSCT run. Traffic from the vscan engine to open, read, and close files to be scanned was of the same order.
- The vscan engine typically issues one or two open requests, one or two reads, and one or two close requests for each scan. The scan process itself is CPU-intensive, but this happens on the vscanner, and more vscanners can be added to increase performance.
- For SMB1, N-blade Exec contexts are used to track requests, including vscan requests if the vscanner is running Windows server 2003. We anticipate that extra pressure on Execs will require changes to ensure that NFS and non-AV-scan-related traffic is not unduly affected. The design we are using provides for this without allocating extra Execs for off-box AV unless they are needed. Details of this are in the design spec.
- There are a number of fairly small structures associated with each avshim connection and each vscan engine connection. These are allocated as normal CIFS structures. There would be one of each for each vserver on a node.

Remaining requirements for CPU and memory are very low, particularly in relation to those of onboard AV, as mentioned in the introduction.

- Essentially zero M-host CPU utilization
- Essentially zero M-host memory footprint
- Almost all vscan code runs in network exempt threads in the N-blade
- No use of Kahuna or Network Legacy domains

15 Feature Interaction Dependencies and Impacts

15.1 FlexCache interactions

- There are no on-disk changes, including metadir files, hence no effect on data transfers or version matching.
- There are no lock manager changes. The privileged accesses through `ontap_admin$` are identical to those of `fpolicy`.
- The off-box vscan configuration on the cache vserver need not be identical to the origin vserver. Hence, there is no need to replicate this configuration information.

15.2 MetroCluster / Vserver DR interactions

- Off-box AV will have no interaction until FS.0. However, once vserver dr is available,
 - The major issue that arises is the question of what configuration to use when switchover has

occurred. That is handled in this design by configuring vscanners in both environment, and allowing a scanner pool to detect which environment the vserver is in.

- We will need to make sure the scanner pools are evaluated in a timely way, to ensure that clients are able to access the vserver quickly after switchover/switchback.
- We plan for a future release to add the ability for an M-host to send a notification, over a secure channel, to the avshim of a standby vscanner so that it will attempt to register with a vserver. That would be ideal in this setting.

15.3 Snaplock interactions

- If a file is found to be infected, the vscanner will attempt to delete, disinfect, or quarantine the file. If that is disallowed by SnapLock, a message will be logged on the vscan engine, and the in-core inode will be updated to reflect that the file is infected, preventing further accesses or scans. However, the on-disk inode will never be modified by vscan operations.

15.4 HA

- The vscanner (avshim and vscan engine) have normal SMB connections with a node. If the node reboots, either for takeover or giveback, the vscanner will have to reconnect, which it will do. There are no special considerations for HA with this feature, beyond normal CIFS interactions.
- Any movement of data, disruptive or non-disruptive, will appear to the off-box vscan feature the same it would appear to any SMB client. No special considerations apply.
- There is no need to notify this facility of these data movement operations. The vscanner(s) will automatically disconnect/reconnect as needed.
- If there are slow pending AV scans, the normal CIFS shutdown logic will terminate the sessions for both avshim and vscan engine, without requiring the entire scan process to complete.
- When the CIFS server comes up on the partner, the configured vscanners should immediately connect, so this should not introduce any noticeable delay before CIFS client requests can be serviced. If there is a problem, and the vscanners do not connect right away, access would be granted or not, depending on the mandatory vscan setting. This is true any time, of course, not just after reboot in an HA situation.
- No new veto checks related to this feature.

15.5 NDU impact

- The external components (avshim, vscan engine) just connect and operate as CIFS clients, so the behavior during NDU is the same as what is seen by other CIFS connections.
- Off-box vscan cannot be enabled unless all nodes are running a version that supports it. This will be enforced using the capability manager.
- There are no on-disk changes, so there are no revert considerations, other than perhaps cleaning up the SMF tables.
 - For the first release, there is no plan to clean up these tables, as this would have to be done in SN.0, and at the time of this writing, a change like that would not be permitted under release guidelines.
- The avshim component can be updated at any time, but the vscanner will not be functional during that time. The same is true of a major update to the vscan engine that requires virus scanning to be disabled during the upgrade. For this reason, it will always be best practice to have more than one vscanner configured.

16 Test Strategies

- During development we have constructed a number of test setups that allow us to test different components separately. However these would not be applicable to actual testing, except for fault injection.
 - The "nvsc" fake vscan engine we have developed can create various scenarios, and additional scenarios could be developed for use by QA. Examples of fault injection using nvsc include
 - Failing to respond to a scan request
 - Sending multiple responses
 - Sending responses to requests that were never sent
 - Introducing arbitrary delays before responding
- It should be possible to adapt much of the existing 7-mode AV testing to make it applicable to clustered ONTAP.
- Security features need to be carefully tested,
 - to make sure that only configured users and IP addresses are capable of using the `vscan` and `ontap_admin$` objects.
 - to confirm that neither the avshim nor the vscan engine can be hijacked by other clients sending them forged MSRPC requests.
- It is specifically *not* required to demonstrate that other applications loaded on the vscanner cannot create insecurity.
- Performance testing must be done to confirm that the goals mentioned in the ERRD are met.
- We should stress the components that handle setup and teardown of avshim notification sessions by attaching many vservers and having them register and unregister repeatedly. Similarly, we should use nvsc to send a flood of version updates, to make sure we handle that properly.
- We will want to test with vscanners running on the following Windows versions: Windows 2008R2, and Windows 2012. As mentioned in the ERRD, Windows 2003 (SMB1) support was planned, if possible, but after consultation with product marketing, we decided not to support it.

17 Reliability

The off-box vscan feature directly affects the reliability and accessibility of the cluster for clients that create vscan requests. For this release, that includes only CIFS clients. Experience with the 7-mode vscan feature shows that failures and misconfigurations of off-box AV is one of the largest sources of disruptions and performance issues of customers who require AV scanning. This is a *major* pain point with our current customers, and relief of this pain is one of the most anticipated benefits that customers and support personnel expected from the onboard AV feature. Improving the reliability and management of off-box vscan has been very much front and center when developing the clustered ONTAP version of this feature.

Some of the problems are due to things that are difficult for us to address in the product, such as,

- Remote vscanners may be multiple hops away over not-always-reliable networks.
- These vscanners may be in a different data center, procured and maintained by one or more different teams, leaving our friend Stormy the storage administrator frustrated when things go wrong in an area he does not control.

But we *can* instrument things much better, and we can provide mechanisms to help alert administrators to problems before they get out of hand. Additionally, we can support increased redundancy, and increased flexibility of configurations, both of which should allow cluster and vserver administrators to set up the product so that it can more reliably and securely serve clients that require virus scanning. Some of these features are:

- An arbitrary number of vscanners can be configured to serve an arbitrary number of vservers. This is theoretically possible with 7-mode vscanners, but we will explicitly test this and ensure that adding vservers scales the performance as expected. (There is some empirical evidence that this does not work as well as expected with 7-mode vscanners.)
- Rulesets can be defined that allow flexibility in configuring which vscanners are to be used under what circumstances:
 - For example, a ruleset can enable a set of vscanners only when scan latencies of the normal vscanners exceeds a threshold. At other times, those vscanners would be serving some other group of vservers, or some other cluster. This allows handling a temporary increase in peak load.
 - For the first release, the only "rule" for rulesets we guarantee to deliver is a hard-coded "true". But in subsequent releases we can add rules such as the one described in the last point, as well as rules like the one described above in the Vserver DR section. And of course, if we can, we will add some rudimentary rules in the first release to enable behavior equivalent to the "primary/secondary" scanner distinction that is available with 7-mode.
- We will provide counters to track latency statistics for each scanner, and we have developed heuristics to favor faster vscanners, rather than the simple round-robin approach used by 7-mode. This is described in the design spec.
- Because this feature has high potential to affect the perceived health of a vserver, it will be important to provide proactive notification when we can detect that performance degradation is due to off-box vscan. **CIFS Health Monitor?** Overrunning the AV resources is a very common customer problem. We will provide SNMP traps and EMS notifications when scanning bogs down too much.
- We will provide console commands and ZAPIs to show all connected vscanners, and identify any displaying signs of overloading.
- We plan to avoid the resource issues that have plagued 7-mode vscan and fpolicy (for example, some of the issues described here). Those details are in the design spec.
- The avshim will report local vscanner conditions to ONTAP which will show when the vscan engine is overtaxed and is not keeping up with the arrival rate of new scan requests. This information will feed the counters and status information in the previously mentioned CLI and ZAPI commands.
- Any signs of network connectivity issues will be available from the CLI or ZAPI by examining the table of vscan connections for one or more vservers. This will help distinguish over-taxed vscanners from vscanners that are completely non-responsive.
- The scanner version map that is described in the design spec allows us to avoid clearing the scan state any time any vscanner is updated, and it also avoids having us traverse all the in-core inodes, looking for scan status bits to clear. This results in less impact when vscan version updates occur (which is often more than once a day).

In addition to the items mentioned above for detecting and reporting problems, we plan to implement heuristics for determining where to send scans, recovery when a vscanner starts misbehaving, etc. These will be documented in the design spec for this feature.

18 Availability and Non-Disruptive Operations (NDO)

The clustered ONTAP off-box vscan feature connects to external vscanners using the SMB protocol. That means the behavior is similar to that of other SMB clients. The avshim component on the vscanner is tasked with maintaining connections with the cluster based on its configuration, and it will attempt to connect to its vserver data LIFs until a successful connection is made. This connection should happen around the same time that other CIFS clients are connecting, so there should never be a lengthy delay, waiting for vscanners.

18.1 7-mode NDU and Cluster NDU (Rolling Upgrades)

- No on-disk structures are modified, so there is no impact from that source.
- The notification protocol is versioned, so after upgrade, the N-blade will continue to operate with prior versions of the avshim.
- Revert is not affected by this feature.
- The feature is not enabled until all nodes are updated. This is enforced by the capability manager.
 - It is worth noting that because SN.0 supports but never invokes the off-box vscan SpinNp changes, there should be no major issues, in principle, with running this feature on a cluster where some of the nodes are running an SN.x variant which does not support off-box AV (SN.0, say), and others do support it (SN.0P1, say). There are four cases:

	D-blade is up-rev	D-blade is down-rev
N-blade is up-rev	normal scanning	no scan D-blade ignores scan bits
N-blade is down-rev	no scan N-blade never sends scan bits	no scan N- and D-blades both down-rev

- We would need to vet this idea carefully, were we to be asked about the feasibility of doing something like this. In particular, there might be issues around having the new RDB tables present on down-rev nodes. And the down-rev nodes would not support the new ZAPIs and CLIs, so at best this would be a short-term transitional mode to facilitate rolling upgrades.

18.2 Unplanned Component Failures

This feature has high potential of introducing a single point of failure, for CIFS clients, at least. The most common way to create this type of failure would be through misconfiguration of vscanners. If no vscanners are available, but AV scanning has been marked as mandatory, no CIFS requests will be served. For this reason, it will be important for customers requiring mandatory scans to have multiple vscanners available.

Another source of problems is loss of network connectivity to an entire pool of vscanners, so if possible, redundant network paths could help avoid that. Alternate rulesets, as described elsewhere in this document, can be configured to automatically activate if one set of vscanners goes offline. This will also reduce the likelihood of vscan failures.

Also as mentioned elsewhere, we will provide detailed, timely messages to administrators to inform them of any issues related to this feature.

18.3 HA Considerations

As previously described, takeover and giveback would both require the vscanner to reconnect before virus scanning could occur. The avshim component of the vscanner is responsible for handling this, which it does by re-trying the connection until it is successful. If the avshim cannot connect, it will emit Windows events and logs, alerting administrators to this condition.

18.4 Maintenance Operations

This feature may require updates from time to time to the avshim component, but if the customer has followed best practices, there will be other vscanners to fall back on while this is done. Our AV partners will also have a similar requirement. Such updates should be very infrequent.

The vscanners themselves will require maintenance and upgrades, and again, if the customer has multiple vscanners, that will not unduly affect availability.

19 Supportability

Common failures, based on 7-mode experience, are expected to be things like

- The AV infrastructure is over-stressed. Scan times are long, and overall system performance is poor, from the point of view of CIFS clients. This might be due to things like
 - Changing application mix on the cluster.
 - If the vscanners are in virtual machines, changing loads on the host OS.
 - A sudden influx of difficult-to-scan files (e.g. a client copies in 1000 1GB zip files).
- Networking issues prevent the vscanners from reliably connecting to the cluster.
- Misconfigured vscanners fail to connect to the cluster, either because they are using the wrong account, or they are coming in from an IP address that is not configured on the cluster.
- A file is infected, but the end user does not realize that, and they continue attempts to copy it. This potentially creates a confused user, and a large amount of logging.
- Internal resource limitations on either the vscanner or in ONTAP cause scans to go slowly, or to lock up.

Features that assist with diagnosing and recovering from these problems include:

- Counters to track average latency and numbers of scans on various vscanners.
- EMS messages when vscan times exceed a threshold.
- EMS messages if a vscanner connection goes down because of networking problems.
- EMS messages if a vscanner voluntarily shuts down.
- EMS messages if a vscanner reports an infected file.
- CLI and ZAPI commands to view connection status and conditions on the vscanners, including things like avshim queue depths, and performance data (CPU utilization, memory utilization, and network utilization) gathered on the vscanner.

Standard supportability features will include

- AutoSupport
- Counters/stats
- EMS logging
- CIFS health monitor?

More advanced supportability features will include

- Node-scope debug counters visible from diag level.
- Tracing to log files. This will be an extension of the existing CIFS tracing mechanism.
- The FS and DS will describe the tables used to keep configuration information.

20 Scalability

A primary advantage of the off-box approach is the ability to scale CPU, memory, and network bandwidth as needed in order to support various numbers of users. As described elsewhere, this design has no pre-defined limits on the number of vscanners that can be supported.

Our experience with 7-mode vscanners, as well as the proof-of-concept that we implemented for clustered ONTAP, both show that performance does indeed scale when more vscanners are added. We will carefully measure and characterize the amount of scaling we see, both for capacity modeling, as well as for recommendations for best practices.

21 Open Issues

Item	Date	Name	Issue	Resolution	Date Resolved
1	Feb 26, 2013	Mark Muhlestein	Because this design has no mechanism for outbound connections to vscanners, changes to LIF configuration are not known on the vscanner. In some cases, that will result in a need for an administrator to restart and possibly reconfigure the avshim on one or more vscanners. We need a mechanism to do that automatically.	We will consider changes to the M-host which would allow it to contact the avshim component on a vscanner and give it new LIF configuration information any time a vscanner would be affected by the change.	Mar 1, 2013

22 Revision History

Version	Date	Name	Change
0.99	18 Feb 2013	Mark Muhlestein	Initial version adapted from ONTAP template.
1.0	28 Feb 2013	Mark Muhlestein	First review version.
1.1	22 Mar 2013	Mark Muhlestein	Incorporated review feedback. Also see talk page.

23 Approvals

These are in the official spec tracker (<http://engmetrics-web.eng.netapp.com/SpecTool/Specs.jsp?suiteId=1&docId=7143>)

NetApp Confidential - Limited Use

Sensitive business information which is intended for use within NetApp. Its unauthorized disclosure could adversely impact NetApp, its customers, its suppliers, its business partners, and/or its employees.



Retrieved from "https://wikid.netapp.com/wikid/index.php?title=CIFS/clustered_ONTAP/Off-box_AV/Architecture_Spec&oldid=6485976"

-
- This page was last modified on 10 November 2013, at 01:21.