



Contribute to NetApp technical content in GitHub

Contributor's Guide

NetApp
December 31, 2024

This PDF was generated from <https://docs.netapp.com/us-en/contribute/index.html> on December 31, 2024. Always check docs.netapp.com for the latest.

Table of Contents

- Contribute to NetApp technical content in GitHub 1
 - Overview 1
 - Request doc changes 1
 - Submit edits to the docs 4
- Style guide for NetApp docs 7
 - Write conversationally 7
 - Write simply 8
 - Write minimally 8
 - Write actively 9
 - Write consistent content 10
 - Use inclusive language 10
 - Get to the point 11
 - Use lots of visuals 11
 - Create scannable content 12
 - Create workflows that help users achieve their goal 12
 - Organize content based on the user's goal 12
 - Write for a global audience 13
 - A to Z guidelines 14
- AsciiDoc reference 22
 - The basics 22
 - Advanced stuff 26
- Policy forbidding the use of AI-generated content in NetApp technical documentation 33
- Legal notices 34
 - Copyright 34
 - Trademarks 34
 - Patents 34
 - Privacy policy 34

Contribute to NetApp technical content in GitHub

The documentation for NetApp's products and services is open source. This enables you to contribute to the content by making improvements, corrections, and suggestions. All you need is a GitHub account and a little initiative.

Overview

You can contribute to our docs by using the following options:

- Select **Request doc changes** to submit general feedback or to ask a question about the content. The NetApp content lead then reviews your request to determine what changes are needed to the docs. This is the most common option.

[View step-by-step instructions for this option.](#)

- Select **Edit this page** to directly edit the content yourself. The NetApp content lead then reviews your edits and merges them.

[View step-by-step instructions for this option.](#)

The following video provides a brief overview of these two options.

[Overview of contributing to docs.netapp.com](#)

The sections below provide step-by-step instructions.

Request doc changes

Submitting a request for a doc change is the most common way to contribute to NetApp docs. After you submit the request, the content lead will confirm that they've received your feedback. You'll get an email notification from GitHub when they do.

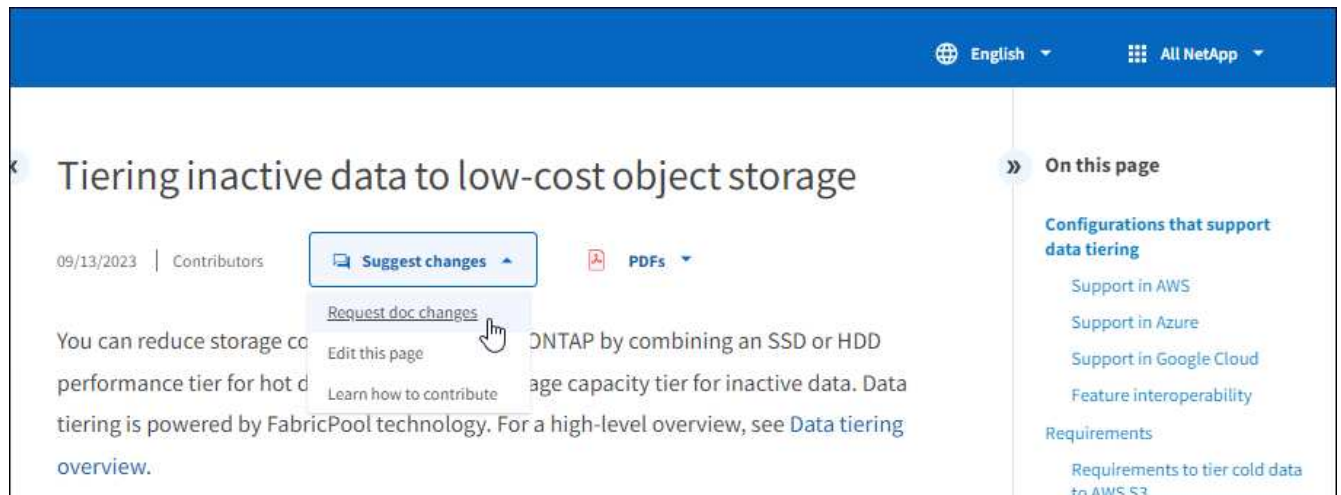
If the content lead agrees that your suggestion can make the content better, they will commit the change shortly after that. You'll receive another notification that your feedback was incorporated.



All comments that you provide are publicly visible. Anyone who navigates to the issues in the GitHub repo can see your comments.

Steps

1. If you don't already have a GitHub account, [create one from github.com](#)
2. Log in to your GitHub account.
3. Using your web browser, open the page on [docs.netapp.com](#) that's related to your feedback.
4. At the top of the page, select **Suggest changes > Request doc changes**.



A new browser tab opens with a GitHub form that you can use to provide details to our doc team.

5. Enter a title, summary, and acknowledge that the issue doesn't contain any sensitive information.

The form is prepopulated with the URL and title of the page. Don't delete this information because we'll need it to understand your request.

Broken link

Welcome and thank you for your contribution.

Anything that you enter here is publicly visible. If you need to provide sensitive information, you can email us at doccomments@netapp.com.

Page URL *

https://docs.netapp.com/us-en/test/task-tiering.html

Page title *

Tiering inactive data to low-cost object storage

Summary *

Required. Describe the problem.

Write

Preview

H B I E < >

The link to AWS documentation is broken.

Attach files by dragging & dropping, selecting or pasting them.

Public issues must not contain sensitive information

Required. Examples of sensitive information include but are not limited to customer information, employee information, unreleased features, release dates, or confidential product information.

☒ This issue contains no sensitive information.

Fields marked with an asterisk (*) are required.

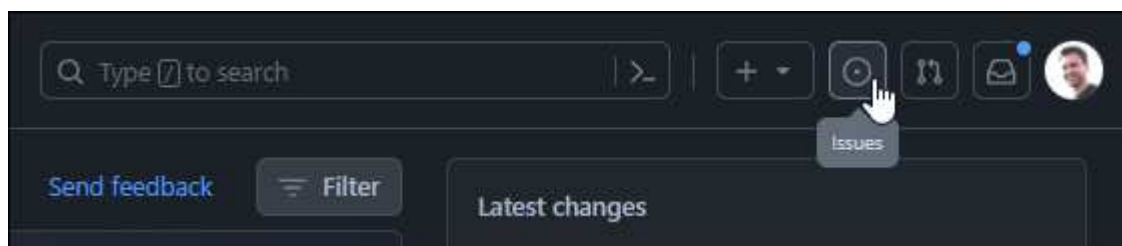
Submit new issue

6. Select **Submit new issue** to create an issue for your request.

After you're done

Opening an issue enables collaboration through GitHub comments. You'll receive email notifications based on the preferences that you indicated in your GitHub account settings.

You can also view the status of the request by selecting **Issues** from the GitHub banner:



Submit edits to the docs

If you're comfortable editing the content yourself, you can submit the exact doc changes that you'd like to see by directly editing the source file.

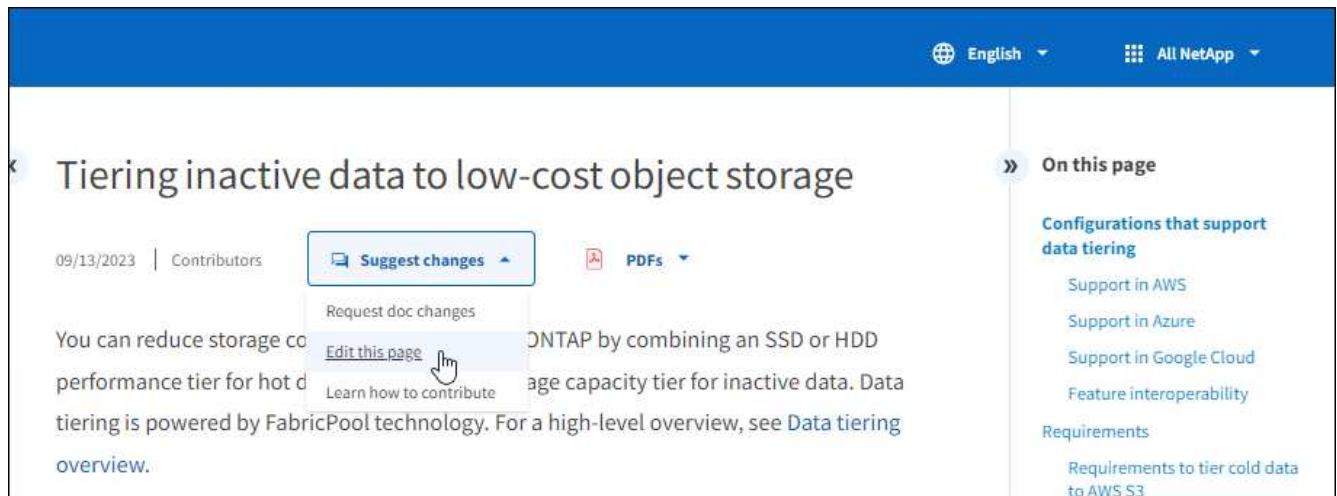
As an outside contributor, you won't be able to publish the change directly. The content lead will review the changes, make any required edits, and then merge the changes. You'll get an email notification from GitHub when this happens.

If you need help with our writing style or source syntax, you can use these resources:

- [Style guide for NetApp docs](#)
- [AsciiDoc syntax](#)

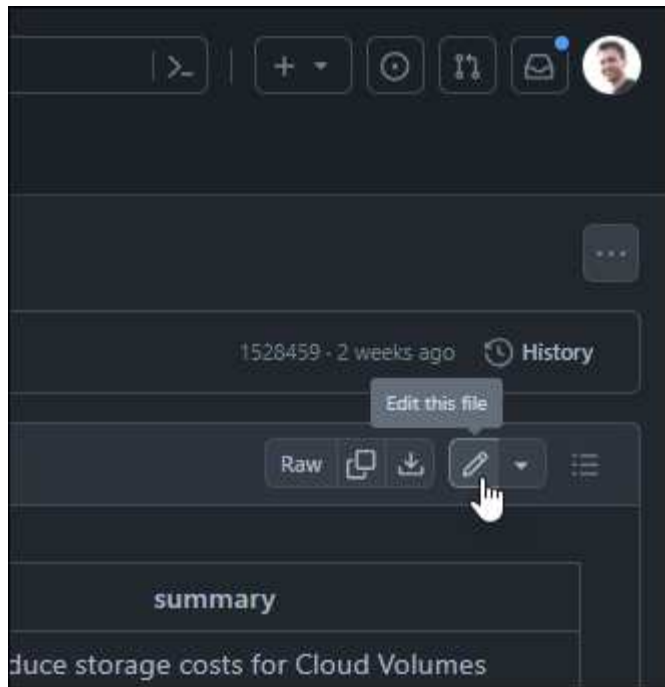
Steps

1. If you don't already have a GitHub account, [create one from github.com](#)
2. Log in to your GitHub account.
3. Using your web browser, open the page on [docs.netapp.com](#) that you want to edit.
4. At the top of the page, select **Suggest changes > Edit this page**.



A new browser tab opens and brings you to the file in the GitHub repository for the documentation site.

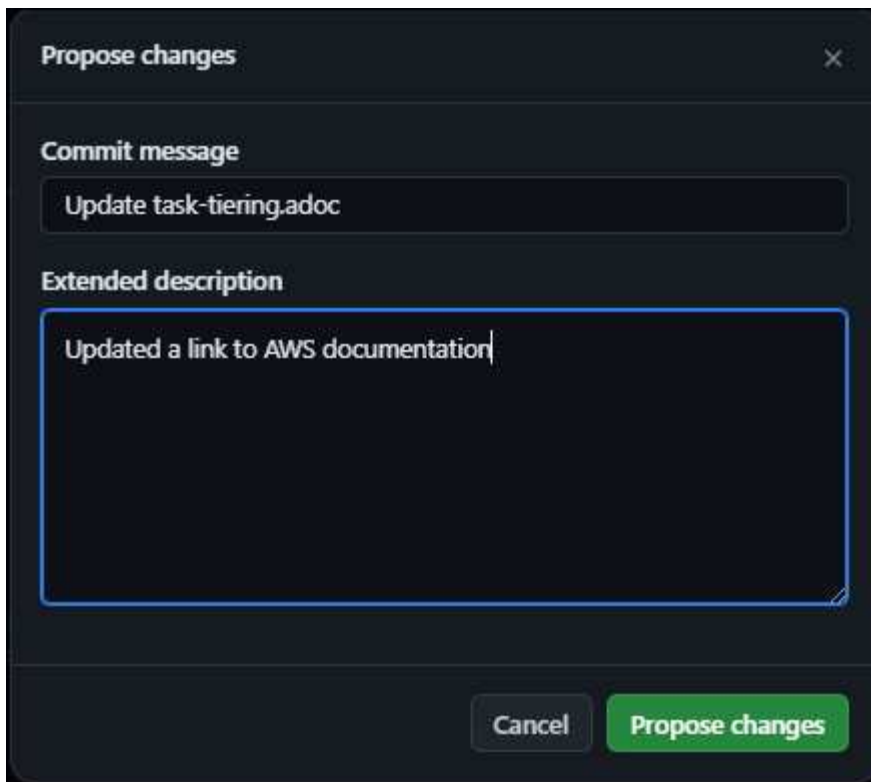
5. Select the pencil icon.



6. If you're prompted to create a fork of the repository, select **Fork this repository**.
7. Edit the content.

The content is written in AsciiDoc, a lightweight markup language. [Learn about AsciiDoc syntax](#).

8. To commit your changes, select **Commit changes** and fill out the form:
 - a. Optionally modify the default commit message.
 - b. Add an optional description.
 - c. Select **Propose changes**.



Propose changes [X]

Commit message

Update task-tiering.adoc

Extended description

Updated a link to AWS documentation

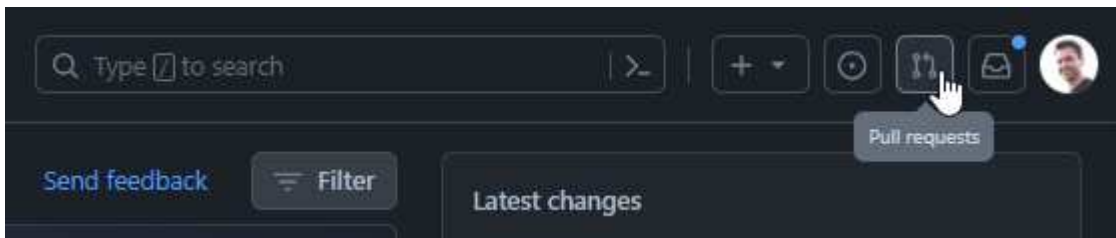
Cancel Propose changes

9. Select **Create pull request**.

After you're done

After you propose the changes, we'll review them, make any necessary edits, and then merge the changes into the GitHub repository.

You can view the status of the pull request by selecting **Pull requests** from the GitHub banner:



Style guide for NetApp docs

Our style is conversational and empathetic, but we stay professional and get to the point. Follow these guidelines when writing content for NetApp docs.

Write conversationally

Write like you speak when you're explaining something to a professional colleague. Writing in a conversational tone helps you connect with your audience.

Too formal	Too casual	Our style
If you experience issues with BlueXP tiering, you can view the health status on the Cluster Dashboard to determine why the error occurred. The health reflects the status of the ONTAP system and the Service Connector.	It's no fun when failures happen. Don't worry though, just check out the Cluster Dashboard to see what happened and what's affected.	When there's a failure, BlueXP tiering displays a "Failed" health status on the Cluster Dashboard. View the status to get information about the failure.
Before you can provision storage, you must discover your ONTAP cluster from BlueXP. After discovery is complete, you must open the working environment to provision storage.	All you do is first discover your ONTAP cluster from BlueXP, then open the working environment to start provisioning storage. Easy!	After you discover your ONTAP cluster from BlueXP, open the working environment to provision storage.
When the Setup wizard starts, follow the instructions in the wizard to set up the node and join it to the cluster. The following steps walk you through the steps in the wizard.	The Setup wizard appears (almost like magic), to guide you through the simple process of setting up the node and joining it to the cluster.	Follow the instructions in the Setup wizard to set up the node and join it to the cluster.
You can choose from among multiple content format types to install and set up your new system. Each format type provides complete instructions. Choose the format type that most closely matches the way you like to learn.	How do you want to set up and install your system? We provide instructions in multiple content format types, but only you know how you like to learn.	Choose a content format type to guide you through installing and setting up your system.

Use contractions

Contractions reinforce a [conversational tone](#), and many contractions are easy to understand and translate.

Expand to learn more

- Use contractions like these, which are easy to understand and translate:

aren't	you're
isn't	we're
wasn't	it's
weren't	let's
didn't	we'll (if future tense is required)
doesn't	won't (if future tense is required)
don't	you'll (if future tense is required)

- Don't use contractions like these, which are hard to understand and translate:

would've	should've
wouldn't've	shouldn't've
could've	couldn't've

Write simply

Avoid big and confusing words. Keep it simple. You're explaining something to a professional colleague, not showing off your vocabulary.

Rather than this: "Dissociate the user from your NetApp Cloud Central account."

Do this: "Remove the user from your NetApp Cloud Central account."

Write minimally

Short, simple sentences make content easier to read or scan. It's okay to use a longer sentence every now and then but follow it with a shorter one. Like this.

Rather than this: "To replicate data between a Cloud Volumes ONTAP system in AWS and ONTAP systems in other networks, you must have a VPN connection between the Amazon VPC and the other network—for example, an Azure VNet or your corporate network."

Do this: "Data replication between networks requires connection through a VPN. For example, between your Amazon VPC and your corporate network or between AWS and Azure."

Ask yourself the following:

- Do users need this content at this place, at this time?
- Does the user interface already guide the user well enough? If it doesn't, what additional guidance can be added succinctly?
- Can I present the content in fewer words without sounding too formal or too casual?

- Can I shorten or simplify a long sentence or break it into two or more sentences?
- Can I use a list to make the content more scannable?
- Can I use a graphic to augment or replace a block of text?

Write actively

Avoiding passive voice is a standard rule for tech writing, but it's especially important to use active voice when you want to sound conversational.

Active voice

Use active voice so that the subject of the sentence performs the verb's action. This typically means that the verb follows the subject of the sentence. Active voice keeps writing crisp and clear. Use active voice and address users directly as "you" unless you have a specific reason to use passive voice.

Here are some examples of good active voice writing. Write like this:

- Provide the required permissions before you deploy your first cluster.
- If you shut down the system improperly the interface displays a warning message.
- NetApp received the contract.

Passive voice

In passive voice, the doer of the action is unclear:

- A warning message is displayed if the system is shut down improperly.
- NetApp was awarded the contract.

Use passive voice when:

- You don't know who or what performed the action.
- You want to avoid blaming users for the results of an action.
- You can't write around it, such as for some prerequisite information.

Imperative mood

Use imperative mood for steps, directives, requests, and headings for lists of user actions:

- "On the Working Environments page, click Discover and select ONTAP Cluster."
- "Rotate the cam handle so that it is flush against the power supply."

Consider using imperative voice to replace passive voice:

Rather than this: "The required permissions must be provided before you deploy your first cluster."

Do this: "Provide the required permissions before you deploy your first cluster."

Avoid using imperative voice to embed steps in conceptual and reference information.

For additional verb conventions, see:

- [Microsoft Writing Style Guide](#)
- [Merriam-Webster Dictionary Online](#)

Write consistent content

"Write like you speak when you're explaining something to a professional colleague" means something different to everyone. Our professional yet conversational style helps connect us to users and increases the frequency of minor inconsistencies among multiple contributing authors:

- Focus on making the content clear and easy to use. If all content is clear and easy to use, minor inconsistencies don't matter.
- Be consistent within the page you're writing.
- Always follow the guidelines in [Write for a global audience](#).

Use inclusive language

NetApp believes that its product documentation should not contain discriminatory, exclusive language. The words that we use can make a difference between forging a positive relationship with our customers or alienating them. Especially with written words, impact is more important than intent.

As you create content for NetApp products, avoid language that can be interpreted as degrading, racist, sexist, or otherwise oppressive. Instead, use language that is accessible and welcoming to everyone who needs to use the documentation. For example, instead of "master/slave" use "primary/secondary."

Use people-first language where we refer first to the person, followed by the disability.

Don't use he, him, his, she, her, or hers in generic references. Instead:

- Rewrite the sentence to use the second person (you).
- Rewrite the sentence to have a plural noun and pronoun.
- Use "the" or "a" instead of a pronoun (for example, "the document").
- Refer to a person's role (for example, reader, employee, customer, or client).
- Use the term "person" or "individual".

Examples of words and phrases that are considered inclusive or exclusive

Inclusive examples	Exclusive examples
Primary/secondary	Master/slave
Allowed list	Whitelist
Blocked list	Blacklist
Stop	Kill
Stop responding	Hang
End or Cancel	Abort
Person hour	Man hour

Inclusive examples	Exclusive examples
Developers need access to servers in their development environments, but they don't need access to the servers in Azure.	A developer needs access to servers in his development environment, but he doesn't need access to servers in Azure.
Person who is blind	Sight-impaired
Person with low vision	Vision-impaired

Get to the point

Each page should start with what's most important to the user. We need to find out what the user is trying to do and focus on helping them achieve that goal. We should also add keywords at the beginning of the sentence to improve scan ability.

Follow these general sentence guidelines:

- Be precise.
- Avoid filler words.
- Be short.
- Use formatted text or bulleted lists to highlight key points.

Examples of getting to the point

Good examples	Bad examples
If your business has strict security policies, use data-in-flight encryption to sync data between NFS servers in different networks.	Cloud Sync can sync data from one NFS server to another NFS server using data-in-flight encryption. Encrypting the data can help if you have strict security policies for transferring data over networks.
Save time by creating a document template that includes the styles, formats, and page layouts you use most often. Then use the template whenever you create a new document.	Templates provide a starting point for creating new documents. A template can include the styles, formats, and page layouts you use frequently. Consider creating a template if you often use the same page layout and style for documents.
Astra Control provides three operational modes that you can assign to your users to carefully control access between Astra Control and your cloud environment.	Astra Control enables you to assign one of three operational modes for users in your AWS accounts. The modes allow you to carefully control access between Astra Control and your cloud estate based on your IT policies.

Use lots of visuals

Most people are visual learners. Use videos, diagrams, and screenshots to improve learning, break up blocks of text, and provide a visual cue to users as to where they are in the task instructions.

- Include a lead-in sentence that describes the image that follows: "The following illustration shows the AC power supply LEDs on the back panel."
- Refer to the location of the illustration as "following" or "preceding," not "above" or "below."

- Use alt text on embedded visuals.
- If the visual pertains to a step, include the visual right after the step and indented to align with the step number.

Best practices on screenshots:

- Include no more than 5 screenshots per task.
- Don't include text in a screenshot. Use numbered callouts instead.
- Be judicious with the screenshots you choose to include. Screenshots can go out of date quickly.

Best practices on videos or animations:

- Videos should be under 5 minutes in length.

Examples

- [Example #1 Learn about AWS credentials and permissions](#)
- [Example #2 Protect your ONTAP volume data with BlueXP backup and recovery](#)
- [Example #3 Create a replication plan \(shows screenshots in a task\)](#)
- [Example #4 Manage credentials in BlueXP video](#)

Create scannable content

Help readers find content quickly by organizing text under section headings and by using lists and tables. Headings, sentences, and paragraphs should be short and easy to read. The most important information should be provided first.

Examples

- [Example #1](#)
- [Example #2](#)

Create workflows that help users achieve their goal

Users read our content to accomplish a specific goal. Users want to find the content they need, accomplish their goals, and go home to their families. Our job is not to document products or features. Our job is to document user goals. Workflows are the most direct way to help users accomplish their goals.

A workflow is a series of steps or subtasks that describes how to achieve a user goal. The scope of a workflow is a complete goal.

For example, the steps to create a volume would not be a workflow, because creating a volume in itself is not a complete goal. The steps to make storage available to an ESX server could be a workflow. The steps would include not only creating a volume, but exporting the volume, setting any necessary permissions, creating a network interface, and so on.

Workflows are derived from customer use cases. A workflow shows only the one best way to achieve the goal.

Organize content based on the user's goal

Help users find information quickly by organizing content based on the goal that the user is trying to achieve.

This standard applies to the table of contents (navigation) for a documentation site, as well as the individual pages that appear on the site.

Organize content as follows:

The first entry in the left-hand navigation (high level)

Organize content around the goals that the user is trying to achieve. For example, the first entry in the navigation for the site might be "Get started" or "Protect data."

The second-level entries in the navigation for the documentation site (medium level)

Organize content around the broad tasks that compose the goals.

For example, the "Get started" section might include the following pages:

- Prepare for installation
- Install and set up <product name>
- Set up licensing
- What you can do next

Individual pages (detailed level)

On each page, organize the content around the individual tasks that compose the broad tasks. For example, the content that users need to prepare for installation or to set up disaster recovery.

A page can describe a single task or multiple tasks. If there are multiple tasks, they should be described in separate sections on the page. Each section should focus on a single learning or doing aspect of the broad task. This might include some conceptual and reference-based information that's required to complete the task.

Write for a global audience

Our documentation is read by many users whose primary language isn't English. We translate our content into other languages using Neural Machine Translation tools or human translation. To support our global audience, we write content that is easy to read and easy to translate.

Follow these guidelines to write for a global audience:

- Write short, simple sentences.
- Use standard grammar and punctuation.
- Use one word for one meaning and one meaning for one word.
- Use common contractions.
- Use graphics to clarify or replace text.
- Avoid embedding text in graphics.
- Avoid having three or more nouns in a string.
- Avoid unclear antecedents.
- Avoid jargon, colloquialisms, and metaphors.
- Avoid nontechnical examples.
- Avoid using hard returns and spacing.

- Don't use humor or irony.
- Don't use discriminatory content.
- Don't use gender-biased language unless you're writing for a specific persona.

A to Z guidelines

acronyms and abbreviations

Use well-known acronyms and abbreviations for familiarity but avoid obscure ones that might negatively affect clarity and findability. For additional conventions for acronyms and abbreviations, see the [Microsoft Writing Style Guide](#).

active voice (versus passive voice)

Refer to [Write actively](#).

admonitions

Admonitions are a powerful tool when used correctly. They can draw attention to important information, provide helpful tips, or warn users about potential hazards. When overused, they lose their impact and can lead to user fatigue. Here are some guidelines to ensure the effective use of admonitions.

Standard admonitions

Three standard admonitions use custom labels. The labels are NOTE, TIP, and CAUTION. These three standard admonitions are formatted distinctly from normal text, and their labels are always written in uppercase in the AsciiDoc source.

- **NOTE**
Use NOTE to highlight important information that must stand out from the rest of the text. However, avoid using NOTE for "nice to know" information that isn't essential for users to understand or complete a task. The purpose of a NOTE is to draw the reader's attention to critical points that they might otherwise overlook.
- **TIP**
Use TIP to provide useful advice or shortcuts that can enhance the user's experience. For example, a TIP might help a user complete a step or task more easily and efficiently. A TIP should be used sparingly, if at all, as our policy is to document the best way to complete a task by default.
- **CAUTION**
Use CAUTION to warn users about conditions or actions that can lead to undesirable outcomes, including personal injury or damage to equipment. CAUTION should be used to draw attention to potential hazards that the user must avoid to prevent harm or disruption.

Best practice admonition

The Best practice admonition is not a custom admonition label but can be used as a standalone formatting convention. Use Best practice to highlight optimal ways to complete tasks or use a product. These are not mere suggestions, but strategies that have been validated by experts or industry standards.

- **What makes a Best practice?**

It's an actionable, task-specific strategy that offers clear benefits and is backed by reliable sources.

- **When can I use Best practice?**

You can use Best practice for all content types and all audiences. Like TIPS, use them sparingly to maintain their significance.

- **How do I format Best practice?**

To use the Best practice format, apply sentence-style capitalization, boldface the term Best practice, followed by a colon and a space.

Present Best practice in a consistent, user-friendly format. This could be a bulleted list, numbered list, or paragraph, depending on the context. For example, **Best practice**: Always test your configuration changes in a staging environment before applying them to production.

Additional Guidelines

- Only use supported admonitions. Any other kind of formatting is not supported.
- Avoid overusing admonitions. Overuse can lead to users skipping over these important sections because they see them as the "junk drawer" of our docs.
- As a rule of thumb, limit the number of admonitions to a maximum of 3 per page.
- Provide clear and concise information within the admonition. The message should be brief and to the point, allowing users to quickly understand the importance of the information provided.
- Avoid AsciiDoc admonitions in a table. If content needs to be identified as a note, tip, or caution, use Note:, Tip:, or Caution: as an inline lead-in to the text.

after (versus "once")

- Use "after" to indicate a chronology: "Turn on your computer after you plug it in."
- Use "once" only to mean "one time."

also

- Use "also" to mean "additionally."
- Don't use "also" to mean "alternatively."

and/or

Choose the more precise term if there is one. If neither term is more precise than the other, use "and/or."

as

Don't use "as" to mean "because."

by using (versus "using" or "with")

- Use "by using" when the entity that is doing the using is the subject: "You can add new components to the repository by using the Components menu."
- You can begin a sentence with either "using" or "with," which are sometimes acceptable with product names: "Using SnapDrive, you can manage virtual disks and Snapshot copies in a Windows environment."

can (versus "might," "may," "should," or "must")

- Use "can" to indicate capability: "You can commit your changes at any time during this procedure."
- Use "might" to indicate possibility: "Downloading multiple programs might affect processing time."
- Don't use "may," which is ambiguous because it could mean either capability or permission.
- Use "should" to indicate a recommended but optional action. Consider using an alternative phrase instead, such as "we recommend."
- Avoid using "must" because it's [passive](#). Consider restating the thought as an instruction using imperative voice. If you use "must," use it to indicate a required action or condition.

capitalization

Use sentence-style capitalization (lowercase) for almost everything. Only capitalize:

- The first word of sentences and headings, including table headings
- The first word of list items, including sentence fragments
- Proper nouns
- Doc titles and subtitles (capitalize all major words and prepositions of five or more letters)
- UI elements, but only if they are capitalized in the interface. Otherwise, use lowercase.

caution notices

Refer to [admonitions](#).

contractions

Use [contractions](#) as part of writing conversationally.

ensure (versus "confirm" or "verify")

- Use "ensure" to mean "to make certain." Include "that," as appropriate: "Ensure that there is sufficient white space around illustrations."
- Never use "ensure" to imply a promise or guarantee: "Use Cloud Manager to ensure that you can provision NFS and CIFS volumes on ONTAP clusters."
- Use "confirm" or "verify" when you mean that the user should double-check something that already exists or has happened already: "Verify that NFS is set up on the cluster."

graphics

Refer to [Use lots of visuals](#).

grammar

Except where noted otherwise, follow the grammar, punctuation, and spelling conventions detailed in:

- [Microsoft Writing Style Guide](#)
- [Merriam-Webster Dictionary Online](#)

if not

Don't use "if not" by itself to refer to the previous sentence:

- **Rather than this:** "The computer should be off. If not, turn it off."
- **Do this:** "Verify that the computer is off."

if (versus "whether" or "when")

- Use "if" to indicate a condition, such as in "if this, then that" constructions.
- Use "whether" when there is a stated or implied "or not" condition. To ease translation, it is often best to replace "whether or not" with "whether" alone.
- Use "when" to indicate a passage of time.

imperative voice

Refer to [Write actively](#).

future functionality or releases

Don't refer to the timing or content of upcoming product releases or features, other than to say that a feature or function is "not currently supported."

KB articles: referring to

Refer to KB (NetApp Knowledgebase) articles in content when appropriate. For resources pages and GitHub content, put the link in running text.

lists

Lists of info are usually easier to scan and absorb than blocks of text. Consider ways to simplify complex info by presenting it in list form. Here are some general guidelines, but use your judgment:

- Make sure that the reason for the list is clear. Introduce the list with a complete sentence, a sentence fragment with a colon, or a heading.
- When using a list inside a list, limit the structure to at most two levels of depth to maintain clarity and readability. If you find yourself needing more levels, consider reorganizing the content to make it easier for users to navigate and understand.
- Any list, including nested lists, should have between two and seven entries. In general, the shorter the info in each entry, the more entries you can add while keeping the list scannable. If a list has multiple entries that contain nested lists, consider using sections or block titles to divide the entire thing into more consumable chunks.
- List entries should be as scannable as possible. Avoid blocks of text that get in the way of keeping list entries scannable.
- List entries should start with a capital letter, and list entries should be grammatically parallel. For example, start each entry with a noun or a verb:
 - If all list entries are complete sentences, end them with periods.
 - If all list entries are sentence fragments, don't end them with periods.

- List entries should be ordered in a logical way, such as alphabetically or chronologically.

localization

Refer to [Write for a global audience](#).

minimalism

Refer to [Write minimally](#).

numbers

- Use Arabic numerals for 10 and all numbers greater than 10, with these exceptions:
 - If you begin a sentence with a number, use a word, not an Arabic numeral.
 - Use words (not numerals) for approximate numbers.
- Use words for numbers that are less than 10.
- If a sentence contains a mixture of numbers less than 10 and greater than 10, use Arabic numerals for all numbers.
- For additional number conventions, see [Microsoft Writing Style Guide](#).

plagiarism

We document NetApp products and the interaction of NetApp products with third-party products. We do not document third-party products. We should never need to copy and paste third-party content into our docs and we should never do it.

prerequisites

Prerequisites identify the conditions that must exist or the actions that users must have completed before they start the current task.

- Identify the nature of the content with a heading, such as "Prerequisites," "Before you begin," or "Before you get started."
- Use passive voice for prerequisite wording if it makes sense to do so:
 - "NFS or CIFS must be set up on the cluster."
 - "You must have the cluster management IP address and the password for the admin user account to add the cluster to Cloud Manager."
- Clarify the prerequisite as needed: "NFS or CIFS must be set up on the cluster. You can set up NFS and CIFS using System Manager or the CLI."
- Consider other ways to present the information, for example whether it would be appropriate to reword the content as the first step in the current task:
 - Prerequisite: "You must have the required permissions before you deploy your first cluster."
 - Step: "Provide the required permissions to deploy your first cluster."

prior (versus "before," "previous," or "preceding")

- If possible, replace "prior" with "before."

- If you can't use "before," use "prior" as an adjective to refer to something that occurred earlier in time or with a higher order of importance.
- Use "previous" to indicate something that occurred at an unspecified time earlier.
- Use "preceding" to indicate something that occurred immediately beforehand.

punctuation

Keep it simple. In general, the more punctuation included in a sentence, the more brain cells it takes to understand.

- Use a serial comma (Oxford comma) before the conjunction ("and" or "or") in a narrative list of three or more items.
- Limit use of semicolons and colons.
- Except where noted otherwise, follow the grammar, punctuation, and spelling conventions detailed in:
 - [Microsoft Writing Style Guide](#)
 - [Merriam-Webster Dictionary Online](#)

since

Use "since" to indicate a passage of time. Don't use "since" to mean "because."

spelling

Except where noted otherwise, follow the grammar, punctuation, and spelling conventions detailed in:

- [Microsoft Writing Style Guide](#)
- [Merriam-Webster Dictionary Online](#)

that (versus "which" or "who")

- Use "that" (without a trailing comma) to introduce clauses that are required for the sentence to make sense.
- Use "that" even if the sentence is clear in English without it: "Verify that the computer is off."
- Use "which" (with a trailing comma) to introduce clauses that add supporting information but are not required for the sentence to make sense.
- Use "who" to introduce clauses referring to people.

trademarks

We don't include trademark symbols in most of our technical content because the legal statements in our templates are sufficient. However, we do follow all usage rules when using [NetApp trademarked terms](#):

- Use trademarked terms (with or without the symbol) only as adjectives, never as nouns, verbs, or verbals.
- Don't abbreviate, hyphenate, or italicize trademarked terms.
- Don't pluralize trademarked terms. If a plural form is required, use the trademarked name as an adjective that modifies a plural noun.
- Don't use a possessive form of a trademarked term. You can use the possessive form of company names,

such as NetApp, when the names are being used in a general sense, rather than as trademarked terms.

user interface

When you are documenting a user interface, rely on the interface as much as possible to guide the user.

General guidelines

Use a simple and minimal style when documenting UIs.

- Assume that the user is using the interface while reading the content:
 - Don't walk the user through a wizard or screen step by step. Only call out important things that are not apparent from the interface.
 - Don't include "click OK" or "click Save" or "the volume is created" or anything else that's obvious to someone doing the task.
 - Assume success. Unless you expect an operation to fail most of the time, do not document the failure path. Assume that the interface provides proper guidance.
- Don't use "click" at all. Always use "select" because that word covers mouse, touch, keyboard, and any other way of making a choice.
- Focus content on a workflow that addresses a customer use case and on getting the user to the right place in the interface to start the workflow.
- Always document the one best way to achieve the user goal.
- If the workflow requires a significant decision, make sure to document a decision rule.
- Use the minimum number of steps necessary for most users most of the time.

Naming UI elements

Avoid documenting to the level of granularity that requires naming UI elements.

Rely on the interface to guide the user through the specifics of the interaction. If you must get that specific, name the label on the element. For example, "Select the desired volume" or "Select 'Use existing volume'." There is no need to name menus or radio buttons or checkboxes, just use the label.

For icons that users must select, use an image of the icon. Don't try to name it. This rule applies to icons like the arrow, pencil, gear, kabob, hamburger, and so on.

Representing displayed labels

Follow the spelling and capitalization used by the user interface when identifying labels.

If a label is followed by ellipses, do not include the ellipses when naming the object. Encourage developers to use title-style capitalization for user interface labels, to make writing about them easier.

Using screen captures

Use screen captures sparingly.

An occasional screen capture ("screenshot") helps users be confident that they are in the right place in an interface when starting or changing interfaces during a workflow. Don't use screen captures to show what data to enter or what value to select.

while (versus "although")

- Use "while" to indicate something occurring in time.
- Use "although" to represent an activity that occurs at nearly the same time or shortly after another activity.

AsciiDoc reference

AsciiDoc is a lightweight markup language, similar to Markdown. We chose AsciiDoc over standard Markdown because it provides more out-of-box capabilities. While it's more powerful, it's still simple to use. Refer to the sections below to get started writing in AsciiDoc.

See the [AsciiDoctor User Manual](#) for additional help.

The basics

You need to know a few things to contribute simple doc updates.

Headings

```
= Page title
== Level 1 section
=== Level 2 section
==== Level 3 section
===== Level 4 section
```

You can have only one page title, but you can have multiple section titles. For example, you might have three level 1 sections that include level 2 and 3 sections:

```
= Page title
== Level 1 section
=== Level 2 section
== Level 1 section
== Level 1 section
=== Level 2 section
==== Level 3 section
```

Bold text

```
*Text*
```

Italic text

```
_Text_
```


Bulleted lists

```
* Item 1
+
Continuation text for the previous list item.

* Item 2
** Item 2a

* Item 3
```



The + is a list continuation. It keeps the text inline with the list item. Omitting the + affects the formatting of that line.

Labeled lists

```
Item 1::
Description 1

Item 2::
Description 2
```

or

```
[horizontal]
Item 1::
Description 1

Item 2::
Description 2
```

When you add [horizontal] above item 1, the label and description appear on the same line. That works well when you have very short descriptions.

Example without [horizontal]

Item 1

Description 1

Item 2

Description 2

Example with [horizontal]

Item 1 Description 1

Item 2 Description 2

Steps

```
.Steps

. Step 1

. Step 2
+
Info for step 2

. Step 3
.. Step 3a
.. Step 3b

. Step 4
```



The + is a list continuation. It keeps the text inline with the list item. Omitting the + affects the formatting of that line.

Images

```
image:file.png["alt text"]
```

alt text means alternative text. It describes the image that appears on the page. The primary use is for visually-impaired users who use screen readers.

Two notes:

1. It's best to enclose alt text in quotes because punctuation like commas can affect the ability to transform the content from AsciiDoc to HTML.
2. The [AsciiDoctor docs](#) state that *block images* should be on their own line with *two* colons:
`image::file.png`

But we prefer to use one colon, as shown above. Using one colon has the same result and it works better with our internal tools.

Videos

Hosted on YouTube:

```
video::id[youtube]
```

Hosted locally in GitHub:

```
video::https://docs.netapp.com/us-en/contribute//media/file.mp4
```

Links

The syntax that you should use depends on what you're linking to:

- [Link to an external site](#)
- [Link to a section on the same page](#)
- [Link to another page in the docs](#)

Link to an external site

```
url[link text^]
```

The ^ opens the link in a new browser tab.

Link to a section on the same page

```
<<section_title>>
```

For example:

```
For more details, see <<Headings>>.
```

The link text can be something other than the section title:

```
<<section_title,Different link text>>
```

For example:

```
<<Headings,Learn the syntax for headings>>.
```

Link to another page in the docs

The file needs to be in the same GitHub repository:

```
xref:{relative_path}<file_name>.html[Link text]
```

To link directly to a section in the file, add a hash (#) and the section's title:

```
xref:{relative_path}<file_name>.html#<section-name-using-dashes-and-all-lower-case>[Link text]
```

For example:

```
xref:{relative_path}style.html#use-simple-words[Use simple words]
```

Notes, tips, and cautions

You might want to draw attention to certain statements by using notes, tips, or caution statements. Format them as follows:

NOTE: text

TIP: text

CAUTION: text

Use each of these sparingly. You don't want to create pages that are full of notes and tips. They become less meaningful if you do.

Here's what each of these looks like when the AsciiDoc content is turned into HTML:



This is a note. It includes extra info that a reader might need to know.



A tip provides useful information that can help a user do something or understand something.



A caution advises the reader to act carefully. Use this in rare circumstances.

Advanced stuff

If you're authoring new content, you'll want to review this section for some nitty-gritty details.

Document headers

Each AsciiDoc file includes two types of headers. The first is for GitHub and the second is for AsciiDoctor, which is the publishing tool that turns the AsciiDoc content into HTML.

The GitHub header is the very first set of content in the .adoc file. It needs to include the following:

```
---
sidebar: sidebar
permalink: <file_name>.html
keywords: keyword1, keyword2, keyword3, keyword4, keyword5
summary: "A summary."
---
```

The keywords and summary directly affect search results. In fact, the summary itself displays in the search results. You should make sure that it's user friendly. The best practice is to have the summary mirror your lead paragraph.



It's best to enclose the summary in quotes because punctuation like colons can affect the ability to transform the content from AsciiDoc to HTML.

The next header goes directly underneath the document title (see [Headings](#)). This header should include the following:

```
:hardbreaks:
:icons: font
:linkattrs:
:relative_path: ./
:imagesdir: {root_path}{relative_path}./media/
```

You won't need to touch any of the parameters in this heading. Just paste it in and forget it.

Lead paragraph

The first paragraph that appears under the document title should include the following syntax directly above it:

```
[.lead]
This is my lead paragraph for this content.
```

[.lead] applies CSS formatting to the lead paragraph, which has a different format than the text that follows it.

Tables

Here's syntax for a basic table:

```
[cols=2*,options="header",cols="25,75"]
|===
| heading column 1
| heading column 2
| row 1 column 1 | row 1 column 2
| row 2 column 1 | row 2 column 2
|===
```

There are *many* ways to format a table. Refer to the [AsciiDoctor User Manual](#) for additional help.



If a cell contains formatted content like bulleted lists, it's best to add an "a" in the column header to enable formatting. For example:
[cols="2,2,4a" options="header"]

See the [AsciiDoc Syntax Quick Reference](#) for more table examples.

Task headings

If you're explaining how to perform a task, you might include introductory information before you get to the steps. And you might need to say what to do after completing the steps. If you do, it's best to organize that information using headers, which enables scanning.

Use the following headings as needed:

What you'll need

The information the user needs to complete the task.

About this task

Some extra contextual info the user might need to know about this task.

Steps

The individual steps to complete the task.

What's next?

What the user should do next.

Each of these should include a . right before the text, like so:

```
.What you'll need
.About this task
.Steps
.What's next?
```

This syntax applies bold text in a larger font.

Command syntax

When providing command input, enclose the command within ``` to apply monospace font:

```
`volume show -is-encrypted true`
```

Here's what that looks like:

```
volume show -is-encrypted true
```

For command output or command examples, use the following syntax:

```
----  
cluster2::> volume show -is-encrypted true  
  
Vserver  Volume  Aggregate  State  Type  Size  Available  Used  
-----  -  
vs1      vol1    aggr2      online  RW   200GB   160.0GB   20%  
----
```

The four dashes enable you to enter separate lines of text that appear together.
Here's the result:

```
cluster2::> volume show -is-encrypted true  
  
Vserver  Volume  Aggregate  State  Type  Size  Available  Used  
-----  -  
vs1      vol1    aggr2      online  RW   200GB   160.0GB   20%
```

Variable text

In commands and command output, enclose variable text within underscores to apply italics.

```
`vserver nfs modify -vserver _name_ -showmount enabled`
```

Here's what that command and the variable text looks like:

```
vserver nfs modify -vserver name -showmount enabled
```



Underscores aren't supported with code syntax highlighting at this time.

Code syntax highlighting

Code syntax highlighting provides a developer-focused solution for documenting the most popular languages.

Output example 1

```
POST https://netapp-cloud-account.auth0.com/oauth/token
Header: Content-Type: application/json
Body:
{
  "username": "<user_email>",
  "scope": "profile",
  "audience": "https://api.cloud.netapp.com",
  "client_id": "UaVhOIXMWQs5i1WdDxauXe5Mqkb34NJQ",
  "grant_type": "password",
  "password": "<user_password>"
}
```

Output example 2

```
[
  {
    "header": {
      "requestId": "init",
      "clientId": "init",
      "agentId": "init"
    },
    "payload": {
      "init": {}
    },
    "id": "5801"
  }
]
```

Supported languages

- bash
- curl
- https
- json
- powershell
- puppet
- python
- yaml

Implementation

Copy and paste the following syntax and then add a supported language and the code:


```
[source,<language>]
<code>
```

For example:

```
[source,curl]
curl -s https:///v1/ \
-H accept:application/json \
-H "Content-type: application/json" \
-H api-key: \
-H secret-key: \
-X [GET,POST,PUT,DELETE]
```

Content reuse

If you have a chunk of content that's repeated across different pages, you can easily write it once and reuse it across those pages. Reuse is possible from within the same repository and across repositories. Here's how it works.

1. Create a folder in your repository named `_include`

For example, take a look at the [Cloud Tiering repository](#).

2. Add a `.adoc` file in that folder that includes the content that you'd like to reuse.

It can be a sentence, a list, a table, one or more sections, and so on. Don't include anything else in the file—no headers or anything.

3. Now go to the files where you'd like to reuse that content.
4. If you're reusing the content from within the *same* GitHub repository, use the following syntax on a line by itself:

```
include::_include/<filename>.adoc[]
```

For example:

```
include::_include/s3regions.adoc[]
```

5. If you're reusing the content in a *different* repository, use the following syntax on a line by itself:

```
include::https://raw.githubusercontent.com/NetAppDocs/<reponame>/main/_include/<filename>.adoc[]
```

For example:

```
include::https://raw.githubusercontent.com/NetAppDocs/cloud-  
tiering/main/_include/s3regions.adoc[]
```

That's it!

If you want to learn more about the include directive, [check out the AsciiDoctor User Manual](#).

Policy forbidding the use of AI-generated content in NetApp technical documentation

At NetApp, we value the expertise of our employees, contractors, and partners in creating, publishing, and maintaining accurate, relevant, useful technical documentation for our customers. We believe that the use of AI-generated content can compromise these values and undermine the integrity and accuracy of our documentation. Therefore, NetApp's policy is to prohibit publication of AI-generated content that has not been verified and validated by a human. This applies to all technical documentation published on docs.netapp.com. Note that this policy is similar to and extends NetApp's policy that forbids the use of languages models to write code for the products it sells.

We expect all NetAppDocs contributors to adhere to the following policy:

- We do not allow publication of customer-facing documentation (how to use NetApp products) generated from AI without human review and validation. This includes written materials, graphics, multimedia, or other materials that are published on docs.netapp.com. All content published on docs.netapp.com must be verified and validated by human content creators.
- Part of the role of content creator for docs.netapp.com involves adhering to the style and contributor guides. This includes staying informed about the latest developments in AI-generated content and their implications for NetApp technical documentation.
- All employees, contractors, and partners are expected to comply with the policy. Repeated violations may result in suspended authoring privileges.

NetAppDocs contributors should keep the following in mind:

- All AI-based content-generating technologies embedded in software you have installed on your work computer, including ones like Microsoft 365 Co-pilot, send your data to the company that hosts the assistive technology. Do not prompt such a technology with company confidential content, including but not limited to specifications for unreleased product features.
- Do not use public AI technologies to generate or edit technical documentation for **previously unreleased** products, versions of products, or features within products.
- Copyright for new technical documentation must remain within NetApp and not be shared with AIs outside NetApp prior to official product release dates and the corresponding publication of related technical documentation.
- Do not send confidential documentation or newly-written content into AI (ChatGPT or any other tool). Any information fed into ChatGPT becomes a part of its training dataset, which in turn becomes property of OpenAI and can be used by them and their users. While much of this data is publicly accessible, we do not want to train the model on our own IP.

Legal notices

Legal notices provide access to copyright statements, trademarks, patents, and more.

Copyright

<https://www.netapp.com/company/legal/copyright/>

Trademarks

NETAPP, the NETAPP logo, and the marks listed on the NetApp Trademarks page are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.

<https://www.netapp.com/company/legal/trademarks/>

Patents

A current list of NetApp owned patents can be found at:

<https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf>

Privacy policy

<https://www.netapp.com/company/legal/privacy-policy/>

Copyright information

Copyright © 2024 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

Trademark information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.