

KNOWLEDGE BASE ARTICLE

Privacy and Security in RSS Reader: How Feeds Are Loaded and What Is Sent



Disclaimer

For the purposes of this Disclaimer:

- **Company** (referred to as either "the Company", "We", "Us" or "Our" in this Disclaimer) refers to Vitextra SIA, LV-1009 Riga, Matīse iela 61 - 25.
- **Document** refers to the Knowledgebase Article.
- **You** means the individual accessing the Document, or the company, or other legal entity on behalf of which such individual is accessing or using the Document, as applicable.
- **Website** refers to Vitextra, accessible from <https://vitextra.com>

Disclaimer

The information contained in the Document is for general information purposes only.

The Company assumes no responsibility for errors or omissions in the contents of the Document.

In no event shall the Company be liable for any special, direct, indirect, consequential, or incidental damages or any damages whatsoever, whether in an action of contract, negligence, or other torts, arising out of or in connection with the use of the contents of the Document. The Company reserves the right to make additions, deletions, or modifications to the contents in the Document at any time without prior notice.

Errors and Omissions Disclaimer

The information given by the Document is for general guidance on matters of interest only. Therefore, errors can occur even if the Company takes every precaution to ensure that the Document's content is current and accurate. Plus, given the changing nature of laws, rules, and regulations, there may be delays, omissions, or inaccuracies in the information contained in the Document.

The Company is not responsible for any errors or omissions, or for the results obtained from the use of this information.

Fair Use Disclaimer

The Company may use copyrighted material that the copyright owner has not explicitly authorized. The Company is making such material available for criticism, comment, news reporting, teaching, scholarship, or research.

The Company believes this constitutes a "fair use" of any such copyrighted material as provided for in section 107 of the United States Copyright law.

If You wish to use copyrighted material from the Document for your own purposes that go beyond fair use, You must obtain permission from the copyright owner.

"Use at Your Own Risk" Disclaimer

All information in the Document is provided "as is", with no guarantee of completeness, accuracy, timeliness or of the results obtained from the use of this information, and without warranty of any kind, express or implied, including, but not limited to warranties of performance, merchantability and fitness for a particular purpose.

The Company will not be liable to You or anyone else for any decision made or action taken in reliance on the information given by the Service or for any consequential, special or similar damages, even if advised of the possibility of such damages.

Contact Us

If you have any questions about this Disclaimer, You can contact Us by email: support@vitextra.com.

Table Of Contents

Introduction	4
Summary for administrators	4
Permissions of the solution	4
What users need	5
How feeds are loaded and what is sent	5
Where the configuration is stored	6
What is kept in the browser	7
How feed content is rendered	7
Requests to Vitextra services	7
Network requirements	8
Recommendations	8
Terms	10

Introduction

A web part that brings content from outside the company onto the intranet raises two questions for an administrator: what leaves the tenant, and what comes in. [RSS Reader](#) is unusual among Vitextra products in that its content does pass through a Vitextra server – the feed service that downloads the feeds. This article states precisely what that request contains, what the web part is permitted to do in your tenant, where its configuration lives, what it keeps in the browser and how it treats the HTML that publishers put into their feeds.

Summary for administrators

Question	Answer
Which API permissions does the package request?	None
Does the web part read SharePoint lists, files or user profiles?	No
Who downloads the feeds?	The Vitextra feed service, on request of the browser
What does the request to the feed service contain?	The feed addresses, the Tenant ID, the object ID of the user, the site address, the product and its version
Is any content of your tenant sent to Vitextra?	No
Where is the configuration stored?	In the web part properties on the page; for the Teams personal app, in the Vitextra service for your tenant
What is kept in the browser?	The last loaded feed items, for up to 30 minutes of the tab session
Is feed HTML executed in the page?	No – it is parsed inertly; only plain text and image addresses are used

Permissions of the solution

RSS Reader is a **SharePoint Framework** web part deployed from the **app catalog**.

- The package requests **no API permissions**. Nothing appears on the **API access** page of the SharePoint admin center, and the web part calls neither Microsoft Graph nor the SharePoint REST API for data.
- From the page context it reads only what every web part has: the theme, the language, the site address (for the license check and the feed request) and the identity of the signed-in user (the object ID, for the same requests).
- It writes nothing to SharePoint, OneDrive or Microsoft Entra ID.

What users need

Task	Required permission
Upload the package to the app catalog, add it to Teams	SharePoint administrator, or owner of the app catalog
Add the web part to a page and configure the feeds	Edit permission on the page
Save the default configuration of the Teams personal app	Edit permission on a page with the web part
Read the feeds, switch sources, filter and search	Read permission on the page

How feeds are loaded and what is sent

Browsers do not allow a script on one website to download a document from another website unless that website permits it with cross-origin headers; feed publishers almost never do. A browser-only RSS web part would therefore fail for most feeds. RSS Reader solves this with a server-side **feed service** at vitextra.com: the browser sends it the list of feed addresses, the service downloads the feeds from the internet, parses them and returns the channels and items as structured data.

Every load and every automatic refresh is one request to the feed service with the following content:

Field	Value
Sources	The addresses of the configured feeds
TenantId	The ID of your Microsoft 365 tenant
UserId	The object ID (GUID) of the signed-in user in Microsoft Entra ID

Field	Value
SiteUrl	The address of the site on which the web part runs
ProductId, ProductName, ProductVersion	The identifier, internal name and version of the product

The response contains the feed items – titles, links, descriptions, dates, authors, categories and image addresses – and the list of sources that could not be downloaded. The service does not receive and does not return any content of your tenant; the user is identified by a GUID, not by name or e-mail address.

Three consequences:

- the **feed addresses** you configure are known to the feed service. Do not use feeds whose address itself contains a secret (a personal token or API key in the query string): such feeds are not meant to be shared with any aggregator, and they are not supported anyway, because the service downloads feeds anonymously;
- feeds that are reachable only inside your network cannot be shown;
- the browser never connects to the publishers' servers for the feed. It does connect to them for the **images** of the items, which are loaded directly by the browser like any picture on a web page – the publisher sees the ordinary image request of the user's browser.

Note

The processing of these requests by Vitextra is described in the [Privacy Policy](#). The request tells the service which feeds a page is configured with, not which articles a user opens – the links to the articles lead straight to the publisher, and no click is reported to Vitextra.

Where the configuration is stored

On a SharePoint page, the configuration – the feeds with their addresses, names and order, the layout and the switches – is stored in the **properties of the web part**, inside the page in the **Site Pages** library. Everyone who can read the page can read the configuration (the feed addresses are visible in the source tabs anyway); page versions keep earlier configurations.

For the Teams personal app, an administrator saves the configuration with **Save Settings** in the **Personal app (Microsoft Teams)** group. This configuration – the same

fields, without the web part title – is stored by the Vitextra service for your tenant and product, because the personal app has no page to store it in. It is read by the personal app of every user in the tenant and overwritten by the next **Save Settings**. It contains settings and feed addresses only, no user data.

In a **Teams channel tab**, the configuration is stored with the tab, like on a page.

What is kept in the browser

After a successful load the web part writes the received items to the **session storage** of the browser, keyed by the set of feed addresses, and uses them for up to 30 minutes to show content instantly on the next visit while fresh items load in the background. The cache contains the feed items as returned by the feed service and nothing about the user; it belongs to the browser tab session and is removed when the tab is closed. The web part does not use local storage, cookies or any other persistent storage.

Which articles a user has opened is **not** recorded anywhere. The dimming of already-opened titles relies on the browser's own memory of visited links, which no script can read.

How feed content is rendered

Feed items contain HTML written by the publisher – descriptions with markup, embedded images, sometimes scripts or tracking pixels. RSS Reader never inserts this HTML into the page:

- the description is parsed with an **inert parser** (**DOMParser**), which neither executes scripts nor loads embedded resources, and only its **plain text** is displayed;
- the image of an item is taken from the feed's media elements or from the first image of the description, and one-pixel images – the typical tracking pixels – are skipped; the image is then shown as an ordinary **** element with the publisher's address;
- links open the original article in a new tab on the publisher's site; the web part adds no tracking parameters.

A malformed item – missing elements, broken markup – is shown with the data it has and does not prevent the other items from rendering.

Requests to Vitextra services

Purpose	What is sent	Where
Feed service	See the table above	vitextra.com

Purpose	What is sent	Where
License check	The Tenant ID, the object ID of the user, the site address, the identifier, name and version of the product	vitextra.com
Teams personal-app configuration	Read on every load of the personal app; written by Save Settings : the configuration fields listed above	vitextra.com
Usage telemetry	An event when the web part loads: product name and version, the ID and address of the site, the object ID of the user, the interface language and whether the web part runs in Teams; technical details of errors	Azure Application Insights
Product files	An ordinary file request for the scripts and icons of the web part	static.vitextra.com, vitextra-storage.azureedge.net

For the same information across all Vitextra products, see [KB10363. Where Vitextra Products Store Your Data: Privacy, Data Residency and GDPR](#).

Network requirements

Allow outbound HTTPS from the users' network (and from the Teams client) to [vitextra.com](#), [static.vitextra.com](#) and [vitextra-storage.azureedge.net](#). Without the first host the web part shows *Sorry, something went wrong...* instead of items; without the other two it does not load at all. The images of the items are loaded from the publishers' hosts; if your proxy blocks a publisher, its items appear without pictures.

Recommendations

- 1 Use feeds of publishers your organization trusts, as you would for any external content on the intranet: the titles, texts and pictures are theirs and change without your review.
- 2 Prefer HTTPS feed addresses with HTTPS images; pictures served over HTTP are blocked by the browser on HTTPS pages.
- 3 Do not configure feeds whose address contains a personal token; they are shared with everyone who can read the page and with the feed service.
- 4 On shared computers, clear the site data of the browser at sign-out as for any intranet page; the feed cache disappears with the tab anyway.

Terms

App Catalog

The App Catalog is a special SharePoint site that stores apps for SharePoint, Teams, and other Microsoft 365 services. Global or SharePoint administrators can upload and manage applications from the AppSource marketplace, custom-developed apps, or third-party apps for the organization's internal use. There can be only one tenant-level App Catalog in a tenant; site collection app catalogs can be added for individual site collections.

SharePoint Framework (SPFx)

SharePoint Framework (SPFx) is a model for building client-side web parts, components, and extensions for SharePoint Online, SharePoint Server (on-premises), and Microsoft Teams. SPFx customizations are based on JavaScript and support the modern user interface natively.