NetWitness® Platform

Version 12.5

Extended Meta User Guide

NETWITNESS

Platform

Contact Information

NetWitness Community at https://community.netwitness.com contains a knowledge base that answers common questions and provides solutions to known problems, product documentation, community discussions, and case management.

Trademarks

RSA and other trademarks are trademarks of RSA Security LLC or its affiliates ("RSA"). For a list of RSA trademarks, go to https://www.rsa.com/en-us/company/rsa-trademarks. Other trademarks are trademarks of their respective owners.

License Agreement

This software and the associated documentation are proprietary and confidential to RSA Security LLC or its affiliates are furnished under license, and may be used and copied only in accordance with the terms of such license and with the inclusion of the copyright notice below. This software and the documentation, and any copies thereof, may not be provided or otherwise made available to any other person.

No title to or ownership of the software or documentation or any intellectual property rights thereto is hereby transferred. Any unauthorized use or reproduction of this software and the documentation may be subject to civil and/or criminal liability. This software is subject to change without notice and should not be construed as a commitment by RSA.

It is advised not to deploy third-party repos or perform any change to the underlying NetWitness Operating System that is not part of the supported NetWitness version. Any such change outside of the NetWitness approved image may result in a service or functionality conflict and require a reimage of the NetWitness system to bring NetWitness back to an optimized functional state. In the event a third-party repo is deployed, or other non-supported change is made by the customer without NetWitness approval, the customer takes full responsibility for any system malfunction until the issue can be remediated through troubleshooting efforts or a reimage of the service.

Third-Party Licenses

This product may include software developed by parties other than RSA. The text of the license agreements applicable to third-party software in this product may be viewed on the product documentation page on NetWitness Community. By using this product, a user of this product agrees to be fully bound by terms of the license agreements.

Note on Encryption Technologies

This product may contain encryption technology. Many countries prohibit or restrict the use, import, or export of encryption technologies, and current use, import, and export regulations should be followed when using, importing or exporting this product.

Distribution

Use, copying, and distribution of any RSA Security LLC or its affiliates ("RSA") software described in this publication requires an applicable software license.

RSA believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED "AS IS." RSA MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Miscellaneous

This product, this software, the associated documentations as well as the contents are subject to NetWitness' standard Terms and Conditions in effect as of the issuance date of this documentation and which can be found at https://www.netwitness.com/standard-form-agreements/.

October, 2024

Contents

Overview

The maximum meta value limit for Decoder and Log Decoder has always been 256 bytes. This means that when a Packet Decoder or Log Decoder (hereafter collectively referred to as “Decoder”) parses raw data and generates meta, the values of that meta cannot exceed 256 bytes. If the raw value exceeds that limit, it will be truncated to the first 256 bytes, and only those bytes will be parsed and indexed.

This presents a challenge because meta such as URL Query Strings or PowerShell Commands sometimes contain values that exceed 256 bytes.

The Extended Meta Feature allows configuring certain meta keys to support values greater than 256 bytes.

Extended Meta can have significant impacts on multiple areas of Core, including (but not limited to) data retention, queries, and aggregation. The purpose of this document is to guide users on how to add extended meta to their environment without impacting performance too drastically.

Performance Considerations

Users must opt into the Extended Meta feature and only specific types of meta should be extended. Extended meta can have an impact on the following areas.

Database Size

Extended meta refers to data written to core databases (for example, metadb and indexdb). Increasing the maximum size of the meta can result in shorter retention periods as session sizes grow. This can lead to more frequent occurrences of meta database size rolls and shorter retention periods for meta and index, especially in environments with high amounts of extended meta.

Query Performance

The performance of queries or expensive operations can be adversely affected, whether initiated by an analyst or an automated process (for example, report, API script). In an environment with many rules or queries containing operations like "contains" and "regex", these operations will naturally take longer to run when used with longer values.

Aggregation

Aggregation is the process of downloading data. Since Extended Meta can increase the size of meta data per session, each round of aggregation may potentially take longer.

In our testing, we found that extending a single impactful meta key (for example, “alias.host” or “url”) from 255 to 4k and with 30% of traffic or logs containing large extended values for those meta has reduced capture throughput by 10% on Decoders and Log Decoders. It also impacted the Query performance on Concentrators and Archivers by minimum of 10% and maximum resulting in query timeouts when querying those meta for large data.

Configuring Extended Meta Keys

Extended meta keys are configured in the index file using the attribute `maxValueLength`.

For example:

```
<key description="Querystring" name="query" format="Text" level="IndexKeys"
maxValueLength="4096" />
```

Here we have defined the meta key `query` to save values up to 4,096 bytes in size. Any values exceeding 4k in length will be truncated to 4k.

Note: A service restart is required before any changes to `maxValueLength` take effect.

The maximum value of maxValueLength is 4k (4,096 bytes).

Index Files

When configuring the `custom index xml` files, the `maxValueLength` setting needs to be added to the source devices' language file only (i.e. `index-decoder-custom.xml` and `index-logdecoder-custom.xml`). It is not required to add the setting to Concentrators, Archivers, and so on.

The `maxValueLenth` setting is also supported in the `*-custom.xml` files (that is, `index-decoder-custom.xml`).

Extended Meta Recommendations

When adding Extended Meta to an environment it is recommended that this be done slowly and incrementally. After each incremental change, users should let the new data work its way through the system to see how performance is impacted.

Specific things to look for include:

- **Database sizes and retention** - Do database sizes appear significantly larger? Do size rollouts appear to be happening more frequently?
- **Query performance** - Are the Extended Meta keys indexed? Are there any queries performing expensive operations, such as contains or regex, on these meta?
- **Aggregation performance** - Are Concentrators and Archivers still able to keep up with aggregation? Are any aggregating devices falling behind?

There are two main techniques users can utilize when deploying Extended Meta in their environment:

Incrementally Increasing Extended Meta Sizes

While there is no configuration error in configuring a key from non-extended right to 4,096 (the maximum size of an Extended Meta key), it is better to gradually increase the size and find the minimum size that gives you the most value. Users may first want to increase the size of a single meta to 500, then maybe 1,000 and so on, and noting the impact that each step has on the environment.

Incrementally Adding Extended Meta

There may be multiple keys that users want to extend, however extending all keys at once makes it difficult to pinpoint problems if they may occur. Instead, users should Extended Meta one or two at a time. Thus, if the system is generating a particular piece of Extended Meta in quantities that are affecting performance, the user should know exactly which meta is causing the issue and then can either reduce the maximum size of that meta or remove it entirely.