Revision Letter

Editor: Jaehoon Paul Jeong
Date: January 26, 2022

OLD: draft-ietf-i2nsf-nsf-monitoring-data-model-12
NEW: draft-ietf-i2nsf-nsf-monitoring-data-model-13

Dear Tim Bray, Kyle Rose, Dale R. Worley, Melinda Shore, Valery Smyslov, and Tom Petch,

I sincerely appreciate your detailed comments to improve our NSF Monitoring YANG Data Model. I have addressed all your comments one by one. I use bold font for your comments and use a regular font for my responses with the prefix "=> [PAUL]".

Expert Review (IANA Registry)
Reviewer: Tim Bray

Dear Authors,

The XML registry expert has identified an issue with the registration in this document:

===

For it to work, (a) the prefix in the alarm-category element MUST be the same as the namespace prefix for urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-monitoring, which means that XML software MUST be chosen that makes the namespace prefix information available. I don't think it's OK for the draft not to say those things.

```
  <alarm-category
    xmlns:nsfmi="urn:ietf:params:xml:ns:yang:\
                ietf-i2nsf-nsf-monitoring">
    nsfmi:memory-alarm
  </alarm-category>
```

=> [PAUL] We have addressed your comments in the beginning of Section 11. The update is as follows:

OLD:
11. XML Examples for I2NSF NSF Monitoring This section shows the XML examples of I2NSF NSF Monitoring data delivered via Monitoring Interface from an NSF.

NEW:

11. XML Examples for I2NSF NSF Monitoring

This section shows the XML examples of I2NSF NSF Monitoring data delivered via Monitoring Interface from an NSF. In order for the XML data to be used correctly, the prefix (i.e., the characters before the colon or 'nsfmi' in the example) in the content of the element that uses the "identityref" type (e.g., /i2nsf-event/i2nsf-system-detection-alarm/alarm-category/) in the YANG module described in this document MUST be the same as the namespace prefix (i.e., 'nsfmi' in the example) for urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-monitoring. Therefore, XML software MUST be chosen that makes the namespace prefix information available.

===

Best regards,

Amanda Baber
IANA Operations Manager

Reviewer: Kyle Rose
Review Results: Ready with Nits

This document has been reviewed as part of the transport area review team's ongoing effort to review key IETF documents. These comments were written primarily for the transport area directors, but are copied to the document's authors and WG to allow them to address any issues raised and also to the IETF discussion list for information.

When done at the time of IETF Last Call, the authors should consider this review as part of the last-call comments they receive. Please always CC tsv-art@ietf.org if you reply to or forward this review.

My summary of the review is that this document does not carry any concerns of particular interest to the transport area.

Coincidentally or not, I [completed a secdir review last week](https://datatracker.ietf.org/doc/review-ietf-i2nsf-nsf-facing-interface-dm-16-secdir-lc-rose-2021-11-22/) on a related document. In that review I had comments re: how YANG was used that may also apply here.

The data model in various areas makes assumptions about the systems being monitored. For instance, in section 6.2.1 ("Access Violation"), the identifying information for the attempted access violation is given as (user, group, IP address). Is this universal? What if the connection was NATed? You might need the port and/or a snapshot of the connection tracking state at the time. Or proxied? It feels like identifying information is highly context-dependent and should be parameterized in an extensible way.

=> [PAUL] Usually, the username is the most important information when an Access Violation is detected. Other identifying information is used to give additional information. An IP address information (which can be easily masqueraded with VPN/proxy or simply with NAT), can be helpful to detect any unusual activity. For example, if the NSF is deployed in South Korea, but a notification of Access Violation is emitted with the IP address information from the US. It will help ring a suspicion whether the user ID has been compromised and security actions can be implemented.

And as you mentioned, the identifying information is context-dependent, so we set the 4 minimum information as the identifying information, i.e., username, group, ip-address, and port number. This identifying information can be extended as needed. We updated the information as follows:

OLD:

```
   The access-violation system event is an event when a user tries to
   access (read, write, create, or delete) any information or execute
   commands above their privilege.

   *  event-name: access-denied.

   *  user: Name of a user.

   *  group: Group(s) to which a user belongs.  A user can belong to
      multiple groups.

   *  ip-address: The IP address of the user that triggered the event.

   *  authentication: The method to verify the valid user, i.e., pre-
      configured-key and certificate-authority.

   *  message: The message to give the context of the event,
      such as "Access is denied".
```

NEW:

```
   The access-violation system event is an event when a user tries to
   access (read, write, create, or delete) any information or execute
   commands above their privilege.  The following information should be
   included in this event:

   *  event-name: access-violation.

   *  identity: The information to identify the attempted access
      violation.  The minimum information (extensible) that should be
      included:

      1.  user: The unique username that attempted access violation.

      2.  group: Group(s) to which a user belongs.  A user can belong to
```

```
         multiple groups.

      3.  ip-address: The IP address of the user that triggered the
          event.

      4.  port-number: The port number used by the user.

   *  authentication: The method to verify the valid user, i.e., pre-
      configured-key and certificate-authority.

   *  message: The message to give the context of the event, such as
      "Access is denied".
```

Relatedly, the same schema for user identifying information is replicated elsewhere in the data model rather than being abstracted. Right after "Access Violation" comes "Configuration Change", which includes the same information.
=> [PAUL] In the same way with the Access Violation above, we added a similar explanation for the other data models that use the identifying information as follows:

OLD:

```
   A configuration change is a system event when a new configuration is
   added or an existing configuration is modified.  The following
   information should be included in this event:

   *  event-name: config-change.

   *  user: Name of a user.

   *  group: Group(s) to which a user belongs.  A user can belong to
      multiple groups.

   *  ip-address: The IP address of the user that triggered the event.

   *  authentication: The method to verify the valid user, i.e., pre-
      configured-key and certificate-authority.

   *  message: The message to give the context of the event, such as
      "Configuration is modified" or "New configuration is added".
```

NEW:

```
   A configuration change is a system event when a new configuration is
   added or an existing configuration is modified.  The following
   information should be included in this event:

   *  event-name: configuration-change.

   *  identity: The information to identify the attempted access
      violation.  The minimum information (extensible) that should be
```

included:

1. user: The unique username that changes the configuration.

2. group: Group(s) to which a user belongs. A user can belong to multiple groups.

3. ip-address: The IP address of the user that triggered the event.

4. port-number: The port number used by the user.

* authentication: The method to verify the valid user, i.e., pre-configured-key and certificate-authority.

* message: The message to give the context of the event, such as "Configuration is modified", "New configuration is added", or "A configuration has been removed".

* changes: Describes the modification that was made to the configuration. The minimum information that must be provided is the name of the policy that has been altered (added, modified, or removed). Other detailed information about the configuration changes is up to the implementation.

One major nit (Is that like jumbo shrimp?): there are a lot of 32-bit counters employed in this data model, many of which will probably overflow quite quickly. While moving to 64-bit counters would probably address most such instances, I cannot find any discussion in the WG's other documents of how counter overflow should be managed.
=> [PAUL] We have updated the counters to 64-bit to increase the limit of the counters. Following RFC 8343 (A YANG Data Model for Interface Management), we use “discontinuity-time” to identify any overflow/discontinuity on the counters data. We added the explanation in the data model that consists of any counters.

NEW:

6.6.1. Interface Counter

o discontinuity-time: The time on the most recent occasion at which any one or more of the counters suffered a discontinuity. If no such discontinuities have occurred since the last re-initialization of the local management subsystem, then this node contains the time the local management subsystem was re-initialized.

6.7.1. Firewall Counter

o discontinuity-time: The time on the most recent occasion at which any one or more of the counters suffered a discontinuity. If no such discontinuities have occurred since the last re-initialization of the local management subsystem, then this node contains the time the local management subsystem was re-initialized.

6.7.2. Policy Hit Counter

o discontinuity-time: The time on the most recent occasion at which any one or more of the counters suffered a discontinuity. If no such discontinuities have occurred since the last re-initialization of the local management subsystem, then this node contains the time the local management subsystem was re-initialized.

Popping up a level, however, I question the utility of standardizing an interface to what the WG charter itself recognizes as a basic set of functions, as anything beyond these basic functions would need to be accessed via custom knobs. Even within flow-based security functions, it's unclear to me (for example) how extensibility for novel or more specific values (even within an existing category) is expected to work in a way that is compatible with the goal of creating a standard interface. If the motivation here is to prevent vendor lock-in, it's not clear to me that this approach will achieve that. If OTOH the goal is to fix an interface for a relatively stable set of functionality that is no longer expected to expand in scope, in a long-term effort (alongside development of a shared software ecosystem) to reduce maintenance costs for all players in that ecosystem, this might be the right direction. Standards almost always lag proprietary implementations, and for good reason.
=> [PAUL] I think that I2NSF standard interfaces are intended for multiple vendors of NSFs to provide their security solutions for basic functionality of network-based security services. For further advanced network security functions for those security vendors, I2NSF working group will work for the extension of the current standard interfaces and also develop new standard interfaces.

I am the assigned Gen-ART reviewer for this draft. The General Area Review Team (Gen-ART) reviews all IETF documents being processed by the IESG for the IETF Chair. Please treat these comments just like any other last call comments.

For more information, please see the FAQ at

<https://trac.ietf.org/trac/gen/wiki/GenArtfaq>.

Document: draft-ietf-i2nsf-nsf-monitoring-data-model-12
Reviewer: Dale R. Worley
Review Date: 2021-11-28
IETF LC End Date: 2021-12-01
IESG Telechat date: not known

Summary:

This draft is on the right track but has open issues, described in the review. It is clear that all of these issues can be fixed appropriately, but they need to be fixed before publication.

Major issues:

This document presents a data model for data being passed between various I2NSF entities. It appears that the author has a thorough understanding of the I2NSF architecture and so has made various references to it in the document. But since the data model definition does not depend on the overall architecture, the document should be revised to either (1) remove unnecessary references to the overall architecture, (2) segregate them in ways that show they are not needed to understand the data model, or (3) carefully referenced back to the documents that define them.
=> [PAUL] We have addressed your guidelines in the following text.

There are also a few points where there seems to be technical issues regarding the definitions of specific data items.

Details:

1. Introduction

Why do we have both "administrative entities" and "Security Controller" here, and "NSF data collector" in section 3? Naively, I would expect that in regard to the definition of the data model presented by the "server side", all "client side" processes would be considered an amorphous group covered by one generic term.
=> [PAUL] We removed the usage of “administrative entities” in them draft and more focused on the “NSF data collector” as the principle for the “client-side”. The Introduction has been revised as follows:

OLD:
According to [RFC8329], the interface provided by a Network Security Function (NSF) (e.g., Firewall, IPS, or Anti-DDoS function) to administrative entities (e.g., Security Controller) to enable remote management (i.e., configuring and monitoring) is referred to as an I2NSF Monitoring Interface. This interface enables the sharing of vital data from the NSFs (e.g., alarms, records, and counters) to the Security Controller through a variety of mechanisms (e.g., queries, notifications, and events). The monitoring of NSF plays an important role in an overall security framework, if it is done in a timely and comprehensive way. The monitoring information generated by an NSF can be a good, early indication of anomalous behavior or malicious activity, such as denial of service attacks (DoS). This document defines a comprehensive information model of an NSF monitoring interface that provides visibility into an NSF for the NSF data collector ~~(e.g., Security Controller)~~. Note that an NSF data collector is defined as an entity to collect NSF monitoring data from

an NSF, such as Security Controller. It specifies the information and illustrates the methods that enable an NSF to provide the information required in order to be monitored in a scalable and efficient way via the NSF Monitoring Interface. The information model for the NSF monitoring interface presented in this document is complementary for the security policy provisioning functionality of the NSF-Facing Interface specified in [I-D.ietf-i2nsf-nsf-facing-interface-dm].

This document also defines a YANG [RFC7950] data model for the NSF monitoring interface, which is derived from the information model for the NSF monitoring interface.

NEW:

According to [RFC8329], the interface provided by a Network Security Function (NSF) (e.g., Firewall, IPS, or Anti-DDoS function) to enable the collection of monitoring information is referred to as an I2NSF Monitoring Interface. This interface enables the sharing of vital data from the NSFs (e.g., events, records, and counters) to the NSF data collector through a variety of mechanisms (e.g., queries and notifications). The monitoring of NSF plays an important role in an overall security framework, if it is done in a timely and comprehensive way. The monitoring information generated by an NSF can be a good, early indication of anomalous behavior or malicious activity, such as denial of service attacks (DoS).

This document defines a comprehensive information model of an NSF monitoring interface that provides visibility into an NSF for the NSF data collector. Note that an NSF data collector is defined as an entity to collect NSF monitoring data from an NSF, such as Security Controller. It specifies the information and illustrates the methods that enable an NSF to provide the information required in order to be monitored in a scalable and efficient way via the NSF Monitoring Interface. The information model for the NSF monitoring interface presented in this document is complementary for the security policy provisioning functionality of the NSF-Facing Interface specified in [I-D.ietf-i2nsf-nsf-facing-interface-dm].

This document also defines a YANG [RFC7950] data model for the NSF monitoring interface, which is derived from the information model for the NSF monitoring interface.

2. Terminology

This document uses the terminology described in [RFC8329].

Given that RFC 8329 doesn't define the terminology, it would be better to expand on this to "This document uses the terminology described in [RFC8329], much of which is defined in the I2NSF terminology document

I2NSF-TERMS]." Indeed, since I2NSF-TERMS is draft-ietf-i2nsf-terminology-05, presumably part of the same effort as this document, why is RFC 8329 being mentioned?

=> [PAUL] The draft-ietf-i2nsf-terminology-05 is expired and no longer continued. I think it is not appropriate to reference to an expired draft, so we reference RFC 8329 instead.

--

There seems to be trouble with terms used in this document. Some of them are mentioned in section 2.2 of RFC 8329, which simply refers to I2NSF-TERMS. Others (e.g. "I2NSF Record") seem like they should be listed in RFC 8329, but aren't, and seem to be entirely undefined. Some of those terms appear in text that may as well be omitted from this document. Ideally, the specialized vocabulary in this document should be listed in this section and proper definitions or references provided for them.

=> [PAUL] I2NSF Event, Record, and sCounter are defined in Section 4.1 to explain the contents clearly.

Other terms are added in Section 2 in the document as follows:

OLD:
Section 2. Terminology The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here. This document uses the terminology described in [RFC8329]. This document follows the guidelines of [RFC8407], uses the common YANG types defined in [RFC6991], and adopts the Network Management Datastore Architecture (NMDA) [RFC8342]. The meaning of the symbols in tree diagrams is defined in [RFC8340].

NEW:
Section 2. Terminology The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here. This document uses the terminology described in [RFC8329]. In addition the following terms are defined in this document: o I2NSF User: An entity that delivers a high-level security policy

to the Security Controller and may request monitoring information via the NSF data collector.

- Monitoring Information: Relevant data that can be processed to know the status and performance of the network and the NSF. The monitoring information in I2NSF environment consists of I2NSF Event, I2NSF Record, and I2NSF Counter (see Section 4.1 for the detailed definition). This information is to be delivered to the NSF data collector.

- Notification: Unsolicited transmission of monitoring information.

- NSF Data Collector: An entity that collects NSF monitoring information from NSFs, such as Security Controller.

- Subscription: An agreement initialized by the NSF data collector to receive monitoring information from an NSF. The method to subscribe is following the method explained in [RFC5277].

This document follows the guidelines of [RFC8407], uses the common YANG types defined in [RFC6991], and adopts the Network Management Datastore Architecture (NMDA) [RFC8342]. The meaning of the symbols in tree diagrams is defined in [RFC8340].

3. Use Cases for NSF Monitoring Data

* **The security administrator with I2NSF User can configure a policy**

"I2NSF User" is not listed in RFC 8329. Also, the placement of "with I2NSF User" suggests that that phrase is some aspect of "security administrator", and you might want to say "The I2NSF User that is the security administrator ...". OTOH, if "with I2NSF User" is some aspect of "can configure", it should probably be placed after "can configure". (Is an I2NSF User a type of "user", as that word is normally used?)

=> [PAUL] The sentence is updated according to your suggestion as follows:

OLD:

The security administrator with I2NSF User can configure a policy that is triggered on a specific event occurring in the NSF or the network [RFC8329] [I-D.ietf-i2nsf-consumer-facing-interface-dm].

NEW:

The I2NSF User that is the security administrator can configure a policy that is triggered on a specific event occurring in the NSF or the network [RFC8329] [I-D.ietf-i2nsf-consumer-facing-interface-dm]

4. Classification of NSF Monitoring Data

This enables security administrators to assess the state of the networks and in a timely fashion.

Likely should delete "and".
=> [PAUL] The word “and” is removed from the sentence.

In essence, these types of monitoring data can be leveraged to

Probably can be simplified to "This monitoring data ...".
=> [PAUL] We change the sentence following your suggestion as follows:

OLD:
In essence, these types of monitoring data can be leveraged to support constant visibility on multiple levels of granularity and can be consumed by the corresponding functions.

NEW:
In essence, this monitoring data can be leveraged to support constant visibility on multiple levels of granularity and can be consumed by the corresponding functions.

As with I2NSF components, every generic system entity can include a set of capabilities that creates information about some context with monitoring data (i.e., monitoring information), composition, configuration, state or behavior of that system entity.

I am sure this could be clarified if it was simplified. I think the meaning is "Every system entity creates information about some context with defined I2NSF monitoring data, and so every entity can be an I2NSF component."
=> [PAUL] We clarified the sentence following your suggestion as follows:

OLD:
As with I2NSF components, every generic system entity can include a set of capabilities that creates information about some context with monitoring data (i.e., monitoring information), composition, configuration, state or behavior of that system entity.

NEW:
Every system entity creates information about some context with defined I2NSF monitoring data, and so every entity can be an I2NSF component.

This

information is intended to be provided to other consumers of information and in the scope of this document, which deals with NSF monitoring data in an automated fashion.

I think this means "This information is can be consumed by other I2NSF components."
=> [PAUL] We updated the sentence with your comments as follows:

OLD:
This information is intended to be provided to other consumers of information and in the scope of this document, which deals with NSF monitoring data in an automated fashion.

NEW:
This information is intended to be consumed by other I2NSF components, which deal with NSF monitoring data in an automated fashion.

4.1. Retention and Emission

I2NSF Event: I2NSF Event is defined as an important occurrence over time,

This should be "an important occurrence at a particular time,". "over time" means that there is an extended period of time over which the event occurs, but I'm sure that I2NSF Events specify only a single instant for "when it happened".
=> [PAUL] The sentence is updated as follows:

OLD:
I2NSF Event is defined as an important occurrence over time, that is, a change in the system being managed or a change in the environment of the system being managed.

NEW:
I2NSF Event is defined as an important occurrence at a particular time, that is, a change in the system being managed or a change in the environment of the system being managed.

Records can be continuously processed by a system entity as an I2NSF Producer and

Up until this point, the description of "record" could apply to any database system. But I suspect that the intended semantics are that Records are generated at particular instants (and are unchanging

afterward), and thus a set of records has an ordering in time based on when they are generated. This is the fundamental characteristic of a "log file". In particular, a database of users does not have this property but a database of user activities does. If Record is intended to be constrained to this situation, that should be stated explicitly.

=> [PAUL] The definition for "I2NSF Record" is updated with a reordering of sentences according to your comments as follows:

OLD:

I2NSF Record: A record is defined as an item of information that is kept to be looked at and used in the future. Unlike I2NSF Event, records do not require immediate attention but may be useful for visibility and retroactive cyber forensic. Depending on the record format, there are different qualities in regard to structure and detail. Records are typically stored in log-files or databases on a system entity or NSF. ~~Records in the form of log-files usually include less structures but potentially more detailed information in regard to the changes of a system entity's characteristics. In contrast, databases often use more strict schemas or data models, therefore enforcing a better structure. However, they inhibit storing information that does not match those models ("closed world assumption").~~ Records can be continuously processed by a system entity as an I2NSF Producer and emitted with a format tailored to a certain type of record. Typically, records are information generated by a system entity (e.g., NSF) that is based on operational and informational data, that is, various changes in system characteristics. The examples of records include as user activities, network/traffic status, and network activity. They are important for debugging, auditing and security forensic of a system entity or the network having the system entity.

NEW:

I2NSF Record: A record is defined as an item of information that is kept to be looked at and used in the future. Typically, records are information generated by a system entity (e.g., NSF) that is based on operational and informational data (i.e., various changes in system characteristics), and are generated at particular instants to be kept without any changes afterward. A set of records has an ordering in time based on when they are generated. Unlike I2NSF Event, records do not require immediate attention but may be useful for visibility and retroactive cyber forensic. Records are typically stored in log-files or databases on a system entity or NSF. The examples of records include as user activities, device performance, and network status. They are important for debugging, auditing, and security forensic of a system entity or the network having the system entity.

I2NSF Counter: [...] When an NSF data collector asks for the value of a counter to it, a system entity emits

Note this sentence is incomplete in the draft.

=> [PAUL] We updated the sentence as follows:

NEW:

When an NSF data collector asks for the value of a counter, a system entity MUST update the counter information and emit the latest information to the NSF data collector.

It might be valuable to note that an I2NSF Counter can be an integer approximation of a value that is actually continuous. (All of the examples that are given are values that are intrinsically integers.) Perhaps add as the 3rd sentence "Other examples are integer approximations to continuous values, such as a processor temperature measured in tenths of a degree or the percentage of a disk that is used."

=> [PAUL] We added your suggestion to the sentence.

OLD:

I2NSF Counter: An I2NSF Counter is defined as a specific representation of continuous value changes of information elements that occur very frequently. Prominent examples are network interface counters for protocol data unit (PDU) amount, byte amount, drop counters, and error counters. Counters are useful in debugging and visibility into operational behavior of a system entity (e.g., NSF). When an NSF data collector asks for the value of a counter to it, a system entity emits

NEW:

I2NSF Counter: An I2NSF Counter is defined as a specific representation of an information element whose value changes very frequently. Prominent examples are network interface counters for protocol data unit (PDU) amount, byte amount, drop counters, and error counters. Other examples are integer approximations to continuous values, such as a processor temperature measured in tenth of a degree or the percentage of a disk that is used. Counters are useful in debugging and visibility into operational behavior of a system entity (e.g., NSF). When an NSF data collector asks for the value of a counter, a system entity MUST update the counter information and emit the latest information to the NSF data collector.

Indeed, the first sentence of this paragraph says "continuous value changes", despite that all of the examples are integer values that cannot change continuously. Perhaps a better phrasing is "a specific representation of an information element whose value changes very

frequently."
=> [PAUL] We updated the description of I2NSF Event following your suggestions as follows:

OLD:
I2NSF Counter: An I2NSF Counter is defined as a specific representation of continuous value changes of information elements that occur very frequently. Prominent examples are network interface counters for protocol data unit (PDU) amount, byte amount, drop counters, and error counters. Counters are useful in debugging and visibility into operational behavior of a system entity (e.g., NSF). When an NSF data collector asks for the value of a counter to it, a system entity emits

NEW:
I2NSF Counter: An I2NSF Counter is defined as a specific representation of an information element whose value changes very frequently. Prominent examples are network interface counters for protocol data unit (PDU) amount, byte amount, drop counters, and error counters. Other examples are integer approximations to continuous values, such as a processor temperature measured in tenth of a degree or the percentage of a disk that is used. Counters are useful in debugging and visibility into operational behavior of a system entity (e.g., NSF). When an NSF data collector asks for the value of a counter, a system entity MUST update the counter information and emit the latest information to the NSF data collector.

The retention of I2NSF monitoring information listed in Section 9 may

It seems like "in Section 9" could/should be omitted.
=> [PAUL] The sentence is updated as follows:

OLD:
The retention of I2NSF monitoring information ~~listed in Section 9~~ may be affected by the importance of the data.

NEW:
The retention of I2NSF monitoring information may be affected by the importance of the data.

4.2. Notifications, Events, and Records

**In consequence, an I2NSF Event is specified to
trigger an I2NSF Policy Rule. Such an I2NSF Event is defined as any
important occurrence over time in the system being managed, and/or in
the environment of the system being managed,**

**This text provides two definitions of "I2NSF Event" which aren't quite
the same. One is "anything that triggers an I2NSF Policy Rule", a
purely technical face. The other is "any important occurrence over
time", which is a human fact. The two definitions coincide only if
the policy rules exactly cover everything that is "important". This
needs to be tracked back to the source definition of "I2NSF Event" and
these sentences revised to match it.**
=> [PAUL] In I2NSF environment, Event is defined as something that triggers the evaluation of Condition clause of the I2NSF Policy Rule [RFC 8329]. The event itself is “important” occurrence that happens in the NSF. The sentence is updated as follows:

OLD:
In consequence, an I2NSF Event is specified to trigger an I2NSF Policy Rule. Such an I2NSF Event is defined as any important occurrence over time in the system being managed, and/or in the environment of the system being managed, which aligns well with the generic definition of Event from [RFC3877].

NEW:
In consequence, an I2NSF Event is specified to trigger the evaluation of the Condition clause of the I2NSF Policy Rule. Such an I2NSF Event is defined as an important occurrence at a particular time in the system being managed, and/or in the environment of the system being managed whose concept aligns well with the generic definition of Event from [RFC3877].

**which aligns well with
the generic definition of Event from [RFC3877].**

**Strictly, this clause says that "an I2NSF Event" "aligns well with the
generic definition of Event", but I think you mean that the *concept*
of an I2NSF Event aligns etc.**
=> [PAUL] The sentence is updated as follows:

OLD:
Such an I2NSF Event is defined as ~~any~~ important occurrence over time in the system being managed, and/or in the environment of the system being managed, which aligns well with the generic definition

of Event from [RFC3877].

NEW:
Such an I2NSF Event is defined as an important occurrence at a particular time in the system being managed, and/or in the environment of the system being managed whose concept aligns well with the generic definition of Event from [RFC3877].

4.3. Unsolicited Poll and Solicited Push

Ideally, an I2NSF User is accessing every relevant information about the I2NSF Component and is emitting I2NSF Events to an NSF data collector (e.g., Security Controller) in a timely manner.

OK, what *is* the model of operations? In this sentence, it seems that an "I2NSF User" is a process that accesses (by some method) information about (in?) an I2NSF Component, and then emits (via I2NSF Events) that data to an NSF data collector. But none of that is laid out in the preceding sections. Indeed "I2NSF User" is not defined, though here it doesn't sound like the usual definition of "user".

The actual mechanism implemented by an I2NSF Component is out of the scope of this document.

In this sentence, it sounds like the Component is the thing that sends the data, whereas just above, it is the User.

In some cases, the collection of information has to be conducted via a login mechanism provided by a system entity.

What is the use of the terms solicited, unsolicited, poll, and push here? Usually, the data source is considered a "server", and the consumer is a "client". If the client makes a request to the server, that is called "solicited" "pulling", and if it happens periodically, it is called "polling". Whereas if the server initiates an interaction to send data, that is called "unsolicited" "pushing". Terminology in this draft doesn't seem to use those conventions, but it doesn't tell what conventions it does use.

=> [PAUL] We updated the whole Section 4.3 according to your comments as follows:

OLD:
4.3 Unsolicited Poll and Solicited Push

The freshness of the monitored information depends on the acquisition method. Ideally, an I2NSF User is accessing every relevant information about the I2NSF Component and is emitting I2NSF Events to an NSF data collector (e.g., Security Controller) in a timely manner. Publication of events via a pubsub/broker model, peer-2-peer meshes, or static defined channels are only a few examples on how a solicited push of I2NSF Events can be facilitated. The actual mechanism implemented by an I2NSF Component is out of the scope of this document.

Often, the corresponding management interfaces have to be queried in intervals or on demand if required by an I2NSF Policy rule. In some cases, the collection of information has to be conducted via a login mechanism provided by a system entity. Accessing records of information via this kind of unsolicited polls can introduce a significant latency in regard to the freshness of the monitored information. The actual definition of intervals implemented by an I2NSF Component is also out of scope of this document.

NEW:

4.3. Unsolicited Poll and Solicited Pull

An important aspect of monitoring information is the freshness of the information. From the perspective of security, it is important to notice the current status of the network. The I2NSF Monitoring Interface provides the means of sending monitored information from the NSFs to an NSF data collector in a timely manner. The method of acquiring the monitoring information can be performed from a client (i.e., NSF data collector) to a server (i.e., NSF) by unsolicited poll or solicited pull.

The solicited pull is a query-based method to obtain information from the NSF. In this method, the NSF will remain passive until the information is requested from the NSF data collector. Once a new request is accepted (with proper authentication), the NSF MUST update the information before sending it to the NSF data collector.

The unsolicited poll is a report-based method to obtain information from the NSF. The report-based method ensures the information can be delivered immediately without any requests. This method is used by the NSF to actively provide information to the NSF data collector. To receive the information, the NSF data collector subscribes to the NSF for the information.

These methods are used for different types of monitoring information. The information that has a high level of urgency (i.e., I2NSF Event) should be provided with the unsolicited poll method, while information that has a lower level of urgency (i.e., I2NSF Record and I2NSF Counter) can be provided with either the solicited pull method or unsolicited poll method.

5. Basic Information Model for Monitoring Data

* vendor-name: The name of the NSF vendor.

Generally, the minimum information needed to identify how to interact with a device is (1) vendor name, (2) device model name/number, (3) software version identifier. Vendor name alone isn't particularly useful.

=> [PAUL] The device model and software version fields are added to the list as follows:

NEW:

* device-model: The model of the device, can be represented by the device model name or serial number. This field is used to identify the model of the device that provides the security service.

* software-version: The version of the software used to provide the security service.

6. Extended Information Model for Monitoring Data

This section covers the additional information associated with the system messages.

What is "system messages"? The term has not been defined or mentioned previously. Is it a special class of "messages"? Indeed, the term seems to not be used elsewhere.

The extended information model is only for the structured data such as events, record, and counters. Any unstructured data is specified with the basic information model only.

There has been no previous discussion of "structured" vs. "unstructured" data.

=> [PAUL] The paragraph explaining about the Extended Information Model is rewritten as follows:

OLD:

This section covers the additional information associated with the system messages. The extended information model is only for the structured data such as events, record, and counters. Any unstructured data is specified with the basic information model only.

Each information has characteristics as follows:

* Acquisition method: ...

NEW:

The extended information model is the specific monitoring data that covers the additional information associated with the detailed information of status and performance of the network and the NSF over the basic information model. The extended information combined with the basic information creates the monitoring information (i.e., I2NSF Event, Record, and Counter).

The extended monitoring information has characteristics for data collection setting as follows:

* Acquisition method: ...

--

The final sentence of this section suggests that the dampening type can be set by the user of the monitoring system. But all occurrences of dampening-type in the below descriptions say either "dampening-type: on-repetition" or "dampening-type: none", which implies that for each type of alarm, only a particular value of dampening-type is allowed.

=> [PAUL] We updated the “dampening-type” to make sure that it allows for two choices.

NEW:

* acquisition-method: subscription
* emission-type: on-change
* dampening-type: on-repetition or no-dampening

Also "dampening-type: none" is invalid (as it is undefined in the model in section 9) and "dampening-type: no-dampening" is probably Intended.

=> [PAUL] The “dampening-type: none” is updated to “dampening-type: no-dampening”.

6.1.1. Memory Alarm

* **severity: The severity of the alarm such as critical, high, medium, and low.**

"such as" implies that there may be other values, whereas section 5 states that there are exactly 4 severities and section 9 agrees. You need to decide what the rule is and align all descriptions of "security" data to that rule.

=> [PAUL] We updated the description for severity to clarify that the severity level value can be only one among of critical, high, medium, or low. The update is as follows:

OLD:

```
   *  severity: The severity of the alarm such as critical, high,
      medium, and low.
```

NEW:

```
   *  severity: The severity level of the message. There are total four
      levels, i.e., critical, high, middle, and low.
```

6.2.2. Configuration Change

Should there be components of the event that describe what change was made to the configuration? The examples for "message" only distinguish creating a new configuration vs. modifying an existing configuration, but that information seems to me to be inadequate for any significant security monitoring.

=> [PAUL] We added a new field to describe the changes made to the configuration. A “list changes” is used to describe the changes that were made to the configuration. The minimum information that should be given is the name of the policy that has been altered. Any detailed information about the changes can be done by extending the list and is up to the implementation. The update is as the following:

NEW:

```
Section 6.2.2 Configuration Change

   *  changes: Describes the modification that was made to the
      configuration. The minimum information that must be provided is the
      name of the policy that has been altered (added, modified, or
      removed). Other detailed information about the configuration changes
      is up to the implementation.


Section 8. YANG Data Model

  case i2nsf-system-detection-event {
    container i2nsf-system-detection-event {
      description
        "This notification is sent when a security-sensitive
         authentication action fails.";
      leaf event-category {
        type identityref {
          base system-event;
        }
        description
          "The event category for system-detection-event";
      }
      uses characteristics;
      uses i2nsf-system-event-type-content;
      uses common-monitoring-data;
```

```
      list changes {
        key policy-name;
        description
          "Describes the modification that was made to the
           configuration. The minimum information that must be
           provided is the name of the policy that has been
           altered (added, modified, or removed).

           This list can be extended with the detailed
           information about the specific changes made to the
           configuration based on the implementation.";

        leaf policy-name {
          type leafref {
            path
              "/nsfintf:i2nsf-security-policy"
             +"/nsfintf:system-policy-name";
          }
          description
            "The name of the policy configuration that has been
             added, modified, or removed.";
        }
      }
    }
  }
```

6.2.3. Session Table Event

The following information should be included in a Session Table Event:

Is "session table event" a known term of art?

=> [PAUL] The explanation has been added for Session Table Event as follows:

OLD:

6.2.3. Session Table Event

The following information should be included in a Session Table Event:

NEW

6.2.3. Session Table Event

Session Table Event is the event triggered by the session table of an NSF. A session table holds the information of the current active sessions. The following information should be included in a Session Table Event:

6.2.4. Traffic Flows

* arrival-rate: Arrival rate of packets of the traffic flow.

Most data for "packets per second" have a twin datum for "bytes per second". Should there be an "arrival-speed" datum for traffic flows?
=> [PAUL] We add “arrival-speed” to the “Traffic Flows” event. The addition is as follows:

NEW:

Section 6.2.4. Traffic Flows

* arrival-rate: Arrival rate of packets of the traffic flow in packet per second.
* arrival-speed: Arrival rate of packets of the traffic flow in bytes per second.

Section 8. YANG Data Model

```
  case i2nsf-traffic-flows {
    container i2nsf-traffic-flows {
      ...
      leaf arrival-speed {
        type uint32;
        units "Bps";
        description
          "The average arrival rate of the flow in bytes per
           second. The average is calculated from the start of
           the NSF service until the generation of this
           record.";
      }
      uses characteristics;
      uses common-monitoring-data;
    }
  }
```

6.3.1. DDoS Detection

* attack-type: Any one of SYN flood, ACK flood, SYN-ACK flood, FIN/RST flood, TCP Connection flood, UDP flood, ICMP flood, HTTPS flood, HTTP flood, DNS query flood, DNS reply flood, SIP flood, SSL flood, and NTP amplification flood.

The module definition gives a fixed set of attack-types, but given that there are 14 described types, it seems likely that additional types will be defined. Some extension mechanism needs to be used, either a catch-all extension type or recogition that users will define additional types.
=> [PAUL] We update the description of “attack-type” according to your comments as follows:

OLD:
attack-type: Any one of SYN flood, ACK flood, SYN-ACK flood, FIN/RST flood, TCP Connection flood, UDP flood, ICMP flood, HTTPS flood, HTTP flood, DNS query flood, DNS reply flood, SIP flood, SSL flood, and NTP amplification flood.

NEW:
attack-type: The type of DDoS Attack, i.e., SYN flood, ACK flood, SYN-ACK flood, FIN/RST flood, TCP Connection flood, UDP flood, ICMP flood, HTTPS flood, HTTP flood, DNS query flood, DNS reply flood, SIP flood, SSL flood, and NTP amplification flood. This can be extended with additional types of DDoS attack.

*** end-time: The time stamp indicating when the attack ended. If the attack is still undergoing when sending out the alarm, this field can be empty.**

The Yang definition seems to make this field mandatory and provide no null value. Perhaps making it optional in the model is the best way of modeling the desired semantics.

=> [PAUL] We updated the data model to make the “end-time” field is optional. The update is as follows:

OLD:

leaf end-time {
type yang:date-and-time;
mandatory true;
description
"The time stamp indicating when the attack ended";
}

NEW:

leaf end-time {
type yang:date-and-time;
description
"The time stamp indicating when the attack ended. If the attack is still undergoing when sending out the notification, this field can be empty.";
}

6.3.2. Virus Event

It's not clear whether this event is for when a virus is found within a packet flow or for when it is found within a host system. Are there two different types of virus events for these? Or does each type use

a subset of the fields of one common event schema?
=> [PAUL] This is used when a virus is detected within a packet flow and host system. As it is possible that the virus somehow infected the host system, the NSF can send a notification using the file-type, file-name, and os fields. We added this explanation in the beginning of the section. We also added fields to properly identify the infected host as follows:

OLD:
6.3.2. Virus Event The following information should be included in a Virus Event: * event-name: detection-virus. * virus: Type of the virus. e.g., trojan, worm, macro virus type. * virus-name: Name of the virus. * dst-ip: The destination IP address of the flow where the virus is found. * src-ip: The source IP address of the flow where the virus is found. * src-port: The source port of the flow where the virus is found. * dst-port: The destination port of the flow where the virus is found. * src-location: The geographical location (e.g., country and city) of the src-ip field. * dst-location: The geographical location (e.g., country and city) of the dst-ip field. * os: The operating system of the host that has the virus. * file-type: The type of the file where the virus is hidden. * file-name: The name of the file where the virus is hidden. * raw-info: The information describing the packet triggering the event. * rule-name: The name of the rule being triggered.

NEW:
6.3.2. Virus Event

This information is used when a virus is detected within the traffic flow or inside the host. The following information should be included in a Virus Event:

- event-name: detection-virus.
- virus-name: Name of the virus.
- virus-type: Type of the virus. e.g., trojan, worm, macro virus type.
- dst-ip: The destination IP address of the flow where the virus is found. This is used when the virus is detected within the traffic flow.
- src-ip: The source IP address of the flow where the virus is found. This is used when the virus is detected within the traffic flow.
- src-port: The source port of the flow where the virus is found. This is used when the virus is detected within the traffic flow.
- dst-port: The destination port of the flow where the virus is found. This is used when the virus is detected within the traffic flow.
- src-location: The geographical location (e.g., country and city) of the src-ip field. This is used when the virus is detected within the traffic flow.
- dst-location: The geographical location (e.g., country and city) of the dst-ip field. This is used when the virus is detected within the traffic flow.
- host: The name or IP address of the host/device that is infected by the virus. This is used when the virus is detected within a host system. If the given name is not IP address, the name can be an arbitrary string including FQDN (Fully Qualified Domain Name). The name MUST be unique in the scope of management domain for identifying the device that has been infected with a virus.
- os: The operating system of the host that has the virus. This is used when the virus is detected within a host system.
- file-type: The type of the file where the virus is hidden. This is used when the virus is detected within a host system.
- file-name: The name of the file where the virus is hidden. This is used when the virus is detected within a host system.
- raw-info: The information describing the flow triggering the event.
- rule-name: The name of the rule being triggered.

```
Section 8. YANG Data Model

  leaf host {
    type union {
      type string;
      type inet:ip-address-no-zone;
    }
    description
      "The name or IP address of the host/device. This is
       used to identify the host/device that is infected by
       the virus. If the given name is not IP address, the
       name can be an arbitrary string including FQDN
       (Fully Qualified Domain Name). The name MUST be unique
       in the scope of management domain for identifying the
       device that has been infected with a virus.";
  }
```

**   * virus: Type of the virus. e.g., trojan, worm, macro virus type.**

It seems like this datum should be named "virus-type". Also, it seems unlikely that virus types form a definitive taxonomy, so this field should be considered less important than "virus-name" (which is likely to be a key into a database of known viruses).

=> [PAUL] We updated the name to “virus-type”. We also change the order of the information to emphasize the virus-name is more important than the virus-type as follows:

OLD:

```
6.3.2 Virus Event

   The following information should be included in a Virus Event:

   *  event-name: detection-virus.

   *  virus: Type of the virus. e.g., trojan, worm, macro
      virus type.

   *  virus-name: Name of the virus.
```

NEW:

```
6.3.2 Virus Event

   The following information should be included in a Virus Event:

   *  event-name: detection-virus.

   *  virus-name: Name of the virus.
```

* virus-type: Type of the virus. e.g., trojan, worm, and macro virus.

6.3.3. Intrusion Event

*** event-name: The name of the event. e.g., detection-intrusion.**

Why is there not a single, definitive event-name value for intrusion events? Or was "i.e." meant rather than "e.g."?
=> [PAUL] We change the “event-name” description to be synchronized with the other “event-name” in the document. The update is as follows:

OLD:
* event-name: ~~The name of the event. e.g.,~~ detection-intrusion.

NEW:
* event-name: detection-intrusion.

*** raw-info: The information describing the flow triggering the event.**

Given there are 8 defined fields that describe the flow, what additional information can raw-info contain? Also, the semantics of this raw-info is different from that in 6.3.2.
=> [PAUL] As the usage of “raw-info” is unknown, and the functionality of the information is also unknown, we removed the “raw-info” field from the document.

Indeed, there is some disalignment in the description of this field:

In section 6, raw-info is listed for:
6.3.2 Virus Event
6.3.3 Intrustion event

In section 8, raw-info is listed for:
i2nsf-nsf-detection-ddos
i2nsf-nsf-detection-virus
i2nsf-nsf-detection-intrusion
i2nsf-nsf-detection-web-attack
i2nsf-nsf-detection-voip-volte

In the model in section 9, raw-info is listed as a component of:
i2nsf-nsf-detection-ddos

grouping i2nsf-nsf-event-type-content, which is used in
i2nsf-nsf-detection-virus
i2nsf-nsf-detection-intrusion
i2nsf-nsf-detection-web-attack
i2nsf-nsf-detection-voip-volte

Only in section is raw-info described as "describing the flow triggering the event".
=> [PAUL] We removed the “raw-info” as the functionality of the field is unknown.

6.3.4. Web Attack Event

* **event-name: The name of event. e.g., detection-web-attack.**

Why is there not a single, definitive event-name value for intrusion events? Or was "i.e." meant rather than "e.g."?

=> [PAUL] We change the “event-name” description to be synchronized with the other “event-name” in the document. The update is as follows:

OLD:
* event-name: ~~The name of the event. e.g.,~~ detection-web-attack.

NEW:
* event-name: detection-web-attack.

* **cookies: The HTTP Set-Cookie header field of the response.**

I would think the cookies header in the request would be of more interest than the cookies header in the response.
=> [PAUL] We updated the description in Section 6.3.4 and in the YANG module according to your comments as follows:

OLD:
Section 6.3.4 cookies: The HTTP Set-Cookie header field of the response. Section 9. YANG Data Model leaf cookies { type string; description "The HTTP Set-Cookie header field of the response";

```
    reference
      "RFC 6265: HTTP State Management Mechanism -
       Set-Cookie";
  }
```

NEW:

```
Section 6.3.4

  cookies: The HTTP Cookie header field of the request from the
  user agent.

Section 8. YANG Data Model

  leaf cookie {
    type string;
    description
      "The HTTP Cookie header field of the request from the
       user agent.";
    reference
      "RFC 6265: HTTP State Management Mechanism - Cookie";
  }
```

6.3.5. VoIP/VoLTE Event

This event type has no event-type field. Is that correct?

=> [PAUL] We added the missing event-name field as “event-name: detection-voip-volte” to synchronize it with the other information in the document.

6.4.3. User Activity Log

6.4.1 and 6.4.3 are only weakly aligned with each other, despite that they describe login and activities of two types of users (administrators and ordinary users). Should these types be unified, or at least their fields compared to better align them?

=> [PAUL] We updated the user identifying information for User Activity and Access Logs in a unified way according to your comments as follows:

OLD:

```
Section 6.4.1

   Access logs record administrators' login, logout, and operations on a
   device.  By analyzing them, security vulnerabilities can be
   identified.  The following information should be included in an
   operation report:
```

* username: The username that operates on the device.

* login-ip: IP address used by an administrator to log in.

* login-role: The login role to specify the privilege level of the user account, e.g., administrator, user, and guest.

* operation-type: The operation type that the administrator execute, e.g., login, logout, configuration, and other.

* input: The operation performed by a user after login. The operation is a command given by a user.

* output: The result after executing the input.

Section 6.4.3 User Activity Log

User activity logs provide visibility into users' online records (such as login time, online/lockout duration, and login IP addresses) and the actions that users perform. User activity reports are helpful to identify exceptions during a user's login and network access activities.

* user: Name of a user.

* group: Group to which a user belongs.

* login-ip-addr: Login IP address of a user.

* authentication: The method to verify the valid user, i.e., pre-configured-key and certificate-authority.

* online-duration: The duration of a user's activeness (stays in login) during a session.

* logout-duration: The duration of a user's inactiveness (not in login) from the last session.

* additional-info: Additional Information for login:

    1. type: User activities. e.g., Successful User Login, Failed Login attempts, User Logout, Successful User Password Change, Failed User Password Change, User Lockout, and User Unlocking.

    2. cause: Cause of a failed user activity.

NEW:

Section 6.4.1 Access Log

Access logs record administrators' login, logout, and operations on a device. By analyzing them, security vulnerabilities can be

identified. The following information should be included in an operation report:

* identity: The information to identify the user. The minimum information (extensible) that should be included:

  1. user: The unique username that attempted access violation.

  2. group: Group(s) to which a user belongs. A user can belong to multiple groups.

  3. ip-address: The IP address of the user that triggered the event.

  4. port-number: The port number used by the user.

* authentication: The method to verify the valid user, i.e., pre-configured-key and certificate-authority.

* operation-type: The operation type that the administrator execute, e.g., login, logout, configuration, and other.

* input: The operation performed by a user after login. The operation is a command given by a user.

* output: The result after executing the input.

Section 6.4.3 User Activity Log

User activity logs provide visibility into users' online records (such as login time, online/lockout duration, and login IP addresses) and the actions that users perform. User activity reports are helpful to identify exceptions during a user's login and network access activities. This information should be included in a user's activity report:

* identity: The information to identify the user. The minimum information (extensible) that should be included:

  1. user: The unique username that attempted access violation.

  2. group: Group(s) to which a user belongs. A user can belong to multiple groups.

  3. ip-address: The IP address of the user that triggered the event.

  4. port-number: The port number used by the user.

* authentication: The method to verify the valid user, i.e., pre-configured-key and certificate-authority.

* online-duration: The duration of a user's activeness (stays in

login) during a session.

* logout-duration: The duration of a user's inactiveness (not in login) from the last session.

* additional-info: Additional Information for login:

1. type: User activities. e.g., Successful User Login, Failed Login attempts, User Logout, Successful User Password Change, Failed User Password Change, User Lockout, and User Unlocking.

2. cause: Cause of a failed user activity.

6.7.2. Policy Hit Counter

* **hit-times: The hit times that the security policy matches the specified traffic.**

Given the Yang definition, I think the wording you want here is "The number of times that the security policy ...".
=> [PAUL] We updated the description as “hit-times: The number of times that the security policy matches the specified traffic.”, following your suggestion.

7. NSF Monitoring Management in I2NSF

It's not clear to me that any of this is needed for the definition of the data model. It seems more to be a higher-level description of the entire I2NSF system, but the details of the data model aren't directly relevant to the higher-level description (as long as the data model provides the required fields) and that the data model isn't directly affected by the higher-level I2NSF system.
=> [PAUL] As you mentioned, the section is not relevant to the details of the data model. We remove this Section as the information is not necessary in this document.

9. YANG Data Model

The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this document are to be interpreted as described in BCP 14 (RFC 2119) (RFC 8174) when, and only when, they appear in all capitals, as shown here.

This module contains the full RFC 8174 text, but the only use of it is the instance of MUST in nsf-name.
=> [PAUL] The usage of the full RFC 8174 text is needed in the YANG module even though only “MUST” is used. The validator returns the following error when the full text is not given.

"ietf-i2nsf-nsf-monitoring.yang:59: warning: the module seems to use RFC 2119 keywords, but the required text from RFC 8174 is not found."

Therefore, we put RFC 8174's full text in the YANG module.

10. I2NSF Event Stream

The following example

It seems like this should start a new paragraph. The preceding text is an overview of the event stream, but the following text is a single example. The example is not what I would expect; it is not an event. I think this text would better state the purpose:

The following example XML shows the capabilities of the event streams generated by an NSF (e.g., "NETCONF" and "I2NSF-Monitoring" event streams) for the subscription of an NSF data collector. The XML examples in this document ...

=> [PAUL] We updated the sentence following your suggestion with a new paragraph as follows:

OLD:
The following example shows the capabilities of the event streams of an NSF (e.g., "NETCONF" and "I2NSF-Monitoring" event streams) by the subscription of an NSF data collector; note that this example XML file is delivered by an NSF to an NSF data collector. The XML examples in this document follow the line breaks as per [RFC8792].

NEW:
The following XML example shows the capabilities of the event streams generated by an NSF (e.g., "NETCONF" and "I2NSF-Monitoring" event streams) for the subscription of an NSF data collector. The XML examples in this document follow the line breaks as per [RFC8792].

--

```
<replayLogCreationTime>
  2021-04-29T09:37:39+00:00
</replayLogCreationTime>
```

It's not clear to me "2021-04-29T09:37:39+00:00" is a value that is applicable to all NSF event streams.

=> [PAUL] The value "2021-04-29T09:37:39+00:00" is just an example to complete the XML. The "replayLogCreationTime" element is to indicate the earliest available logged notification. The detailed explanation can be available from RFC 5277. As this section is intended to only introduce the necessary requirement for subscribing to the I2NSF

monitoring information, we added a sentence to refer to RFC 5277 for the detailed explanation on Event Streams and XML.

NEW:
The following XML example shows the capabilities of the event streams generated by an NSF (e.g., "NETCONF" and "I2NSF-Monitoring" event streams) for the subscription of an NSF data collector. Refer to [RFC5277] for more detailed explanation of Event Streams. The XML examples in this document follow the line breaks as per [RFC8792].

15. Contributors

Chaehong Chung Department of Electronic, Electrical and Computer Engineering Sungkyunkwan University 2066 Seo-ro Jangan-gu Suwon, Gyeonggi-do 16419 Republic of Korea EMail: darkhong@skku.edu

[and others]

For clarity, between the name and the affiliation should be a comma or dash.

=> [PAUL] We separate the contributor name and affiliation with a dash (-) for clarity.

[END]

Reviewer: Valery Smyslov
Review result: Ready with Issues

I am the assigned ART directorate reviewer for this document. These comments were written primarily for the benefit of the ART area directors. Document editors and WG chairs should treat these comments just like any other last call comments.

The document defines an information model and the YANG data model for an interface used for monitoring Network Security Functions in the I2NSF framework.

Issues.

1. The YANG Data Model contains human-readable strings, like "src-user", "message", etc. From description of thees fields they seem to contain a free-form text with no indication in which language it is written. Section 4.2 of BCP 18 requires that protocols that transfer text MUST provide for carrying information about the language of that text (e.g. via language tags).
=> [PAUL] We provide a new field to identify the human language used in the information. The field is as follows:

NEW:

```
Section 5. Basic Information Model for Monitoring Data

   o  language: It describes the human language intended for the user, so
      that it allows a user to differentiate the language that is used
      in the notification.  This field is not mandatory, but required
      when the implementation provides more than one human language for
      the human-readable string fields.


Section 8. YANG Data Model

  notification i2nsf-event {
    description
      "Notification for I2NSF Event.";

    leaf language {
      type string {
        pattern
            "^((en-GB-oed|i-ami|i-bnn|i-default|"
          + "i-enochian|i-hak|i-klingon|i-lux|i-mingo|i-navajo|i-pwn|"
          + "i-tao|i-tay|i-tsu|sgn-BE-FR|sgn-BE-NL|sgn-CH-DE)|"
          + "(art-lojban|cel-gaulish|no-bok|no-nyn|zh-guoyu|zh-hakka|"
          + "zh-min|zh-min-nan|zh-xiang)|"
          + "(([A-Za-z]{2,3}(-[A-Za-z]{3}(-[A-Za-z]{3}){0,2})?)|"
          + "[A-Za-z]{4}|[A-Za-z]{5,8}"
          + "(-[A-Za-z]{4})?"
          + "(-[A-Za-z]{2}|[0-9]{3})?"
          + "(-[A-Za-z0-9]{5,8}|[0-9][A-Za-z0-9]{3})*"
          + "(-[0-9A-WY-Za-wy-z](-[A-Za-z0-9]{2,8})+)*"
          + "(-x(-[A-Za-z0-9]{1,8})+)?)|"
          + "x(-[A-Za-z0-9]{1,8})+)$";
      }
      description
        "The value in this field describes the human language
         intended for the user, so that it allows a user to
         differentiate the language that is used in the
         notification. This field is not mandatory, but required
         when the implementation provides more than one human
         language for the human-readable string fields,
         e.g., /i2nsf-nsf-event/i2nsf-nsf-detection-ddos/message.

         This field uses the language-tag production in Section 2.1
         in RFC 5646. See the document for more details.";
      reference
        "RFC 5646: Tags for Identifying Languages";
    }

    ...
  }

  notification i2nsf-log {
    description
```

```
    "Notification for I2NSF log. The notification is generated
     from the logs of the NSF.";

  leaf language {
    type string {
      pattern
          "^((en-GB-oed|i-ami|i-bnn|i-default|"
        + "i-enochian|i-hak|i-klingon|i-lux|i-mingo|i-navajo|i-pwn|"
        + "i-tao|i-tay|i-tsu|sgn-BE-FR|sgn-BE-NL|sgn-CH-DE)|"
        + "(art-lojban|cel-gaulish|no-bok|no-nyn|zh-guoyu|zh-hakka|"
        + "zh-min|zh-min-nan|zh-xiang)|"
        + "(([A-Za-z]{2,3}(-[A-Za-z]{3}(-[A-Za-z]{3}){0,2})?)|"
        + "[A-Za-z]{4}|[A-Za-z]{5,8}"
        + "(-[A-Za-z]{4})?"
        + "(-[A-Za-z]{2}|[0-9]{3})?"
        + "(-[A-Za-z0-9]{5,8}|[0-9][A-Za-z0-9]{3})*"
        + "(-[0-9A-WY-Za-wy-z](-[A-Za-z0-9]{2,8})+)*"
        + "(-x(-[A-Za-z0-9]{1,8})+)?)|"
        + "x(-[A-Za-z0-9]{1,8})+)$";
    }
    description
      "The value in this field describes the human language
       intended for the user, so that it allows a user to
       differentiate the language that is used in the
       notification. This field is not mandatory, but required
       when the implementation provides more than one human
       language for the human-readable string fields,
       e.g., /i2nsf-nsf-log/i2nsf-system-res-util-log/message.

       This field uses the language-tag production in Section 2.1
       in RFC 5646. See the document for more details.";
    reference
      "RFC 5646: Tags for Identifying Languages";
  }

  ...
}

container i2nsf-counters {
  config false;
  description
    "The state data representing continuous value changes of
     information elements that occur very frequently. The value
     should be calculated from the start of the service of the
     NSF.";

  leaf language {
    type string {
      pattern
          "^((en-GB-oed|i-ami|i-bnn|i-default|"
        + "i-enochian|i-hak|i-klingon|i-lux|i-mingo|i-navajo|i-pwn|"
        + "i-tao|i-tay|i-tsu|sgn-BE-FR|sgn-BE-NL|sgn-CH-DE)|"
        + "(art-lojban|cel-gaulish|no-bok|no-nyn|zh-guoyu|zh-hakka|"
        + "zh-min|zh-min-nan|zh-xiang)|"
        + "(([A-Za-z]{2,3}(-[A-Za-z]{3}(-[A-Za-z]{3}){0,2})?)|"
```

```
          + "[A-Za-z]{4}|[A-Za-z]{5,8}"
          + "(-[A-Za-z]{4})?"
          + "(-[A-Za-z]{2}|[0-9]{3})?"
          + "(-[A-Za-z0-9]{5,8}|[0-9][A-Za-z0-9]{3})*"
          + "(-[0-9A-WY-Za-wy-z](-[A-Za-z0-9]{2,8})+)*"
          + "(-x(-[A-Za-z0-9]{1,8})+)?)|"
          + "x(-[A-Za-z0-9]{1,8})+)$";
      }
      description
        "The value in this field describes the human language
         intended for the user, so that it allows a user to
         differentiate the language that is used in the
         notification. This field is not mandatory, but required
         when the implementation provides more than one human
         language for the human-readable string fields,
         e.g., /i2nsf-counters/system-interface/message.

         This field uses the language-tag production in Section 2.1
         in RFC 5646. See the document for more details.";
      reference
        "RFC 5646: Tags for Identifying Languages";
    }
    ...
  }
```

2. There are a number of 32-bit counters in the model. In high performance networks they would wrap around after a relatively short period of time. It is not clear how this situation is handled.

=> [PAUL] To prevent the counters from wrapping around quickly, the type has been replaced with 64-bit for counters.

Reviewer: Melinda Shore
Review result: Not Ready

I've marked this "not ready" only because of the quality of the writing, which is both unidiomatic and ungrammatical throughout the document. But, the draft has been through working group last call, and if the working group is good with it, I'm good with it - I'm here to do a security review, and it's basically fine in that regard.

=> [PAUL] During the revision with comments from other reviewers, I believe that the quality of the writing of this document is significantly improved.

A couple of nits:

In section 6, it seems to me that by having two different dampening messages you risk having both no-dampening and on-repetition active at the same time (implementers don't always make good decisions). Setting on-repetition to an impossible value (say, -1) could serve the same purpose as no-dampening and

avoid possible implementation errors.
=> [PAUL] dampening-type is string, which can have “on-repetition” or “no-dampening”. A wrong setting for “on-repetition” can be avoided by the careful implementation of this YANG data module. Also, dampening-period for on-repetition has the type of uint32, so it has at least zero.

I'm curious why you’re monitoring system things (cpu, disk), since presumably those are also being monitored elsewhere.
=> [PAUL] It is possible that the system resources (CPU, memory, etc.) are being monitored in different ways. But we still provide their monitoring data in the module as they are common ones to be monitored in a device. If these fields are missing, it may seem that the module does not fully provide the monitoring information that can be obtained from an NSF.

In the security considerations section you may want to discuss some of the limitations of relying on the transport protocol to protect the data, particularly around data authenticity, etc.
=> [PAUL] To handle the limitations of transport protocols (e.g., NETCONF and RESTCONF) for monitoring data, there is a discussion for secure transport protocols (e.g., SSH and HTTPS) for the protection and authentication of monitoring data as follows.

> The YANG module described in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the required secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the required secure transport is TLS [RFC8446].
>
> The NETCONF access control model [RFC8341] provides a means of restricting access to specific NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

Two thoughts.

The YANG module imports from RFC8343 which must then be a Normative Reference but I cannot see such.
=> [PAUL] We have added RFC 8343 to the Normative Reference of the draft.

The URL in the YANG module for contact should be 'datatracker' and not 'tools'
=> [PAUL] We have updated the URL to “datatracker”.

Tom Petch

Thanks for your valuable comments.

Best Regards,
Jaehoon (Paul) Jeong