Unauthorised Guide to RSA Security Analytics (aka NetWitness) 10.6

by Timothy A. Underhay

Copyright © 2022

This publication is licensed under Creative Commons Attribution-NoDerivatives 4.0 International.

Disclaimer

Although the author has made every effort to ensure that the information in this book was correct at press time and whilst this publication is designed to provide accurate information in regard to the subject matter covered, the author assumes no responsibility for errors, inaccuracies, omissions, or any other inconsistencies herein and hereby disclaim any liability to any party for any and all loss, damage, or disruption caused by any information contained herein.

By using this publication, you agree to hold the author blameless for any and all damage, loss, or disruption perceived to be caused by the use of any information contained herein.

USE AT YOUR OWN RISK!

Creative Commons Attribution-NoDerivatives 4.0 International Public License

By exercising the Licensed Rights (defined below), You accept and agree to be bound by the terms and conditions of this Creative Commons Attribution-NoDerivatives 4.0 International Public License ("Public License"). To the extent this Public License may be interpreted as a contract, You are granted the Licensed Rights in consideration of Your acceptance of these terms and conditions, and the Licensor grants You such rights in consideration of benefits the Licensor receives from making the Licensed Material available under these terms and conditions.

Section 1 – Definitions.

a. **Adapted Material** means material subject to Copyright and Similar Rights that is derived from or based upon the Licensed Material and in which the Licensed Material is translated, altered, arranged, transformed, or otherwise modified in a manner requiring permission under the Copyright and Similar Rights held by the Licensor. For purposes of this Public License, where the Licensed Material is a musical work, performance, or sound recording, Adapted Material is always produced where the Licensed Material is synched in timed relation with a moving image.
b. **Copyright and Similar Rights** means copyright and/or similar rights closely related to copyright including, without limitation, performance, broadcast, sound recording, and Sui Generis Database Rights, without regard to how the rights are labeled or categorized. For purposes of this Public License, the rights specified in Section 2(b)(1)-(2) are not Copyright and Similar Rights.
c. **Effective Technological Measures** means those measures that, in the absence of proper authority, may not be circumvented under laws fulfilling obligations under Article 11 of the WIPO Copyright Treaty adopted on December 20, 1996, and/or similar international agreements.
d. **Exceptions and Limitations** means fair use, fair dealing, and/or any other exception or limitation to Copyright and Similar Rights that applies to Your use of the Licensed Material.
e. **Licensed Material** means the artistic or literary work, database, or other material to which the Licensor applied this Public License.
f. **Licensed Rights** means the rights granted to You subject to the terms and conditions of this Public License, which are limited to all Copyright and Similar Rights that apply to Your use of the Licensed Material and that the Licensor has authority to license.
g. **Licensor** means the individual(s) or entity(ies) granting rights under this Public License.
h. **Share** means to provide material to the public by any means or process that requires permission under the Licensed Rights, such as reproduction, public display, public performance, distribution, dissemination, communication, or importation, and to make material available to the public including in ways that members of the public may access the material from a place and at a time individually chosen by them.
i. **Sui Generis Database Rights** means rights other than copyright resulting from Directive 96/9/EC of the European Parliament and of the Council of 11 March 1996 on the legal protection of databases, as amended and/or succeeded, as well as other essentially equivalent rights anywhere in the world.
j. **You** means the individual or entity exercising the Licensed Rights under this Public License. **Your** has a corresponding meaning.

Section 2 – Scope.

a. **License grant**.
    1. Subject to the terms and conditions of this Public License, the Licensor hereby grants You a worldwide, royalty-free, non-sublicensable, non-exclusive, irrevocable license to exercise the Licensed Rights in the Licensed Material to:
        A. reproduce and Share the Licensed Material, in whole or in part; and
        B. produce and reproduce, but not Share, Adapted Material.
    2. Exceptions and Limitations. For the avoidance of doubt, where Exceptions and Limitations apply to Your use, this Public License does not apply, and You do not need to comply with its terms and conditions.
    3. Term. The term of this Public License is specified in Section 6(a).
    4. Media and formats; technical modifications allowed. The Licensor authorizes You to exercise the Licensed Rights in all media and formats whether now known or hereafter created, and to make technical modifications necessary to do so. The Licensor waives and/or agrees not to assert any right or authority to forbid You from making technical modifications necessary to exercise the Licensed Rights, including technical modifications necessary to circumvent Effective Technological Measures. For purposes of this Public License, simply making modifications authorized by this Section 2(a)(4) never produces Adapted Material.
    5. Downstream recipients.
        A. Offer from the Licensor – Licensed Material. Every recipient of the Licensed Material automatically receives an offer from the Licensor to exercise the Licensed Rights under the terms and conditions of this Public License.
        B. No downstream restrictions. You may not offer or impose any additional or different terms or conditions on, or apply any Effective Technological Measures to, the Licensed Material if doing so restricts exercise of the Licensed Rights by any recipient of the Licensed Material.
    6. No endorsement. Nothing in this Public License constitutes or may be construed as permission to assert or imply that You are, or that Your use of the Licensed Material is, connected with, or sponsored, endorsed, or granted official status by, the Licensor or others designated to receive attribution as provided in Section 3(a)(1)(A)(i).
b. **Other rights**.

    1. Moral rights, such as the right of integrity, are not licensed under this Public License, nor are publicity, privacy, and/or other similar personality rights; however, to the extent possible, the Licensor waives and/or agrees not to assert any such rights held by the Licensor to

the limited extent necessary to allow You to exercise the Licensed Rights, but not otherwise.
2. Patent and trademark rights are not licensed under this Public License.
3. To the extent possible, the Licensor waives any right to collect royalties from You for the exercise of the Licensed Rights, whether directly or through a collecting society under any voluntary or waivable statutory or compulsory licensing scheme. In all other cases the Licensor expressly reserves any right to collect such royalties.

Section 3 – License Conditions.

Your exercise of the Licensed Rights is expressly made subject to the following conditions.

a. **Attribution**.

1. If You Share the Licensed Material, You must:

A. retain the following if it is supplied by the Licensor with the Licensed Material:
i. identification of the creator(s) of the Licensed Material and any others designated to receive attribution, in any reasonable manner requested by the Licensor (including by pseudonym if designated);
ii. a copyright notice;
iii. a notice that refers to this Public License;
iv. a notice that refers to the disclaimer of warranties;
v. a URI or hyperlink to the Licensed Material to the extent reasonably practicable;
B. indicate if You modified the Licensed Material and retain an indication of any previous modifications; and
C. indicate the Licensed Material is licensed under this Public License, and include the text of, or the URI or hyperlink to, this Public License.

For the avoidance of doubt, You do not have permission under this Public License to Share Adapted Material.
2. You may satisfy the conditions in Section 3(a)(1) in any reasonable manner based on the medium, means, and context in which You Share the Licensed Material. For example, it may be reasonable to satisfy the conditions by providing a URI or hyperlink to a resource that includes the required information.
3. If requested by the Licensor, You must remove any of the information required by Section 3(a)(1)(A) to the extent reasonably practicable.

Section 4 – Sui Generis Database Rights.

Where the Licensed Rights include Sui Generis Database Rights that apply to Your use of the Licensed Material:

a. for the avoidance of doubt, Section 2(a)(1) grants You the right to extract, reuse, reproduce, and Share all or a substantial portion of the contents of the database, provided You do not Share Adapted Material;
b. if You include all or a substantial portion of the database contents in a database in which You have Sui Generis Database Rights, then the database in which You have Sui Generis Database Rights (but not its individual contents) is Adapted Material; and
c. You must comply with the conditions in Section 3(a) if You Share all or a substantial portion of the contents of the database.

For the avoidance of doubt, this Section 4 supplements and does not replace Your obligations under this Public License where the Licensed Rights include other Copyright and Similar Rights.

Section 5 – Disclaimer of Warranties and Limitation of Liability.

a. **Unless otherwise separately undertaken by the Licensor, to the extent possible, the Licensor offers the Licensed Material as-is and as-available, and makes no representations or warranties of any kind concerning the Licensed Material, whether express, implied, statutory, or other. This includes, without limitation, warranties of title, merchantability, fitness for a particular purpose, non-infringement, absence of latent or other defects, accuracy, or the presence or absence of errors, whether or not known or discoverable. Where disclaimers of warranties are not allowed in full or in part, this disclaimer may not apply to You.**
b. **To the extent possible, in no event will the Licensor be liable to You on any legal theory (including, without limitation, negligence) or otherwise for any direct, special, indirect, incidental, consequential, punitive, exemplary, or other losses, costs, expenses, or damages arising out of this Public License or use of the Licensed Material, even if the Licensor has been advised of the possibility of such losses, costs, expenses, or damages. Where a limitation of liability is not allowed in full or in part, this limitation may not apply to You.**

c. The disclaimer of warranties and limitation of liability provided above shall be interpreted in a manner that, to the extent possible, most closely approximates an absolute disclaimer and waiver of all liability.

Section 6 – Term and Termination.

a. This Public License applies for the term of the Copyright and Similar Rights licensed here. However, if You fail to comply with this Public License, then Your rights under this Public License terminate automatically.
b. Where Your right to use the Licensed Material has terminated under Section 6(a), it reinstates:

1. automatically as of the date the violation is cured, provided it is cured within 30 days of Your discovery of the violation; or
2. upon express reinstatement by the Licensor.

For the avoidance of doubt, this Section 6(b) does not affect any right the Licensor may have to seek remedies for Your violations of this Public License.
c. For the avoidance of doubt, the Licensor may also offer the Licensed Material under separate terms or conditions or stop distributing the Licensed Material at any time; however, doing so will not terminate this Public License.
d. Sections 1, 5, 6, 7, and 8 survive termination of this Public License.

Section 7 – Other Terms and Conditions.

a. The Licensor shall not be bound by any additional or different terms or conditions communicated by You unless expressly agreed.
b. Any arrangements, understandings, or agreements regarding the Licensed Material not stated herein are separate from and independent of the terms and conditions of this Public License.

Section 8 – Interpretation.

a. For the avoidance of doubt, this Public License does not, and shall not be interpreted to, reduce, limit, restrict, or impose conditions on any use of the Licensed Material that could lawfully be made without permission under this Public License.
b. To the extent possible, if any provision of this Public License is deemed unenforceable, it shall be automatically reformed to the minimum extent necessary to make it enforceable. If the provision cannot be reformed, it shall be severed from this Public License without affecting the enforceability of the remaining terms and conditions.
c. No term or condition of this Public License will be waived and no failure to comply consented to unless expressly agreed to by the Licensor.
d. Nothing in this Public License constitutes or may be interpreted as a limitation upon, or waiver of, any privileges and immunities that apply to the Licensor or You, including from the legal processes of any jurisdiction or authority.

Table of contents

1. Foreward
2. About This Course
3. Foundational Concepts
4. Base Components of SA
5. SA User Interface - First Glance
6. SA UI - Administration
7. Building an SA Host From Nil
8. SA Platform (Hardware & OS)
9. Upgrades
10. SA UI - Investigation
11. SA Core Internals
12. The Index
13. Packet Decoder
14. Application Rules, BPF Filters, & Feeds
15. Log Decoder
16. Log Collection
17. Concentrator
18. Broker
19. RSA Live
20. Thick Investigator Client
21. Notifications
22. SA Security
23. Data Privacy
24. Event Stream Analysis
25. Incident Management
26. Reporting
27. Dashboards
28. Licensing
29. SA Language Packs
30. Querying With the NetWitness API
31. Working With NwConsole
32. Puppet
33. Health & Wellness
34. Event Source Monitoring
35. Integrated Active Directory Authentication
36. Audit Logging

1. Foreward

What is this book?

In the spring of 2016, the (foolish) ambition took hold of me to write, a theretofore non-existent, comprehensive training course and compilation of knowledge for the RSA [1] Security Analytics (SA) platform. SA was complex, confusing, and wielded by knowledgeable minds, eminently powerful. Unfortunately, running this leviathan of a security platform requires not just security knowledge, but also a generous mix of proprietary and system administration knowledge.

Having been on the inside of this platform for the better part of six years, first having joined NetWitness, and then post-acquisition RSA, and then subsequently without the further "inconvenience" of regular paid remuneration, I set off. A year and change later, the task was finished.

This is the written fruit of those labours.

What Happened?

I failed to consider one very salient possibility -- that despite manifold conversations to the contrary, not a single company was willing to actually pay a small third-party services company for training on a product they had just spent tens or even hundreds of thousands of dollars in purchasing. I travelled the world trying to sell it. There were ChromeBooks for an entire mobile classroom, and a rather cool transport case which gathers dust in my garage to this very day. **I couldn't even give it away for free.**

The course withered on the vine, and has sat in a backup file for five plus years. What a blunder!

Discussions were held with an, ahem, large, unnamed company with the ostensible goal of delivering this training as a partner. Dismayingly, a meeting set up in good faith quickly degenerated into a petty intel-gathering session, replete with barely-veiled threats. One person genuinely sounded as though they were crying. Of all the meetings of my career, this was more surreal than most. Rather than see it as the opportunity it was, to deliver the comprehensive training that their customers and own employees were asking for, and to enable them for success, this group chose to take it as a personal insult, acting as if their jobs were being threatened.

This book is what that training services team of that rather large company didn't want their customers to see.

Why Now?

Flash-forward to 2022. Why publish a book now for a product version anchored in 2016? I've become aware that other than the occasional software project or Github contribution, I need to share more of the knowledge that I've been blessed with having the opportunity to gather. And secondly, it's still packed with loads of info that's relevant to the NetWitness of 2022. Not much of the Linux stuff, mind you -- CentOS 6 is history, Puppet is long since gone, and new parts have been bolted on to the NetWitness ecosystem.

Why Not Update It?

Four reasons:

1. It would take too long.
2. There is no financial incentive in so doing.
3. Other than my free and open-source project Atomic Fishbowl (see https://github.com/KensingtonTech/atomic-fishbowl), I haven't worked in the packet forensics business for years. I now work in the SOAR product space.
4. My wife would kill me.

Other than this foreward, this book is untouched from the training course's final form. Sorry, but the lab environment is also long gone, but all references to it have been left in tact. It would be ludicrously obsolete by now, anyway.

Disclaimer and Release

This book is not authorised by RSA Security or its parent company.

Nothing herein shall be construed to constitute official guidance on any matter concerning the RSA Security Analytics or NetWitness platforms. By continuing to use this guide, you agree to hold Kensington Technology Associates, Limited, and Timothy Underhay blameless for any and all faults or

1. Security Analytics was originally known as NetWitness, and subsequent to this book was renamed back to NetWitness. Make up your bloody mind, RSA!

2. About This Course

I Want You To...

- Become proficient Administrators and Operators of RSA Security Analytics.
- Understand how the various pieces of SA fit together.
- Open fewer support cases because you knew what to look for, or at the very least make your support case more expeditious.
- Have the same or better formation in Security Analytics than the rest of the industry.
- Open up your careers with new skills in a very hot security industry and threat landscape.

What This Course Does

- It teaches students the purpose of Security Analytics and what problems it tries to solve.
- Teaches how Security Analytics works.
- Gives the student hands-on experience with the Security Analytics product.
- Provides an insider's perspective on the product and its history.

Target Audience

- SA Administrators, both new and somewhat seasoned.
- New and prospective SA customers.
- Professionals with good technical proficiency.
- Linux/Unix skills on the command line are a major plus.

What This Course Doesn't Teach

Security Analytics is an enormous platform, so we aim to pack as much into this four-day course as we possibly can. There will still be some items that the allotted time won't allow for:

- Linux/Unix skills - some knowledge is assumed.
- Networking skills - knowledge of the OSI model and TCP/IP is assumed, including frames, packets, protocols, and ports.
- Analytical and hunting skills, beyond a few basic examples.
- Advanced analytics training, such as ESA, Esper rules and syntax.
- Security Analytics Archiver.
- RSA Warehouse & Data Science Modules.
- Extensive Log Collection configuration.
- Disk operations.
- Custom Parser development.

Classroom Rules

- We ask for your consideration of the learning environment. Your day job is extremely important, but please let's agree to stow and silence laptops and phones whilst class is in session.
- There will be a ten minute break approximately every 1.5 hours. Please let's stick to the timer.
- Lunch will be two hours to allow for your work commitments.
- If you absolutely must take a call, please do so outside of the classroom.
- Recording of any kind by a student is **strictly prohibited**.
- All proprietary and/or confidential information learned by the instructor is confidential and protected under the contract governing this course.

Lab Environment

In an effort to provide the most realistic experience possible for our students, every student will have access to his/her own virtualised lab environment, running nearly all parts of the RSA Security Analytics product stack. This is hosted in a cloud environment accessible over VPN via the Chromebook situated at your desk. Please be sure to use the same laptop for the duration of the course, so as to minimise the chance of cross-contamination of labs.

You will find instructions for connecting to your lab environment next to your Chromebook.

In case you're wondering why we're using Chromebooks, the Chromebook is a lightweight solution with a lightweight, Linux-like OS with the perfect web browser for our work - Chrome. They provide all the necessary tools for working with our lab environment.

ASK QUESTIONS!

It's active discussion that makes this course engaging, insightful, and beneficial. I encourage you to ask questions at any time! I may want to table a question until a more apropos time in the course to discuss it, but I will make every attempt to do so. If you think of a question during a break (maybe you're too shy to ask), write it on a sticky note and put it on the question area.

3. Foundational Concepts

Constituent Components of Security Analytics

RSA Security Analytics (herein referred to as 'SA') is comprised of a number of building blocks which enable the capture, parsing, indexing, querying, reporting on, and display of captured security data.

Types of Ingested Security Data

SA is concerned with two basic types of data: **packet data** and **log data**. It is capable of capturing streams of either of these data types. Packet data can be captured from any point in your organisation's network, inside or outside the firewall. Log Data can in theory come from any type of computer source: network devices, security devices, network services and servers, and even SA services themselves.

Steps of SA Data Processing

All data which finds its way into SA goes through certain stages before it can be used for detecting threats:

1. **Capture** - is the process of reading packet or log data off of the wire and into memory
2. **Assembly** - the process of separating packet data and log data into individual sessions
3. **Parsing** - The process of generating meta from assembled packet or log streams
4. **Rules & feeds** - Generate additional meta on sessions based on other values, such as a rule or a feed from an external data source (e.g. a list of users, DHCP feed, etc). Can also be used to truncate or drop traffic based on certain characteristics.
5. **Database storage** - Save packet, log, meta, and session data to disk for future retrieval.
6. **Indexing** - Create a searchable index of metadata which can be used to quickly drill through and find sessions matching certain characteristics

What is Meta?

Put simply, meta is data that describes data. **Meta is the most fundamental forensic unit of the SA architecture**. It is through drilling through and searching on this meta that most security threats are discovered with SA. It can be nearly any characteristic of your network or log traffic. Some examples of meta are:

- Source IP address
- Destination IP Address
- MAC address
- TCP or UDP
- Event source type
- Username
- Service name
- Source city
- Destination Country
- Service Type (HTTP, SSH, HTTPS, SMTP, etc)
- User Agent
- OS Type

Service Type (13 values)
DNS (>100,000 - 64%) - OTHER (93,857) - HTTP (37,888) - SSL (12,611) - POP3 (5,811) - DHCP (119) - FTP (85) - NETBIOS (63) - RTP (31) - SMTP (23) - SSH (12) - GNUTELLA (2) - AOL IM (2)

Hostname Aliases (20 of 20+ values)
www.google.com (10,292) - star.c10r.facebook.com (7,769) - safebrowsing.cache.l.google.com (7,492) - hipchat.com (5,934) - sb.l.google.com (4,467) - pop.gmx.com (4,418) - 3c-bs.gmx.com (3,398) - host-90-236-3-35.mobileonline.telia.com (3,358) - mycompany.com (3,184) - clients.l.google.com (3,147) - googlemail.l.google.com (2,367) - 90.236.3.35 (2,173) - tomshardware.co.uk (1,807) - a72.dscg10.akamai.net (1,605) - a51.dscg10.akamai.net (1,598) - 8.notify.dropbox.com (1,509) - navigator-bs.gmx.com (1,430) - *.dropbox.com (1,349) - www.technologieduluth.com (1,274) - www.tomshardware.co.uk (915) ... **show more**

Source IP Address (20 of 20+ values)
192.168.0.51 (>100,000 - 75%) - 192.168.0.54 (126,531) - 192.168.0.2 (22,221) - 192.168.0.53 (15,743) - 54.235.163.229 (3,553) - 212.71.235.158 (1,363) - 67.207.202.9 (558) - 192.168.0.1 (380) - 177.71.226.28 (277) - 217.195.49.146 (116) - 217.195.49.112 (86) - 192.168.0.50 (84) - 218.77.79.43 (64) - 217.195.49.103 (45) - 188.138.17.205 (36) - 61.91.45.50 (31) - 95.199.31.163 (30) - 103.10.197.187 (29) - 209.130.172.178 (27) - 95.213.143.180 (24) ... **show more**

Destination IP address (20 of 20+ values)
192.168.0.1 (>100,000 - 66%) - 8.8.8.8 (13,585) - 192.168.0.2 (7,481) - 31.13.64.1 (6,265) - 192.168.0.54 (5,735) - 212.227.17.187 (3,396) - 64.233.161.108 (3,297) - 208.239.76.34 (3,212) - 31.13.91.2 (2,947) - 217.72.201.130 (2,945) - 212.227.17.171 (2,513) - 224.0.0.252 (2,311) - 80.239.178.178 (1,750) - 80.239.178.187 (1,588) - 213.155.151.151 (1,352) - 213.155.151.149 (1,288) - 31.13.93.3 (1,219) - 213.155.151.155 (1,104) - 212.227.111.53 (1,085) - 213.155.151.150 (1,066) ... **show more**

Source IPv6 Address (7 values)
fe80:0:0:0:b536:88f0:ac41:1d7a (2,418) - fe80:0:0:0:eef4:bbff:fe4f:b245 (31) - fe80:0:0:0:260:6eff:fe42:b635 (22) - 0:0:0:0:0:0:0:0 (22) - fe80:0:0:0:eef4:bbff:fe4f:c0d7 (6) - fe80:0:0:0:eef4:bbff:fe51:89ec (3) - fe80:0:0:0:eef4:bbff:fe4f:c07e (3)

Destination IPv6 address (12 values)
ff02:0:0:0:0:0:1:3 (2,316) - ff02:0:0:0:0:0:0:fb (48) - ff02:0:0:0:0:0:0:c (39) - ff02:0:0:0:0:0:0:16 (35) - ff02:0:0:0:0:0:1:2 (27) - ff02:0:0:0:0:0:0:2 (16) - ff02:0:0:0:0:1:ff41:1d7a (8) - ff02:0:0:0:0:0:0:1 (8) - ff02:0:0:0:0:1:ff4f:c0d7 (3) - ff02:0:0:0:0:1:ff4f:b245 (3) - ff02:0:0:0:0:1:ff51:89ec (1) - ff02:0:0:0:0:1:ff4f:c07e (1)

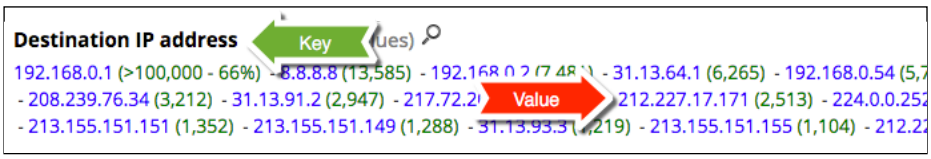
Action Event (5 values)
get (32,170) - put (5,549) - login (1,491) - sendfrom (174) - sendto (173)

As you can see, meta in SA is directional, meaning that it is capable of differentiating between the the source and destination sides of a session, and can optionally be configured to separate meta into separate source and destination categories. For example, every session will have a source and destination IP address. SA has a meta category (called a 'key' or 'meta key', or in legacy parlance a 'language key') called ip.src to identify the IP address for the source side of a session, and a second key called ip.dst to identify the destination side of a session.

With any SA installation, there will be a complete set of out-of-the-box meta keys and values that you can instantly start drilling on.

Keys and Values, You Say?

Every meta key is populated by individual meta values. 'Key' refers to the type of meta, whereas 'value' refers to the the individual drillable values contained within a given key. E.g. 'ip.src' would be a key, and '192.168.1.1' is a value. Every captured session contains a number of meta key/value pairs.



Once you're in possession of the right meta, finding threats is often just a matter of drilling through data until you find the proverbial needle in the haystack. Queries can also be run on SA's dataset based on combinations of meta, such as 'service = 22 && port.dst != 22'. You can also build rules based on meta to create additional meta.

Parsers - Where Meta Is Born

Meta is generated by **parsers**, and the process of generating meta is called **parsing**. Parsers are code which look inside sessions for specific things like protocol type, IP addresses, log fields, email subject line, encoding type, etc. The list of things which are capable of being parsed is nearly endless. SA ships with a number of parsers out-of-the-box, but additional parsers can be downloaded through a service called RSA Live, or new parsers can also be written in one of several languages.

Meta can also be generated through the use of **feeds**, which are applied during parsing through something called the *feed parser*, and through **application rules** (app rules), which are applied after parsing.

Sessions

Security Analytics receives its security data in the form of streams of either packets or logs (***technical note:*** *when referring to the capture of packets, we are in fact more specifically talking about* ***frames*** *of data, as the frame header is also captured by SA. Please keep this in mind when we are talking about 'packets'*). A stream of packets will typically contain millions of packets per day, with packets having many different source and destination IP addresses. Many conversations or 'flows' will be occurring simultaneously within this packet stream. In order to make sense of this jumble of packet data that is captured from the wire, SA employs a process called 'assembly', whereby a single large packet stream is split into many smaller discrete

streams based on certain characteristics, such as source and destination IP address, TCP flags, source and destination port, and others. In SA parlance, these smaller streams of associated traffic are called 'sessions'. Once a session has been created, it is easy to find all packets associated with a particular network conversation.

Whereas meta is the most fundamental forensic unit of SA, **the session is the most fundamental unit of data storage in SA.** Sessions contain packets and meta.

2016-06-02T16:40:59	Network	4 KB	192.168.0.2 -> 213.155.151.149	34127 -> 80
2016-06-02T16:40:59	Network	5 KB	192.168.0.2 -> 213.155.151.151	44537 -> 80
2016-06-02T16:41:04	Network	4 KB	192.168.0.51 -> 54.187.115.34	41767 -> 80
2016-06-02T16:41:20	Network	4 KB	192.168.0.51 -> 54.187.115.34	41771 -> 80
2016-06-02T16:41:29	Network	4 KB	192.168.0.51 -> 54.187.109.244	43992 -> 80
2016-06-02T16:41:40	Network	107 KB	192.168.0.53 -> 204.79.197.200	1032 -> 80

Sessions

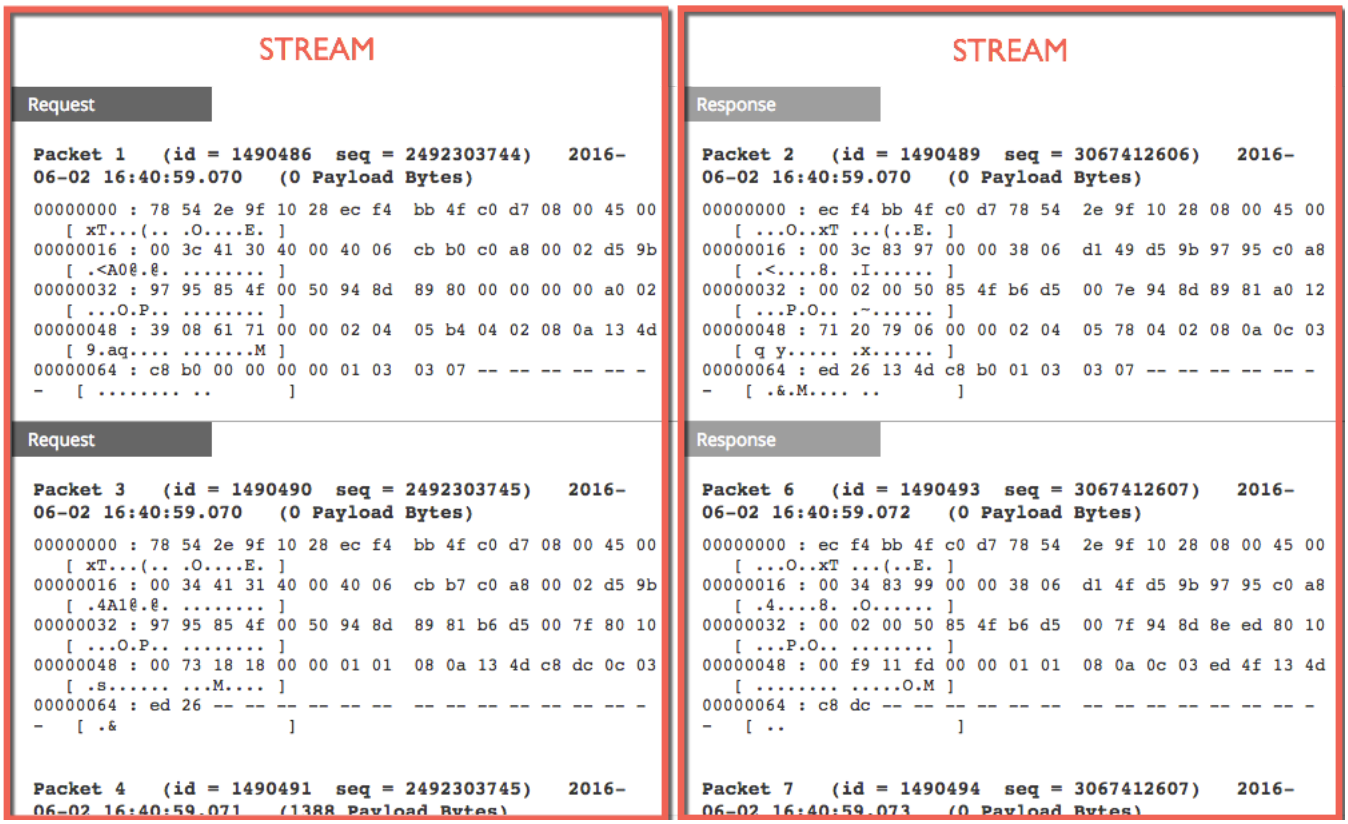
Note: For the sake of consistency, both packet and log sessions contain 'packets'. Whereas a packet session will typically have many packets of actual network data, a log session will have only contain a single 'packet' which is actually one log message.

Streams

Within packet sessions, SA also employs a concept called 'streams'. A stream describes all packets associated with one side of a network session. For instance, in a session in which host 1.1.1.1 is communicating with host 2.2.2.2, all packets sent by 1.1.1.1 will be one stream, and all packets sent by 2.2.2.2 will be the other stream. There can be either one or two streams in a given session.

A typical TCP session will have two streams, as the TCP protocol is connection-oriented and requires a three-way handshake to establish a session. Thus each side of the connection will be sending packets. The exceptions to this would be when the remote end of a connection isn't responding to a SYN packet, or if an asymmetry exists in the capture or routing infrastructure between the two endpoints. If there is a TCP session with data (packet payload) being pushed but only a single stream, this is indicative of a problem in capture (Tap) or routing.

A UDP session can have either one or two streams, as UDP is a connectionless protocol. The remote side of a UDP session is not required to respond to the source side.



Finding Stuff - The Index

The ability to capture packets and logs of the wire, parse them, and save them is useless without also having the ability to find data within the vast data set. The index is a layer that sits atop the lower-level packet, meta, and session databases, which stores a subset of meta values which can be quickly queried/searched.

The SA index stores meta keys and sometimes also values, and also stores session ID's which contain a given meta key and/or value. The index provides a fast way of finding sessions that contain given meta characteristics, and is searched through the use of **queries.**

With some edge-case exceptions, typically only Concentrators and Archivers are the only SA Core services to contain a proper index.

Stay Tuned

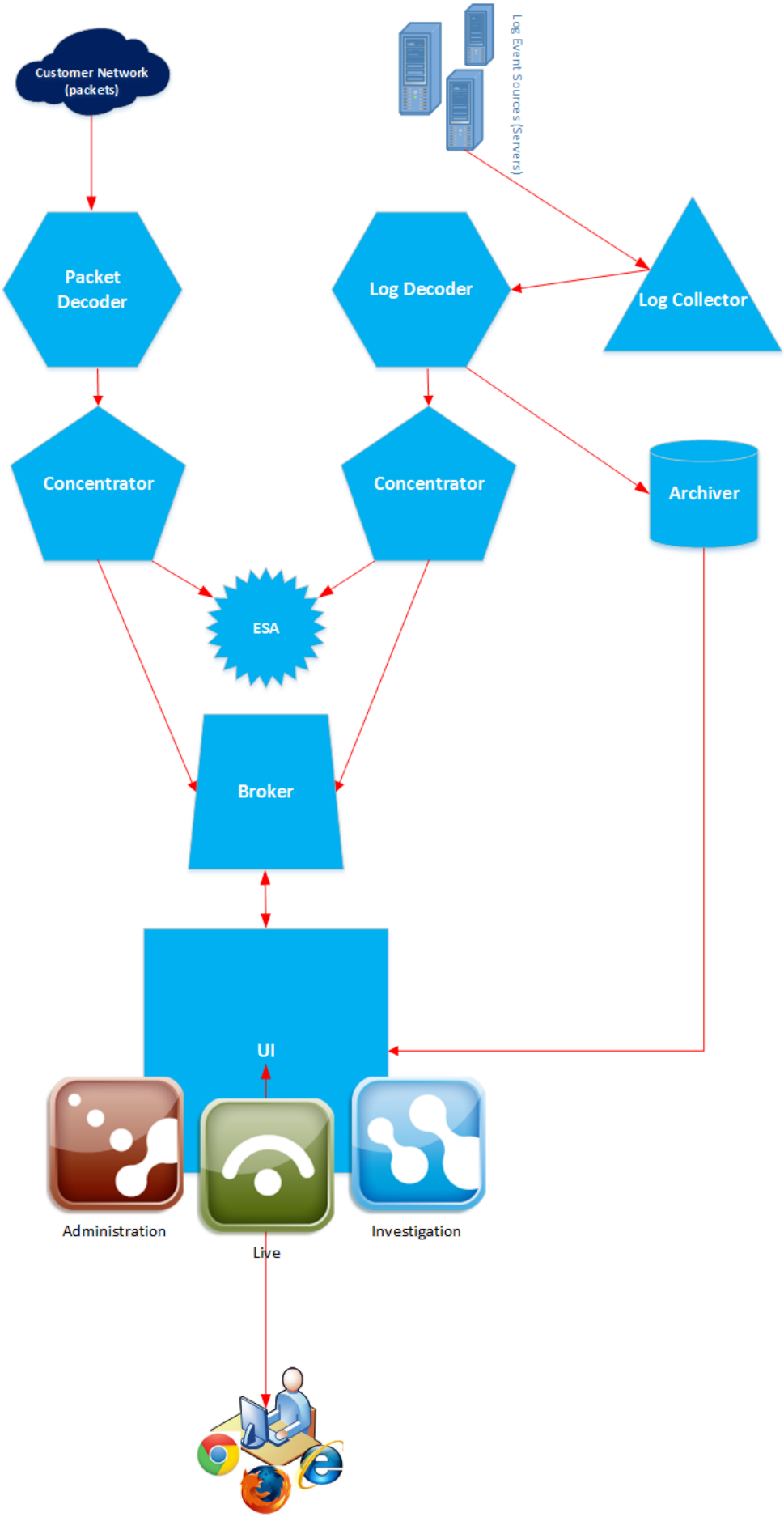
More on all of these concepts soon to follow...

4. Base Components of SA

Introduction

Chapter 1 described foundational concepts which will support the understanding of the SA product stack. This chapter builds on that knowledge to reveal where these concepts are implemented in the stack.

Security Analytics is a suite of interconnected services which work together to enable organisations to ultimately find security threats in their midst.



SA Stack

SA Core Services

The base components of SA, which collect data and enable its storage and indexing, are known as **SA Core**. You might sometimes hear these referred to as 'NextGen', as NetWitness NextGen was the original name of the product before NetWitness was purchased by RSA. SA Core is essentially the original NetWitness product stack (with some new additions), though it has been updated considerably since the NetWitness acquisition of 2011. SA

Core is made up of these component services:

Packet Decoder (or simply 'Decoder') - Is responsible for capturing packet data off the wire, then parsing and storing the packets and also meta. This is the highest component in the SA product stack.

Log Decoder (LD) - Similar to Packet Decoder, but captures log Data instead of packet data. Instead using a NIC in promiscuous mode, Log Decoder runs a syslog receiver. Its parsing mechanism uses RSA enVision log parsers rather than the LUA parsers used by Packet Decoder (though technically it is capable of running LUA parsers as well, though this is seldom done in practice).

Log Collector - Responsible for collecting logs and forwarding them to either a Log Decoder or another downstream Log Collector. There is always a Log Collector co-located on the same host as a Log Decoder - this is called a Local Collector. Log Collectors which are not co-located with a Log Decoder are called either Remote Log Collectors (RLC's) or Virtual Log Collectors (VLC's). A remote collector always sends its data to a local collector rather than sending it directly to the Log Decoder via syslog.

Log Collectors can collect logs from both syslog and non-syslog event sources such as Check Point, Netflow, ODBC, Windows, and more.

Concentrator - Is the SA service responsible for indexing meta and servicing queries against it. Concentrators connect to one or more upstream Decoders or Log Decoders and collect the meta stored in the Decoder's meta database. The process of transferring meta from an upstream SA Core service is called **aggregation**.

Broker - Enables the querying of multiple upstream Concentrators (or even other Brokers) simultaneously. This allows an Analyst to focus an investigation across broad parts of an SA deployment. A Broker is a fairly lightweight service, and is essentially a query proxy. Brokers can reside on their own SA server, and are sometimes packaged with the SA GUI or Malware Analysis services.

Archiver - One of the newest members of the SA Core stack, Archiver is similar to a Concentrator in that it contains an index but also stores packet and log data. Its purpose is to facilitate long-term storage of customer data. It supports a tiered storage model, in which data can be stored in either hot, warm, or cold storage.

Warehouse Connector - Is responsible for transferring meta data and optionally Log Data to an RSA Warehouse (a Hadoop cluster running either MapR or Pivotal HD).

Legacy Appliance Service - An almost-dead holdover from the NetWitness days, the Appliance Service was responsible for providing certain low-level information on the underlying hardware, such as physical or logical disk state. It also provides an interface for running certain tasks on the host OS, such as setting the system clock, NTP server, configuring syslog forwarding, etc. **In practice, the Appliance Service is hardly used at all today**, but has technically not been removed from the product stack. Most of its functionality has been supplanted by newer functionality in the SA product, and is likely to be removed in the future.

Running SA Core Services

Since CentOS 6 was adopted for SA, all SA Core services run under the Upstart system daemon ('start', 'stop', 'restart', etc). SA Core services can also be run directly from the command line for debugging. E.g. to run the Decoder service directly from the command line, with all log output going to the console, simply execute /usr/sbin/NwDecoder.

SA Java Services

Additional SA services which sit atop SA Core have no formal designation as such, but for the sake of discussion we shall call them SA Java Services, for the simple reason that they are mostly written in Java. This is distinct from the SA Core Services, which are written in C++. SA's Java services are as follows:

The UI (or GUI) - Provides web front-end interface for Security Analytics, and is what you login to with your web browser to interact with the SA product. This runs under jettysrv. We will refer to it as "the UI". This service also interacts with a lot of 'glue' services, which are described below. You will sometimes hear this erroneously referred to as either a Broker or just 'SA'; 'Broker' because the UI host will often have a Broker service installed on it.

RSA Live - Technically is not a separate service and is part of the UI service. It's a significant part of SA, nonetheless, and deserves mention here. RSA Live (formerly NetWitness Live) is a content distribution system provided by RSA whereby content like feeds, parsers, rules, and reports can be downloaded and pushed to your SA infrastructure.

Reporting Engine (RE) - The Reporting Engine is a service which is co-located with the UI. It is responsible for generating charts, alerts, and graphs on its configured data sources. A RE data source can be either an SA Core service, RSA Warehouse (specifically Hadoop Hive), or an IPDB database.

Event Stream Analysis (ESA) - ESA (sometimes pronounced "ee-sah") is SA's correlation engine. It employs a complex but powerful event processing engine and language called Esper to detect events that simple application rules cannot.

Incident Management (IM) - IM is a SOC workflow management tool for creating and prioritising security incidents.

Malware Analysis (MA) - Originally called NetWitness Spectrum, SA Malware Analysis is a tool for detecting malware being transferred over the wire. It integrates with packet Decoders and uses signature-less IOC's (indicators of compromise) for detection. It can also make use of sandbox technology, and community (online) resources for determining if a file is malicious.

MA comes in two flavours - **co-located** or **full**. The co-located version is installed on the UI server by default if it is an RSA appliance. It is a light version of full MA which can only handle ad hoc file submissions rather than processing files automatically as they are captured by SA. The full version of MA operates on the stream of sessions captured by SA and does not require files to be submitted manually by an analyst. Most SA customers are not using

either, as SA's Malware Analysis has not seen significant updates in some time, and there are serious competitors in the network Malware detection marketplace.

The 'SA' Host

The 'SA' host is sometimes called the 'head' or 'head-end', or UI host. It houses the UI, puppetmaster, SMS, Reporting Engine, Incident Management, and usually the Co-lo Malware services.

SA 'Glue' Services

Puppet / Puppet Master - Puppet is a third-party technology used for deploying and synchronising the configuration of systems and services across multiple hosts. Nearly all SA nodes run puppet. The head-end host also runs puppet master, which controls the puppet deployment. Though puppet can be modified by knowledgeable administrators, it is implemented in such as way as to have minimal end-user interaction.

MCollective - MCollective is a third-party "under the bonnet" service which is used to execute remote commands on other SA hosts by the UI. This is typically not invoked directly by an end-user except for advanced troubleshooting. Runs alongside puppet.

RabbitMQ - An open source technology used for message queueing, which implements the AMQP Protocol, or Advanced Message Queuing Protocol. Used to transport data between SA hosts, such as logs from Log Collectors, or audit logs, Health & Wellness statistics, and several other things.

System Management Service (SMS) - This is an RSA service which is tightly integrated with the UI which handles a number of back-end and administrative tasks, such as Health & Wellness, Event Source Management, and hosts and services.

CollectD - A third-party service which is used to collect statistics to be used for SA's Health & Wellness. RSA has written a number of plugins to extend collectd for SA purposes. On the SA Host, it also stores health statistics in databases called RRD's (Round-robin databases), which are used by the Health & Wellness feature to generate statistical graphs.

TokuMX - A third-party commercial implementation of MongoDB. Runs on the SA Host as a back-end for several system services, such as SMS and Puppet Master. This is typically not interacted with by end-users except in advanced troubleshooting.

'Orbital Services'

Other services which are not technically part of the SA product stack as they are managed separately, but integrate tightly Security Analytics are:

RSA ECAT - ECAT is a power endpoint security product which uses signature-less methodology to detect malware running on endpoint computers.

RSA Warehouse - RSA Warehouse is a Hadoop-based product which is used for big-data analytics on security data. It can be used with data science modules or for reporting on large data sets. Hadoop is a distributed cluster-oriented technology with a distributed file system which can be accessed by all members of the cluster simultaneously. Hadoop is designed to run on inexpensive hardware, and performance can be increased linearly by adding nodes to a cluster.

RSA originally supported a MapR-based Hadoop distribution as its Warehouse solution, but has since transitioned to Pivotal HD-based solution. Both solutions are supported by RSA.

RSA SecOps - Incident Management on steroids. SecOps runs on the RSA Archer platform. Used by SOC's to prioritise and respond to incidents and events. Implements all aspects, workflows, and personas of a typical SOC. SecOps can be configured to drill back into SA.

These three products are outside the scope of this course but deserve mention.

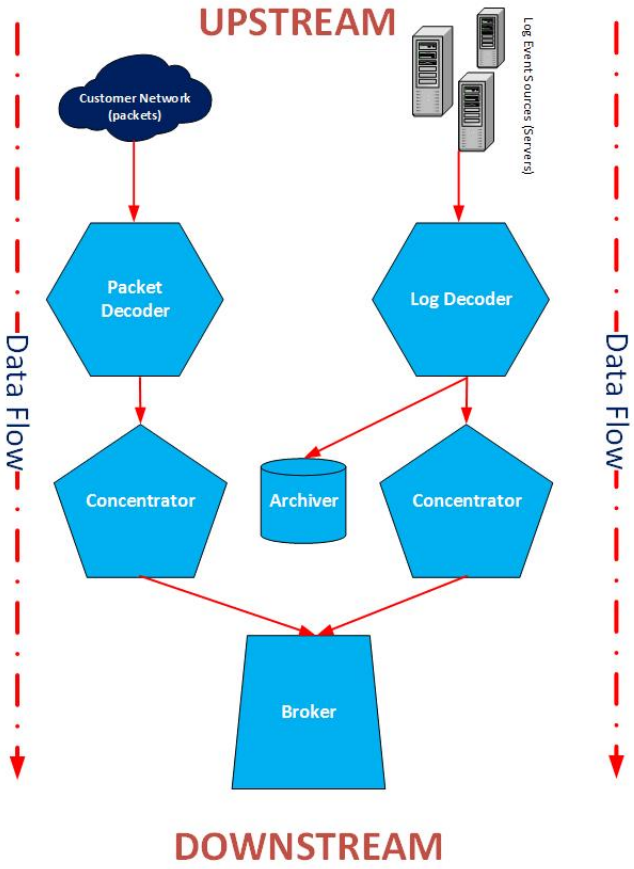
Hybrids

Hybrids are specialised hosts which run both a Decoder (or Log Decoder) service and a Concentrator service under the same OS. These are typically used in either limited-budget situations where there isn't enough money to purchase a separate Decoder/Concentrator pair, and/or in low-bandwidth situations where the expected load on the system will be quite low. There is usually a performance penalty with running a hybrid, as there is more contention for system resources. This is especially true of RSA Series 4 and Series 4S hybrids which unlike a normal Concentrator, do not have SSD's for their index.

Directional Flow of Data

Think of SA as a river of data. At the north end of the river (upstream) is your organisation's data - its network, servers, PC's, etc. This data then flows downstream into and through SA, going through a number of layers before you, the user, can extract value from it. For the purpose of discussion, in this course, we'll stick to these upstream / downstream semantics.

Customer Environment (LAN and/or Log Sources, ECAT)
Log Collector
Decoder (Log or Packet)
Concentrator / Archiver / ESA / Warehouse



Broker
GUI / Reporting Engine / Incident Management / RSA Live / Malware Analysis / IPDB Extractor *

* The IPDB Extractor service is a service provided only for the use of RSA's legacy enVision customers, and is an extreme outlier. Here, we mention its existence, but will not delve into it.

5. SA User Interface - First Glance

Before we can get to the nitty-gritty of the SA Core services, we must first have some understanding of the user interface with which we'll usually interact with it. The UI of Security Analytics is a Jettysrv-driven thin client written in Java, which has largely replaced the functionality of the former NetWitness thick clients.

The Security Analytics user interface service **has no official name,** unlike many other products (Check Point SmartDashboard, etc). You may hear it referred to as "the UI", "the GUI", "the web UI", "SA", or even "the Broker" (those last two are misnomers and shouldn't be used). In this course, we'll refer to it as "the UI". The SA UI is simply the web tier or application used for interacting with the rest of the SA product stack, for purposes of generating reports, administration, investigation, etc.

Supported Browsers

SA works best with the Google Chrome browser, and SA is extensively tested with it. Firefox support is also very good. Problems are known to exist whilst using Microsoft browsers, though the most recent versions of IE should still work, albeit slowly.

UI Performance

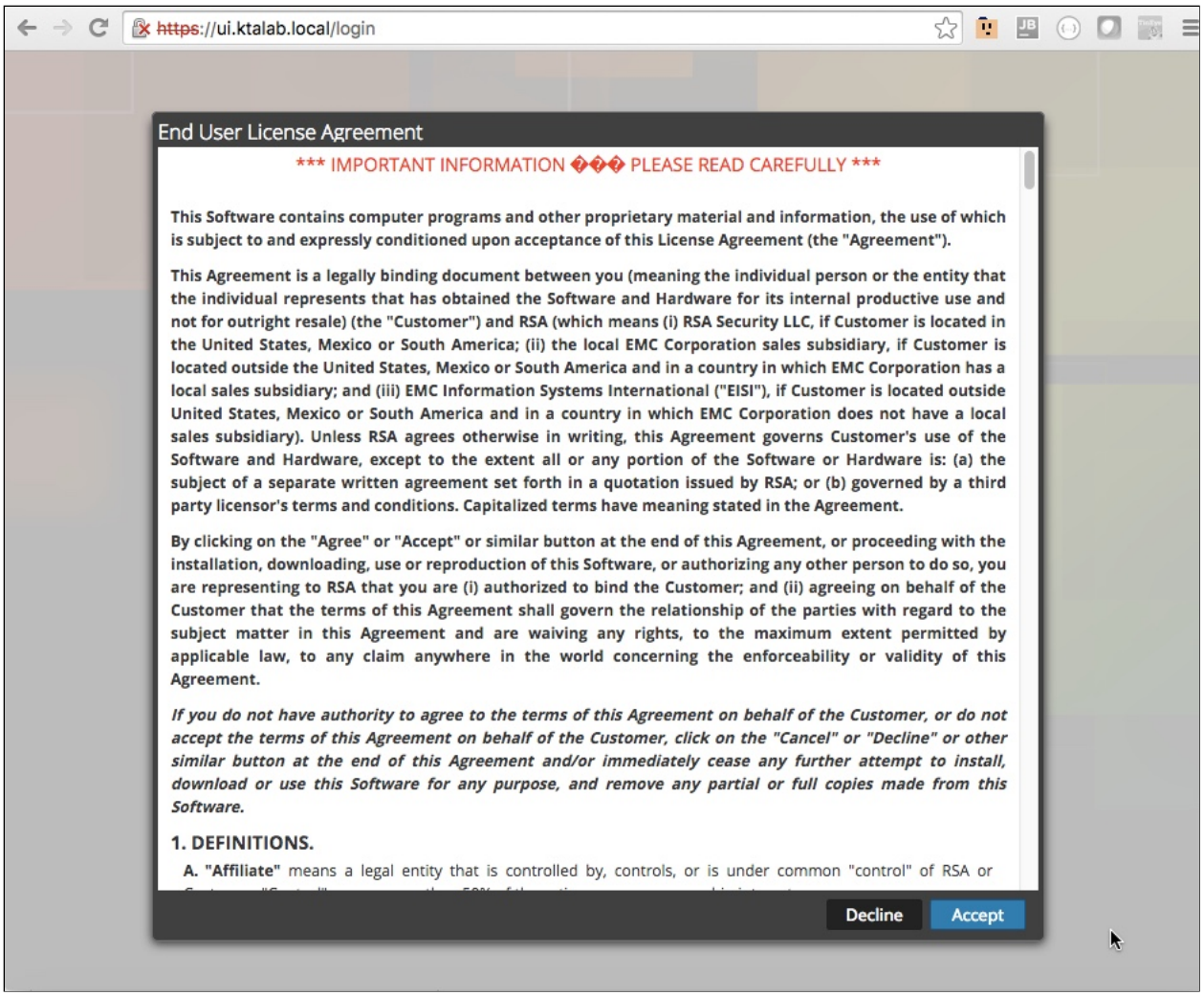
RSA does not provide guidance on how many users may simultaneously use the UI. A good rule of thumb is to keep it at 10 or below users per UI instance.

High Availability

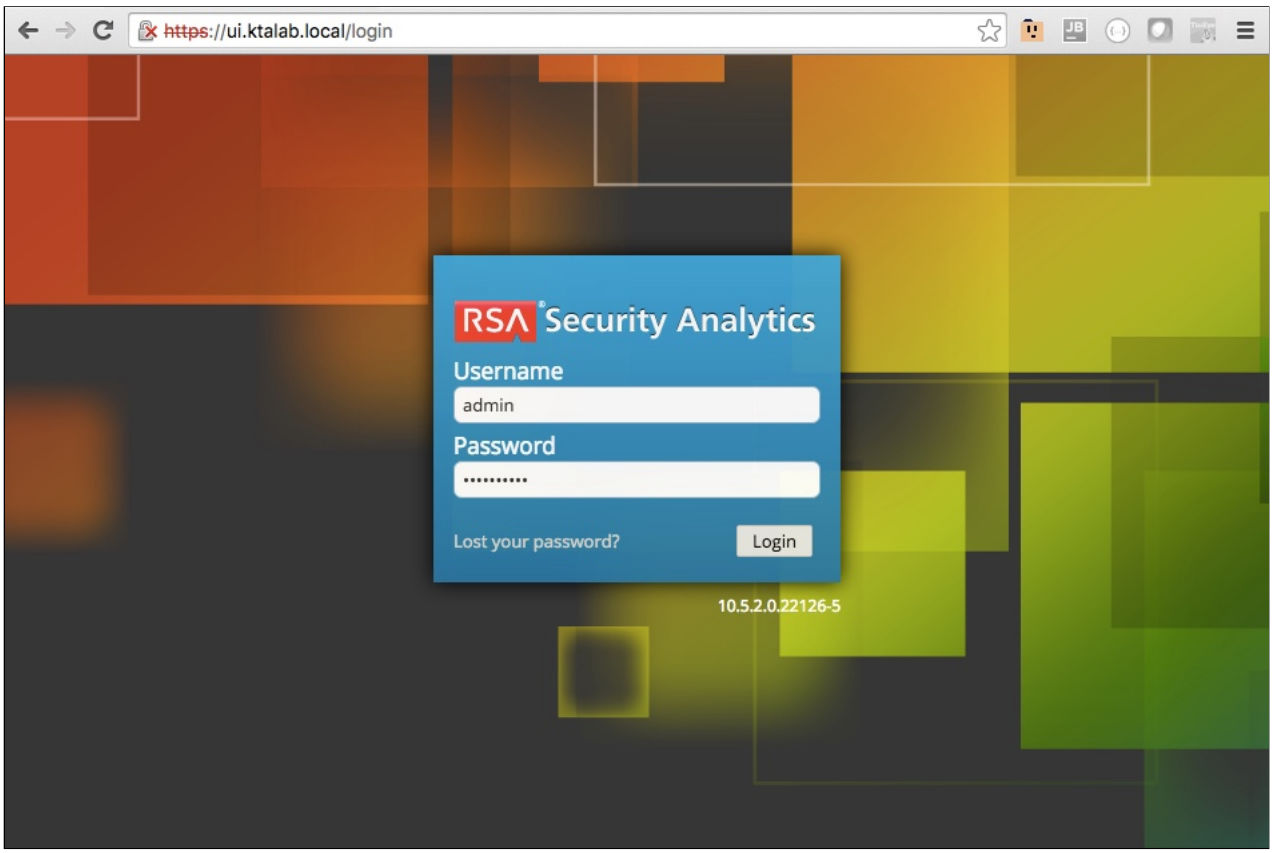
The UI unfortunately does not yet possess any high-availability functionality. If more than one instance of the UI is needed for either redundancy or performance reasons, another SA host must be provisioned. In this scenario, only one UI instance should be the master for all other SA hosts in a deployment, to simplify administration. It is not yet possible to synchronise the configuration between two UI instances, therefore all provisioned users and groups must be kept in sync manually between disparate UI instances. Secondary SA nodes should only be used for investigation and reporting. It is possible to allow SA Core services to implicitly trust connections from secondary UI instances, but this is an advanced procedure. Either Kensington Technology Associates or RSA can assist you in this matter.

Logging In

Before we delve into the UI, let's first log in to it. Open up your Chrome browser and enter **https://ui.ktalab.local** into the URL bar. You will get an SSL error about an invalid certificate. You should choose to ignore the error and continue. You'll first see an End-User License Agreement (EULA).

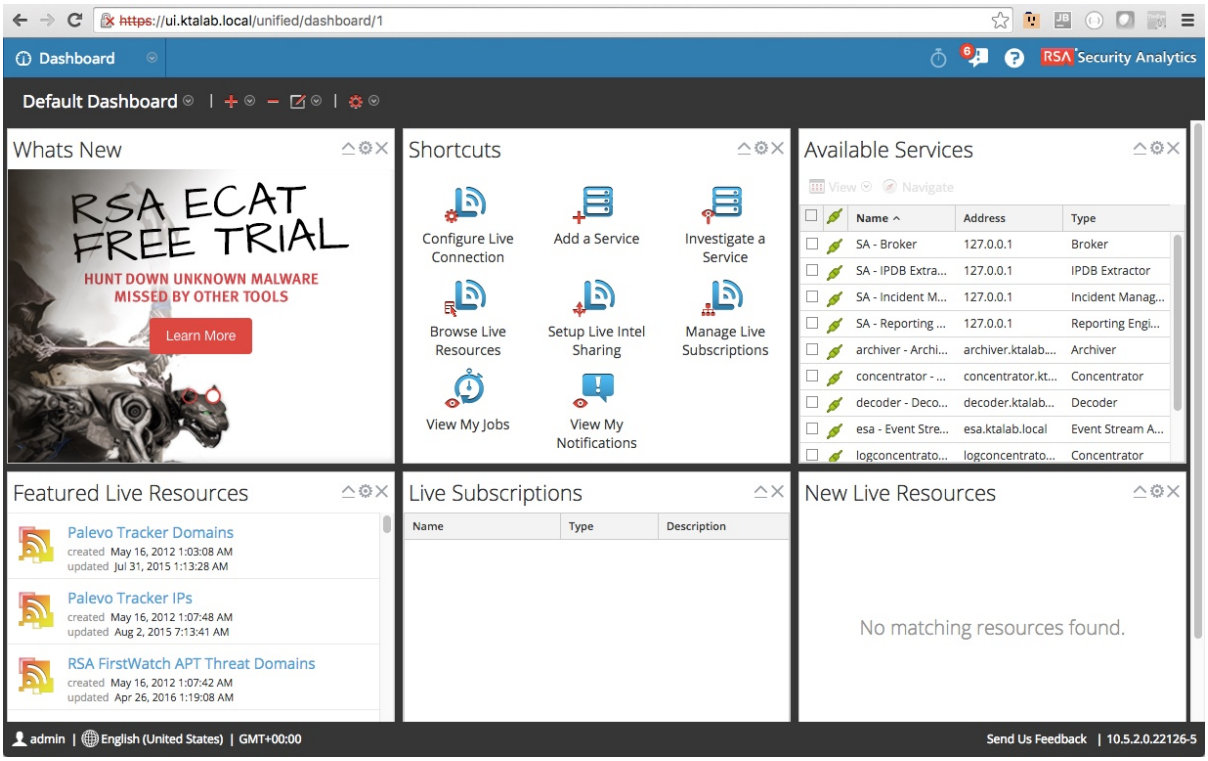


Now click the '**Accept**' button at the bottom of the dialog. You'll now see the following login screen:



Now enter the username '**admin**' and the password '**netwitness**'. These are the default administrative credentials for Security Analytics, including SA Core services. You will now be presented with the default landing page, or dashboard.

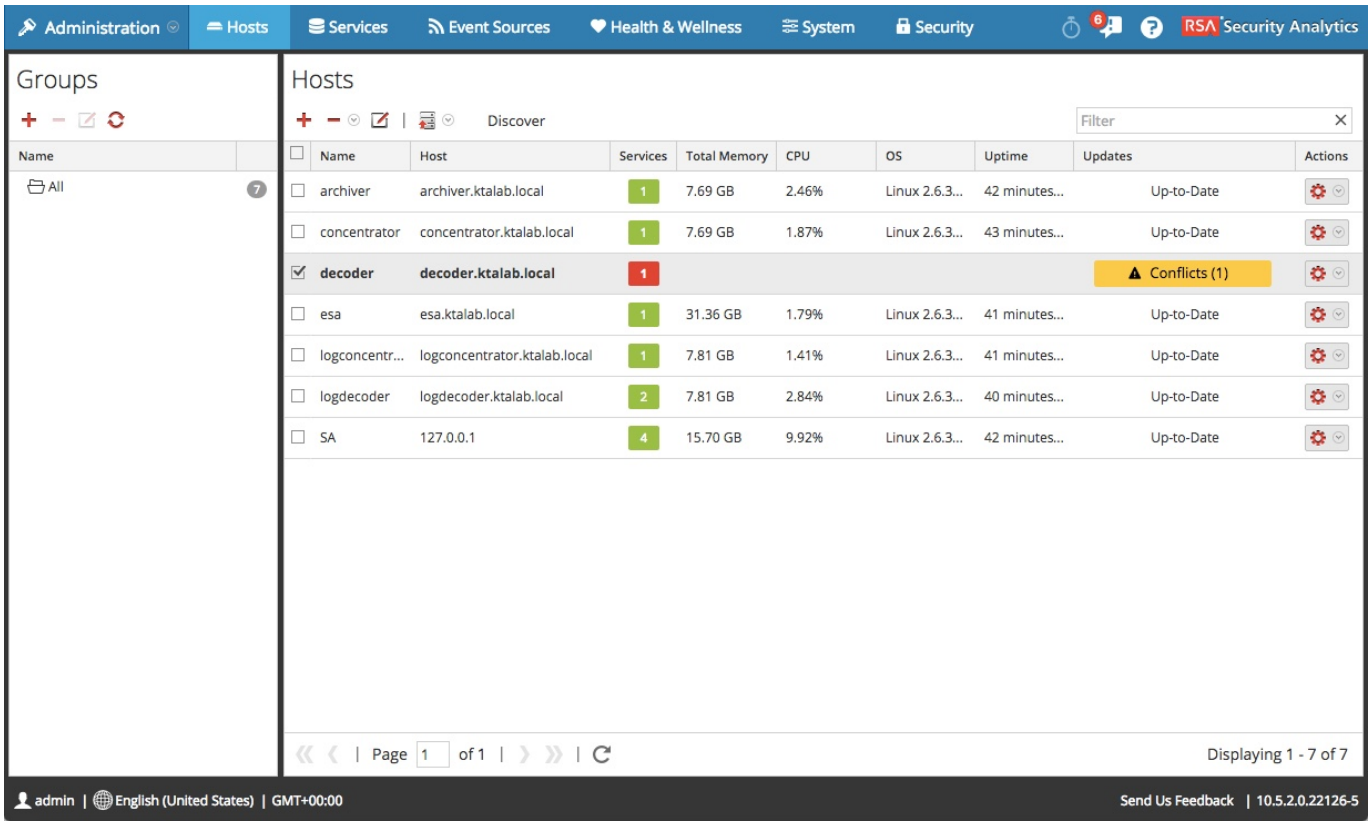
Dashboard



The dashboard page is the starting point for nearly all interactions with SA. It consists of Dashlets, which are small applets which can be customised a great deal to show a user information that is pertinent to his/her role, such as charts and graphs, information on attackers, etc. It is possible to configure multiple dashboards, which can then be switched between to get different views into the data. Many SOC's use these dashboards in their communal monitors for analysts to get a quick glimpse of the current threats to the organisation.

Administration Module

The Administration Module is the primary point of interaction for the SA Administrator persona. In here, an Administrator can provision new hosts, configure services, the settings of the UI, check the health of SA components, manage log event sources, and configure SA security. It can be accessed by clicking any option under Home->Administration.



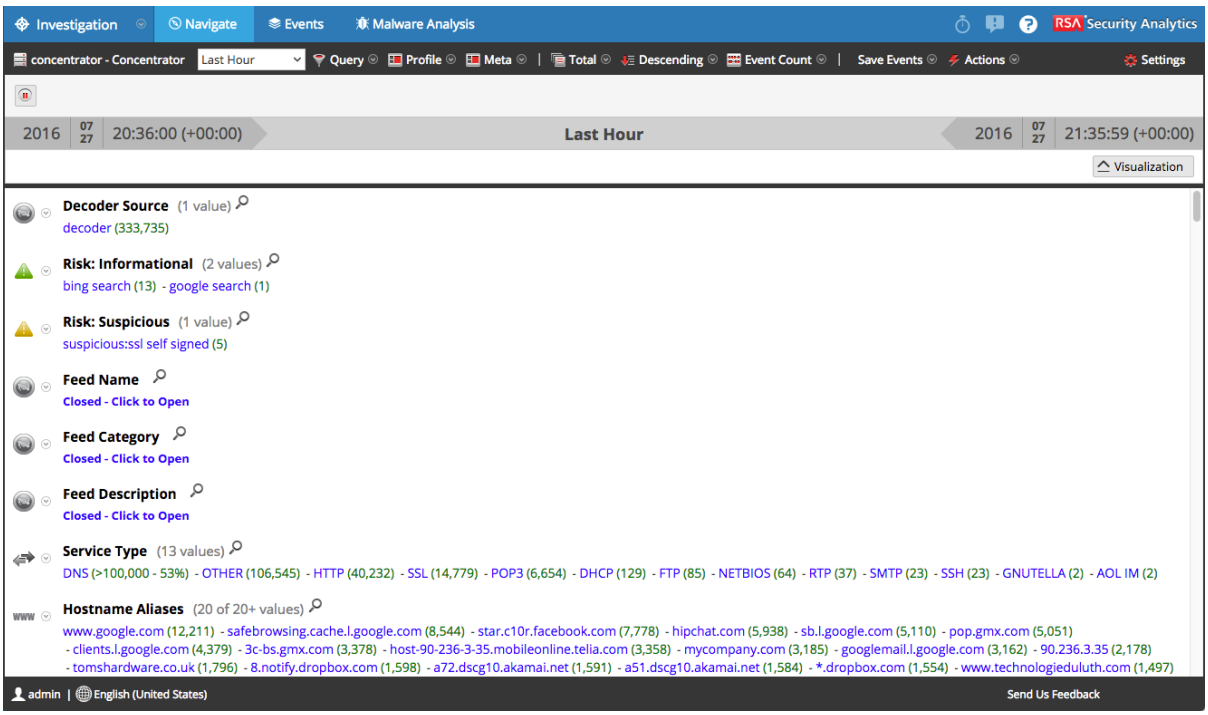
Administration is used by Administrators to change configurations, monitor health and wellness, configure event sources, and also system security.

Investigation Module

The Investigation module of SA allows users to conduct forensic investigations by drilling through the meta of captured sessions.

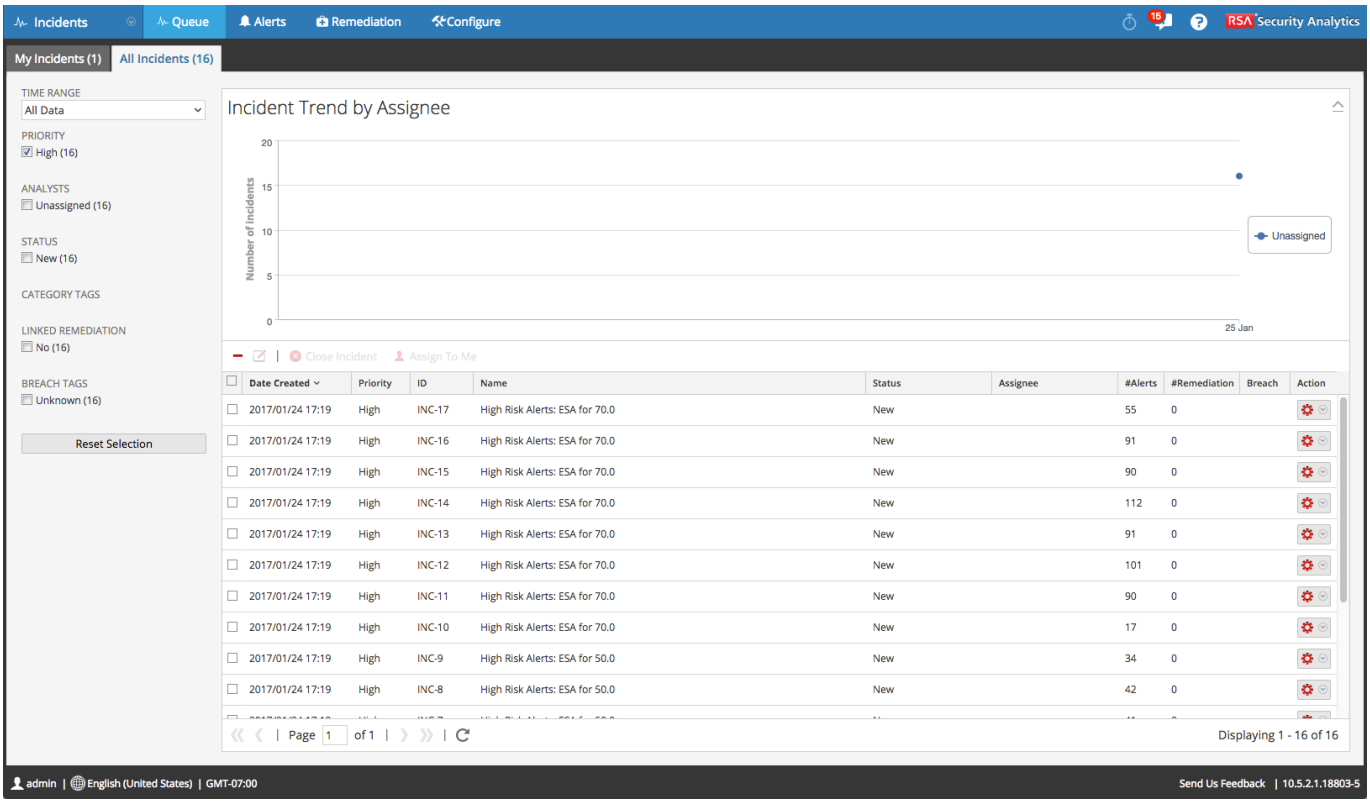
It also is the primary point of contact for Malware Analysis.

It is in this module that sessions can be reconstructed after they have been captured.



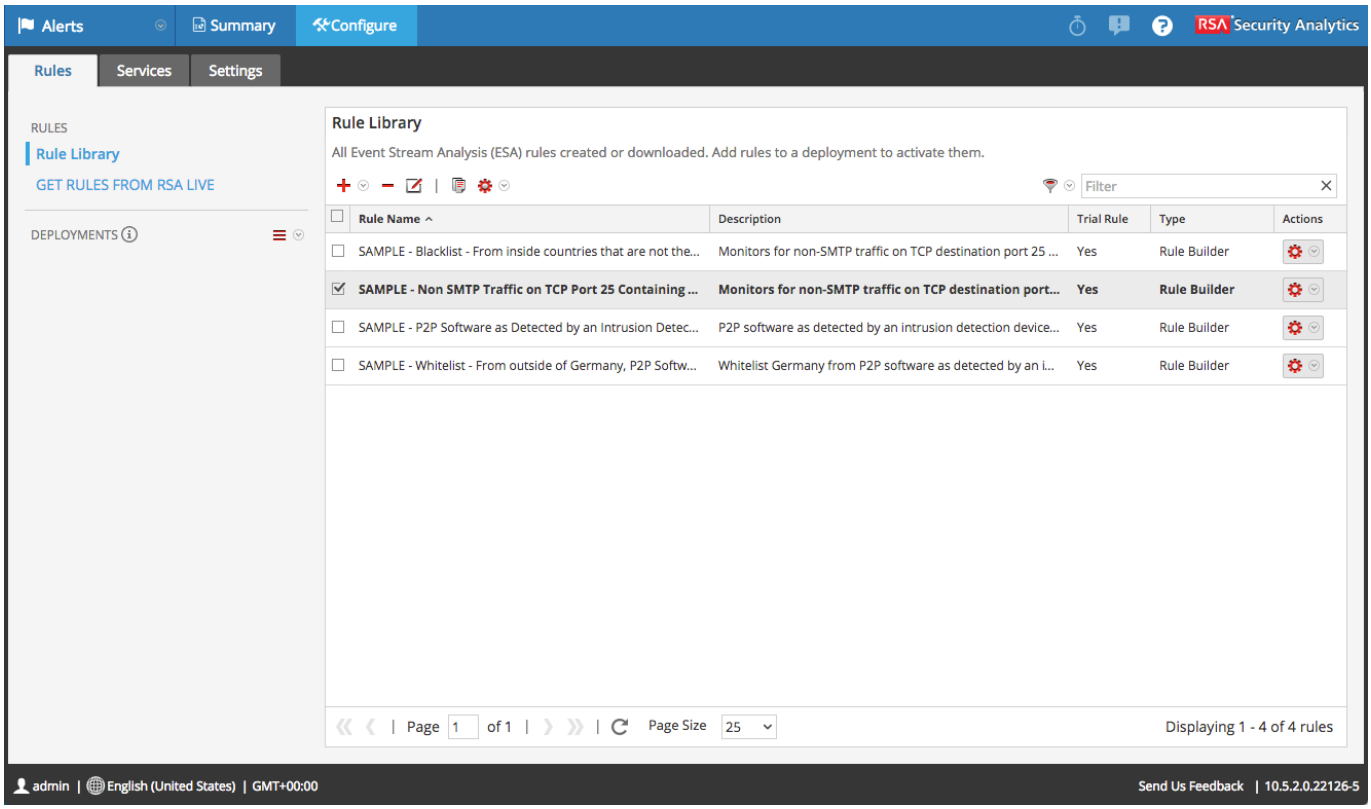
Incidents Module

The Incidents module of SA, sometimes referred to as IM or Incident Management, is a relatively lightweight application for creating and tracking detected security incidents.



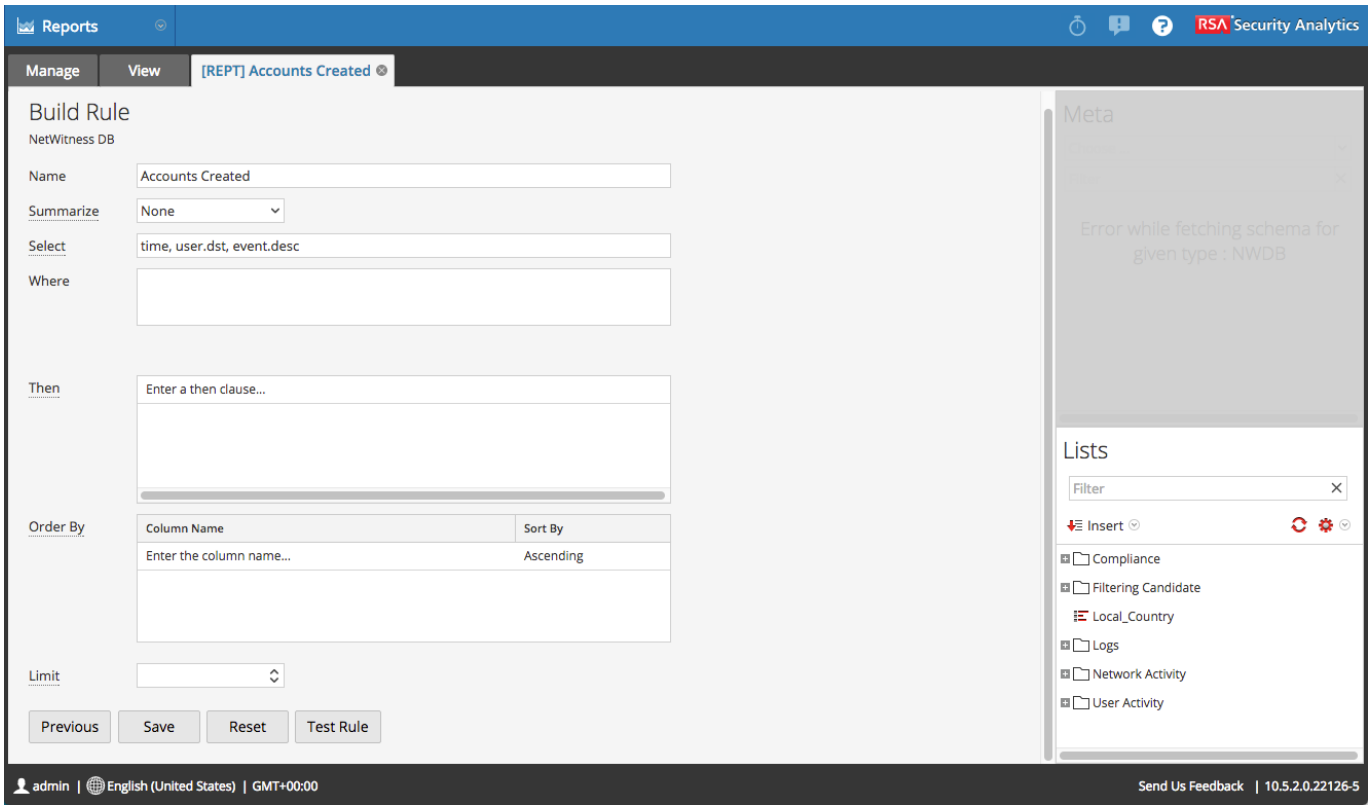
Alerts Module

The Alerts module of SA enables administrators to generate security alerts based on advanced correlative rules. It is the main control point for Event Stream Analysis' (ESA) rules.



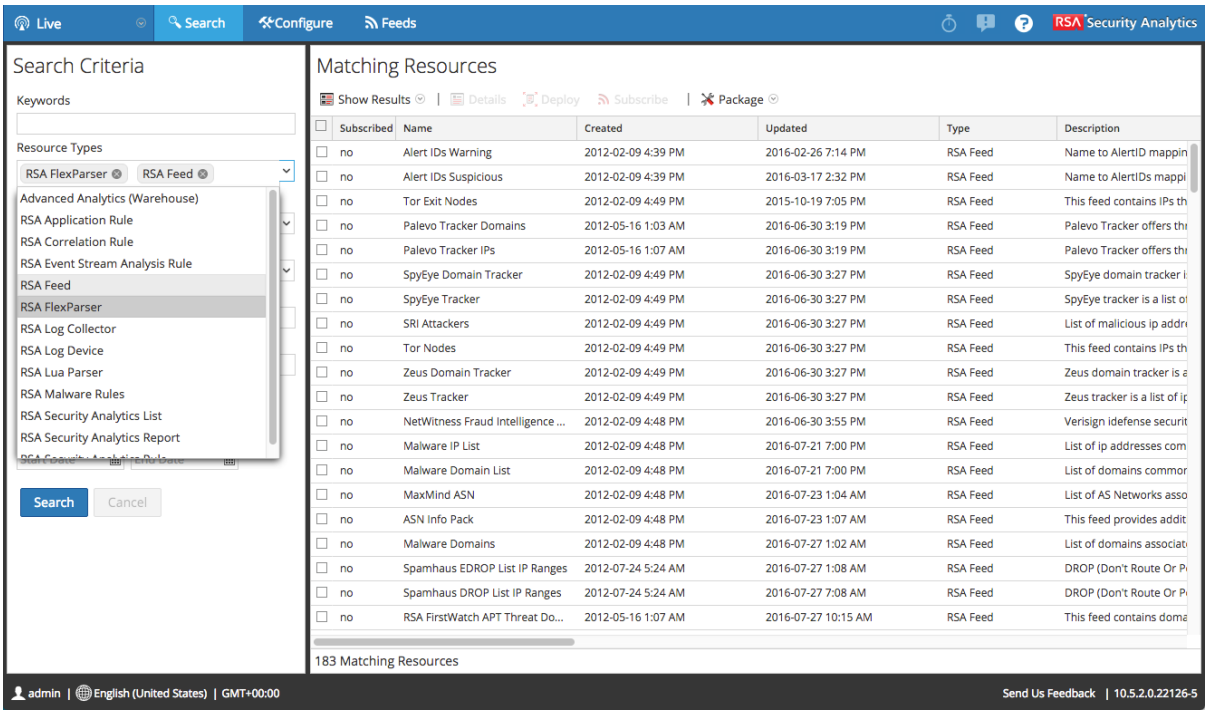
Reports Module

SA's Reporting module interfaces with the Reporting Engine, or 'RE' for short, to generate charts, reports, and alerts against NWDB (SA Core) and RSA Warehouse (Hadoop) data sources.



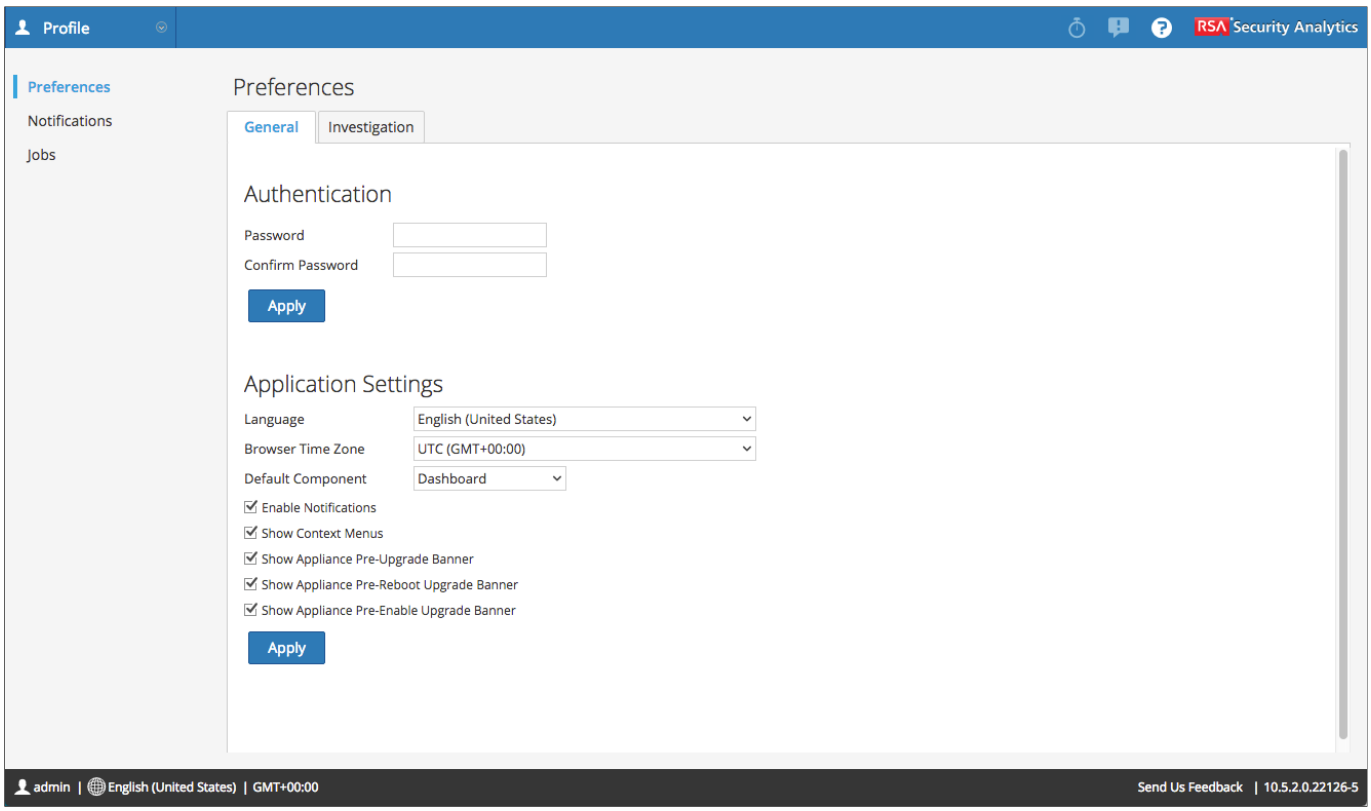
Live Module

The Live module is the primary interface for RSA Live. RSA Live (formerly NetWitness Live) is a subscription-based service which allows organisations to download and deploy pre-built content to assist in their security use-cases. This includes threat intelligence from RSA and partners. Live content includes parsers, feeds, rules, reports, and more. The Live module can deploy these resources and also subscribe to them, so that updates are automatically downloaded and deployed to your SA infrastructure.



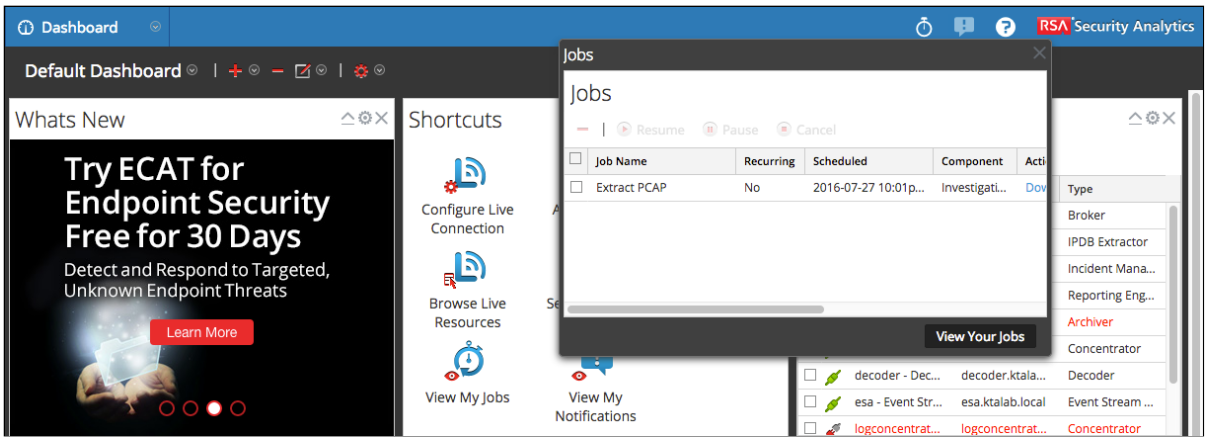
User Profile

The SA User Profile is used for configuring basic attributes of the currently logged-in user. Here, the user can change his/her password, change the time zone, and select the interface language (if a Language Pack has been installed). There are some more advanced options here, as well, but these are not frequently used.



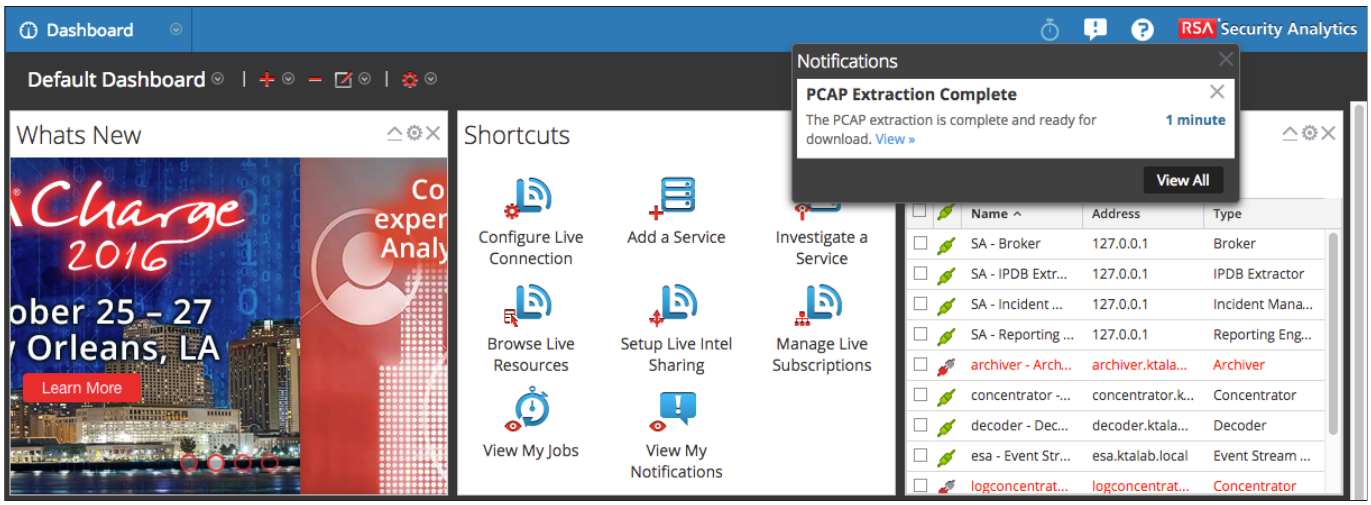
Jobs Panel

The Jobs panel is a drawer which is available at the top of SA's interface, which shows background tasks which have been submitted to the system, such as extracting a PCAP in the Investigation module. This tells you the jobs and their state, and in some instances provides a link for the job's output, such as an extracted PCAP file.



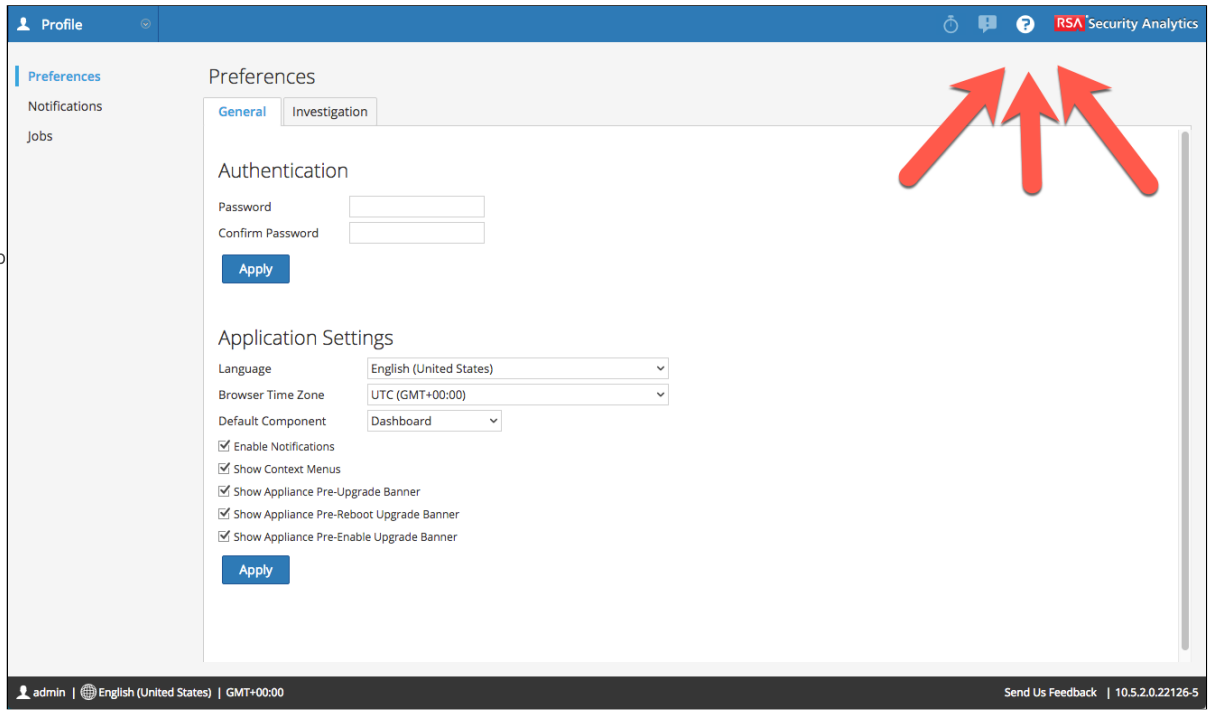
Notifications Panel

The Notifications panel flashes when a job has completed or when there is other pertinent information for the user.



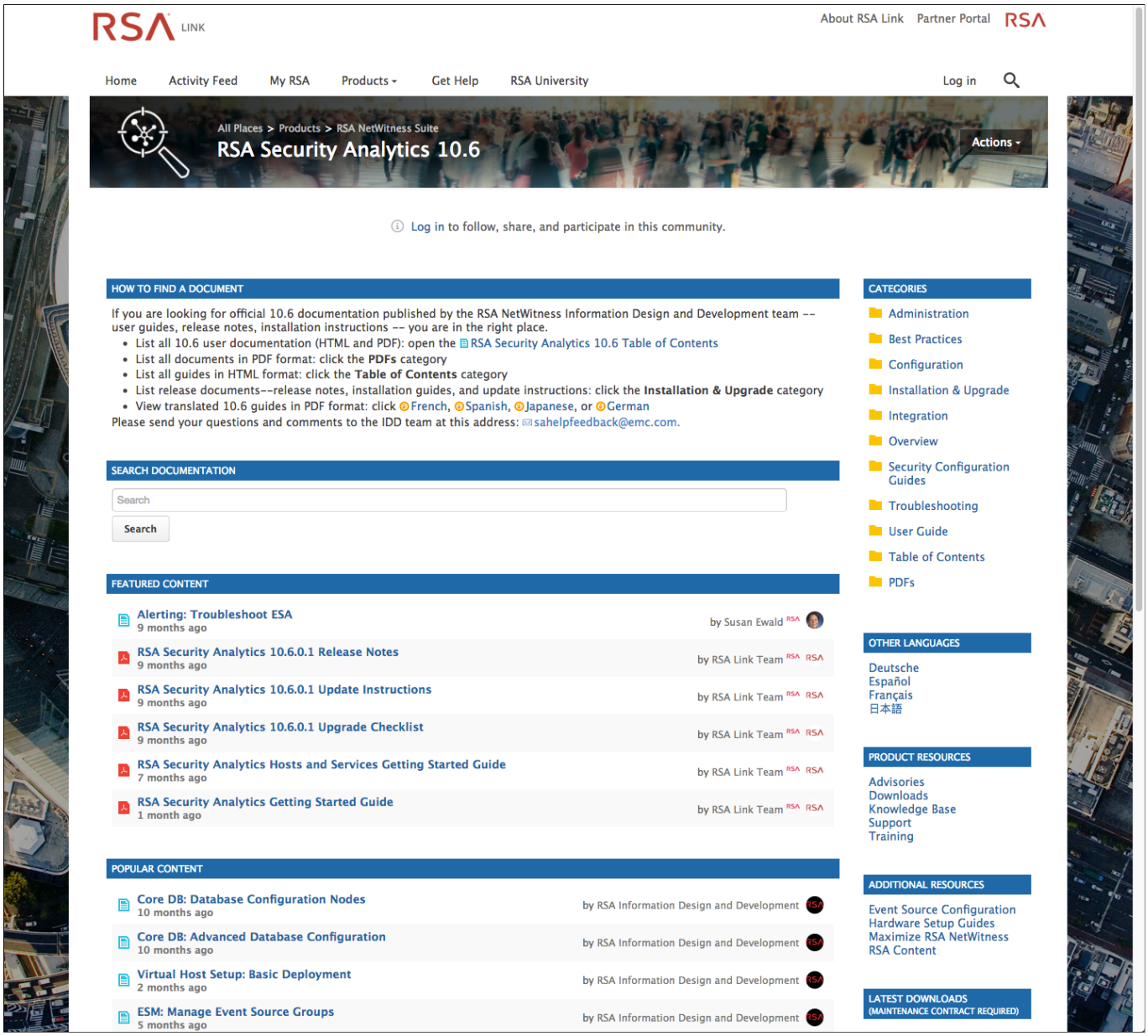
Contextual Help

Contextual help is available from nearly every page in the UI. Simply click the button at the top of the screen to be taken to the relevant online help topic.



Online Help Resources for Security Analytics

Documentation for Security Analytics is available from RSA Link at https://community.rsa.com and at the time of writing, http://sadocs.emc.com, though this may soon be retired. You can browse through the product documentation for multiple SA releases here. Some release documentation is also available in languages other than English.



RSA Link Documentation (version 10.6 pictured)

Investigation Tab - User Preferences Panel

Overview

This topic introduces the features of the Profile view > Preferences panel > Investigations tab.

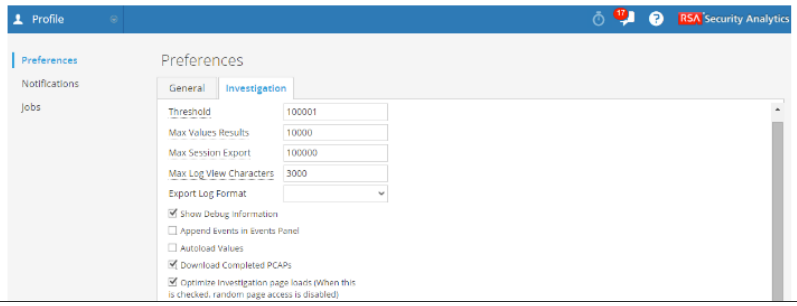
Context

In the Profile view > Preferences panel > Investigation tab, users can set several preferences that affect the performance and behavior of Security Analytics when analyzing data, viewing events, and reconstructing events in Investigation.

Procedures related to this tab are described in Configure Navigate View and Events View.

To access this tab:

1. In the **Security Analytics** menu, select **Profile**.
2. In the **left navigation panel**, select **Preferences.**
3. In the **Preferences** panel, select the **Investigations** tab.



Last modified

04:00, 7 Dec 2015

Support and Service

Web
RSA SecurCare Online

Phone
1-800-995-5095
Option 3

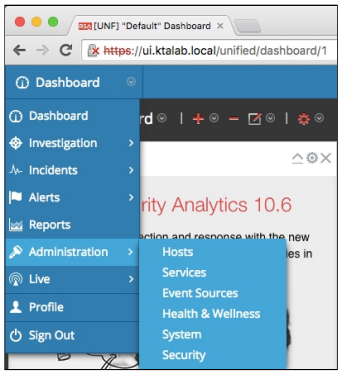
International Contacts

Email
support@rsa.com

Community
RSA Security Analytics

Documentation Questions
sahelpfeedback@emc.com

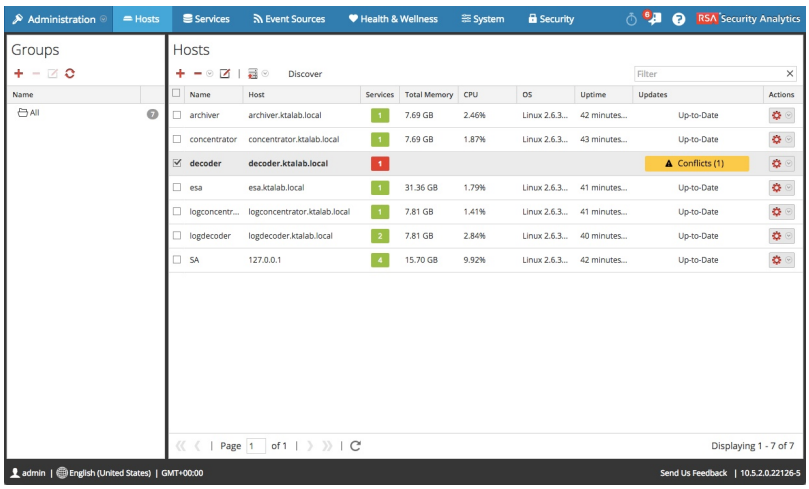
6. SA UI - Administration



Now let's explore the Administration module a bit. Notice in the upper-left hand corner of the screen, there is a drop-down menu with the word 'Dashboard' on it. We'll borrow a bit from Microsoft and call this the **Start Menu**. Now click on the Start Menu, then hover over '**Administration**', and then click on '**Hosts**'.

Hosts vs. Services

For those who are familiar with Security Analytics 10.x through 10.3, one new concept that has been introduced is that of **Hosts** versus **Services**. Prior to SA 10.4, there was a single concept called '**Devices**', in which all services were listed, but there was only an implied Appliance backing it. This made the concept of managing the host hardware and operating system very difficult to manage in the UI, and thus a separation was introduced in SA 10.4, then called '**Appliances**' and 'Services'. In SA 10.5, 'Appliances' was renamed to '**Hosts**' in order to avoid any confusion which might arise from the support of third-party hardware or virtual machines.

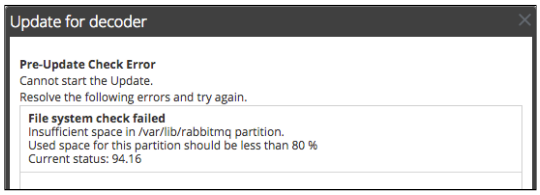


The Administration->Hosts Page

The main functions of the **Administration**->**Hosts** page are:

- Checking for and performing system updates on SA hosts. It also displays errors when a problem condition is believed to exist with the host OS that would prevent an update from succeeding.
- Adding new hosts to SA and enabling them.

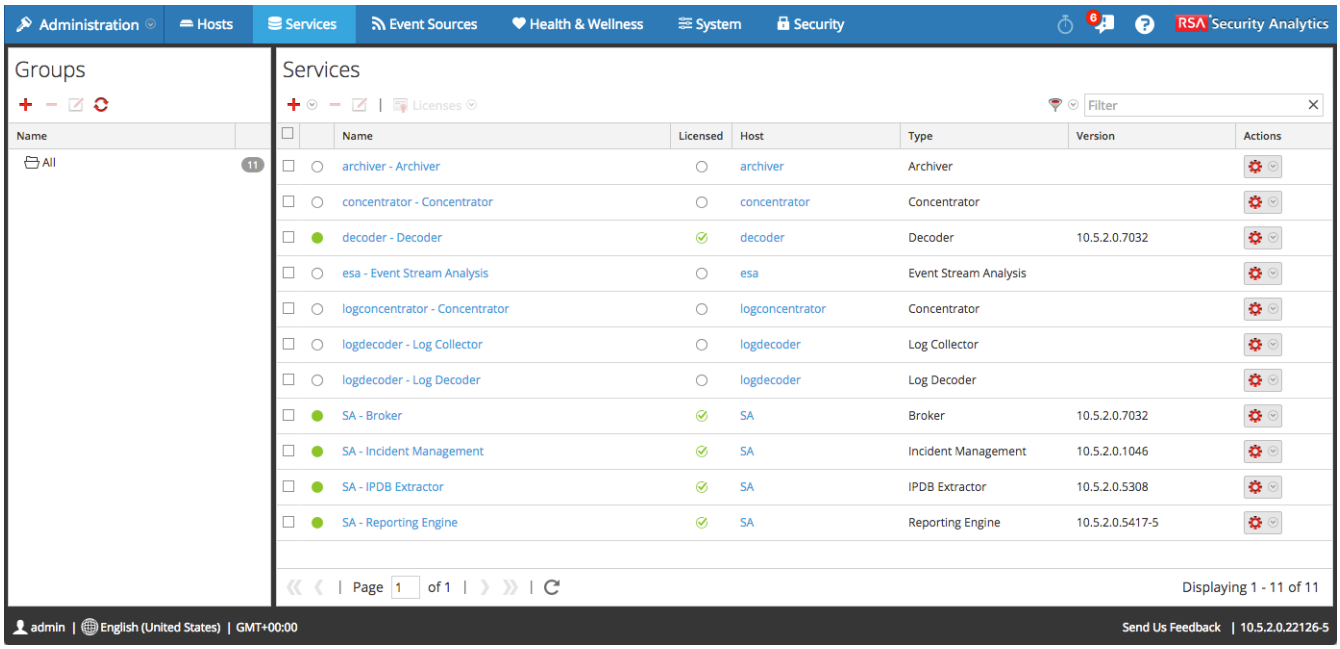
Remember that a '**Host**' in SA parlance refers to the OS and hardware that an SA service or services are running on. The Hosts page is tied into every connected SA appliance via the SMS Service, Puppet and MCollective (remember those glue services we talked about earlier?).



Notice the 'Conflicts' message in the Updates column next to Decoder. This tells us that SA has found something wrong with this particular Decoder. If we click on the 'Conflicts' message, it will pop up a dialog telling us the problem. We won't concern ourselves with this particular problem for now - it's just for illustration.

Now for the Services Page

Now let's click on the **'Services'** tab. Here we can see the actual services running in our SA deployment, without regard to the underlying OS or host. 'Services' in SA refer to the main component services which SA is comprised of: SA Core Services, ESA, Reporting Engine, Incident Management, & Malware Analysis. Glue services aren't shown here.



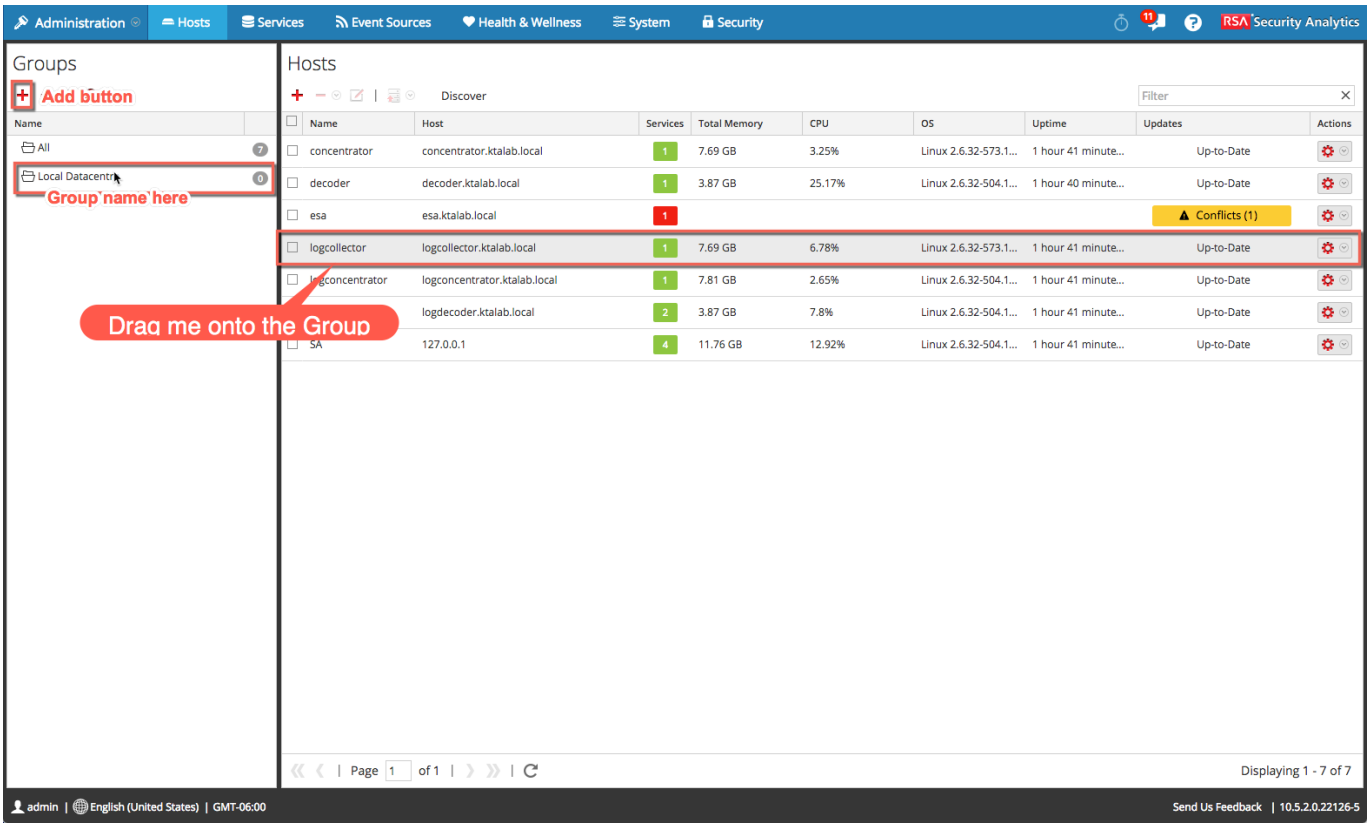
The **Administration->Services** page is the launchpad to the configuration of SA's main services. On this page, we can see whether a service is running, whether it is licensed, which host it's running on, and the service's version. We can also see Groups of services, which can be defined by an Administrator. We can can also use the Services page to apply licenses to services.

Host & Service Groups

In both the Hosts and Services views of Administration, objects can belong to groups. Groups are used by Live and also Health & Wellness policies. All one must do to create a group is click the + button in the Groups pane, and enter a name for the group.

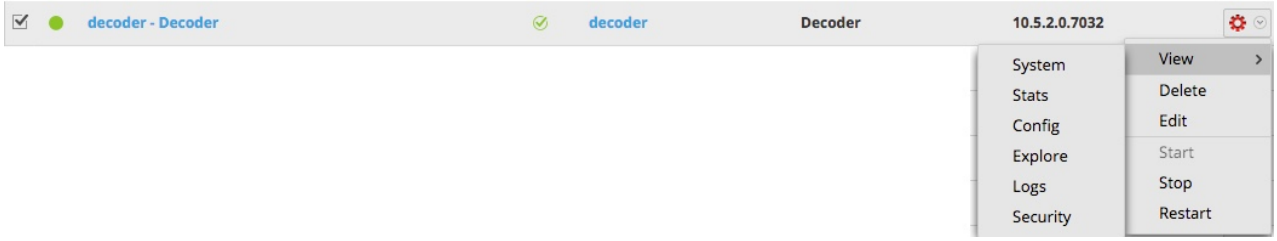
To add a host or service to the group, simply drag it onto the group.

Click on a group name to filter the view so as to show only the group's members.



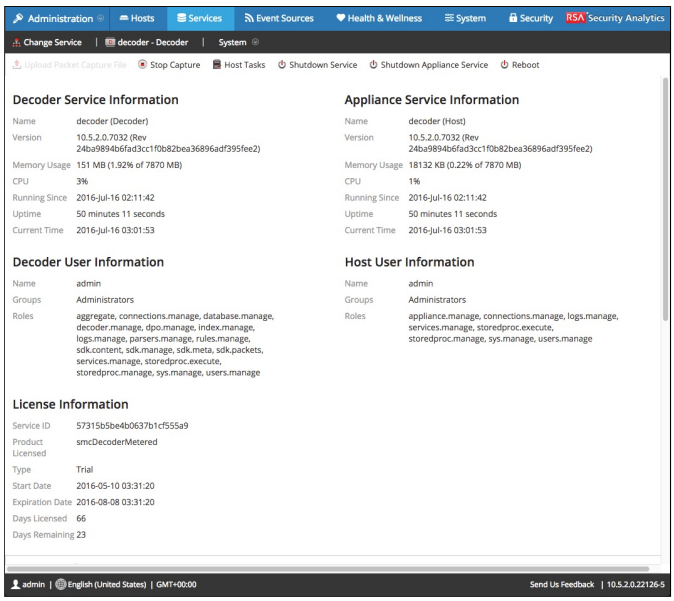
Configuring a Service - The System Tab

Let's have a look at the configuration of one of our services. From **Administration->Services**, click on the gear icon to the right of "decoder - Decoder". This brings up the **Actions** menu for the service.

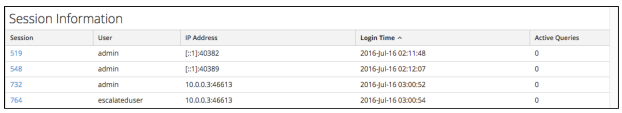


Now click on '**System**'. This takes you to the landing page for the configuration of our Packet Decoder service.

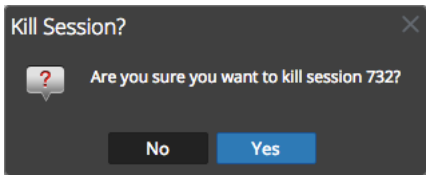
Have a look-see here. We can see information pertaining to the Decoder service itself, which user we're connecting with, details about the license, and also information about the mystical Appliance Service. Remember the Appliance Service from earlier. This is one of the two pages within SA in which you can work with this old service.



Farther below, is information on active sessions (or connections) into the Decoder service.



By clicking on one of the session ID's in this list, you can forcefully kill a logged-in session. **Hint: this may not actually be a good idea**



Towards the top of the screen, there is a bar that allows you to perform some actions on the service and host. You can start and stop capture (on a Decoder), reboot the host, shutdown the service (this actually just restarts the service), and perform tasks on the host OS ('**Host Tasks**'). This bar will vary between different types of services.

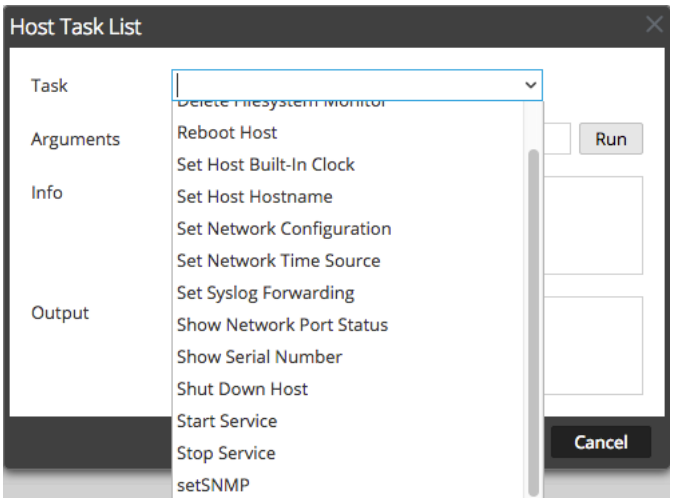
Host Tasks

The Host Tasks button in the **Administration->Services->Services->System** accesses functionality provided by the legacy Appliance Service to get info from the host OS and also perform certain OS-related tasks such as:

- Set the host clock
- Set the OS hostname
- Set NTP time source
- Set Syslog forwarding
- Show an interface status
- Show host serial number
- Shutdown the host
- Start or stop a service

Several items are not typically used:

- Enable or disable the SNMP service (snmpd) - SA's Health & Wellness supercedes the legacy NetWitness SNMP capability
- Add or delete a filesystem monitor - this was used for disk monitoring in the old NwAdministrator thick client



Navigating Between Service Views

In the black-ish bar near the top of the System page, we can switch between different views of the presently selected SA service by clicking the drop-down next to the page title.

Let's click on **Stats** to see a stats view of the presently selected Decoder.

The Service Stats Page

The Stats page gives us the ability to read and visualise internal statistics which are published by the selected service. We can see them as gauges, live timeline charts, and historical timeline charts. This page replicates some of the functionality of the newer Health & Wellness functionality of SA. Expect that sometime in the future these experiences will be unified.

Notice the 'drawer' on the far right hand side of the page. Click on it to see all the statistics available on this service.

Assembler Rate Client Goodput (current)
Stat Name:assembler.client.goodput.rate
Path:/decoder/stats/assembler.client.goodput.rate

Assembler Rate Client Goodput (maximum)
Stat Name:assembler.client.goodput.rate.max
Path:/decoder/stats/assembler.client.goodput.rate.max

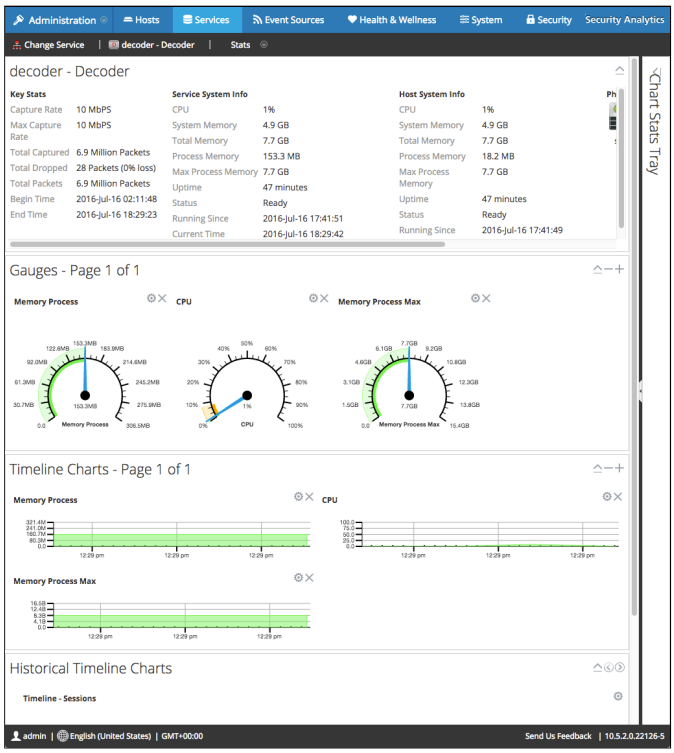
Assembler Rate Meta (current)
Stat Name:assembler.meta.rate
Path:/decoder/stats/assembler.meta.rate

Assembler Rate Meta (maximum)
Stat Name:assembler.meta.rate.max
Path:/decoder/stats/assembler.meta.rate.max

Assembler Rate Packet (current)
Stat Name:assembler.packet.rate
Path:/decoder/stats/assembler.packet.rate

Assembler Rate Packet (maximum)
Stat Name:assembler.packet.rate.max
Path:/decoder/stats/assembler.packet.rate.max

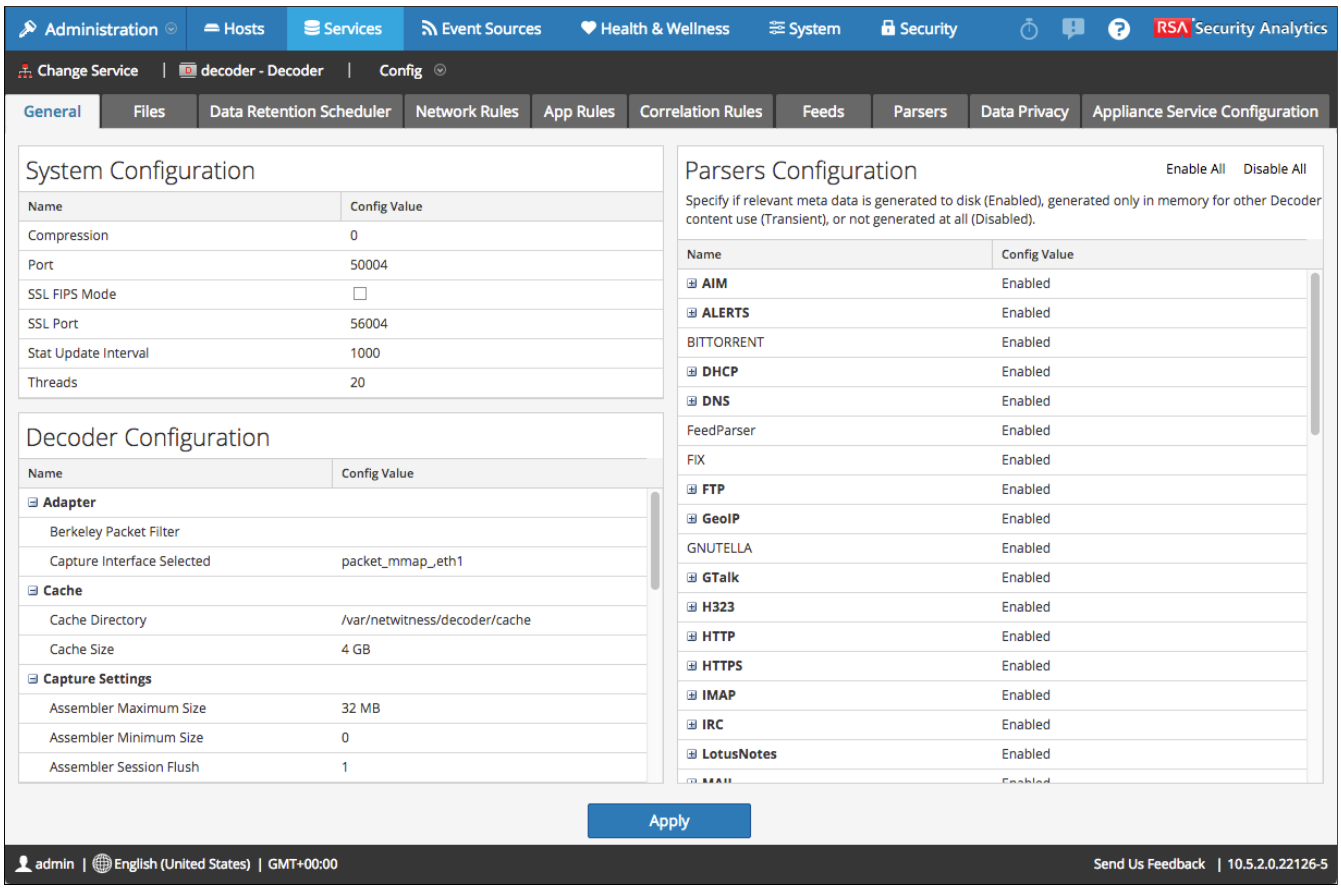
Assembler Rate Server Goodput (current)
Stat Name:assembler.server.goodput.rate
Path:/decoder/stats/assembler.server.goodput.rate



You can page through these stats and drag them over to the gauges or charts on the left-hand side of the page. Try dragging "Packet Capture Queue" into the Timeline Charts. These stats may have little meaning to you now, but some of them can be essential to troubleshooting performance problems with an SA Core service.

The Service Config Page

Let's see if you can move over to the Config page of the Decoder service.



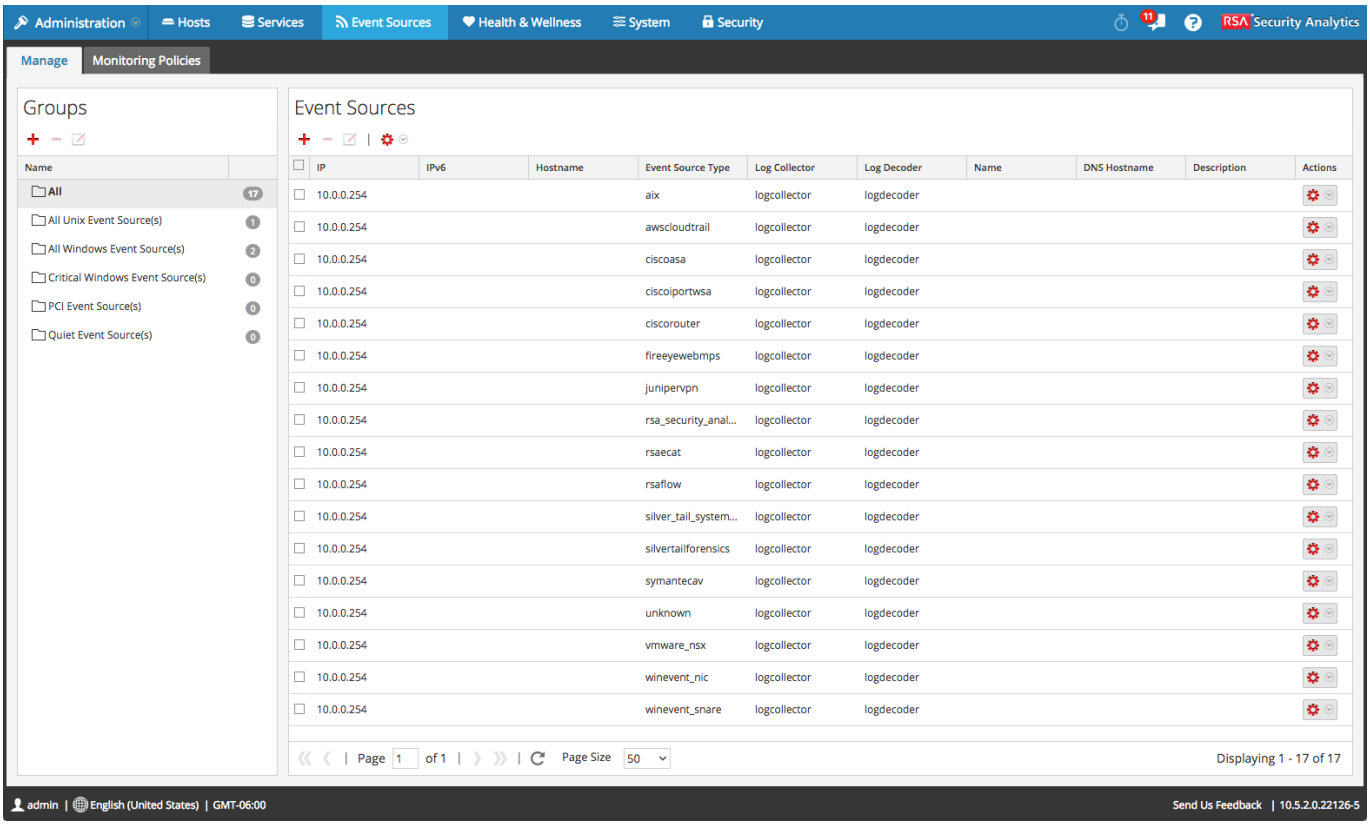
This page is where an SA Administrator would go to change the configuration of an individual SA Service. One can configure simple settings like port numbers and capture autostart, which NIC to capture from and which parsers to enable, and more advanced settings like buffer sizes, BPF filters (Berkeley Packet Filters) and threads. We can also define rules to apply when data is captured.

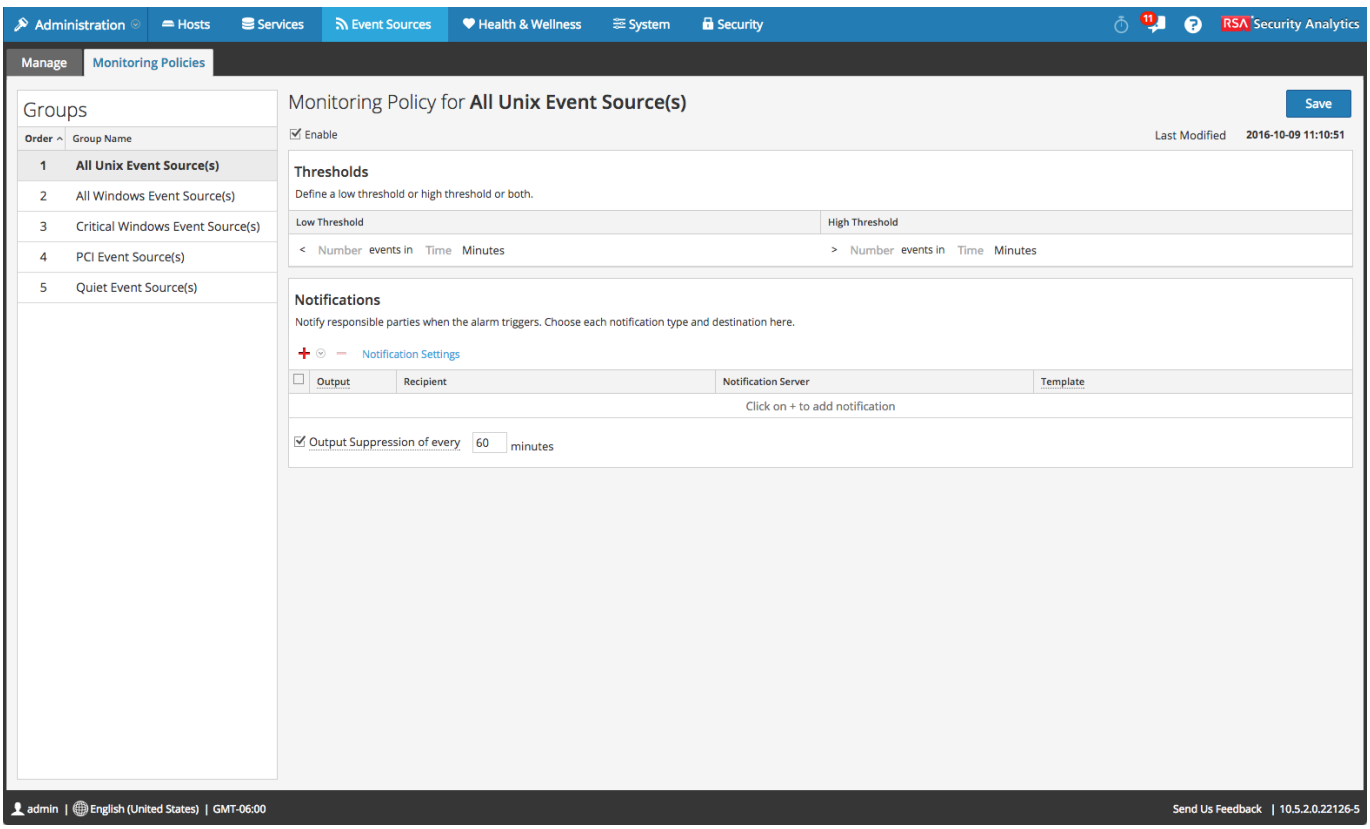
Top Tip: At the time of writing, SA does not possess the ability to keep service configurations in sync, so each SA service must be configured individually.

Have a click through the various tabs to familiarise yourselves with them. Check out the **Files** and **Appliance Service Configuration** tabs.

Event Sources

The Event Sources tab deals with alerting Administrators when log event sources, if a logs deployment, generate too few or too many logs.





Health & Wellness

Health & Wellness deals with the health of SA's software and hardware components. It provides a rollup view, alarms view, and health policy alarming configuration.

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Alarms | Monitoring | Policies | System Stats Browser | Event Source Monitoring | Settings

Time	State	Severity	Rule Name	Service	Hostname	Stat	Value	Id
2016-11-04 12:17:12 PM	Active	Critical	Concentrator Meta Rate Zero	Concentrator	logconcentrator	Concentrator/Meta Rate (current)	0	309-1819-0009
2016-11-04 11:55:25 AM	Active	Critical	Log Decoder Capture Rate Zero	Log Decoder	logdecoder	Capture/Capture Packet Rate (current)	0	309-1757-0008
2016-11-04 11:21:40 AM	Cleared	Critical	Decoder Capture Rate Zero	Decoder	decoder	Capture/Capture Packet Rate (current)	0	309-1723-0005
2016-11-04 11:20:50 AM	Cleared	Critical	Decoder Service in Bad State	Decoder	decoder	ProcessInfo/Service State	Initializing Datab...	309-1721-0001
2016-11-04 11:20:50 AM	Cleared	Critical	Decoder Packet Capture Pool Depleted	Decoder	decoder	Pool/Packet Capture Queue	0	309-1723-0004
2016-11-04 11:20:50 AM	Cleared	Critical	Decoder Capture Not Started	Decoder	decoder	Capture/Capture Status	stopped	309-1721-0000
2016-11-04 11:20:01 AM	Cleared	Critical	Concentrator Meta Rate Zero	Concentrator	concentrator	Concentrator/Meta Rate (current)	0	309-1722-0003
2016-11-02 04:23:23 PM	Cleared	Critical	Log Decoder Service Stopped	Log Decoder	logdecoder	ProcessInfo/Service Status	starting	307-2223-0004
2016-10-31 09:45:10 AM	Cleared	Critical	Concentrator Service in Bad State	Concentrator	concentrator	ProcessInfo/Service State	Initializing Datab...	305-1545-0001
2016-10-31 09:45:10 AM	Cleared	Critical	Concentrator Aggregation Stopped	Concentrator	concentrator	Concentrator/Status	stopped	305-1545-0000
2016-10-09 03:08:50 PM	Cleared	Critical	Concentrator Aggregation Stopped	Concentrator	logconcentrator	Concentrator/Status	stopped	283-2109-0004
2016-10-09 02:29:55 PM	Cleared	Critical	Log Decoder Log Capture Pool Depleted	Log Decoder	logdecoder	Pool/Packet Capture Queue	0	283-2031-0001
2016-10-09 02:29:55 PM	Cleared	Critical	Log Decoder Capture Not Started	Log Decoder	logdecoder	Capture/Capture Status	stopped	283-2030-0000
2016-10-09 11:24:45 AM	Cleared	Critical	Broker Aggregation Stopped	Broker	SA	Broker/Status	stopped	283-1724-0000
2016-11-04 11:20:51 AM	Cleared	High	Concentrator Not Consuming From Service	Concentrator	concentrator	Devices/Status decoder.ktalab.local:50004	offline	309-1721-0002
2016-11-02 04:23:33 PM	Cleared	High	Concentrator Not Consuming From Service	Concentrator	logconcentrator	Devices/Status logdecoder.ktalab.local:50002	offline	307-2224-0005
2016-10-31 09:46:19 AM	Cleared	High	Broker Not Consuming From Service	Broker	SA	Devices/Status concentrator.ktalab.local:50005	offline	305-1547-0005
2016-10-30 03:29:15 PM	Cleared	High	Broker Not Consuming From Service	Broker	SA	Devices/Status logconcentrator.ktalab.local:50005	offline	304-2130-0000
2016-10-30 03:27:37 PM	Cleared	High	High System Swap Utilization	Host	SA	SystemInfo/Swap Utilization	59.25%	304-2132-0001
2016-10-09 11:24:45 AM	Cleared	High	Broker Session Rate Zero	Broker	SA	Broker/Session Rate (current)	0	283-1726-0001

Alarm Details

Page 1 of 1 | Auto Refresh | Items 1 - 20 of 20

admin | English (United States) | GMT-06:00 | Send Us Feedback | 10.5.2.0.22126-5

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Alarms | Monitoring | Policies | System Stats Browser | Event Source Monitoring | Settings

Policies

- Archiver 1
- Broker 1
- Concentrator 1
- Decoder 1
- Event Stream Analysis 1
  - *SA Event Stream Analysis Monitorin...*
- Host 1
  - ***SA Host Monitoring Policy***
- IPDB Extractor 1
- Incident Management 1
- Log Collector 1
- Log Decoder 1
- Malware Analysis 1
- Reporting Engine 1
- Warehouse Connector 1
- Workbench 1

Host: **SA Host Monitoring Policy**

Save

Rules and suppression schedules of Out-of-the-box policies cannot be directly modified. Duplicate the policy if you wish to so modify them

Enable

Last Modified: **2015-10-28 06:00:00 PM**

Hosts

Choose the hosts, services, and groups that your health policy applies to.

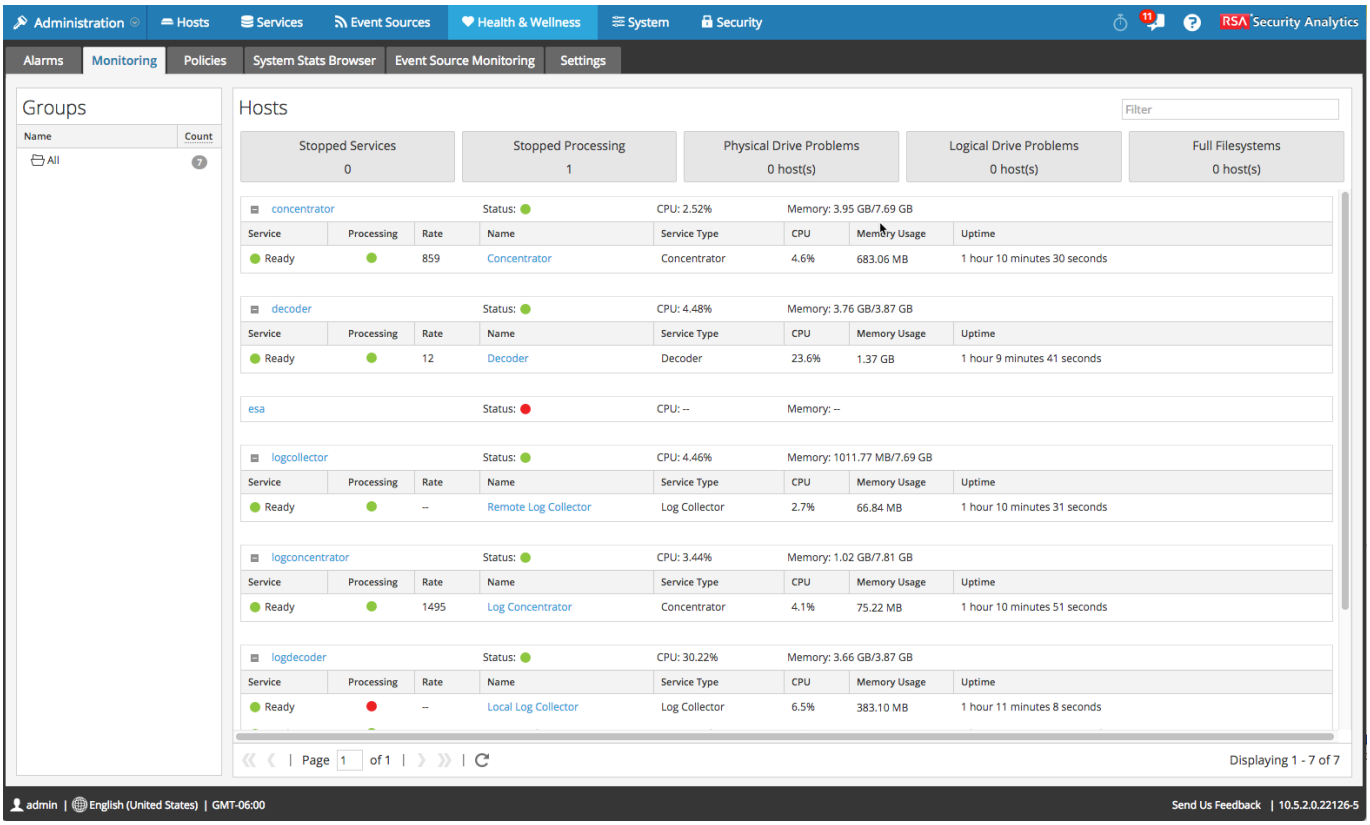
Name	Group	Type
All	7	Group

Rules

Define the conditions under which you want to trigger an alarm for the Security Analytics health problems (definition includes severity, statistic the alarm applies to, threshold, and threshold at which the alarm clears). After you define the alarm rule, enable or disable the alarm.

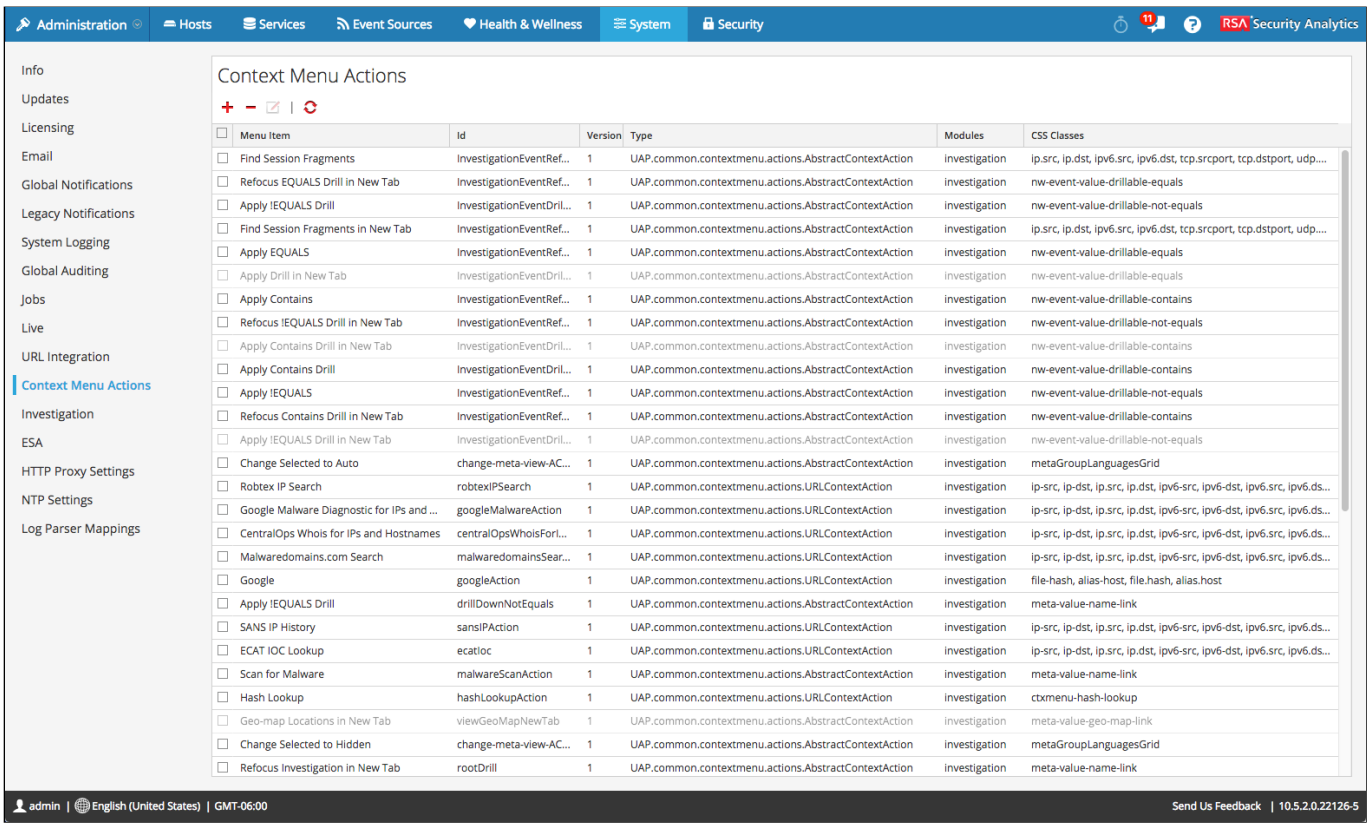
Enable	Name	Severity	Category	Statistic	Threshold
●	Critical Usage on Ra...	Critical	FileSystem	Mounted Filesystem Disk Usage Percent	Alarm > 75% for 0 MINUTES
●	Filesystem is Full	Critical	FileSystem	Mounted Filesystem Disk Usage Percent	Alarm = 100% for 0 MINUTES
●	High Filesystem Usage	High	FileSystem	Mounted Filesystem Disk Usage Percent	Alarm > 95% for 0 MINUTES
●	High System Swap U...	High	SystemInfo	Swap Utilization	Alarm > 50% for 5 MINUTES
●	High Usage on Rabbi...	High	FileSystem	Mounted Filesystem Disk Usage Percent	Alarm > 60% for 0 MINUTES
●	Host Unreachable	Critical	ProcessInfo	Appliance Down	Alarm = True for 0 MINUTES
●	Power Supply Failure	High	SystemInfo	Power Supply Status	Alarm !Regex Match ^ok$\|^ns$|^Presence detected$ for 0 MINUTES
●	RAID Logical Drive D...	High	DiskRaid	Logical Drive State	Alarm Regex Match .*Degraded.* for 0 MINUTES
●	RAID Logical Drive F...	Critical	DiskRaid	Logical Drive State	Alarm Regex Match .*Offline.*|.*Failed.*|.*Unknown.* for 0 MINUTES
●	RAID Logical Drive R...	Medium	DiskRaid	Logical Drive State	Alarm Regex Match .*Rebuild.* for 0 MINUTES
●	RAID Physical Drive ...	Critical	DiskRaid	Physical Drive State	Alarm !Regex Match .*Online.*|.*Hotspare.*|.*Rebuild.* for 0 MINUTES
●	RAID Physical Drive ...	High	DiskRaid	Physical Drive Predictive Failure Count	Alarm > 1 for 0 MINUTES
●	RAID Physical Drive ...	Low	DiskRaid	Physical Drive State	Alarm Regex Match .*Rebuild.* for 0 MINUTES

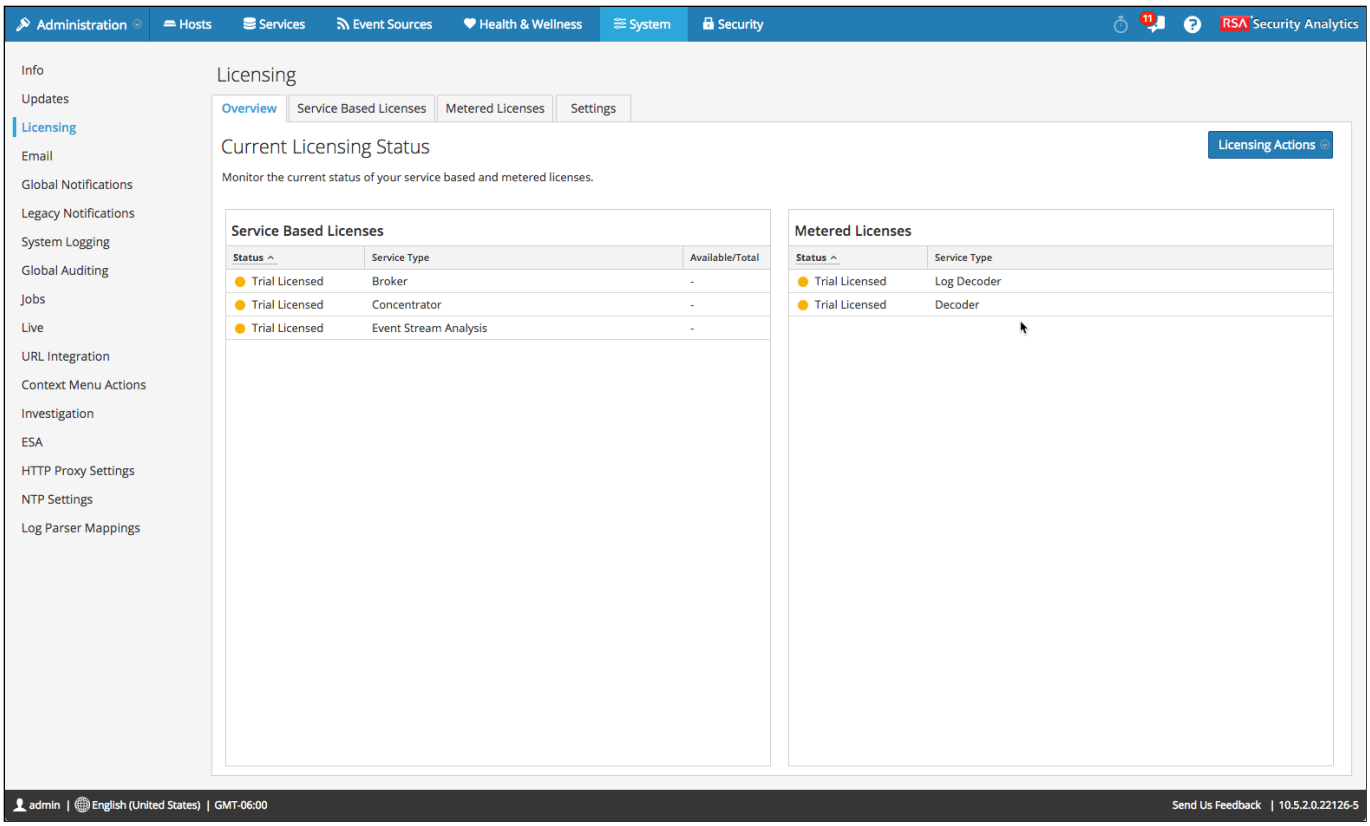
admin | English (United States) | GMT-06:00 | Send Us Feedback | 10.5.2.0.22126-5

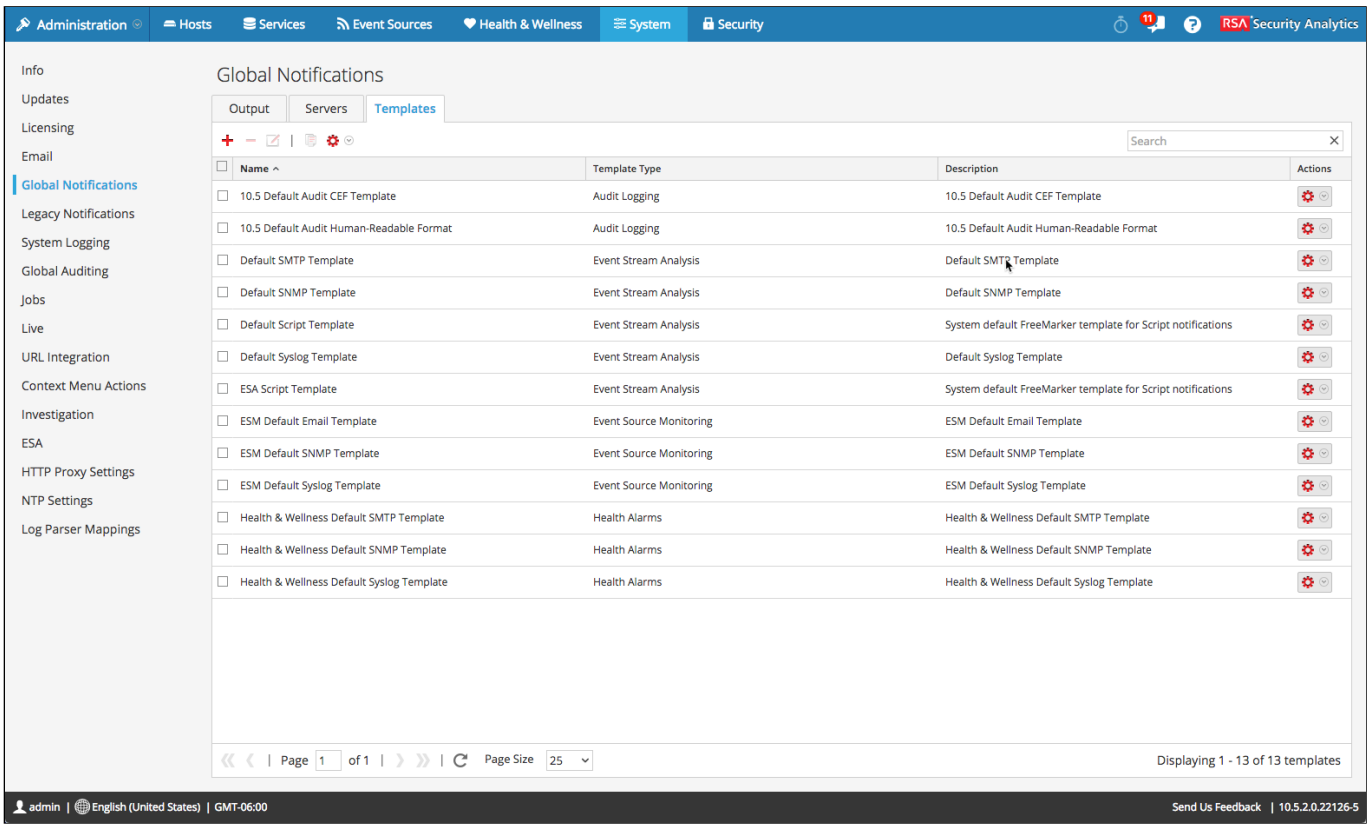


System

The System tab concerns certain global settings of SA. From here, one configures licensing, audit logging, notifications, Live, and certain other settings.







Security

As the name implies, the Security tab concerns SA users, roles, authentication, and password policies.

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Users | Roles | External Group Mapping | Settings | Login Banner

Password Strength

Minimum Required

- Minimum Password Length 0 Characters
- Uppercase Characters 0 Characters
- Lowercase Characters 0 Characters
- Decimal Digits 0 Characters
- Special Characters (~!@#$%^&*_-+=`|(){[]:;"'<>,.?/) 0 Characters
- Non-Latin Alphabetic Characters 0 Characters
- Password May Not Contain Username

Apply

Security Settings

Lockout Period 10 minutes

Idle Period 0 minutes

Session Timeout 0 minutes

Case Insensitive User Name

Max Login Failures 999

Global Default User Password Expiration Period 0 days

Notify User 5 Days Prior to Password Expiry

Apply

External Authentication

- Active Directory
- PAM

Apply | Test

admin | English (United States) | GMT-06:00 | Send Us Feedback | 10.5.2.0.22126-5

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Users | Roles | External Group Mapping | Settings | Login Banner

Enable | Disable | Unlock

Username	Name	Email Address	Roles	External	Description
admin	Administrator		Administrators	no	System Administrator
jsmith	Joe Smith	jsmith@somewhere.sw	Analysts	no	

Page 1 of 1 | Displaying 1 - 2 of 2

admin | English (United States) | GMT-06:00 | Send Us Feedback | 10.5.2.0.22126-5

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Users | Roles | External Group Mapping | Settings | Login Banner

Name	Description	Permissions
Analysts	The SOC Analysts persona is ce...	Dashlet Access - Unifed RSA First Watch Dashlet, View and Manage Incidents, Export List, Navigate Events, Define Rule, Delete Jobs, Dashlet Access - Reporting Recent Report Dashlet, Search Live Resources, Acc...
Operators	The System Operators Persona...	Dashlet Access - Unifed RSA First Watch Dashlet, Dashlet Access - Live Updated Resources Dashlet, Modify ESA Settings, View Health & Wellness Stats Browser, Manage SA Email, Manage SA Notifications, Mana...
SOC_Managers	The persona for SOC Managers...	Dashlet Access - Unifed RSA First Watch Dashlet, View and Manage Incidents, Export List, Navigate Events, Define Rule, Delete Jobs, View Event Sources, Dashlet Access - Reporting Recent Report Dashlet, Searc...
Malware_Analysts	The persona of Malware Analy...	View Health & Wellness Policy, Access Investigation Module, Download Malware File(s), View and Manage Incidents, Navigate Values, Create Incidents from Investigation, Navigate Events, View Malware Analysi...
Data_Privacy_Officers	The persona of Data Privacy Of...	Dashlet Access - Unifed RSA First Watch Dashlet, View and Manage Incidents, Export List, Delete Alerts and incidents, Navigate Events, Define Rule, Delete Jobs, Dashlet Access - Reporting Recent Report Dashle...
Administrators	The System Administrators per...	View and Manage Incidents, Export List, Delete Alerts and incidents, Define Rule, View Event Sources, Dashlet Access - Reporting Recent Report Dashlet, Search Live Resources, View Rules, Access View, View Re...

Page 1 of 1 | Displaying 1 - 6 of 6

admin | English (United States) | GMT-06:00 | Send Us Feedback | 10.5.2.0.22126-5

7. Building an SA Host From Nil

Building an SA host is fairly easy. All one needs are ISO files, a USB stick or DVD, and console access to your physical or virtual hardware.

High-Level SA Building Procedure

ISO's are generally only available for the first code revision of every minor SA release, but not for every patch or service pack. E.g. ISO's are only available for 10.4.0.2, 10.5.0.0, 10.6.0.0, etc. The overall process of bringing an SA host online is usually thus:

1. Image the host with a buildstick, DVD, or ISO file
2. Provision the host with the SA server
3. Upgrade the host to the latest code revision

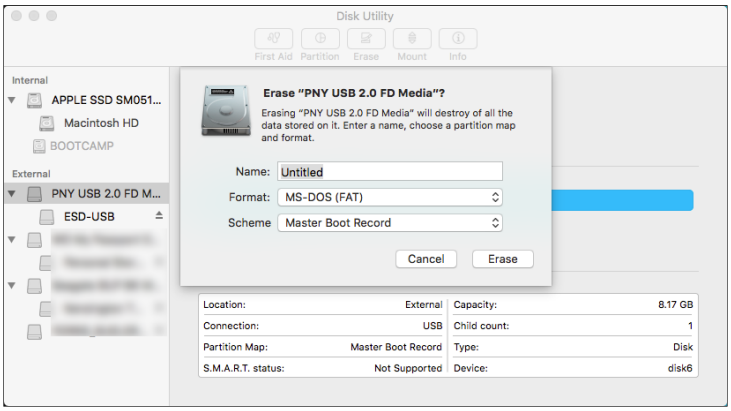
Building the Buildstick

If using a physical host, the most common method of imaging the host is to create what is called a buildstick, which is a bootable USB disk which builds the SA host, including OS installation. Four things are needed to create the buildstick:

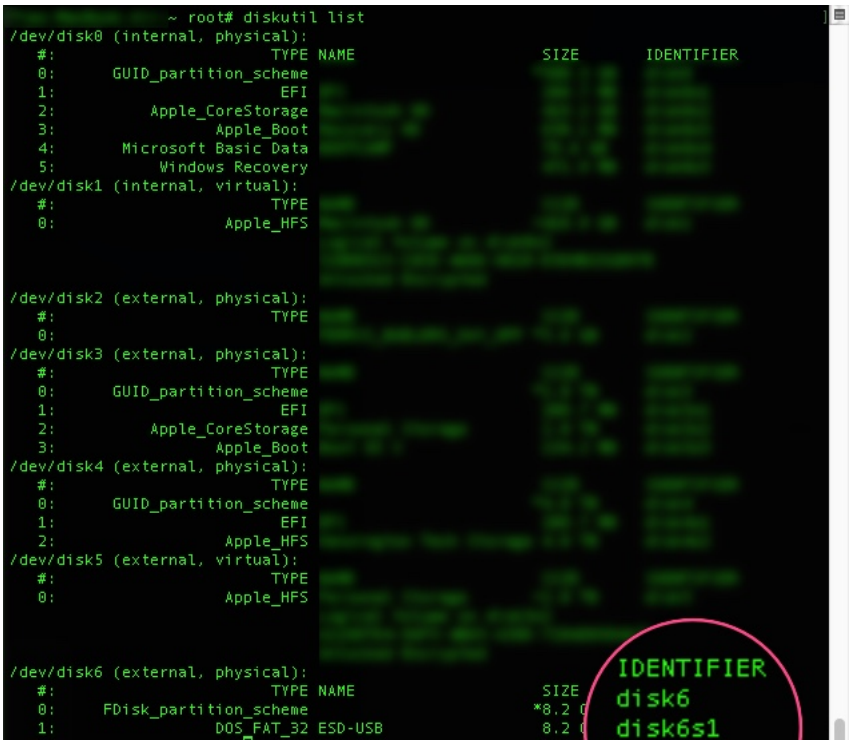
- A USB stick with a minimum of 8GB
- Two SA ISO files. One is used to image the USB stick, and the other is copied to the stick after it has been imaged
- A zip file (sastickboot.zip) containing a folder called '[BOOT]'
- A free utility called Unetbootin, which is available for both Mac and Windows

Once you have these, proceed through these steps. This procedure nearly is identical for Mac and Windows:

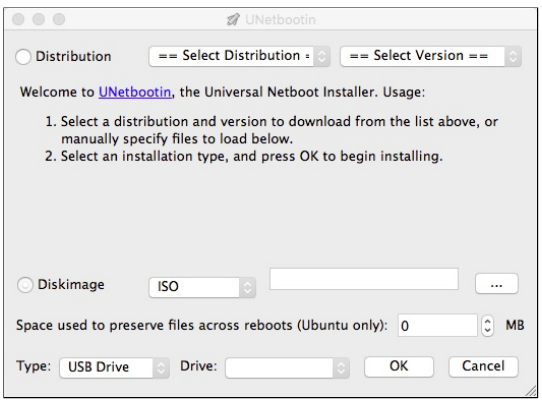
1. Ensure that your USB stick is partitioned using legacy MBR as opposed to the newer GUID scheme. Ensure that it is formatted with the FAT32 filesystem. You can give the partition any name you want. On Mac, this is done with Disk Utility, and on Windows with Explorer



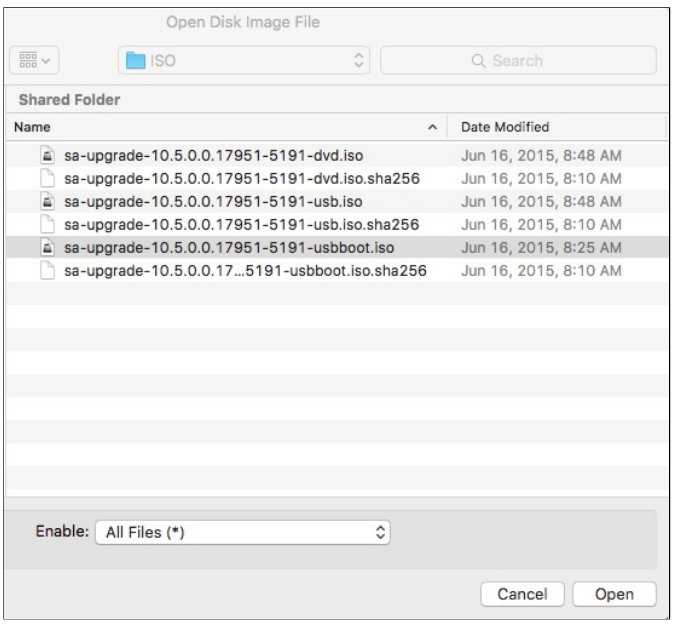
2. If using a Mac, use '**diskutil list**' from the Terminal to confirm which block device your USB stick is using (i.e. /dev/disk6s1)



3. Launch Unetbootin, then choose the button for Diskimage.

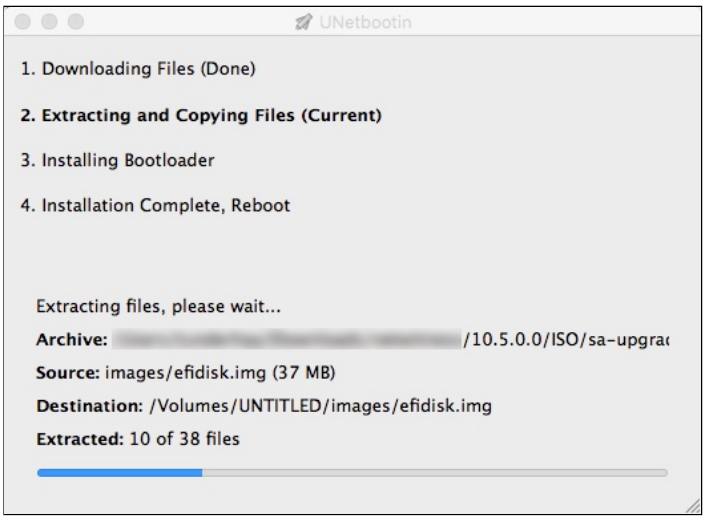


4. Click the button to choose a file to the right of Diskimage, then choose the file sa-upgrade-<version>-**usbboot**.iso and click Open

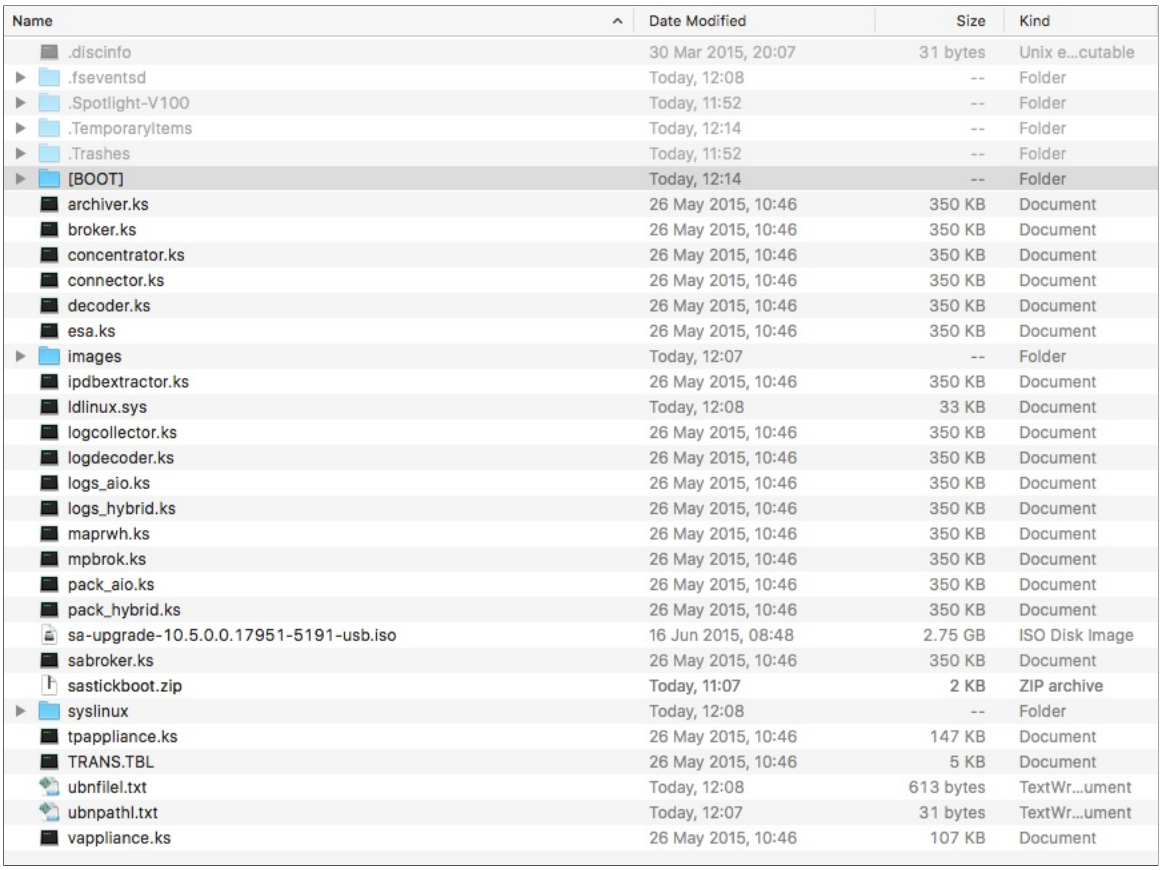


5. Select your USB disk from the Drive dropdown. **MAKE SURE YOU GET THIS STEP RIGHT!**

6. Then click OK to start the imaging.



7. Once the imaging is complete, exit Unetbootin.
8. Copy the file **sa-upgrade-*<version>*-usb.iso** to the root of the buildstick
9. Copy the file **sastickboot.zip** to the root of the buildstick and extract it
10. Your buildstick is now complete and should look like this:



Name	Date Modified	Size	Kind
.discinfo	30 Mar 2015, 20:07	31 bytes	Unix e...cutable
.fseventsd	Today, 12:08	--	Folder
.Spotlight-V100	Today, 11:52	--	Folder
.TemporaryItems	Today, 12:14	--	Folder
.Trashes	Today, 11:52	--	Folder
[BOOT]	Today, 12:14	--	Folder
archiver.ks	26 May 2015, 10:46	350 KB	Document
broker.ks	26 May 2015, 10:46	350 KB	Document
concentrator.ks	26 May 2015, 10:46	350 KB	Document
connector.ks	26 May 2015, 10:46	350 KB	Document
decoder.ks	26 May 2015, 10:46	350 KB	Document
esa.ks	26 May 2015, 10:46	350 KB	Document
images	Today, 12:07	--	Folder
ipdbextractor.ks	26 May 2015, 10:46	350 KB	Document
ldlinux.sys	Today, 12:08	33 KB	Document
logcollector.ks	26 May 2015, 10:46	350 KB	Document
logdecoder.ks	26 May 2015, 10:46	350 KB	Document
logs_aio.ks	26 May 2015, 10:46	350 KB	Document
logs_hybrid.ks	26 May 2015, 10:46	350 KB	Document
maprwh.ks	26 May 2015, 10:46	350 KB	Document
mpbrok.ks	26 May 2015, 10:46	350 KB	Document
pack_aio.ks	26 May 2015, 10:46	350 KB	Document
pack_hybrid.ks	26 May 2015, 10:46	350 KB	Document
sa-upgrade-10.5.0.0.17951-5191-usb.iso	16 Jun 2015, 08:48	2.75 GB	ISO Disk Image
sabroker.ks	26 May 2015, 10:46	350 KB	Document
sastickboot.zip	Today, 11:07	2 KB	ZIP archive
syslinux	Today, 12:08	--	Folder
tpappliance.ks	26 May 2015, 10:46	147 KB	Document
TRANS.TBL	26 May 2015, 10:46	5 KB	Document
ubnfilel.txt	Today, 12:08	613 bytes	TextWr...ument
ubnpathl.txt	Today, 12:07	31 bytes	TextWr...ument
vappliance.ks	26 May 2015, 10:46	107 KB	Document

Prerequisites

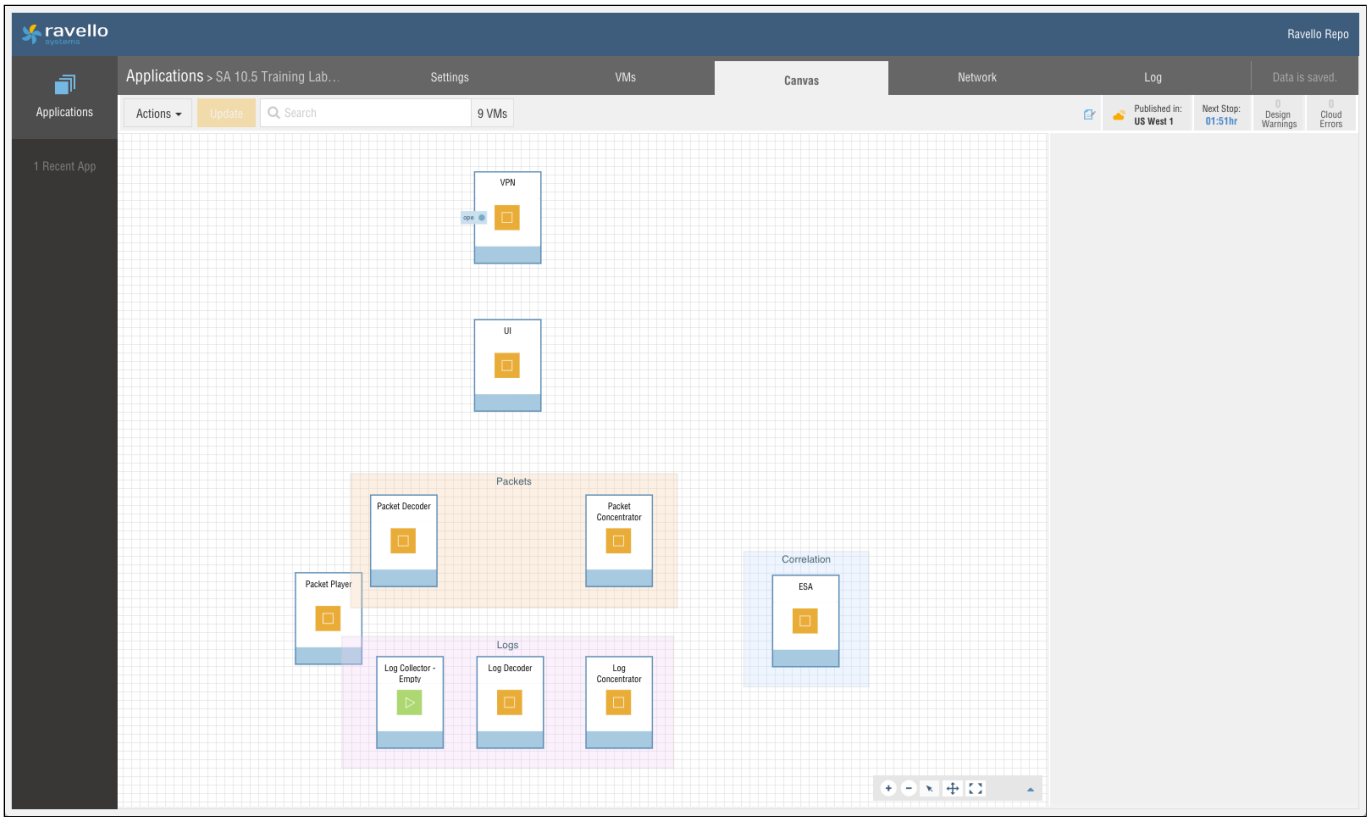
- Before building your SA environment, you should have your IP address scheme planned precisely. I've provided a spreadsheet with all needed info on the your lab environment.
- If your SA hosts are located on different networks, you should ensure that any necessary ports on your network firewall are opened between the SA hosts. See http://sadocs.emc.com/0_en-us/089_105InfCtr/100_Dep/10_NetPrt for documentation on which ports to open. As SA is a hodge-podge of many different services, there are many ports to be opened.
- Critically, you must decide how name resolution will be achieved in your SA environment. Will you be using DNS to resolve your SA hostnames? If so, ensure that all SA hostnames are resolvable before adding them to your SA UI. If you're not using DNS to resolve SA hostnames, ensure that all SA hosts have been added to the /etc/hosts file of your UI host. *Using DNS is generally less error-prone.*

Lab Exercise - Imaging a Host

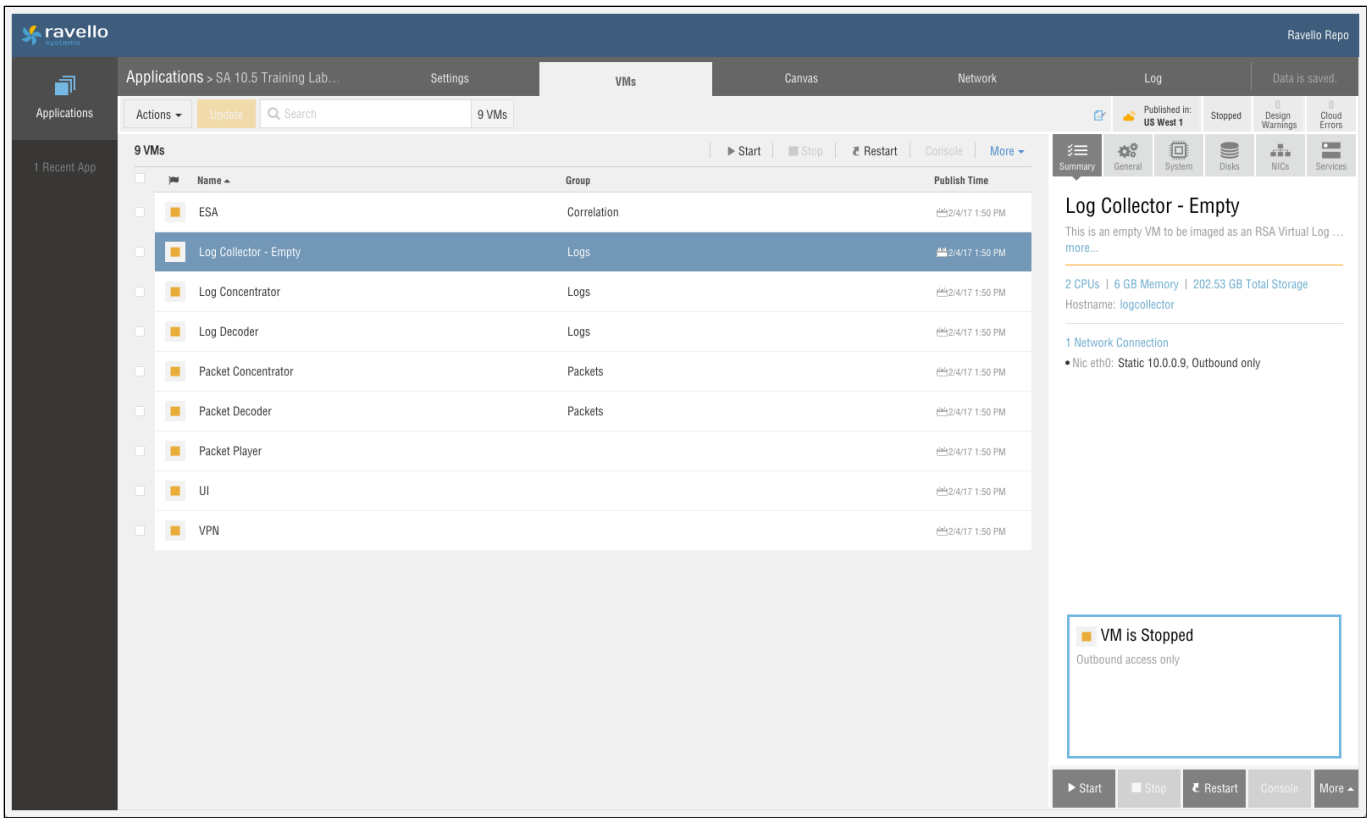
Before we start using our lab environment, we need to give it one missing piece - namely a Log Collector. In the lab, you'll find an empty virtual machine called "**Log Collector - Empty**", which has the SA 10.5.0.0 DVD ISO mounted and ready to boot.

Lab Management

As you know, every student has his or her own dedicated lab environment. I am giving you the power to control your lab environment. On your Chromebook, you'll find a bookmark in the bookmarks bar called "Lab Management". Click on it. This takes you to the Ravello application management console, where we can control our lab VM's: namely starting and stopping them, and accessing their local consoles. Ravello Systems is the cloud provider we use to host our lab environment.



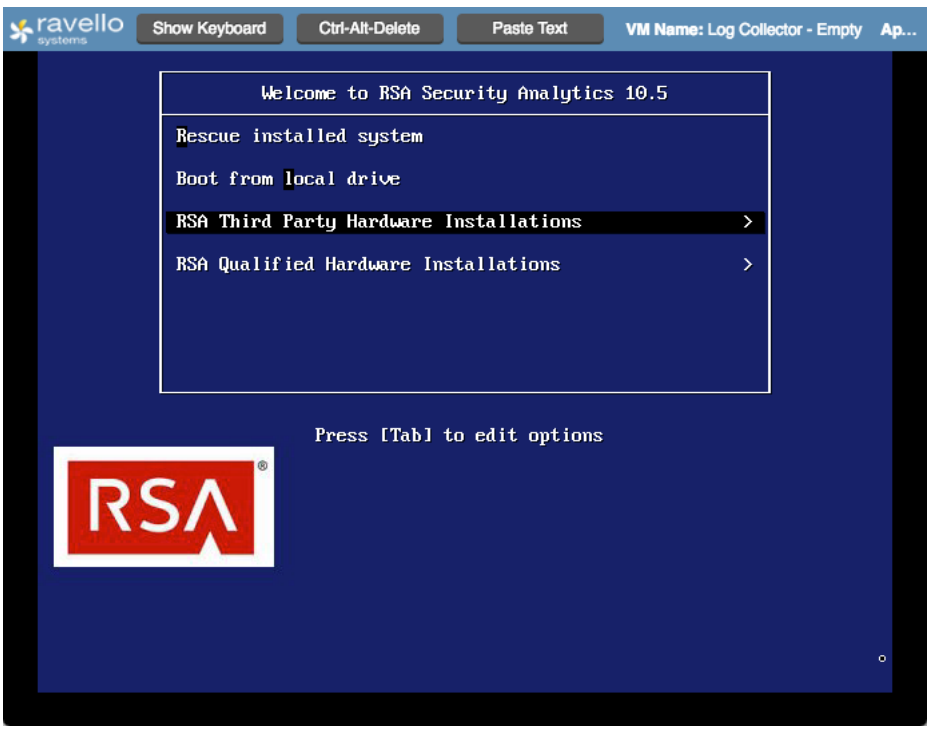
Click on the **VMs** tab, and then click on **Log Collector - Empty**.



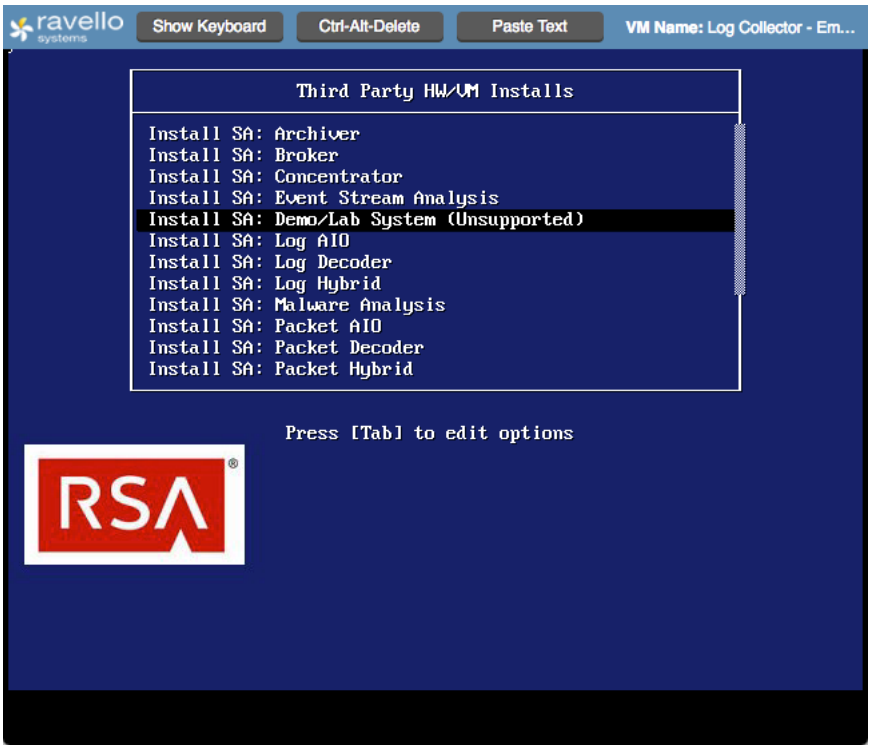
Click the ▶ Start button, either in the toolbar near the top, or on the VM pane to the right. The status will change to VM is Starting. It will take a few minutes for the VM to start, and when it does, click the Console button.

Please also start the VM's named **UI, VPN, Packet Decoder,** and **Packet Concentrator.**

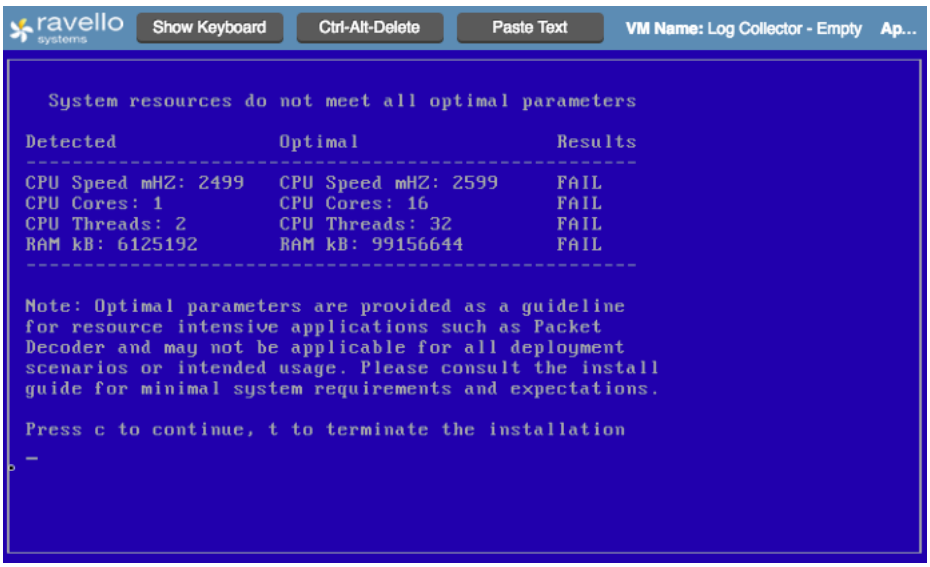
Now you're in the local console of the VM, just the same as if you were sitting in front of it with monitor, keyboard, and mouse.



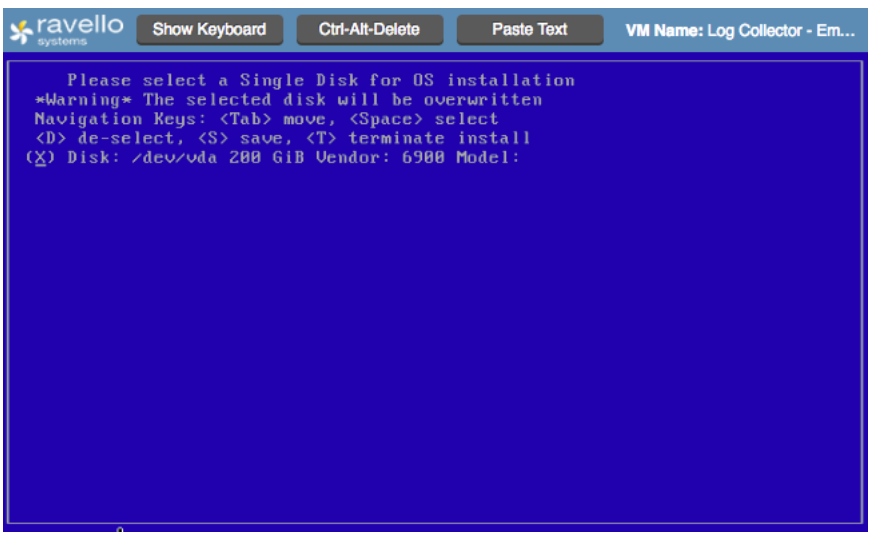
This is the boot menu for the installation DVD. Using your keyboard, select **RSA Third Party Hardware Installations** and hit **<Enter>**. **Third Party Hardware** is what you choose for when you're not installing on RSA Appliances. Virtual machines fall under this. **Qualified Hardware** is what you choose for an actual RSA Appliance.



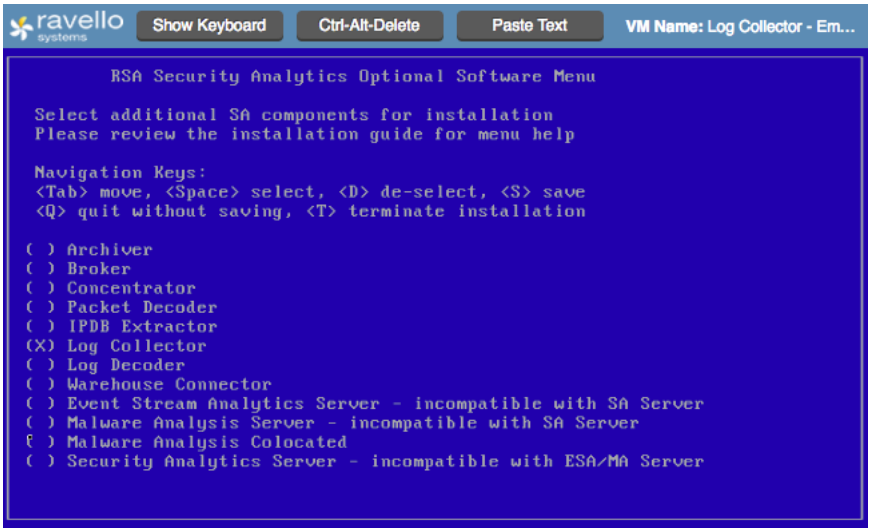
It will take a few minutes for Linux to boot into the installer. Now select **Demo/Lab System (Unsupported)**. As the option suggest, it's most appropriate for lab environments where supportability isn't an issue. This option allows you to choose the individual software components you wish to install, and it will not enforce minimum hardware requirements needed to obtain RSA Support. This is fine for our purposes.



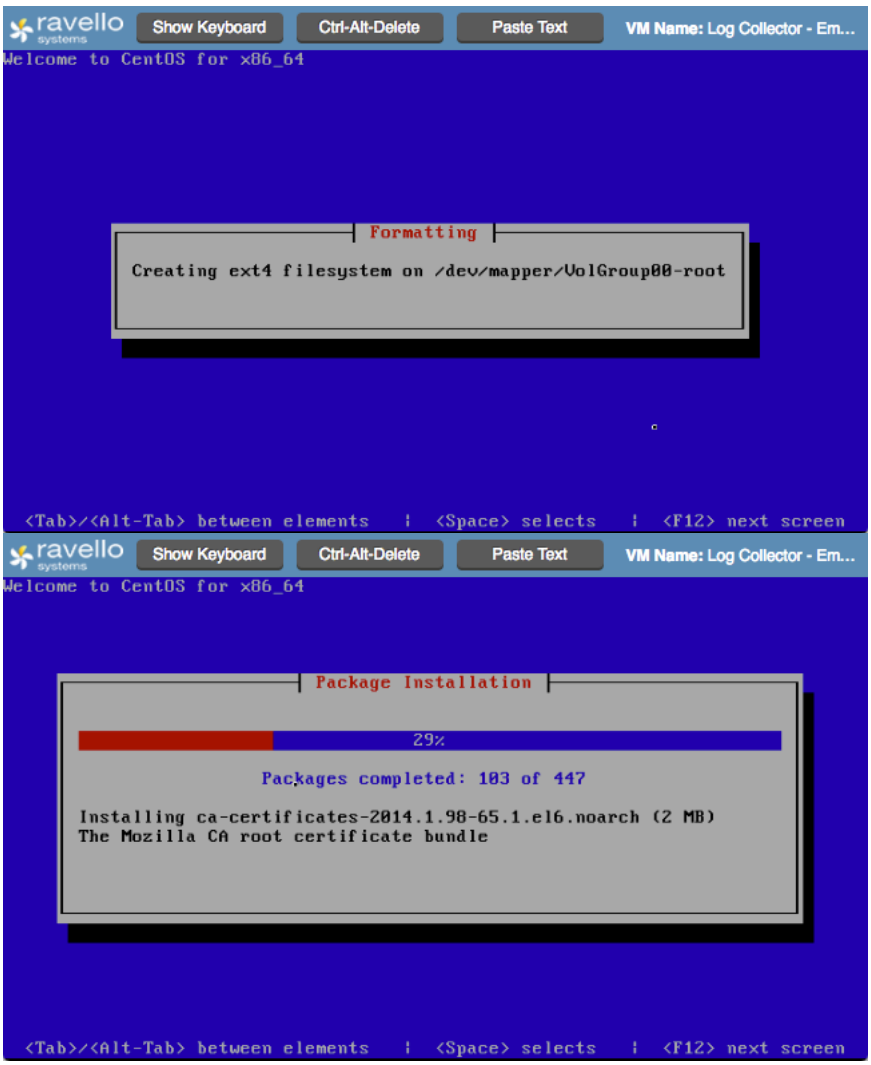
The installer will first display a warning about the VM not meeting optimal parameters. That's perfectly fine for our training lab, so press the letter '**c**' to continue.



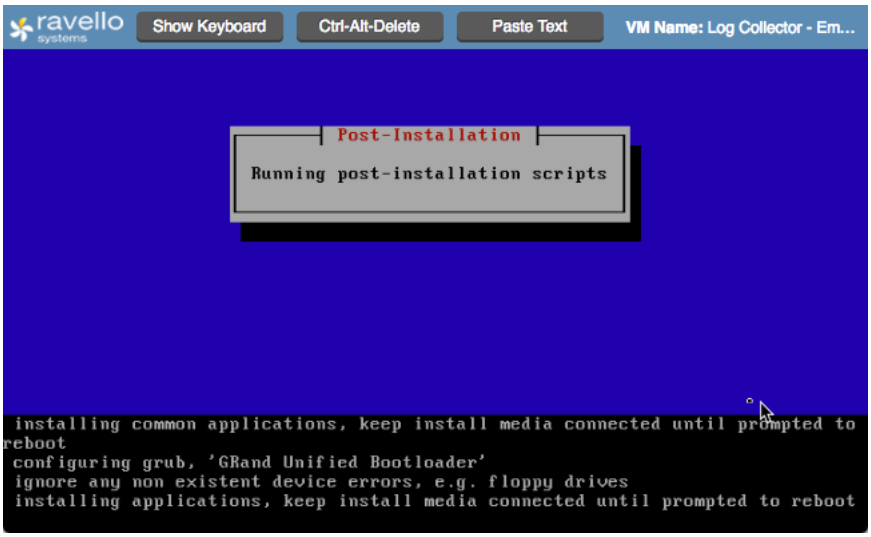
You'll be asked to select a disk you wish to install on. There's only one disk in our VM, so just hit the <**Space Bar>** and then type the letter '**s**'.



You'll now be prompted to choose which components you'd like to install. Use the **Tab** key to scroll down to Log Collector, hit the <**Space Bar>** and then type the letter '**s**'.



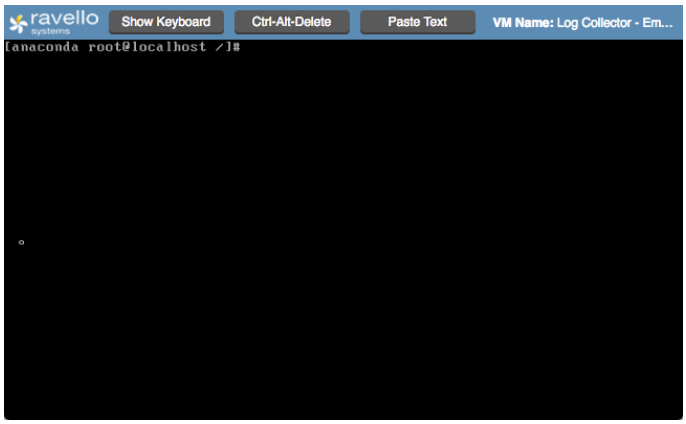
The installation will now proceed. You will see partitions being created and filesystem formatting take place. After a few minutes, you will see a progress bar for package installation. Sit back and let it do its thing - the rest of the installation may take 10 to 15 minutes.



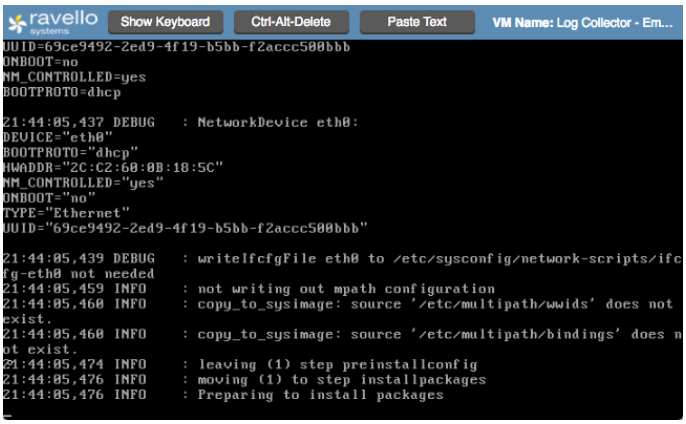
Once the packages have been installed, it will then run some post-installation scripts to prepare the system for its first run.

Accessing the TTY Consoles During Installation

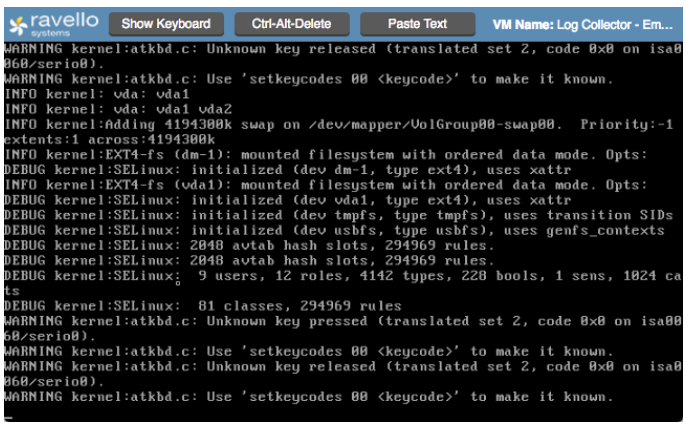
FYI, whilst the install is running, you can gain access to a shell prompt by hitting Alt-F2 to switch TTY's. Hit Alt-F1 to switch back to the installer. Hitting Alt-F3, Alt-F4, and Alt-F5 will display system messages that can help track progress through the installer.



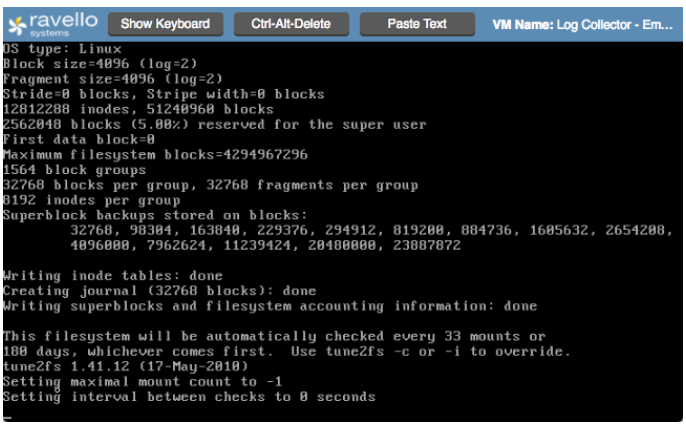
Alt-F2



Alt-F3

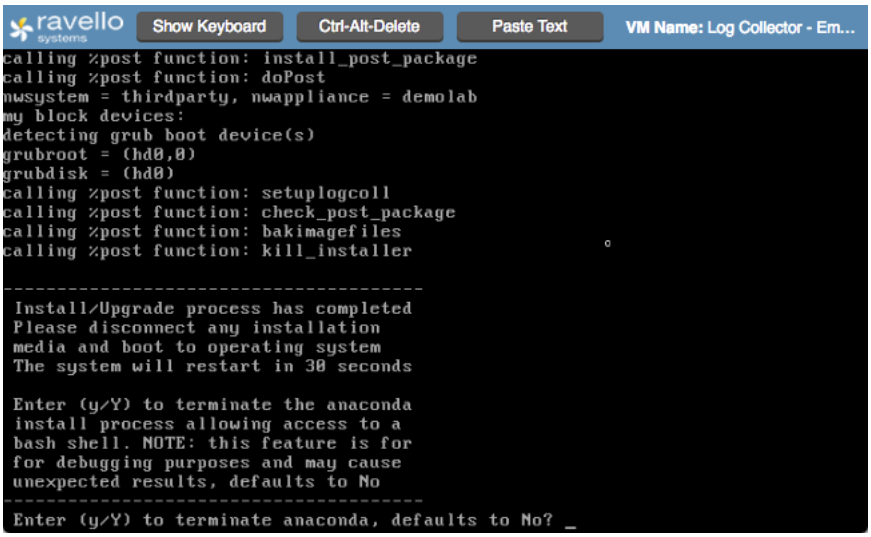


Alt-F4

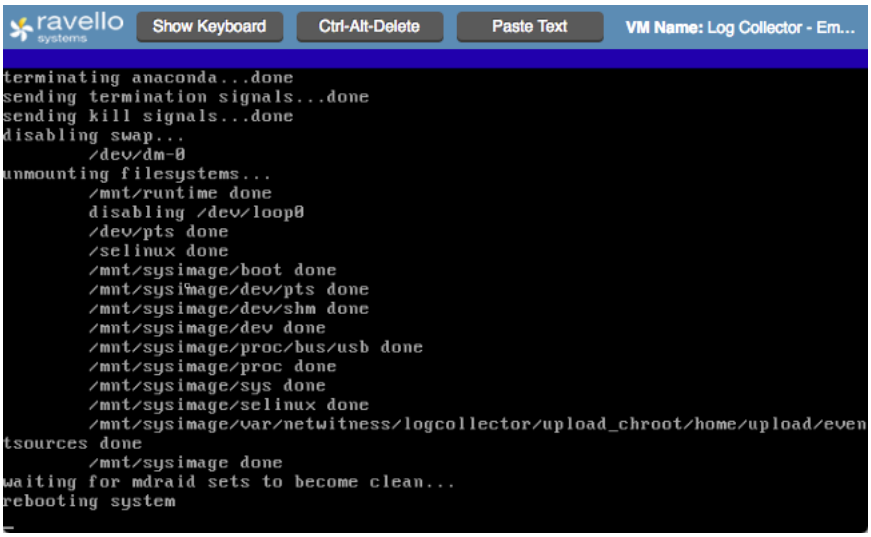


Alt-F5

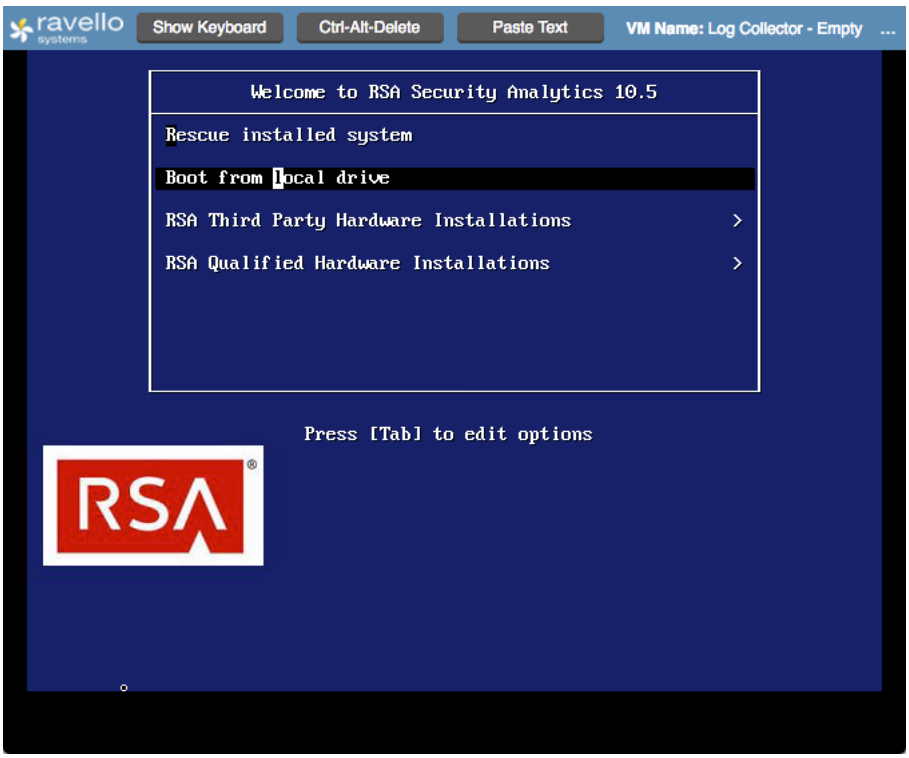
Finishing the Installation



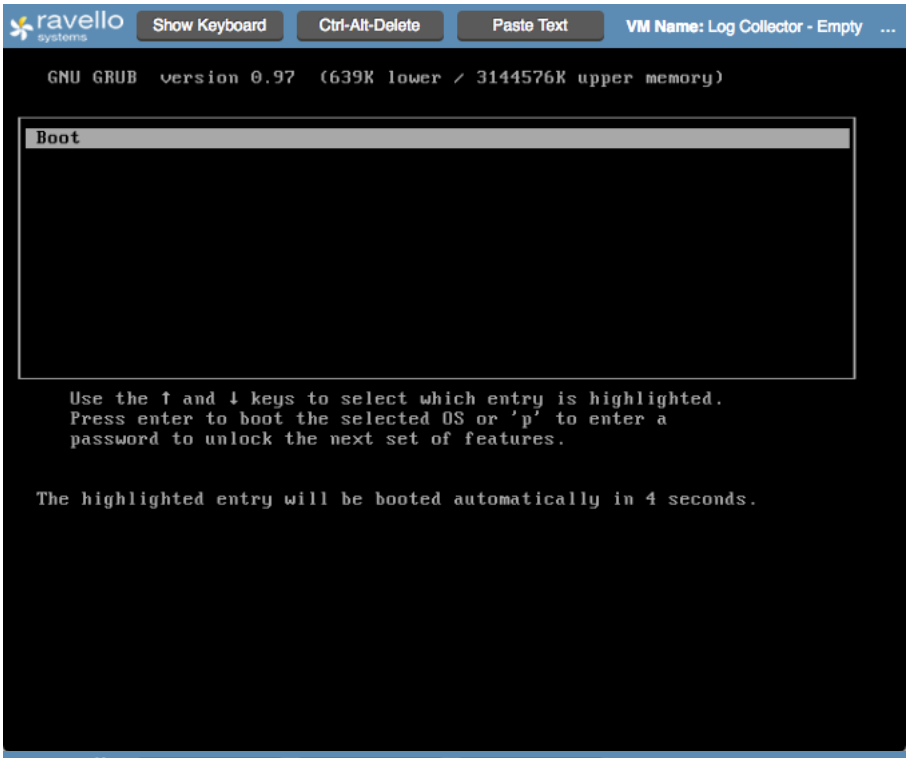
At the end, you will be told that the installation is complete, and you can type '**y**' or '**n**' to continue. Typing '**y**' will take you to a command prompt, which would enable you to customise the installed OS prior to booting it for the first time. We shan't do that, though. Type '**n**' and hit **<enter>**. The system will reboot!

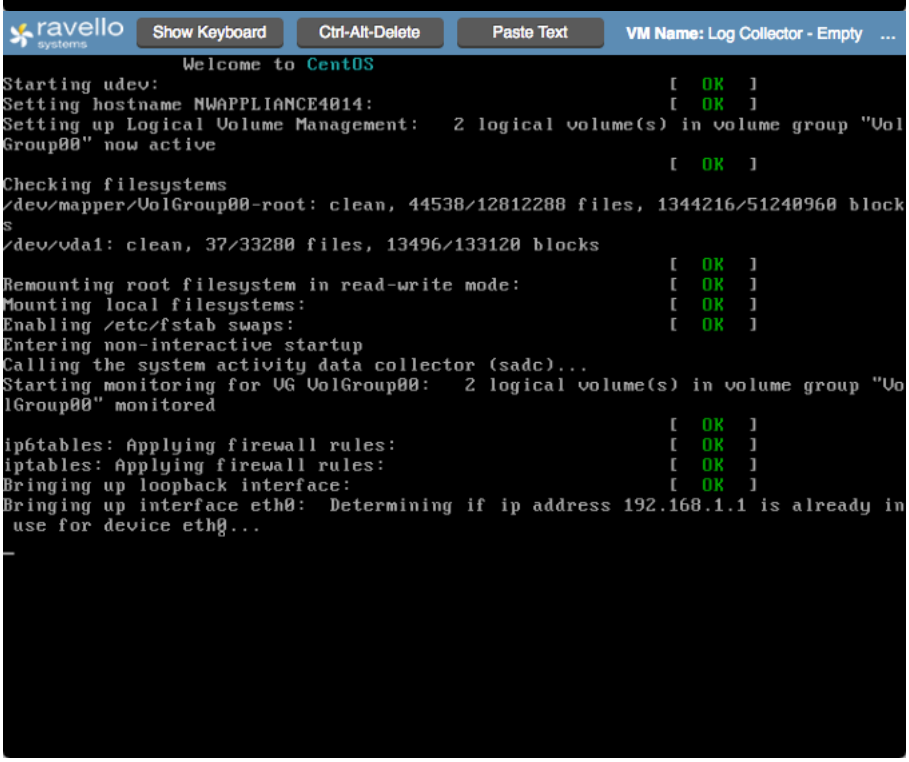


Logging in and Initial Setup



Because we've still got the DVD ISO mounted, the system will boot back to the DVD boot menu. Choose **Boot from local drive**.





You'll see the Grub bootloader and the Linux boot sequence.

```
ravello systems   Show Keyboard   Ctrl-Alt-Delete   Paste Text   VM Name: Log Collector - Empty ...

RSA Security Analytics Appliance
Kernel 2.6.32-504.1.3.el6.x86_64 on an x86_64
NWAPPLIANCE4014 login: root
Password: _
```

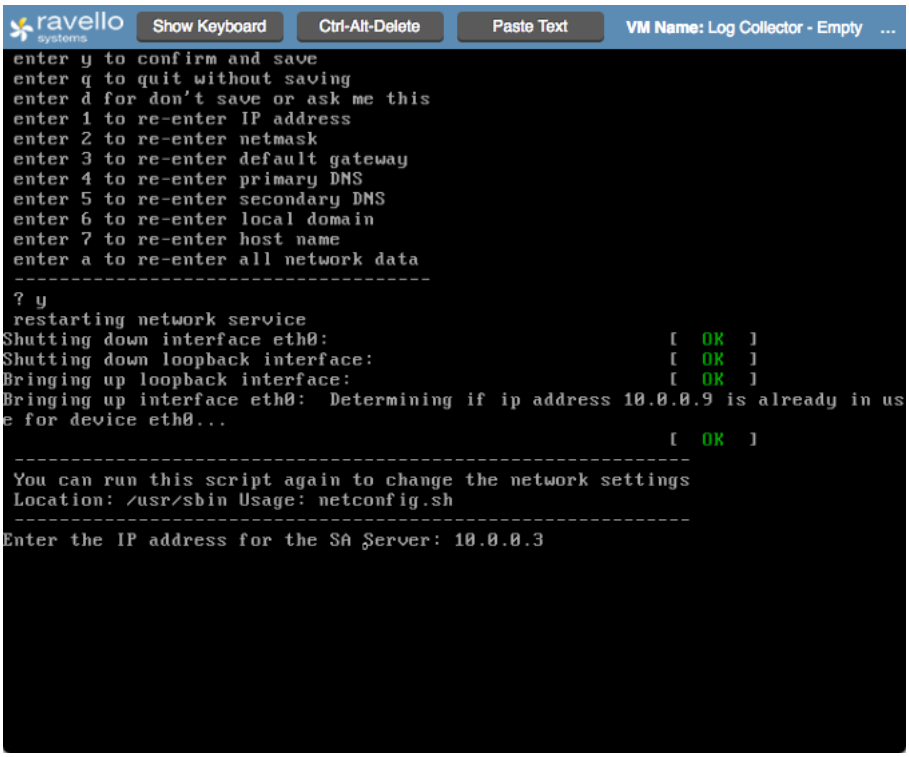
The default credentials to log in are: **root** / **netwitness** .

```
-----------------------------------------------------------------
This script configures the management, network interface settings
<CTRL><C> to Cancel Without Saving

Please enter the requested network settings, or enter for none
You will be prompted to confirm selections or quit before saving
-----------------------------------------------------------------

please enter system IP address or d for DHCP? 10.0.0.9
please enter system netmask? 255.255.255.0
please enter default gateway? 10.0.0.1
please enter primary DNS server IP address? 10.0.0.1
please enter secondary DNS server IP, press enter for none?
please enter local domain name, press enter for none? ktalab.local
please enter unqualified host Name, press enter for none? sa_
```

wong hostname!

```
you entered the following network parameters
IP Address: 10.0.0.9
Netmask: 255.255.255.0
Default Gateway: 10.0.0.1
Primary DNS: 10.0.0.1
Secondary DNS: 255.255.255.0
Local Domain: ktalab.local
Host Name: logcollector
-------------------------------------
enter y to confirm and save
enter q to quit without saving
enter d for don't save or ask me this
enter 1 to re-enter IP address
enter 2 to re-enter netmask
enter 3 to re-enter default gateway
enter 4 to re-enter primary DNS
enter 5 to re-enter secondary DNS
enter 6 to re-enter local domain
enter 7 to re-enter host name
enter a to re-enter all network data
-------------------------------------
? _
```

Immediately upon login, you will be prompted by a script to enter networking information for the host. Use the information provided in your spreadsheet. There is no secondary DNS server in the lab.



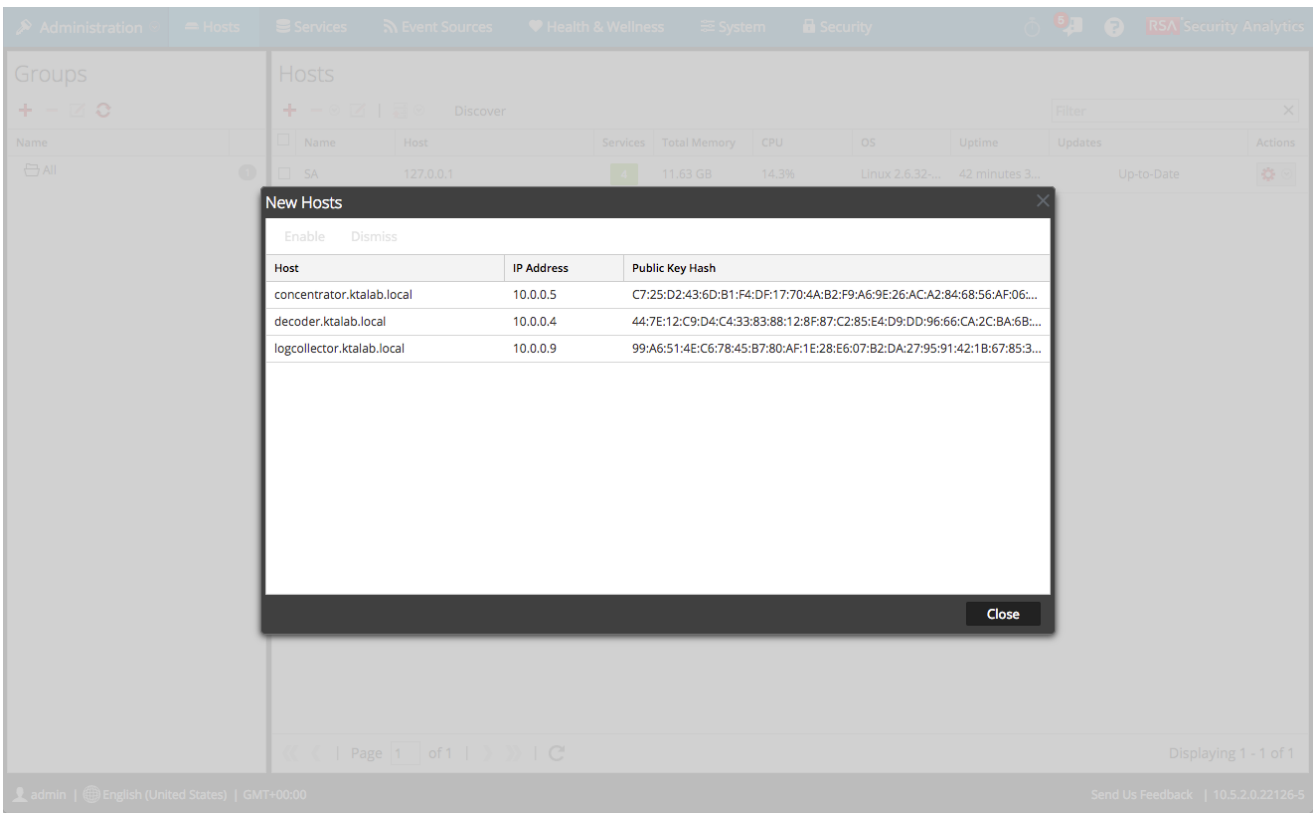
At the end, you will be prompted to type the IP address for the SA Server (the UI serveer). Enter '**10.0.0.3**' and **<enter>** to commit your network configuration. Do **NOT** type the hostname 'sa' here!

At this point, the local puppet service will be configured in the background to point the SA server, and the **puppet** service will be started. We're done with the local console now.

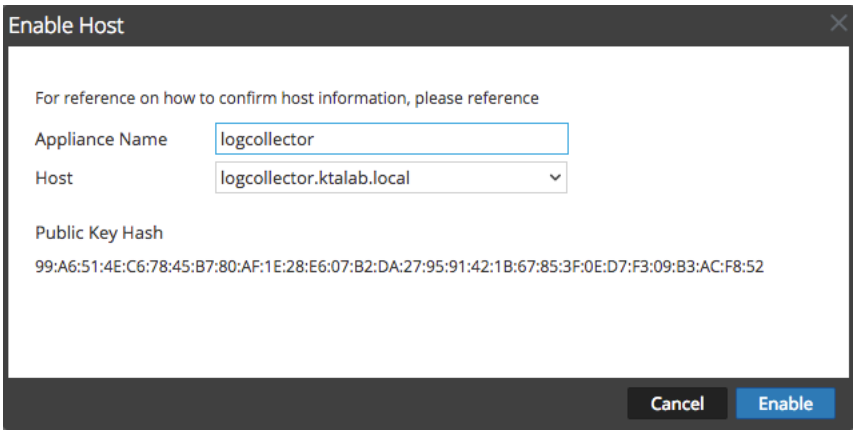
Provisioning the Host in SA

Now that we've built the system and pointed it to the SA server, it must now be provisioned within the SA server. Every system in an SA deployment must be controlled by a single SA UI server, or **puppet master**. To do this, we must provision, or Enable it in your SA user interface.

1. There's a bookmark in your Chromebook's bookmarks bar called **SA**. Click this.
2. Log in to Security Analytics with the credentials **admin** / **netwitness**.
3. Using the 'Start Menu' in the upper-left hand corner (which by default says '**Dashboard**'), navigate to **Administration->Hosts**.

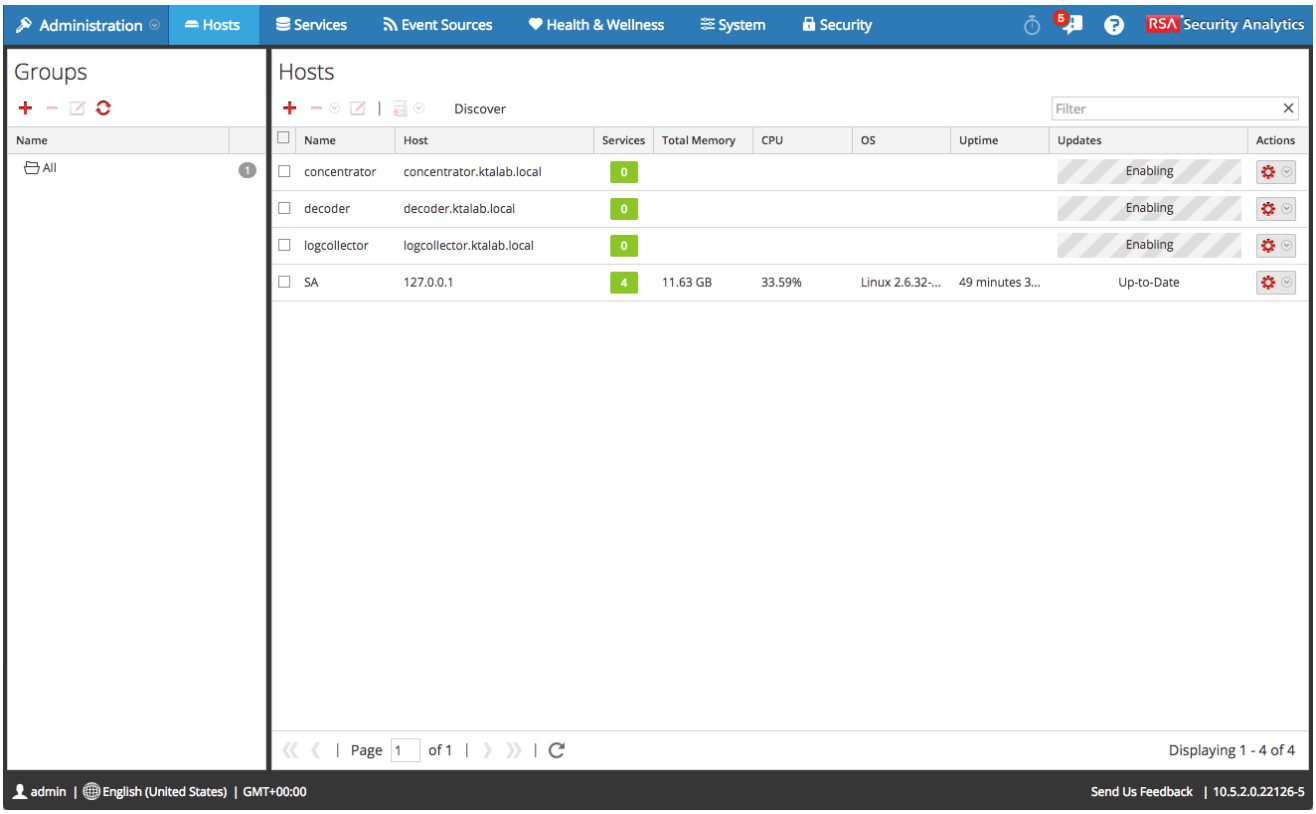


4. You should now see a screen similar to this. Click the host **logcollector.ktalab.local** and click **Enable**.

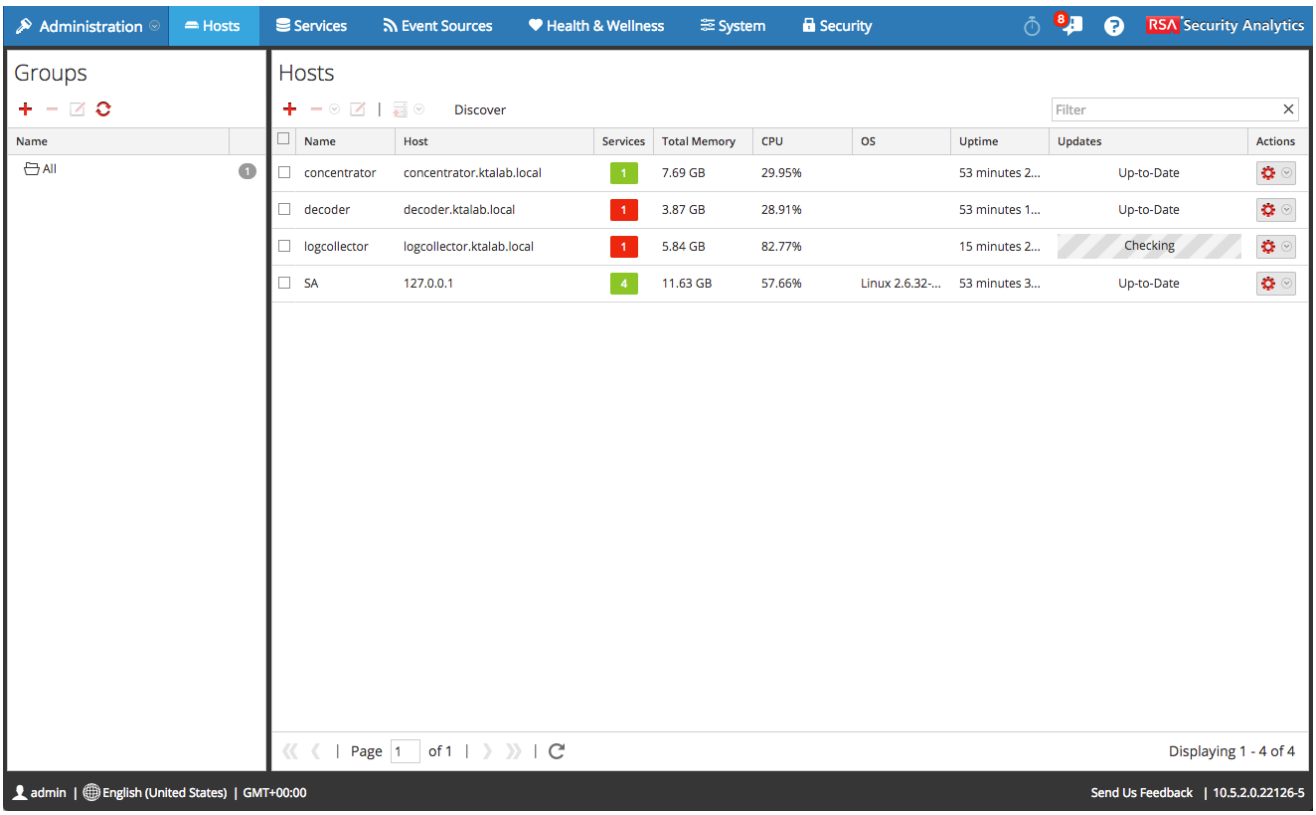


5. Delete the ***.ktalab.local*** suffix from the **Appliance Name** so that it reads just '**logcollector**'. This will make it slightly easier to read later on. Click **Enable** again.

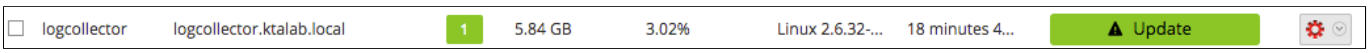
6. Repeat the above steps for **concentrator** and **decoder**.



7. On the **Administration->Hosts** page of SA, you will see the three appliances you just enabled, with the word ***Enabling*** under the **Updates** column. Except for **logcollector**, this text will change to **Up-To-Date**. You will also see the number under the services column for each host increment. They will be red at first, but they will ultimately turn green.



For **logcollector**, the **Updates** column will ultimately display '**Update**'. This is because the DVD we used to install this host with contains version 10.5.0.0 of SA, but our UI is running version 10.5.2.0. The other pre-built hosts in our lab are already on 10.5.2.0. We'll upgrade the **logcollector** host in the lesson on upgrades.



You Mentioned Puppet

There are two main services that serve as the administrative glue between SA hosts:

- **puppet -** The SA server host runs a service called **puppetmaster**, and all SA hosts run a service called **puppet-agent**. Puppet uses a client/server model to ensure that needed configurations are rolled out to each host. The **puppetmaster** contains **recipes** which define how a host node should behave.
- **mcollective** - mcollective is a type of remote procedure call (RPC) service, where the SA host can trigger commands to be run remotely on each SA node.

These two services are entirely separate, but SA uses them together to control what's going on on SA Hosts, including provisioning, and upgrades.

What Happens When We Enable an SA Host?

When we first IP an SA host and tell it the hostname of the SA server, it starts the **puppet** service on the host. Under the bonnet, this is really called **puppet-agent**, which can also be run from the command line with **puppet-agent -t**. On its first run, the puppet agent will generate an SSL keypair (stored in **/var/lib/puppet/ssl**) and a CSR (certificate signing request), which it automatically sends to the **puppetmaster**.

The **puppetmaster** is also a CA (Certificate Authority). Until a CSR generated by a puppet agent has been signed by the puppetmaster, the agent will just sit there, doing nothing. You can see repeated messages in /var/log/messages like this:

```
Feb  6 21:25:24 logcollector puppet-agent[1289]: Did not receive certificate
Feb  6 21:25:34 logcollector puppet-agent[1289]: Did not receive certificate
Feb  6 21:25:44 logcollector puppet-agent[1289]: Did not receive certificate
Feb  6 21:25:54 logcollector puppet-agent[1289]: Did not receive certificate
Feb  6 21:26:04 logcollector puppet-agent[1289]: Did not receive certificate
Feb  6 21:26:14 logcollector puppet-agent[1289]: Did not receive certificate
```

But when we sign the certificate request with the puppetmaster, the puppet agent suddenly springs to life:

```
Feb  6 21:32:37 logcollector puppet-agent[1289]: Starting Puppet client version 3.6.2
Feb  6 21:32:38 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/facts.d]/mode) mode changed '0755' to '0775'
Feb  6 21:32:39 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/facts.d/applianceVersion.sh]/ensure) defined content as
'{md5}bd28d8db285f1eba04a07cba339afddd'
Feb  6 21:32:39 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/facts.d/rabbitmq_disk_limit.sh]/ensure) defined content as
'{md5}024146c308f810f4ea7801a8167fb4d1'
Feb  6 21:32:39 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/facts.d/rabbitmq_plugins_dir.sh]/ensure) defined content as
'{md5}51a1cd5e71760131d27c3e6cccf0d697'
Feb  6 21:32:39 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/facts.d/nw_admin_path.sh]/ensure) defined content as
'{md5}a25c56de3e8587371aba1c79b3568169'
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet]/ensure) created
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/provider]/ensure) created
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/provider/java_ks]/ensure) created
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/provider/file_line]/ensure) created
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/util]/ensure) created
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/util/ipcidr.rb]/ensure) defined content as
'{md5}e1160dfd6e73fc5ef2bb8abc291f6fd5'
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/provider/firewall.rb]/ensure) defined content as
'{md5}32d2f5e5dcc082986b82ef26a119038b'
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/provider/firewallchain]/ensure) created
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/provider/firewall]/ensure) created
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/provider/firewall/iptables.rb]/ensure) defined
content as '{md5}470fdc741c7d60843573b23a2c9fc321'
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/provider/java_ks/keytool.rb]/ensure) defined
content as '{md5}6ac5a79da78281e667515e383151584f'
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/parser]/ensure) created
Feb  6 21:32:42 logcollector puppet-agent[1564]: (/File[/var/lib/puppet/lib/puppet/parser/functions]/ensure) created
```

...and so on.

How do we sign a request, then? Remember when we clicked Enable on Administration->Hosts in the UI? This action signed the certificate. It also does a few things internally within the SA app.

Top Tip: **Never, EVER,** sign an SA host certificate using the puppet command line tool. We won't even tell you how to do this, as we don't want you to mess up your configuration. Doing this won't allow the UI to update its configuration to properly support the host you've enabled.

Install 'acpid'

Once we've finished, there's one last thing I'd like to you do, namely install '**acpid**' on the VM and start it. '**acpid**' is a system daemon which will allow the VM to shutdown gracefully when we stop it from the cloud management console. From a shell prompt, type:

```
yum --enablerepo RSASoftware install -y acpid
service acpid start
```

Output:

```
[root@logcollector ~]# yum --enablerepo RSASoftware install -y acpid
Loaded plugins: fastestmirror
Setting up Install Process
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Running transaction check
---> Package acpid.x86_64 0:1.0.10-2.1.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================================================
 Package                                Arch                                   Version
Repository                                     Size
================================================================================================================================
Installing:
 acpid                                  x86_64                                 1.0.10-2.1.el6
RSASoftware                                    36 k

Transaction Summary
================================================================================================================================
Install       1 Package(s)

Total download size: 36 k
Installed size: 73 k
Downloading Packages:
acpid-1.0.10-2.1.el6.x86_64.rpm
                   |  36 kB     00:00
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : acpid-1.0.10-2.1.el6.x86_64
                                    1/1
  Verifying  : acpid-1.0.10-2.1.el6.x86_64
                                    1/1

Installed:
  acpid.x86_64 0:1.0.10-2.1.el6


Complete!

[root@logcollector ~]# service acpid start
Starting acpi daemon:                                      [  OK  ]
```

Notable Puppet Files and Directories

- **/etc/puppet** - Stores puppet configuration
- **/etc/puppet/puppet.conf** - Stores high-level puppet configuration. Mainly just the certificate name, which is in fact the puppet node ID.
- **/etc/puppet/csr_attributes.yaml** - Stores attributes specific to a puppet certificate request. This stores the FQDN and hostname of the puppet client, which are important for the CSR.
- **/etc/puppet/scripts** - Stores RSA-specific scripts which are used by puppet
- **/etc/puppet/modules** - Puppetmaster-specific. Contains puppet recipes.
- **/var/lib/puppet** - Stores additional puppet configuration
- **/var/lib/puppet/node_id** - Stores the puppet node ID, referenced by all RSA puppet scripts.
- **/var/lib/puppet/ssl** - Contains all puppet certificates and CSR;s

8. SA Platform (Hardware & OS)

RSA Appliances

Until relatively recently, running Security Analytics required purchasing 'appliances' from RSA. These appliances are Dell Linux servers provisioned to run SA. They are not appliances in the same sense as a black box solution - they run a full Linux OS with all the requisite complexity thereof. RSA has undertaken efforts to simplify the installation and upgrade process, however at this juncture, interaction with the CLI for administration is almost inevitable.

RSA Appliance Types

- **Conventional** - These are the majority of physical appliances sold by RSA. They usually are built to run a single major SA service. I.e. a Packet Decoder appliance, Log Decoder, Concentrator, SA UI, ESA, Archiver. Conventional Concentrators, Archivers, and Decoders all require the addition of external storage in the form of a DAC or SAN.
- **Hybrid** - Hybrids are appliances designed to run both a Decoder service (packet or logs) and Concentrator service. They have a smaller footprint and lower price point to convential Decoder/Concentrator pairs, which must use DAC or SAN storage. Hybrids are typically used for lower-bandwidth areas or customers with more limited security budgets.
- **All-In-One** - These are hybrid appliances which also run the SA UI service. AIO's are fully self-sufficient.

Other Appliance Types

- Virtual Hardware - SA is supported running in virtual machines under the VMware ESXi platform. VM's can be built using a DVD ISO file - the same ISO used to build a standard appliance. The expectation is that performance will be lesser than a physical appliance.
- Third-Party Hardware

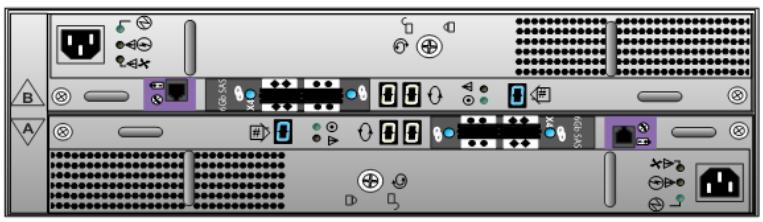
Storage

DAC' and JBOD's

Most SA Core appliances, except for Brokers, ESA, and Hybrids, have the majority of their storage located on external hardware, called DAC's (Direct Attached Capacity) or JBOD's (a legacy term - Just a Bunch of Disks). The standard DAC is 3U in height and contains 15 drive slots. These are attached to a RAID controller on the appliance via a SAS cable. SAS stands for Serial Attached SCSI, which is a standard for high-speed disk access.

The drives in the DAC's are then arranged into RAID5 logical drives, which are then partitioned via LVM (Logical Volume Manager) and ultimately used as the main database storage for SA Core.

3U DAC, 15 Drive - Front

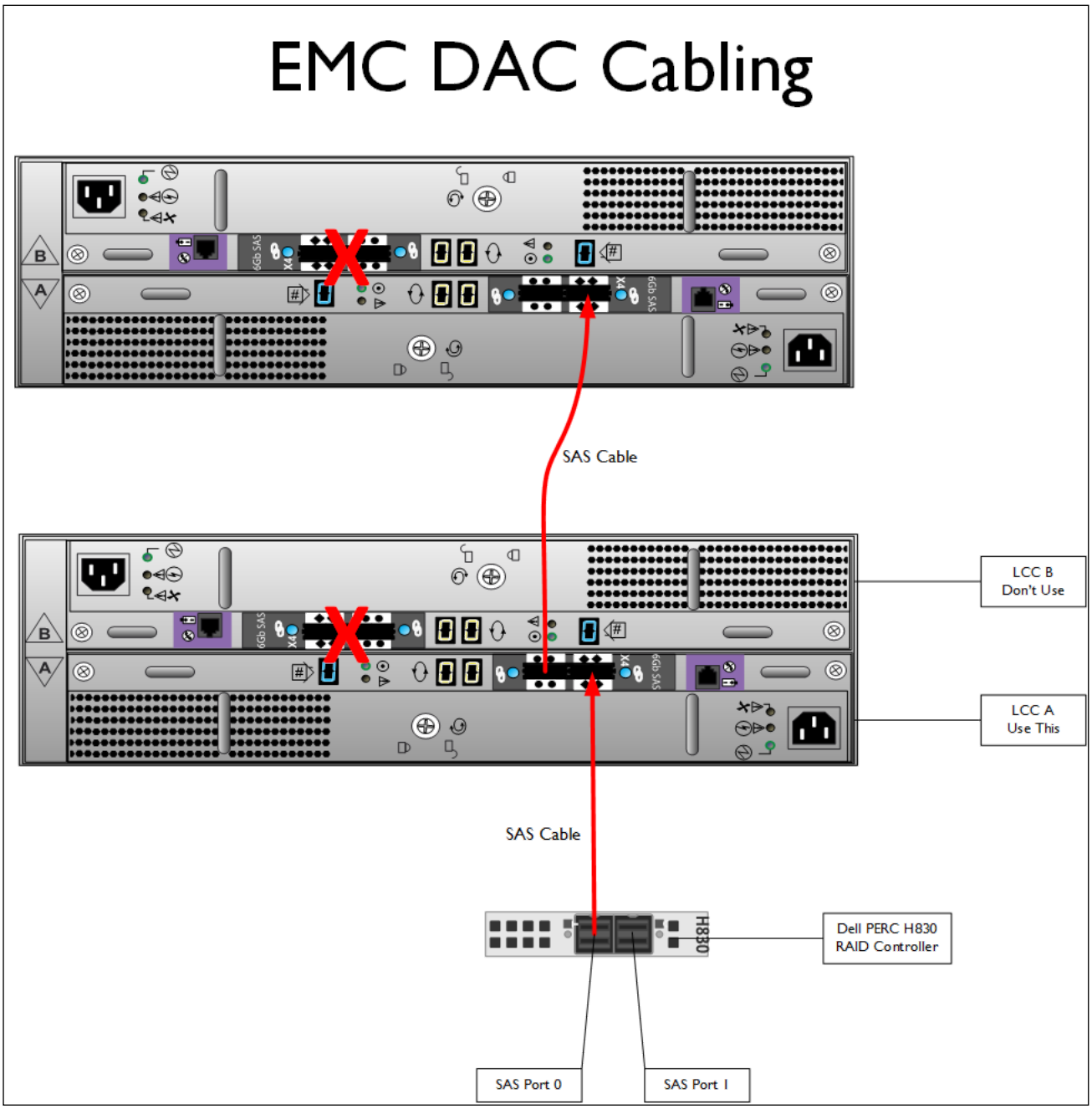


3U DAC, 15 Drive - Rear

EMC 15 Drive DAC

DAC's can be daisy-chained together by making use of their two SAS ports. You will notice that there are two Link Control Cards, or LCC's, on each DAC, with the top being LCC B and the bottom being LCC A. Between the two LCC's, there are four SAS ports in total on a DAC, but only the two bottom ports (LCC A) are used. The top two ports (LCC B) are only used if the DAC is connected to a SAN as a DAE (Disk Array Enclosure), in which case they would serve as a redundant path if the A channel were to fail.

When daisy-chaining DAC's, the right port is the uplink port (towards the RAID controller) and the left port is the downlink port (away from the RAID controller, towards downstream DAC's).



SAN

SA supports SAN-based database storage via use a PCIe HBA card. SAN configuration is beyond the scope of this course.

RSA Appliance Hardware

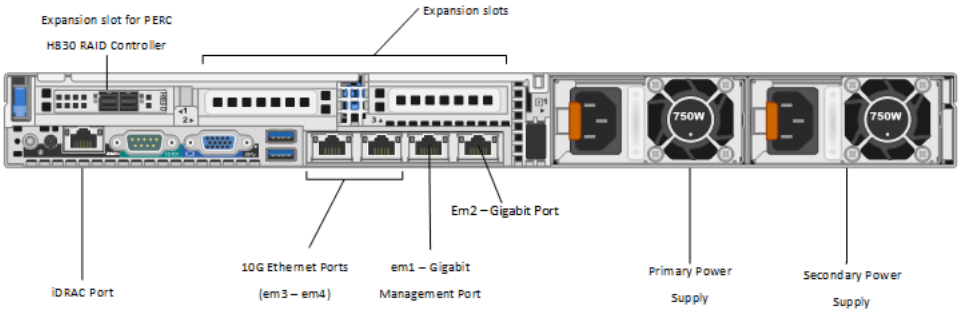
Series 5

With one exception, S5 appliances are based on Dell PowerEdge R630 servers. Series 5 hybrids are the exception to this, which are based on the Dell PowerEdge R730xd server. The ESA version of series 5 hardware is a variant of the standard R630 configuration, with more disks and memory, and a different CPU.

Series 5 hardware comes with two Gigabit copper ports, and two 10G copper ports. The order of the ports is: **3(10G) 4(10G) 1(1G) 2(1G).** This card can optionally be swapped for a fibre card.

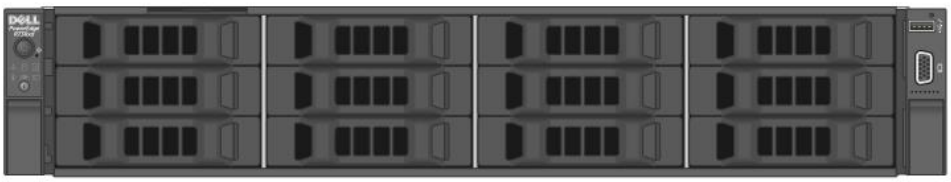
Series 5 Appliance Front / Dell PowerEdge R630

Series 5 Appliance Back / Dell PowerEdge R630

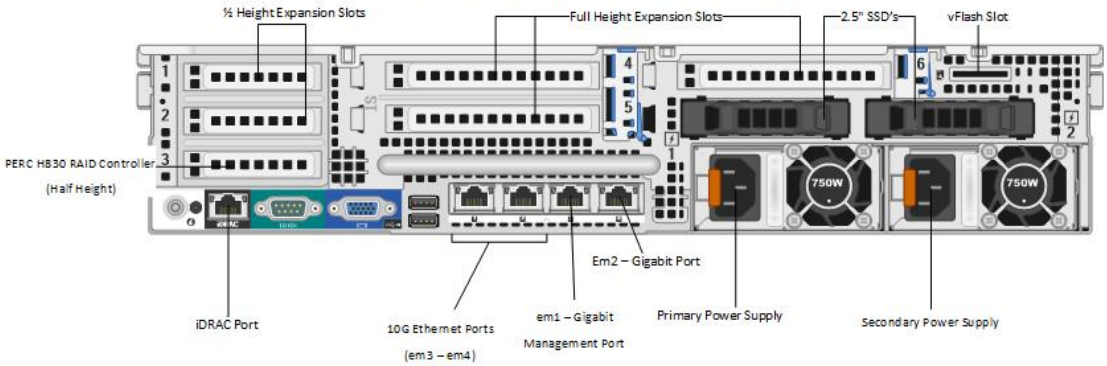


Standard Series 5 Appliance (Dell R630)

Series 5 Hybrid Appliance Front / Dell PowerEdge R730xd



Series 5 Hybrid Appliance Back / Dell PowerEdge R730xd



Series 5 Hybrid Appliance (Dell R730xd)

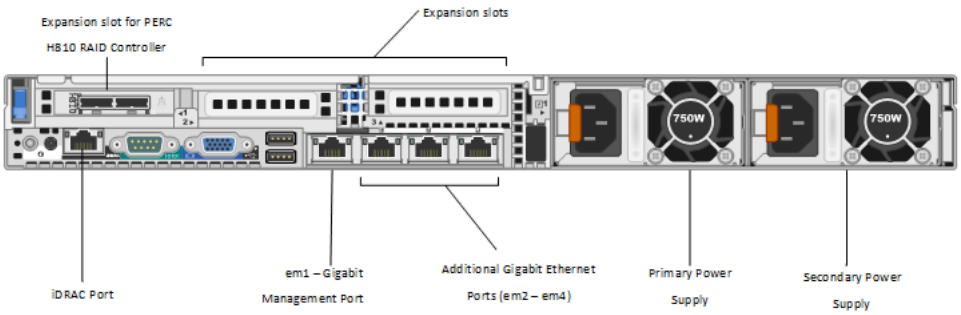
Series 4S

The series 4S (sometimes called 'S4S') appliance is based on the Dell PowerEdge R620 chassis with 10 hard drive slots. It is a 1U server. The series 4S is equipped with four built-in hard drives drives (two 146 GB drives for the OS and two 1TB drives) for running the OS, which are attached to a Dell PERC H710P mini on the motherboard, and a Dell PERC H810 RAID controller for attaching external storage (a SAN HBA card is optionally availabile). The appliance must be attached to external storage to be used as a Decoder, Concentrator, or Archiver, as there is not enough internal drive space to store the databases.

Note on 4S Hybrids: Please note that series 4S hybrids have ten internal 1 TB drives, as opposed to the standard two 146 GB drives and two 1TB drives.

Series 4S Appliance Front / Dell PowerEdge R620

Series 4S Appliance Back / Dell PowerEdge R620



Series 4S Hybrids

Series 4S hybrids have a significant flaw, namely that they do not have an SSD for their Concentrator index. Series 3 hybrids had markedly better query performance than series 4S because series 3 hybrids had an SSD for their index. This mistake has been rectified in series 5 hybrids.

4S SD Card Issues

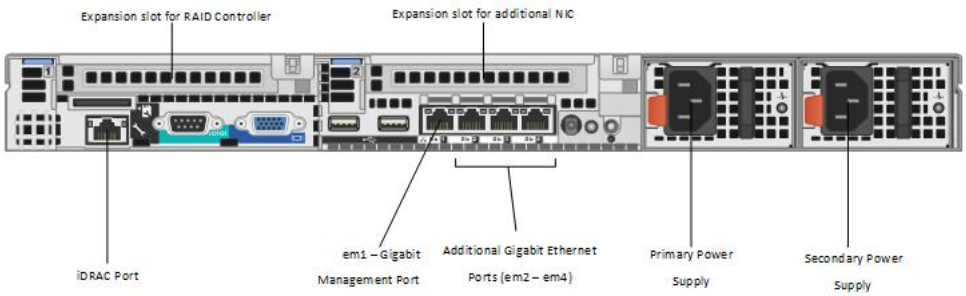
Early series 4S appliances ran their operating systems off of internal SD Card drives instead of conventional hard drives. Many of these drives failed in the field, even leading to entire appliance failures. If your appliance is running from SD Cards but hasn't crashed yet, performance penalties can still be incurred. We recommend re-imaging your appliances so as to no longer run off of SD Cards as soon as possible. This is a non-trivial task - please contact Kensington Technology Associates to assist in this process, or contact RSA. These SD Cards are not accessible from the exterior of the appliance: the case cover must be removed, which could well void your warranty.

Series 4

Based on Dell PowerEdge R610 servers, these are a less beefy version of the series 4S, with only six internal drives, less RAM and CPU capacity. All other details are significantly the same.

Series 4 Appliance Front / Dell PowerEdge R610

Series 4 Appliance Back / Dell PowerEdge R610



Capture Cards

All appliances come with a built in 4-port NIC, which can be used for management and capture. Optionally, an additional PCIe NIC can be added for capture, which can be either 1G copper, 1G fibre, or 10G card.

Swapping a Copper NIC for a Fibre NIC

The all-copper NIC of series 5 appliance can optionally be replaced with a fibre-based card. The fibre card has 4 ports - 2x10G fibre and 2x1G copper. After you swap the cards, there will be a wee bit of a problem, though. The appliance will not detect the ports in the correct order, and you won't be able to access the system over the network:

Detected (Incorrect) Port Order, After Swapping NIC Cards:

10g fibre	10g fibre	1g copper	1g copper
em1 MAC: *N*	em2 MAC: *N*+1	em3 MAC: *N*+2	em4 MAC: *N*+3

Correct Port Order:

10g fibre	10g fibre	1g copper	1g copper
em3	em4	em1 (management)	em2

We say this is incorrect because the third port from the left, the first copper port, should always be the management interface. The reason that CentOS gets this 'wrong' for our purposes because of the way its **Udev** service works. **Udev** is the service which is responsible for mapping hardware to OS device names in **/dev**. **Udev** assigns network interface names ('em1', 'em2', etc) based on the MAC address of the NIC, with the numerically lowest (in hex) MAC address getting the lowest value 'em1', and increasing from there. By MAC address, that is the way the fibre NICs' interfaces are labeled, from left to right: 1 2 3 4. What we have to do, then, is instruct Udev and the OS to switch the order of the NIC's to what we need them to be. This can be slightly tedious, as you will usually only have local console access to an appliance after you've swapped out the NIC's.

Correcting the Order:

1. Run "**ifconfig -a"** from the console and write down the MAC addresses of em1, em2, em3, and em4. You may need run "ifconfig em***X***' on each individual interface in order to fit it all on the screen, or pipe it through the less command to paginate the output: "**ifconfig -a | less**"

Let's say that these are the MAC addresses we're writing down (these are not taken from a real system):

Fibre NIC Detected Layout:

em1	2C:C2:60:6D:93:18	*N*
em2	2C:C2:60:6D:93:19	*N*+1
em3	2C:C2:60:6D:93:1A	*N*+2
em4	2C:C2:60:6D:93:1B	*N*+3

2. On paper, let's now juxtapose these so that they're in the proper order, using the knowledge that the MAC's of the physical ports from left to right are *N* to *N*+3. We'll use this new order in later steps

Fibre NIC Corrected Layout:

em1	2C:C2:60:6D:93:1A	*N*+2
em2	2C:C2:60:6D:93:1B	*N*+3
em3	2C:C2:60:6D:93:18	*N*
em4	2C:C2:60:6D:93:19	*N*+1

3. Using either **vim** or **nano**, edit the file **/etc/udev/rules.d/70-persistent-net.rules**. This file defines rules which udev uses to label the interfaces. It will almost certainly be a mess, containing eight entries rather than the standard four - the reason being that it will have entries for the old copper NIC and also for the new fibre NIC. You can either:

- edit the file to remove the lines for the old copper NIC and then edit the remaining lines to change the interface names to the correct order. this can be tedious.
- delete the contents of the file and type the new lines in manually. This can also be tedious but less confusing.
- You can even delete the contents of the file and reboot the system to allow it to be recreated, but unnecessary reboots are usually considered undesirable.

This is what the file should look like when it's done. Comments are omitted:

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="2c:c2:60:6d:93:1a", ATTR{type}=="1", KERNEL=="eth*", NAME="em1"

SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="2c:c2:60:6d:93:1b", ATTR{type}=="1", KERNEL=="eth*", NAME="em2"

SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="2c:c2:60:6d:93:18", ATTR{type}=="1", KERNEL=="eth*", NAME="em3"

SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="2c:c2:60:6d:93:19", ATTR{type}=="1", KERNEL=="eth*", NAME="em4"
```

All you've done is matched the MAC address values in **ATTR{address}=="*<someMACinLowercase>*"** to the **NAME="em*N*"**

Top Tip: You must use **all lowercase** in the 70-persistent-net.rules file. The entries won't take effect if they are in uppercase.

4. Now edit the files **/etc/sysconfig/network-scripts/ifcfg-em*N***, where N is 1, 2, 3, or 4. The **HWADDR=** entry of each file will have the original copper NIC MAC addresses. You must update the **HWADDR=** line of each of the four files to match the new interface order:

Original ifcfg-em1:

```
DEVICE=em1
HWADDR=12:34:56:78:9A:BC
TYPE=Ethernet
UUID=2e01534a-cbbe-40ec-beb4-c16c27ad065f
BOOTPROTO=static
IPADDR=10.0.0.4
NETMASK=255.255.255.0
GATEWAY=10.0.0.1
NM_CONTROLLED=no
ONBOOT=yes
```

New ifcfg-em1:

```
DEVICE=eth0
HWADDR=2C:C2:60:6D:93:1A
TYPE=Ethernet
UUID=2e01534a-cbbe-40ec-beb4-c16c27ad065f
BOOTPROTO=static
IPADDR=10.0.0.4
NETMASK=255.255.255.0
GATEWAY=10.0.0.1
NM_CONTROLLED=no
ONBOOT=yes
```

New ifcfg-em2:

```
DEVICE=eth1
HWADDR=2C:C2:60:6D:93:1B
TYPE=Ethernet
UUID=d7c420a0-73f4-40fb-9df8-b7cc1f798de9
ONBOOT=no
NM_CONTROLLED=yes
BOOTPROTO=dhcp
```

...and do so for the remaining two files **ifcfg-em3** and **ifcfg-em4**.

5. Reboot the system by typing '**reboot**' from the console. If you've done everything properly, the NIC's will be in the correct order at next boot, and the management interface should be accessible.

CentOS Operating System

The only supported operating system for Security Analytics is CentOS version 6. CentOS 6 is a freely-available redistribution of RedHat Enterprise Linux 6. Customers should NOT install their own OS updates as these can potentially break SA. RSA provides system updates in its software upgrades and security patches. With the exception of RSA's own software, these are simply CentOS packages that have been re-signed with RSA's own GPG key.

Changing an SA Host's IP Address

Changing the IP address of an SA host can be a very frustrating experience for customers, as it isn't a straightforward procedure. Rest easy, though, as we will get you through this process relatively unscathed. In our example, we are changing the IP address of our concentrator host from 10.0.0.5 to 10.0.0.14.

The procedure:

1. Download the script **Puppet_agent_reset_certs_non_interact_V0.51-ktal.sh** from http://kensingtontechlimited.com/files/Puppet_agent_reset_certs_non_interact_V0.51-ktal.sh. Upload it to the host to be re-IP'd using **scp**. Be sure to use Ken-Tech's version of this script rather than RSA's, if you want to avoid trouble.

2. From the root shell of the host to be re-IP'd (**not** your UI host!), execute the script thusly:

```
# ./Puppet_agent_reset_certs_non_interact_V0.51-ktal.sh
Stopping puppet agent:                                     [  OK  ]
Stopping node sa@localhost ...
Resetting node sa@localhost ...
Stopping rabbitmq-server: rabbitmq-server.
Stopping collectd:                                         [  OK  ]
Shutting down mcollective:                                 [  OK  ]
killing Erlang.............Killed
This is not a Packet Decoder
This is not a Broker
This is not a Log Decoder
This is not a WareHouse Connector
This is not a Log Decoder
This is not an Archiver
This is not an Archiver/Workbench
This is a core appliance Decoder, Concentrator or Archiver
[root@concentrator ~]#
```

This will remove most of the puppet configuration related to the host, including its SSL certificates, and it will also reset the RabbitMQ configuration back to an initial state. It will leave the puppet-agent of the host in a stopped state. We want to leave it this way for now.

3. In the UI, navigate to **Administration->Hosts**. Select the SA Host you wish to re-IP , and then click the dropdown, and then click **Remove and Repurpose Host**. This removes the host's entry from SA. It also revokes all SSL certificates related to the host from the Puppet Master, and it removes any RabbitMQ links to the host.

4. Update name resolution:

- If you are using **/etc/hosts** for name resolution of your SA hosts, update the file so that the host's entry points to its new IP address. It may be necessary to do this on all of your SA hosts!
- If not using **/etc/hosts**, update the DNS entry for the host to point to its new IP address.

5. On the host to be re-IP'd, update the entries **IPADDR**, **NETMASK**, and **GATEWAY** by updating **/etc/sysconfig/network-scripts/ifcfg-em1** or **ifcfg-eth0** (if a VM).

```
DEVICE=eth0
HWADDR=12:34:56:78:9A:BC
TYPE=Ethernet
UUID=2e01534a-cbbe-40ec-beb4-c16c27ad065f
BOOTPROTO=static
IPADDR=10.0.0.14
NETMASK=255.255.255.0
GATEWAY=10.0.0.1
NM_CONTROLLED=no
ONBOOT=yes
```

6. On the host to be re-IP'd, edit the file **/etc/hosts** and update the IP address entry next to the local hostname (not the loopback entry for 127.0.0.1, though!).

```
# Created by NetWitness Installer on Tue May 10 03:36:09 UTC 2016
127.0.0.1 concentrator localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 concentrator localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.0.14 concentrator.ktalab.local concentrator
10.0.0.3 puppetmaster.local
```

Top Tip: If your host is being moved to a location on your network where the UI's address is now behind a NAT IP, as opposed to its real IP, then also update the **puppetmaster.local** entry in **/etc/hosts** to match the NAT IP of the SA UI host.

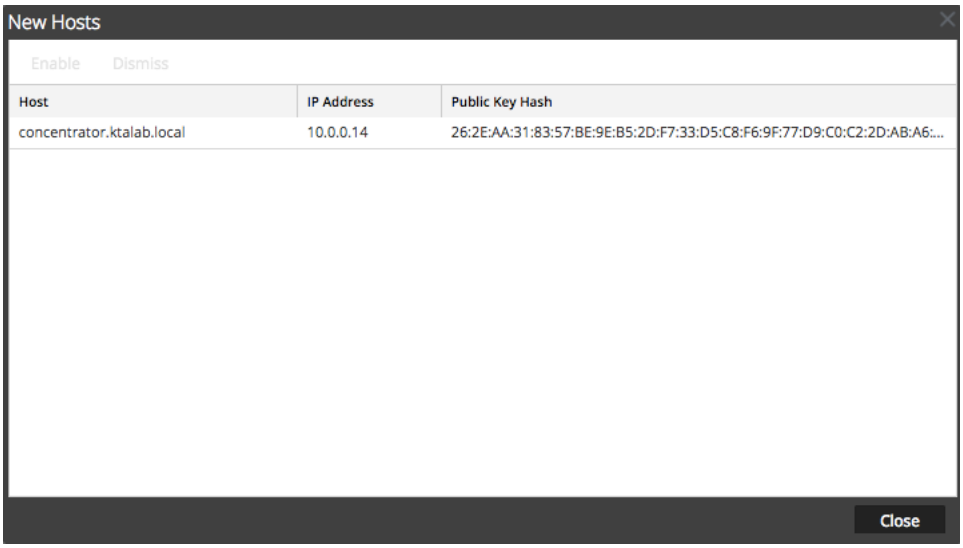
7. On the host to be re-IP'd, edit **/etc/puppet/csr_attributes.yaml** and update '**ipaddress=*<IP>***' inside the line which starts with '**1.2.840.113549.1.9.7**' so that it now lists the updated IP:

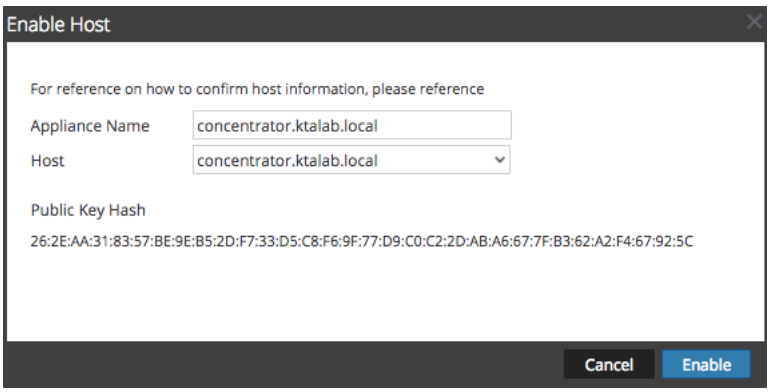
```
custom_attributes:
 1.2.840.113549.1.9.7: fqdn=concentrator.ktalab.local,ipaddress=10.0.0.14,type=base
```

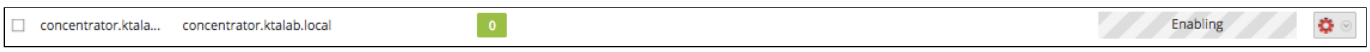
8. If you must physically move the host or its network cabling, you may now shutdown the host with '**shutdown -h now**'. Or if leaving it physically connected, you may reboot it by issuing the '**reboot**' command.

9. Once the re-IP'd host is back online and with network connectivity, navigate back to the **Administration->Hosts** page of the UI, then either refresh the page or click the **Discover** button. You may now proceed with normal provisioning procedure for an SA host, which will re-establish all necessary RabbitMQ links and re-execute puppet on the updated host.

On the UI:







Watching puppet-agent's initial run on the re-IP'd Concentrator:

```
Feb 18 17:53:06 concentrator puppet-agent[1736]: Did not receive certificate
Feb 18 17:53:16 concentrator puppet-agent[1736]: Did not receive certificate
Feb 18 17:53:26 concentrator puppet-agent[1736]: Starting Puppet client version 3.8.2
Feb 18 17:53:49 concentrator puppet-agent[2105]: (/Stage[main]/Ssh/Exec[fix-ssh]/returns) executed successfully
Feb 18 17:53:51 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/File[sa.pem]/ensure) defined content as
'{md5}7d888b5d01dffa2d9d36b8d47fb673e9'
Feb 18 17:54:04 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[create-system-vhost]/returns) executed successfully
Feb 18 17:54:07 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[set-carlos-federation-policy]/returns) executed
successfully
Feb 18 17:54:12 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[create-sa_node_id-user]/returns) executed successfully
Feb 18 17:54:15 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[set-sa_node_id-system-vhost-permissions]/returns)
executed successfully
Feb 18 17:54:18 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[create-sa-vhost]/returns) executed successfully
Feb 18 17:54:21 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[set-guest-user-sa_vhost-permissions]/returns) executed
successfully
Feb 18 17:54:24 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[set-sa_node_id-user-sa_vhost-permissions]/returns)
executed successfully
Feb 18 17:54:26 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[set-carlos-federation-sa_vhost-policy]/returns)
executed successfully
Feb 18 17:54:29 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[set-guest-user-system-vhost-permissions]/returns)
executed successfully
Feb 18 17:54:34 concentrator puppet-agent[2105]: (/Stage[main]/Yumconfig/Exec[disable-Centos-Repos]/returns) executed successfully
Feb 18 17:54:39 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[create-node_id-user]/returns) executed successfully
Feb 18 17:54:42 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[set-node_id-user-sa_vhost-permissions]/returns)
executed successfully
Feb 18 17:54:45 concentrator puppet-agent[2105]: (/Stage[main]/Rabbitmq/Exec[set-node_id-system-vhost-permissions]/returns) executed
successfully
Feb 18 17:54:46 concentrator puppet-agent[2105]: (/Stage[main]/Mcollective/File[/etc/mcollective/facts.yaml]/content) content
changed '{md5}54e9f2a4e09ca998ed7dc520e908a5b8' to '{md5}786c201a16943660292ee30b775554d3'
Feb 18 17:54:47 concentrator puppet-agent[2105]: Finished catalog run in 59.32 seconds
```

If after the initial run of the puppet-agent (which is evidenced by the text "***Finished catalog run in N seconds***" in **/var/log/messages)**, the UI shows **Enable** in red in the updates column with a value of zero in the Service column, you may need to run "**service mcollective restart**" on the host being re-IP'd, and then click on **Enable** on the **Hosts** page one time, which will should complete the provisioning.

Restarting mcollective:

```
[root@concentrator ~]# service mcollective restart
Shutting down mcollective:                                 [  OK  ]
Starting mcollective:                                      [  OK  ]
```

puppet-agent logs on host being re-IP'd after restarting mcollective and then clicking Enable:

```
Feb 18 18:11:56 concentrator puppet-agent[7813]: Caught USR1; storing reload
Feb 18 18:11:57 concentrator puppet-agent[7813]: Processing reload
Feb 18 18:12:28 concentrator puppet-agent[10118]: (/Stage[main]/Ssh/Exec[fix-ssh]/returns) executed successfully
Feb 18 18:12:30 concentrator puppet-agent[10118]: (/Stage[main]/Appliance/Base::Trusted_connection[appliance-trusted-
connection]/Exec[copy-appliance-key_and_cert]/returns) executed successfully
Feb 18 18:12:31 concentrator puppet-agent[10118]: (/Stage[main]/Appliance/Base::Trusted_connection[appliance-trusted-
connection]/Exec[copy-appliance-trusted-connections_sa_node_cert]/returns) executed successfully
Feb 18 18:12:32 concentrator puppet-agent[10118]: (/Stage[main]/Appliance/Base::Trusted_connection[appliance-trusted-
connection]/Exec[copy-appliance-trusted-connections_ca_cert]/returns) executed successfully
Feb 18 18:12:41 concentrator puppet-agent[10118]: (/Stage[main]/Concentrator/Base::Trusted_connection[concentrator-trusted-
connection]/Exec[copy-concentrator-trusted-connections_node_cert]/returns) executed successfully
Feb 18 18:12:42 concentrator puppet-agent[10118]: (/Stage[main]/Concentrator/Base::Trusted_connection[concentrator-trusted-
connection]/Exec[copy-concentrator-trusted-connections_sa_node_cert]/returns) executed successfully
Feb 18 18:12:43 concentrator puppet-agent[10118]: (/Stage[main]/Concentrator/Base::Trusted_connection[concentrator-trusted-
connection]/Exec[copy-concentrator-key_and_cert]/returns) executed successfully
Feb 18 18:12:43 concentrator puppet-agent[10118]: (/Stage[main]/Concentrator/Base::Trusted_connection[concentrator-trusted-
connection]/Exec[copy-concentrator-trusted-connections_ca_cert]/returns) executed successfully
Feb 18 18:12:44 concentrator puppet-agent[10118]: (/Stage[main]/Concentrator/Base::Trusted_connection[concentrator-trusted-
connection]/Exec[tc-restart-nwconcentrator]) Triggered 'refresh' from 4 events
Feb 18 18:13:03 concentrator puppet-agent[10118]: (/Stage[main]/Appliance/Base::Trusted_connection[appliance-trusted-
connection]/Exec[copy-appliance-trusted-connections_node_cert]/returns) executed successfully
Feb 18 18:13:04 concentrator puppet-agent[10118]: (/Stage[main]/Appliance/Base::Trusted_connection[appliance-trusted-
connection]/Exec[tc-restart-nwappliance]) Triggered 'refresh' from 4 events
Feb 18 18:13:09 concentrator puppet-agent[10118]: (/Stage[main]/Yumconfig/Exec[disable-Centos-Repos]/returns) executed successfully
Feb 18 18:13:18 concentrator puppet-agent[10118]: Finished catalog run in 51.75 seconds
```

Provisioning Complete:

Yum Repository

SA updates are distributed in the form of a yum repository, which is hosted locally on the SA server host. The packages in the repository can generally be synchronised from either the RSA Live yum repo, or via an update archive (zip) distributed by RSA. More on upgrades will come later in the course.

Network (Firewall) Ports

Because the SA product stack is built on so many different software components, the number of network ports which must be opened between SA hosts is staggering. We shall endeavour to list them all here for your reference.

Management Connections

Source: All Management Workstation Hosts **Destination:** All SA Hosts

Port	Description
TCP 22	SSH / SFTP / SCP

Source: All Management Workstation Hosts **Destination:** UI Host

Port	Description
TCP 443	HTTPS Management connectivity to SA UI

UI Connections

Source: All Non-UI Hosts **Destination:** UI Host

Port	Description
TCP 80	Yum Repository
TCP 8140	puppet-agent to puppet-master connection
TCP 61614	mcollective
UDP 123	Network Time Protocol (NTP)

Source: UI Host **Destination**: All Non-UI Hosts

Port	Description
TCP 5671	Rabbitmq-Server

Source: UI Host **Destination**: All SA Core Hosts (Decoders, Log Decoders, Concentrators, Brokers, Archivers, Remote Log Collectors, Warehouse Connectors)

Port	Description
TCP 56006	Legacy Appliance Service

Source: UI Host **Destination**: Standalone Broker Hosts

Port	Description
TCP 56003	Connection to Broker service over SSL

Source: UI Host **Destination**: Concentrator Hosts

Port	Description
TCP 56005	Connection to Concentrator service over SSL

Source: UI Host **Destination**: Packet Decoder Hosts

Port	Description
TCP 56004	Connection to Decoder service over SSL
TCP 56020	Connection to Warehouse Connector service over SSL (only if using Warehouse Connector)

Source: UI Host **Destination**: Log Decoder Hosts

Port	Description
TCP 56002	Connection to Log Decoder service over SSL
TCP 56001	Connection to local Log Collector service over SSL
TCP 56020	Connection to Warehouse Connector service over SSL (only if using Warehouse Connector)
TCP 514	(Optional) SA audit logs over syslog (only if configured to send to remote Log Collector)

Source: UI Host **Destination**: Remote Log Collector (VLC) Hosts

Port	Description
TCP 56001	Connection to remote Log Collector service over SSL
TCP 514	(Optional) SA audit logs over syslog (only if configured to send directly to Log Decoder)

Source: UI Host **Destination**: Archiver Hosts

Port	Description
TCP 56008	Connection to Archiver service over SSL
TCP 56007	Connection to Workbench service over SSL

Source: UI Host **Destination**: Standalone Warehouse Connector Hosts

Comment: *A very rare scenario*

Port	Description
TCP 56020	Connection to Warehouse Connector service over SSL

Source: UI Host **Destination**: Event Stream Analysis (ESA) Hosts

Port	Description
TCP 50030	Connection to ESA service over SSL
TCP 27017	Incident Management connection to MongoD service

Source: UI Host **Destination**: Standalone Malware Analysis Hosts

Port	Description
TCP 60007	Connection to Malware Analysis service

Source: UI Host **Destination**: Standalone Reporting Engine Hosts

Comment: *A very rare scenario*

Port	Description
TCP 60007	Connection to Reporting Engine service

Source: UI Host **Destination**: ECAT Host

Port	Description
TCP 443	Connection to ECAT Service

Notification and Audit Connections to Your Internal Infrastructure

Source: UI Host **Destination**: Your SMTP Server Host(s)

Comment: *Port dependent on your environment*

Port	Description

Port	Description
TCP 25 or TCP 587 or TCP 465 or other	(optional) Email SMTP connectivity for alerts (health & wellness, incident management, alerts, reports, licensing, log event source management)

Source: UI Host **Destination**: Your SNMP Trap Receiver Host(s)

Port	Description
UDP or TCP 162	(optional) SNMP traps for Health & Wellness and other alerting.

Source: UI Host **Destination**: Dedicated Non-SA Syslog Audit Syslog Receiver Host(s)

Port	Description
TCP or UDP 514	(Optional) SA Audit logs over syslog (if customer requires audit logs to be sent to a third-party syslog receiver)

Source: UI Host **Destination**: Dedicated Non-SA Syslog Notification Syslog Receiver Host(s)

Port	Description
TCP or UDP 514	(Optional) SA notifications over syslog (Health & Wellness, Event Source Monitoring, Licensing, Reporting, Alerts)

Source: ESA Host(s) **Destination**: Your SMTP Server Host(s)

Comment: *Port dependent on your environment*

Port	Description
TCP 25 or TCP 587 or TCP 465 or other	(optional) Email SMTP connectivity for ESA alert notifications

Source: ESA Host(s) **Destination**: Your SNMP Trap Receiver Host(s)

Port	Description
UDP or TCP 162	(optional) SNMP trap notifications for ESA alerts

Source: ESA Host(s) **Destination**: Dedicated Non-SA Syslog Notification Syslog Receiver Host(s)

Port	Description
TCP or UDP 514	(Optional) SA syslog notifications for ESA alerts

Device-to-Device Connections

Source: Concentrator Host **Destination**: Packet Decoder Host

Comment: *You **REALLY** shouldn't be running this through a firewall anyway as Decoders and Concentrators should be on the same LAN)*

Port	Description
TCP 50004	Unencrypted connection to Packet Decoder service (Optional) for aggregation
TCP 56004	SSL connection to Packet Decoder service (Optional) for aggregation. Slower than unencrypted

Source: Log Concentrator Host **Destination**: Log Decoder Host

Comment: *You **REALLY** shouldn't be running this through a firewall anyway as Decoders and Concentrators should be on the same LAN)*

Port	Description
TCP 50002	Unencrypted connection to Log Decoder service (Optional) for aggregation
TCP 56002	SSL connection to Log Decoder service (Optional) for aggregation. Slower than unencrypted

Source: Dedicated Broker Host **Destination**: All Upstream Concentrator Hosts

Port	Description
TCP 50005	Unencrypted connection to Concentrator service (Optional) for aggregation

Port	Description
TCP 56005	SSL connection to Concentrator service (Optional) for aggregation. Slower than unencrypted, though for Broker connections, the difference is negligible

Source: Dedicated Broker Host **Destination**: All Upstream Broker Hosts

Port	Description
TCP 50003	Unencrypted connection to Broker service (Optional) for aggregation
TCP 56003	SSL connection to Broker service (Optional) for aggregation. Slower than unencrypted, though for Broker connections, the difference is negligible

Source: Dedicated Broker Host **Destination**: Upstream Archiver Hosts

Port	Description
TCP 50008	Unencrypted connection to Archiver service (Optional) for aggregation
TCP 56008	SSL connection to Archiver service (Optional) for aggregation. Slower than unencrypted, though for Broker connections, the difference is negligible
TCP 50007	Unencrypted connection to Workbench service (Optional) for aggregation.
TCP 56007	Connection to Workbench service (Optional) over SSL for aggregation. Slower than unencrypted, though for Broker connections, the difference is negligible

Source: Archiver Host **Destination**: Upstream Log Decoder Host(s)

Port	Description
TCP 50002	Unencrypted connection to Log Decoder service (Optional) for aggregation
TCP 56002	SSL connection to Log Decoder service (Optional) for aggregation. Slower than unencrypted

Source: ESA Host **Destination**: Upstream Concentrator Host(s)

Port	Description
TCP 50005	Unencrypted connection to Concentrator service (Optional) for aggregation
TCP 56005	SSL connection to Concentrator service (Optional) for aggregation. Slower than unencrypted

Source: Remote Log Collector Host **Destination**: Upstream Log Decoder Host(s)

Comment*: If using push model for Remote Log Connector connectivity (this is the more likely scenario)*

Port	Description
TCP 5671	RabbitMQ/AMQP connectivity for pushing logs

Source: Log Decoder Host **Destination**: Upstream Remote Log Collector Host(s)

Comment*: If using pull model for Remote Log Connector connectivity (this is quite rare)*

Port	Description
TCP 5671	RabbitMQ/AMQP connectivity for pulling logs

Source: Standalone Malware Analysis Host **Destination**: Upstream Concentrator Host(s)

Port	Description
TCP 50005	Unencrypted connection from Malware Analysis-local Broker to Concentrator service (Optional) for aggregation
TCP 56005	SSL connection from Malware Analysis-local Broker to Concentrator service (Optional) for aggregation. Slower than unencrypted, though for Broker connections, the difference is negligible

Source: Warehouse Connector **Destination**: RSA Warehouse HDFS IP

Comment: *This might be a cluster Virtual IP (VIP) rather than a physical host, depending on configuration*

Port	Description
TCP/UDP 2049	NFS
TCP/UDP 111	NFS rpcbind
TCP 22	SSH / SCP / SFTP
TCP 50070	WebHDFS connectivity (Hadoop HDFS web UI)

Source: RSA ECAT Host(s) **Destination**: Log Decoder Host

Comment*: If pushing RSA ECAT logs directly to a Log Decoder*

Port	Description
TCP or UDP 514	Syslog connection for ECAT logs

Source: RSA ECAT Host(s) **Destination**: Remote Log Collector Host

Comment*: If pushing RSA ECAT logs to a remote Log Collector*

Port	Description
TCP or UDP 514	Syslog connection for ECAT logs

Source: RSA ECAT Host(s) **Destination**: UI Host

Comment*: If pushing RSA ECAT events into SA Incident Management*

Port	Description
TCP 5671	RabbitMQ/AMQP connection for ECAT events

IPDB & RSA enVision Connections

Only if you need connectivity to a legacy RSA enVision installation for reporting purposes, or if you are using RSA Z-Connector as part of an SA Migration. *This is likely not you.*

Source: UI Host **Destination**: Standalone IPDB Extractor Host

Comment: *This is extremely rare as this service is usually installed on the UI)*

Port	Description
TCP 56025	Connection to IPDB Extractor service over SSL

Source: Host running IPDB Extractor (could be SA host or a standalone IPDB extractor) **Destination**: enVision IPDB Host(s)

Port	Description
TCP & UDP 135	MS Netbios
TCP & UDP 138	MS Netbios
TCP & UDP 139	MS Netbios
TCP 445	MS CIFS

Source: enVision Local Collector Host **Destination**: SA Remote Log Collector Host

Comment: *If using Z-Connector, and it is pushing to a Log Collector*

Port	Description
TCP 514	Syslog connection for RSA Z-Connector

Source: enVision Local Collector Host **Destination**: SA Log Decoder Host

Comment: If using Z-Connector, and it is pushing directly to a Log Decoder

Port	Description

Port	Description
TCP 514	Syslog connection for RSA Z-Connector

Note: Ports for RSA Archer SecOps have been omitted above.

9. Upgrades

Upgrading SA used to be a difficult experience, but the experience has improved somewhat in recent releases. Still, upgrades are by far the most flakey part of the SA experience. I'll start this lesson with a piece of advise which is critical to a successful upgrade of SA:

READ THE UPGRADE DOCUMENTATION

READ THE UPGRADE DOCUMENTATION

READ THE UPGRADE DOCUMENTATION

The Release Notes and Upgrade Guides are your most important resource when upgrading SA. **EVERY SINGLE RELEASE** has different instructions, therefore this lesson will be a bit on the generic side.

Yum and RPM

'**yum**' is RedHat's main system tool for maintaining system packages, be it installing, removing, or upgrading them. The packages are contained within **.rpm** files, and yum works with these package files. RPM stands for RedHat Package Manager. Yum relies on the 'rpm' tool to do its work. RPM maintains an RPM database, which keeps track of installed packages and files.

Yum relies primarily on **yum repositories** to obtain its packages. A yum repository is a file store, usually shared via an HTTP URL, which contains rpm files, and also **repodata** files, which lets yum know which rpm files are in the repository, and their properties.

Yum is version aware, and most packages have dependencies. Very often, packages must be upgraded in groups. For example. most CentOS 6.1 packages depend on other CentOS 6.1 packages, and not CentOS 6.0 packages. So when updating the CentOS version from 6.0 to 6.1, all of the packages must be updated together. The concept also holds true for SA.

The Yum Repository

Being that SA is a CentOS-based system, it follows that SA should use yum, and it does. Every SA UI server has a local yum repository (or repo), which contains packages needed to upgrade SA. This repo can be the source of many a problem. These problems usually manifest when mixing packages of different versions within the repository, which is easy to do. Therefore, READ THE UPGRADE DOCUMENTATION. It will *usually* (but perhaps not always) consist of instructions to clear the repository before proceeding.

SA's yum repository is located in **/var/netwitness/srv/www/rsa/updates**. It contains either two or three subdirectories:

- **RemoteRPMs** - Contains system update packages which have been synced from the public RSA repository
- **repodata** - The yum repository metadata file store
- **SAUserUploaded** - Contains rpm files which have been uploaded through the **Manual Updates** tab of **Administration->System->Updates**.

The repository is made available through a lightweight HTTP server called **lighttpd**, which listens on TCP port 80. All SA systems in a deployment will obtain their updates from the Yum repo that lives on the SA server.

Live Repository

There is also a 'public' SA Yum repository available on the Internet through RSA Live, from which the SA server can be configured to sync its local repository to. Public is in quotes because it requires Live credentials to connect to it. I strongly recommend that you give the 'public' repo it a wide berth.

If you want things to go smoothly, **ALWAYS USE MANUAL UPDATES**!

Supported Upgrade Paths

One of the most important things to watch for when reading the upgrade documentation is the bit on upgrade paths. Depending on the particular release, it may be necessary to upgrade to an interim SA version before upgrading to the version you want to go to. This can mean that if you're behind on your updates, **one upgrade can actually be several upgrades**!

For instance, to upgrade from version 10.5.0.0 of SA to version 10.5.2.1, you must first upgrade to version 10.5.2.0. Don't attempt a direct upgrade, or things could go seriously wrong!

Network Ports

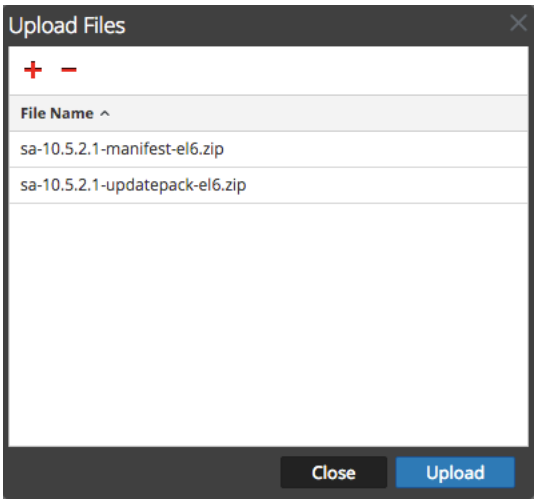
Also, pay attention to the upgrade instructions for any changes in network ports for the release you're upgrading to. You may need to update your firewall rules prior to installing an update. On minor updates, this isn't usually an issue, but for larger releases it's possible that a some new port requirements will come up.

Upgrade Guardrails - Manifest Files

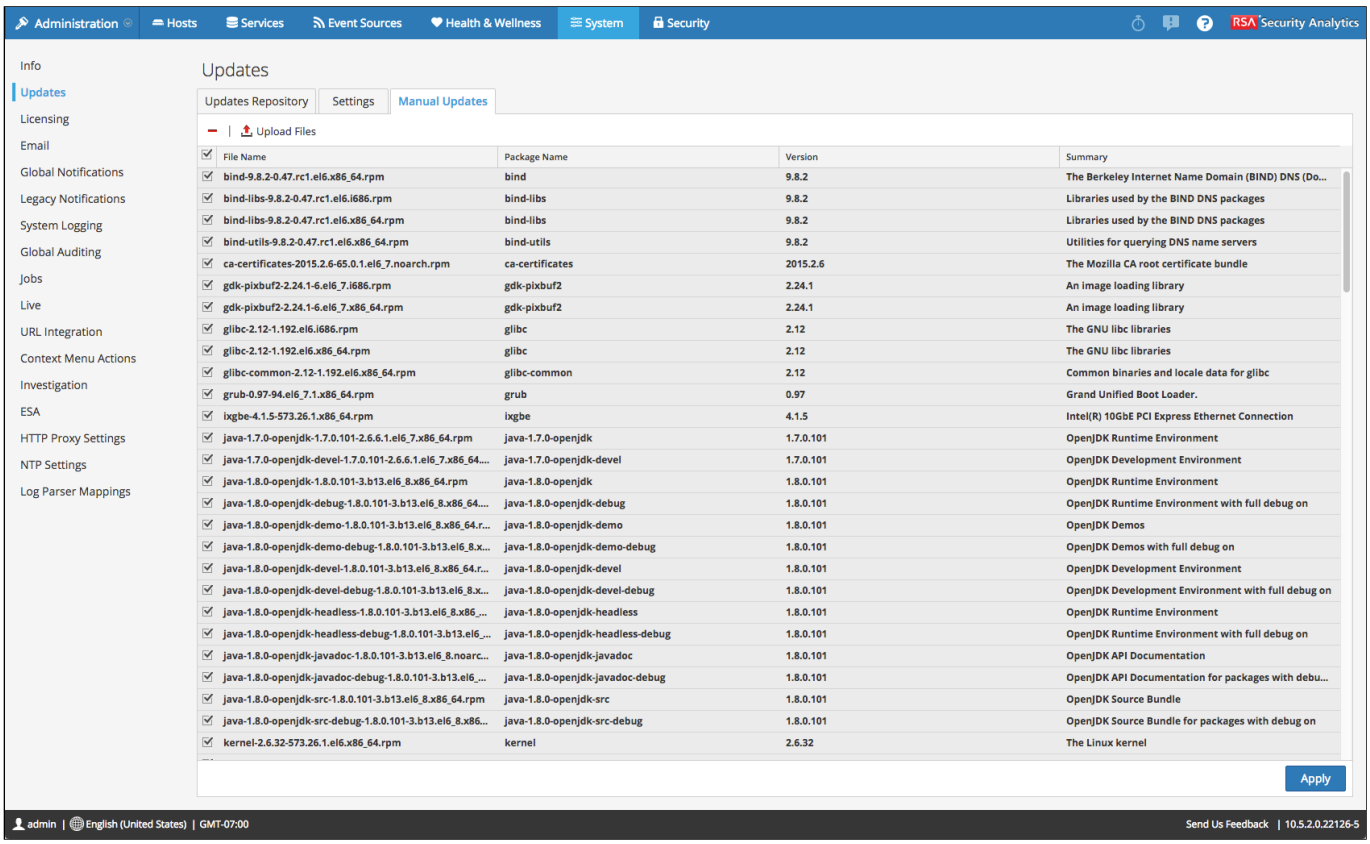
SA 10.5.2.0 introduced some 'guardrails' which help SA install the update packages in the correct order. They also help prevent bad things from happening if something isn't right prior to the upgrade. Though by no means bulletproof, it is a marked improvement over prior releases. These guardrails are implemented in upgrade manifest files, which are uploaded manually through the UI.

Uploading Upgrade Packages to the Yum Repo

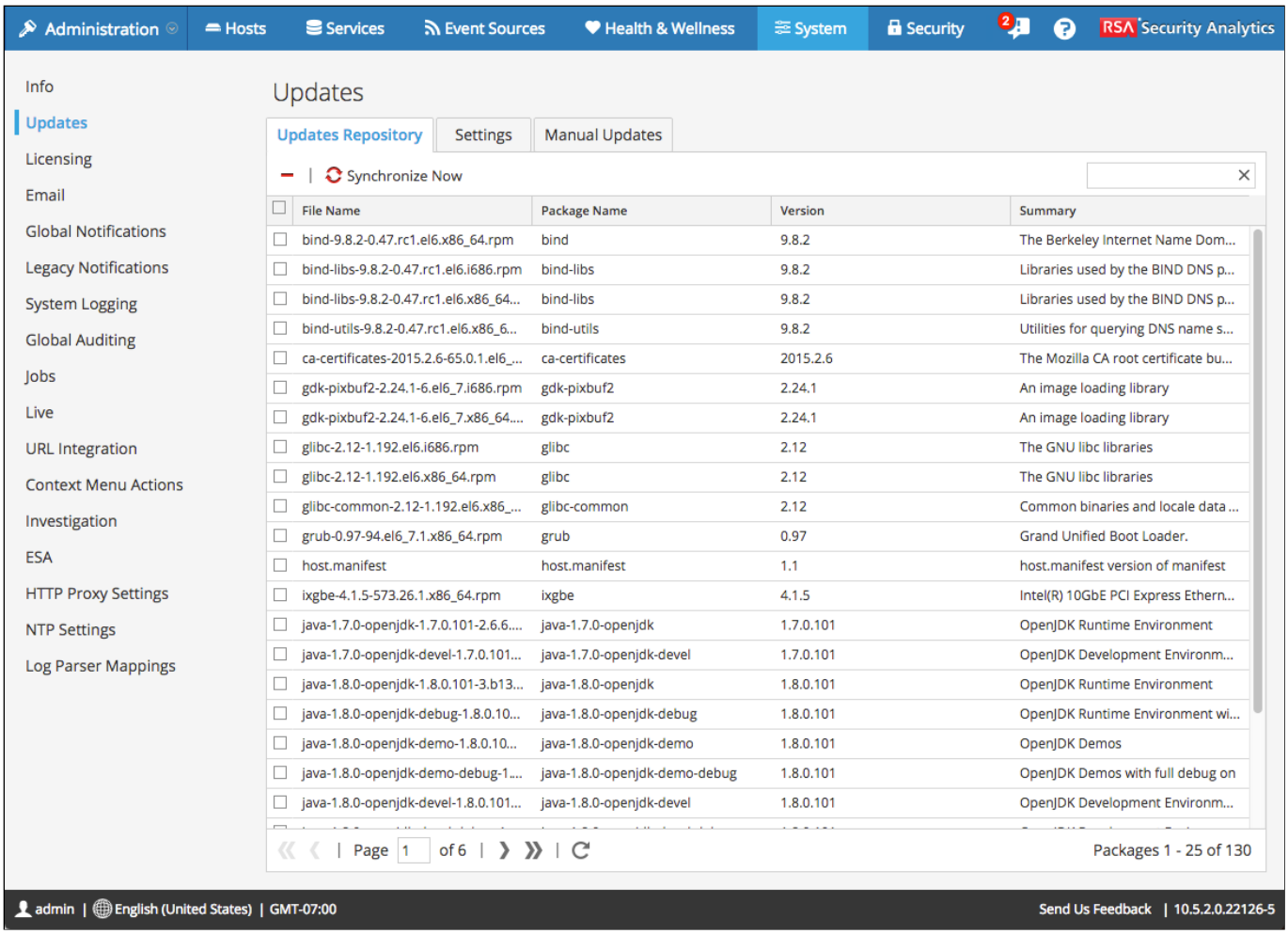
The best way to upload packages to the SA yum repo is to use the **Manual Updates** tab of **Administration->System->Updates**. Simply click on Upload Files, then +, and choose the update package zip file or files, and also the manifest files. Click **Upload** to upload the files.



You'll see update packages populated in Manual Updates. Select all packages, and then hit **Apply** to add them to the repository. This will move them into **/var/netwitness/srv/www/rsa/updates/SAUserUploaded**, and will regenerate the repo metadata.

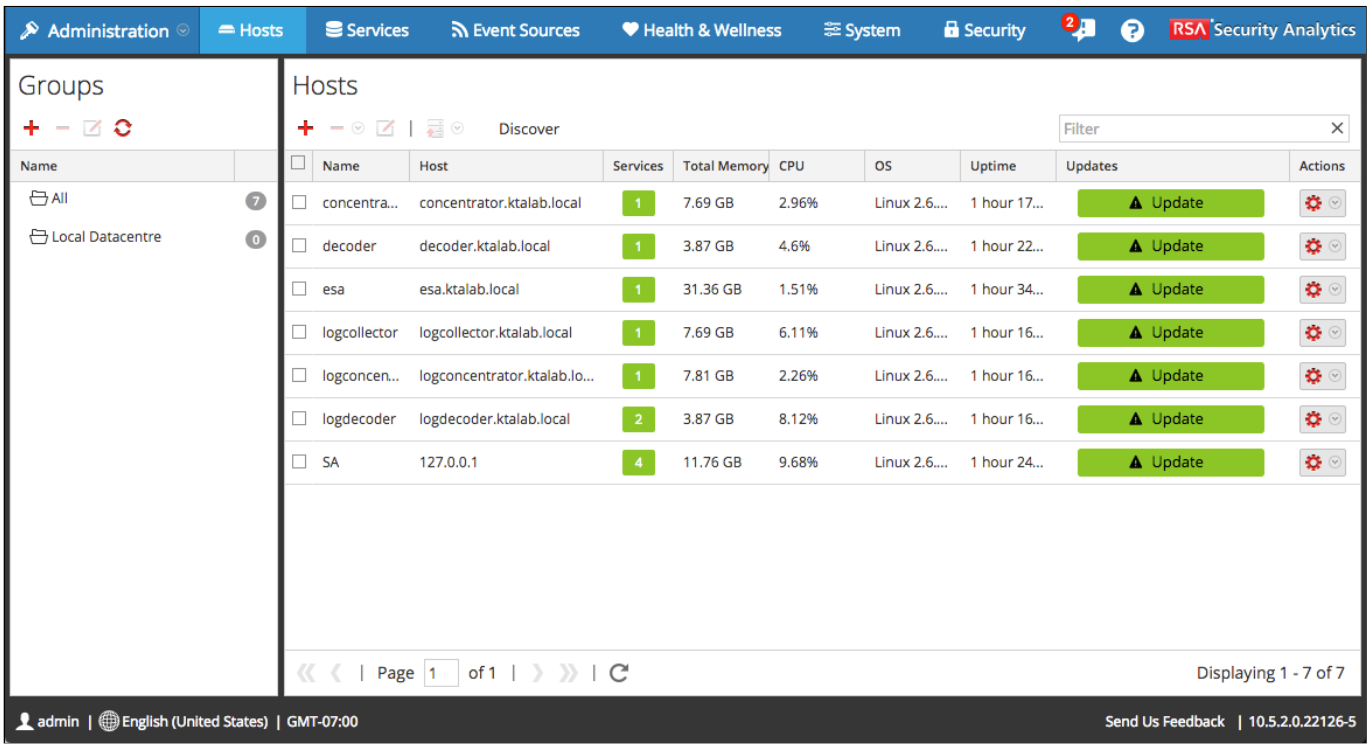


The files should now appear under the Updates Repository tab of **Administration->System.** In theory, you're now ready to trigger the update. That's **IF YOU'VE READ THE REST OF THE UPGRADE DOCUMENTATION**!



Triggering an Upgrade

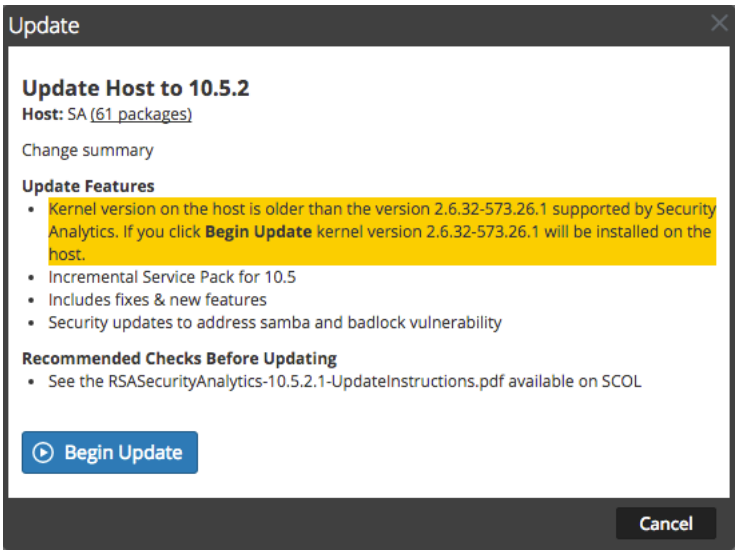
Upgrades are triggered on from **Administration->Hosts** in the UI. If the upgrade packages and the manifest files have been installed properly in the yum repo, you should see the word Update on a green background in the Updates column.



Click on the green box to start the process of updating of an SA host. Upgrade SA Hosts in this order:

1. SA UI Server
2. All other SA hosts

A dialog box will pop up, from where you you click the Begin Update button to kick off the upgrade. If an upgrade contains a kernel update, you'll see this text in yellow warning you of this.

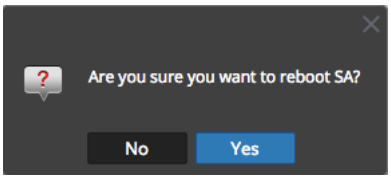


You should now see the text in the Updates column change to **Initiating Update**, and then a progress bar should appear saying Updating 39 of 61 packages. If something goes wrong with the update, the column's text will change to let you know. You may see the services column indicator turn amber or red during the installation. This is normal, and it happens because services are restarted as part of the upgrade.

One may also track the update progress by tailing **/var/log/messages** on the host that's being upgraded. You may see other messages related to the upgrade, for instance, for puppet.

```
Dec 13 00:33:21 sa yum[16707]: Updated: nss-util-3.21.0-0.3.el6_7.x86_64
Dec 13 00:33:21 sa yum[16707]: Updated: nss-sysinit-3.21.0-0.3.el6_7.x86_64
Dec 13 00:33:21 sa yum[16707]: Updated: nss-3.21.0-0.3.el6_7.x86_64
Dec 13 00:33:22 sa yum[16707]: Updated: nss-tools-3.21.0-0.3.el6_7.x86_64
Dec 13 00:33:39 sa yum[16879]: Updated: 1:java-1.7.0-openjdk-1.7.0.101-2.6.6.1.el6_7.x86_64
Dec 13 00:33:48 sa yum[17223]: Updated: nwsdk-10.5.2.1.5345-5.el6.x86_64
Dec 13 00:33:55 sa yum[17248]: Updated: nwappliance-10.5.2.1.5345-5.el6.x86_64
Dec 13 00:34:17 sa collectd[2105]: ESMAggregator: Dispatched 21 stats (in 1 messages) and 21 rrd stats in 0 seconds
Dec 13 00:34:20 sa yum[17416]: Updated: rsa-sms-server-10.5.2.1.3083-5.x86_64
Dec 13 00:34:31 sa yum[18234]: Updated: ntpdate-4.2.6p5-10.el6.centos.1.x86_64
Dec 13 00:34:33 sa yum[18234]: Updated: ntp-4.2.6p5-10.el6.centos.1.x86_64
Dec 13 00:34:34 sa ntpd[1685]: ntpd exiting on signal 15
Dec 13 00:34:34 sa ntpd[18411]: ntpd 4.2.6p5@1.2349-o Tue May 31 10:09:21 UTC 2016 (1)
Dec 13 00:34:34 sa ntpd[18412]: proto: precision = 0.897 usec
Dec 13 00:34:34 sa ntpd[18412]: 0.0.0.0 c01d 0d kern kernel time sync enabled
Dec 13 00:34:34 sa ntpd[18412]: Listen and drop on 0 v4wildcard 0.0.0.0 UDP 123
Dec 13 00:34:34 sa ntpd[18412]: Listen and drop on 1 v6wildcard :: UDP 123
Dec 13 00:34:34 sa ntpd[18412]: Listen normally on 2 lo 127.0.0.1 UDP 123
Dec 13 00:34:34 sa ntpd[18412]: Listen normally on 3 eth0 10.0.0.3 UDP 123
Dec 13 00:34:34 sa ntpd[18412]: Listen normally on 4 lo ::1 UDP 123
Dec 13 00:34:34 sa ntpd[18412]: Listen normally on 5 eth0 fe80::2ec2:60ff:fe2f:556d UDP 123
Dec 13 00:34:34 sa ntpd[18412]: Listening on routing socket on fd #22 for interface updates
Dec 13 00:34:35 sa ntpd[18412]: 0.0.0.0 c016 06 restart
Dec 13 00:34:35 sa ntpd[18412]: 0.0.0.0 c012 02 freq_set kernel -0.057 PPM
Dec 13 00:34:35 sa ntpd[18412]: 0.0.0.0 c615 05 clock_sync
Dec 13 00:34:36 sa collectd[2105]: GenericJMXConfMBean: queryNames failed: java.rmi.ConnectException: Connection refused to host: 10.0.0.3; nested exception is: #
012#011java.net.ConnectException: Connection refused
Dec 13 00:34:42 sa yum[18433]: Updated: 32:bind-libs-9.8.2-0.47.rc1.el6.x86_64
Dec 13 00:34:42 sa yum[18433]: Updated: 32:bind-utils-9.8.2-0.47.rc1.el6.x86_64
Dec 13 00:34:50 sa yum[18607]: Updated: kexec-tools-2.0.0-286.el6_7.2.x86_64
Dec 13 00:34:58 sa yum[18824]: Updated: openssh-5.3p1-118.1.el6_8.x86_64
Dec 13 00:34:59 sa yum[18824]: Updated: openssh-server-5.3p1-118.1.el6_8.x86_64
Dec 13 00:34:59 sa yum[18824]: Updated: openssh-clients-5.3p1-118.1.el6_8.x86_64
Dec 13 00:35:10 sa yum[19082]: Updated: rsa-collectd-sms-10.5.2.1.3083-5.el6.x86_64
Dec 13 00:35:17 sa collectd[2105]: ESMAggregator: Dispatched 21 stats (in 1 messages) and 21 rrd stats in 0 seconds
Dec 13 00:35:21 sa yum[19271]: Updated: libxml2-2.7.6-21.el6_8.1.x86_64
Dec 13 00:35:21 sa yum[19271]: Updated: libxml2-python-2.7.6-21.el6_8.1.x86_64
Dec 13 00:35:37 sa yum[19475]: Updated: 1:java-1.8.0-openjdk-headless-1.8.0.101-3.b13.el6_8.x86_64
Dec 13 00:35:38 sa yum[19475]: Updated: 1:java-1.8.0-openjdk-1.8.0.101-3.b13.el6_8.x86_64
Dec 13 00:35:44 sa yum[19730]: Updated: libssh2-1.4.2-2.el6_7.1.x86_64
Dec 13 00:35:55 sa yum[20015]: Updated: libtdb-1.3.8-1.el6_7.x86_64
Dec 13 00:36:17 sa collectd[2105]: ESMAggregator: Dispatched 21 stats (in 1 messages) and 21 rrd stats in 0 seconds
Dec 13 00:36:18 sa yum[20267]: Updated: kernel-firmware-2.6.32-573.26.1.el6.noarch
Dec 13 00:36:28 sa yum[20599]: Updated: openssl-1.0.1e-48.el6_8.1.x86_64
Dec 13 00:36:28 sa yum[20599]: Updated: openssl-1.0.1e-48.el6_8.1.i686
Dec 13 00:36:47 sa yum[20676]: Installed: kernel-2.6.32-573.26.1.el6.x86_64
Waiting for data... (interrupt to abort)
```

If the update is successful, the Updates column will turn amber, and will display the text **Reboot Required**. Reboot Required. Now click on the Reboot Required text, and click Yes in the dialog that opens.



The Updates column now changes to Rebooting. If you're updating the SA UI server, you may have to wait a while for the server to reboot and the UI to come up again. Just wait and click your browser's refresh button a lot!

Top Tip: If you want to watch the startup messages of your UI service, AKA ***jettysrv***, you can tail the UI log file at **/var/lib/netwitness/uax/logs/sa.log. /var/log/messages** may also display interesting text. After a reboot, puppet will run and automatically perform whatever new tasks are necessary as part of the upgrade.

```
2016-12-13 01:14:19,933 [Live Deprecation Handler Task 423707561] INFO  com.rsa.smc.sa.live.job.task.DeprecationHandlerTask - Total app rules to remove:0
2016-12-13 01:14:20,135 [Live Deprecation Handler Task 423707561] ERROR com.rsa.smc.sa.nextgen.service.DefaultFeedService - Failed to process message delete for /
decoder/parsers  com.rsa.netwitness.carlos.transport.TransportException: Resource nwwikileaksdomains.feed does not exists
2016-12-13 01:14:20,171 [Live Deprecation Handler Task 423707561] WARN  com.rsa.smc.sa.live.job.task.DeprecationHandlerTask - Error removing Live Deprecated Resou
rce from service Decoder, error: Failed to delete nwwikileaksdomains.feed
2016-12-13 01:14:20,380 [Endpoint Availability Resolver 1688395682] ERROR com.rsa.smc.sa.core.appliance.service.DefaultApplianceService - Failed to load appliance
 with id null
2016-12-13 01:14:22,171 [Endpoint Service Resolver 652501809] WARN  com.rsa.netwitness.carlos.clients.nextgen.nw.NwClientPipeBase - escalateduser@logdecoder.ktala
b.local:56002/sys/stats received error: Message get was not recognized by /sys/stats
2016-12-13 01:14:23,010 [Provision Daemon 707272654] INFO  com.rsa.smc.sa.core.appliance.service.DefaultApplianceService - Updating host logcollector.ktalab.local
 with 1 endpoints
2016-12-13 01:14:23,123 [Endpoint Availability Resolver 1688395682] INFO  com.rsa.smc.sa.esa.service.DefaultEsaCentralMetaService - Imported meta types from end p
oint: 57fab33ce4b08196ad7a3885/ESA
2016-12-13 01:14:23,799 [Endpoint Service Resolver 652501809] WARN  com.rsa.netwitness.carlos.clients.nextgen.nw.NwClientPipeBase - escalateduser@127.0.0.1:56025/
sys/stats received error: Message get was not recognized by /sys/stats
2016-12-13 01:14:24,252 [Endpoint Service Resolver 652501809] WARN  com.rsa.netwitness.carlos.clients.nextgen.nw.NwClientPipeBase - escalateduser@127.0.0.1:56003/
sys/stats received error: Message get was not recognized by /sys/stats
2016-12-13 01:14:24,442 [System Monitoring Service ApplianceConfig 1401190672] WARN  com.rsa.smc.sa.admin.service.DefaultSystemMonitoringService - Unknown system
monitoring component type getIPDBEndpointLabel
2016-12-13 01:14:24,622 [Endpoint Availability Resolver 1688395682] INFO  com.espertech.esper.core.service.EPServiceProviderImpl - Initializing engine URI 'testCE
PEngine' version 4.11.0
2016-12-13 01:14:24,803 [Endpoint Service Resolver 652501809] WARN  com.rsa.netwitness.carlos.clients.nextgen.nw.NwClientPipeBase - escalateduser@decoder.ktalab.l
ocal:56004/sys/stats received error: Message get was not recognized by /sys/stats
2016-12-13 01:14:26,230 [Endpoint Service Resolver 652501809] WARN  com.rsa.netwitness.carlos.clients.nextgen.nw.NwClientPipeBase - escalateduser@logconcentrator.
ktalab.local:56005/sys/stats received error: Message get was not recognized by /sys/stats
2016-12-13 01:14:26,666 [Endpoint Service Resolver 652501809] WARN  com.rsa.netwitness.carlos.clients.nextgen.nw.NwClientPipeBase - escalateduser@logcollector.kta
lab.local:56001/sys/stats received error: Message get was not recognized by /sys/stats
2016-12-13 01:14:27,583 [Endpoint Service Resolver 652501809] WARN  com.rsa.netwitness.carlos.clients.nextgen.nw.NwClientPipeBase - escalateduser@concentrator.kta
lab.local:56005/sys/stats received error: Message get was not recognized by /sys/stats
2016-12-13 01:14:27,714 [Endpoint Service Resolver 652501809] WARN  com.rsa.netwitness.carlos.clients.nextgen.nw.NwClientPipeBase - escalateduser@logdecoder.ktala
b.local:56001/sys/stats received error: Message get was not recognized by /sys/stats
2016-12-13 01:14:28,945 [Provision Daemon 707272654] INFO  com.rsa.smc.sa.core.appliance.service.DefaultApplianceService - Updating host logdecoder.ktalab.local w
ith 2 endpoints
2016-12-13 01:14:29,316 [Endpoint Availability Resolver 1688395682] INFO  com.rsa.smc.sa.esa.service.DefaultEsaCentralMetaService - Imported meta types from end p
oint: 57fab33ce4b08196ad7a3885/ESA
2016-12-13 01:14:33,103 [Provision Daemon 707272654] INFO  com.rsa.smc.sa.core.appliance.service.DefaultApplianceService - Updating host logconcentrator.ktalab.lo
cal with 1 endpoints
2016-12-13 01:14:36,725 [Provision Daemon 707272654] INFO  com.rsa.smc.sa.core.appliance.service.DefaultApplianceService - Updating host concentrator.ktalab.local
 with 1 endpoints
2016-12-13 01:14:39,596 [Provision Daemon 707272654] INFO  com.rsa.smc.sa.core.appliance.service.DefaultApplianceService - Updating host esa.ktalab.local with 1 e
ndpoints
2016-12-13 01:14:42,365 [Provision Daemon 707272654] INFO  com.rsa.smc.sa.core.appliance.service.DefaultApplianceService - Updating host decoder.ktalab.local with
 1 endpoints
2016-12-13 01:14:45,392 [main] INFO  com.rsa.smc.sa.esa.service.DefaultOOTBContentService - Loading ESA Starter Pack rules
2016-12-13 01:14:45,395 [main] INFO  com.rsa.smc.sa.esa.service.DefaultOOTBContentService - ESA Starter Pack rules previously loaded: 4
2016-12-13 01:14:45,396 [main] INFO  com.rsa.smc.sa.esa.service.DefaultOOTBContentService - ESA Starter Pack rules loaded: 0
Waiting for data... (interrupt to abort)
```

```
Dec 13 00:53:52 sa puppet-agent[2505]: (/Stage[main]/Rsa-sms-server/Service[rsa-sms]) Dependency Java_ks[puppetca] has failures: true
Dec 13 00:53:52 sa puppet-agent[2505]: (/Stage[main]/Rsa-sms-server/Service[rsa-sms]) Skipping because of failed dependencies
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Rsa-sms-server/Service[rsa-sms]) Triggered 'refresh' from 1 events
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jetty-logging.xml]) Dependency Java_ks[carlos] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jetty-logging.xml]) Dependency Java_ks[puppetca] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jetty-logging.xml]) Skipping because of failed dependencies
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jmx-SAServer.conf]) Dependency Java_ks[carlos] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jmx-SAServer.conf]) Dependency Java_ks[puppetca] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jmx-SAServer.conf]) Skipping because of failed dependencies
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jetty-started.xml]) Dependency Java_ks[carlos] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jetty-started.xml]) Dependency Java_ks[puppetca] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jetty-started.xml]) Skipping because of failed dependencies
Dec 13 00:53:58 sa puppet-agent[2505]: (/Firewall[2 Yum Port]) Dependency Java_ks[carlos] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Firewall[2 Yum Port]) Dependency Java_ks[puppetca] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Firewall[2 Yum Port]) Skipping because of failed dependencies
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jetty-ssl.xml]) Dependency Java_ks[carlos] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jetty-ssl.xml]) Dependency Java_ks[puppetca] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/File[jetty-ssl.xml]) Skipping because of failed dependencies
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/Service[jettysrv]) Dependency Java_ks[carlos] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/Service[jettysrv]) Dependency Java_ks[puppetca] has failures: true
Dec 13 00:53:58 sa puppet-agent[2505]: (/Stage[main]/Saserver/Service[jettysrv]) Skipping because of failed dependencies
Dec 13 00:53:59 sa init: jettysrv main process (2740) terminated with status 143
Dec 13 00:54:03 sa puppet-agent[2505]: (/Stage[main]/Saserver/Service[jettysrv]) Triggered 'refresh' from 1 events
Dec 13 00:54:03 sa puppet-agent[2505]: (/Stage[main]/Reporting-engine/Service[rsasoc_re]) Dependency Java_ks[carlos] has failures: true
Dec 13 00:54:03 sa puppet-agent[2505]: (/Stage[main]/Reporting-engine/Service[rsasoc_re]) Dependency Java_ks[puppetca] has failures: true
Dec 13 00:54:03 sa puppet-agent[2505]: (/Stage[main]/Reporting-engine/Service[rsasoc_re]) Skipping because of failed dependencies
Dec 13 00:54:05 sa init: rsasoc_re main process (910) killed by TERM signal
Dec 13 00:54:05 sa puppet-agent[2505]: (/Stage[main]/Reporting-engine/Service[rsasoc_re]) Triggered 'refresh' from 1 events
Dec 13 00:54:05 sa puppet-agent[2505]: (/Firewall[1 SA Port]) Dependency Java_ks[carlos] has failures: true
Dec 13 00:54:05 sa puppet-agent[2505]: (/Firewall[1 SA Port]) Dependency Java_ks[puppetca] has failures: true
Dec 13 00:54:05 sa puppet-agent[2505]: (/Firewall[1 SA Port]) Skipping because of failed dependencies
Dec 13 00:54:15 sa puppet-agent[2505]: (/Stage[main]/Rsa-sms-server/File[MessageBusReadModule.conf]) Dependency Java_ks[carlos] has failures: true
Dec 13 00:54:15 sa puppet-agent[2505]: (/Stage[main]/Rsa-sms-server/File[MessageBusReadModule.conf]) Dependency Java_ks[puppetca] has failures: true
Dec 13 00:54:15 sa puppet-agent[2505]: (/Stage[main]/Rsa-sms-server/File[MessageBusReadModule.conf]) Skipping because of failed dependencies
Dec 13 00:54:25 sa puppet-agent[2505]: (/Stage[main]/Mcollective/File[/etc/mcollective/facts.yaml]/content) content changed '{md5}e227c920268de730d63111bb45b54ce8
' to '{md5}7ae5512266c32d7b36cb725119d8a2cf'
Dec 13 00:54:25 sa puppet-agent[2505]: (/Stage[main]/Collectd/Service[collectd]) Dependency Java_ks[carlos] has failures: true
Dec 13 00:54:25 sa puppet-agent[2505]: (/Stage[main]/Collectd/Service[collectd]) Dependency Java_ks[puppetca] has failures: true
Dec 13 00:54:25 sa puppet-agent[2505]: (/Stage[main]/Collectd/Service[collectd]) Skipping because of failed dependencies
Dec 13 00:54:26 sa puppet-agent[2505]: Finished catalog run in 211.01 seconds
Dec 13 00:54:48 sa collectd[2110]: ESMAggregator: Dispatched 21 stats (in 1 messages) and 21 rrd stats in 0 seconds
Dec 13 00:55:48 sa collectd[2110]: ESMAggregator: Dispatched 21 stats (in 1 messages) and 21 rrd stats in 0 seconds
Dec 13 00:56:16 sa puppet-master[1757]: (Scope(Class[Base])) puppetmaster.local
Dec 13 00:56:17 sa puppet-master[1757]: Compiled catalog for f11f6740-7ad4-45d3-baa0-18516dea1b16 in environment production in 2.47 seconds
Dec 13 00:56:29 sa puppet-master[1757]: (Scope(Class[Base])) puppetmaster.local
Dec 13 00:56:30 sa puppet-master[1757]: Compiled catalog for 4fe924bb-8ed9-4f65-a731-8767f0a37992 in environment production in 4.28 seconds
Waiting for data... (interrupt to abort)
```

Note: Whilst performing our update from 10.5.2.0 to 10.5.2.1, we see many errors in /var/log/messages regarding keytool:

```
Dec 13 00:52:15 sa puppet-agent[2505]: (/Stage[main]/Carlos/Java_ks[puppetca]/ensure) change from absent to latest failed: Execution
of 'keytool -importcert -noprompt -alias puppetca -file /var/lib/puppet/ssl//certs/ca.pem -keystore
/etc/alternatives/jre/lib/security/cacerts -trustcacerts' returned 1:
```

To fix this: we run "**alternatives --config java**", and enter '**1**', and then "**service puppet restart**":

```
[root@sa ~]# alternatives --config java

There is 1 program that provides 'java'.

  Selection    Command
-----------------------------------------------
*+ 1           /usr/lib/jvm/jre-1.8.0-openjdk.x86_64/bin/java

Enter to keep the current selection[+], or type selection number: 1

[root@sa ~]# service puppet restart
Stopping puppet agent:                                     [  OK  ]
Starting puppet agent:                                     [  OK  ]
```

Kernel Updates

Some SA updates contain updates to the Linux kernel. The UI will notify you when a kernel update is part of an upgrade, as it will require a system reboot to take effect.

Top Tip: Very frequently during upgrades, for an unknown reason, the kernel upgrade will install but will not be activated in Grub. You must therefore edit **/boot/grub/grub.conf** to change the '**default**' entry to match the updated kernel version. ***Always check this before rebooting your SA host during the install!*** In this screenshot, grub is pointed to the old kernel version:

```
password --md5 $1$K7uqK$bUbRkTIRK5zem3tYXF7ai0
default 1
timeout 5
title CentOS (2.6.32-573.12.1.el6.x86_64)
root (hd0,0)
kernel /vmlinuz-2.6.32-573.12.1.el6.x86_64 root=/dev/mapper/VolGroup00-root ro nodmraid quiet
 crashkernel=auto SYSFONT=latarcyrheb-sun16 LANG=en_US.UTF-8 KEYTABLE=us
initrd /initramfs-2.6.32-573.12.1.el6.x86_64.img
title Boot
root (hd0,0)
kernel /vmlinuz-2.6.32-504.1.3.el6.x86_64 root=/dev/VolGroup00/root ro nodmraid quiet
initrd /initramfs-2.6.32-504.1.3.el6.x86_64.img
~
~
~
~
~
~
~
```

...so change "**default 1**" to "**default 0**", and the next time your system boots, it will boot the first kernel entry in grub.conf.

```
password --md5 $1$K7uqK$bUbRkTIRK5zem3tYXF7ai0
default 0
timeout 5
title CentOS (2.6.32-573.12.1.el6.x86_64)
root (hd0,0)
kernel /vmlinuz-2.6.32-573.12.1.el6.x86_64 root=/dev/mapper/VolGroup00-root ro nodmraid quiet
 crashkernel=auto SYSFONT=latarcyrheb-sun16 LANG=en_US.UTF-8 KEYTABLE=us
initrd /initramfs-2.6.32-573.12.1.el6.x86_64.img
title Boot
root (hd0,0)
kernel /vmlinuz-2.6.32-504.1.3.el6.x86_64 root=/dev/VolGroup00/root ro nodmraid quiet
initrd /initramfs-2.6.32-504.1.3.el6.x86_64.img
~
~
~
~
~
~
~
-- INSERT --
```

The Danger of Rebooting

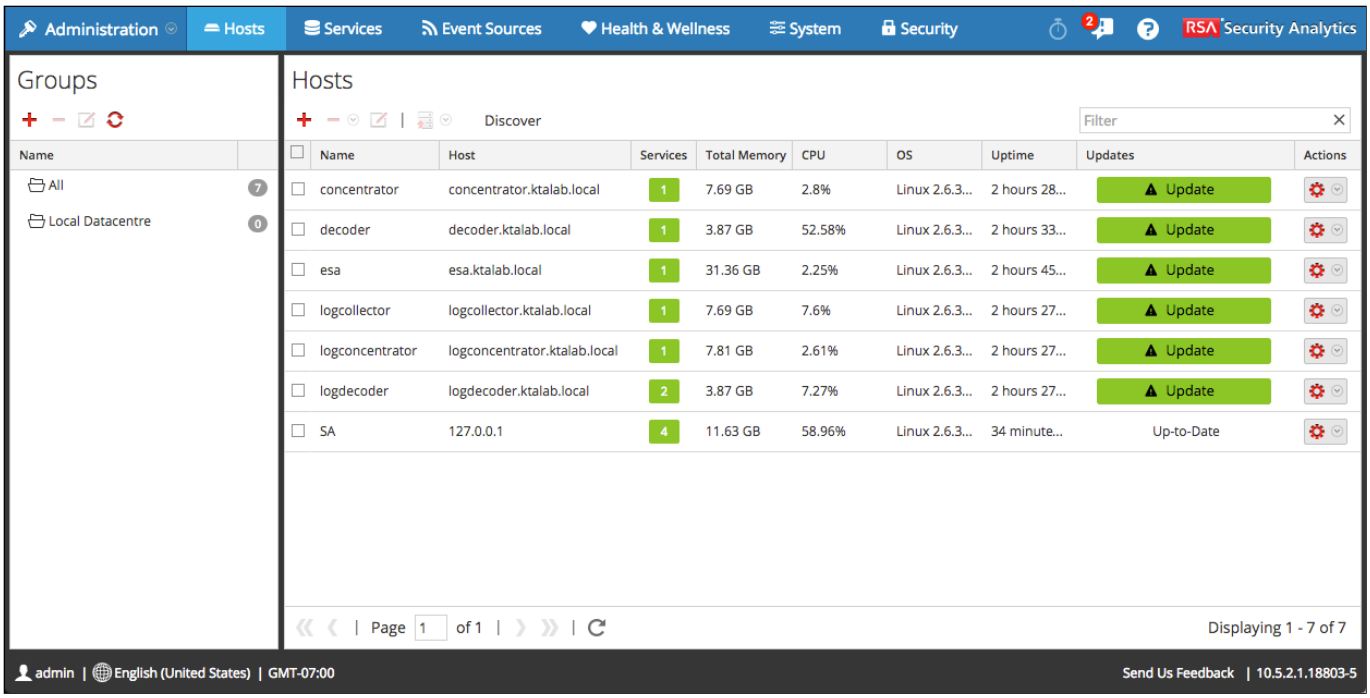
Rebooting an SA host, especially an appliance, is an inherently risky operation. Here are some of the things which can go wrong during a reboot:

- Host kernel panics whilst shutting down. Only a hard system reset can resolve this
- Host fails RAM test during POST
- RAID controller encounters an error initialising a virtual drive
- A problem is encountered mounting a filesystem during boot, which causes boot to halt, and drops to maintenance mode

What's the lesson to be learned here? **Make doubly-sure that you have console access to the server before rebooting**! I strongly recommend the Dell iDRAC, which is built into every SA appliance that RSA sells. You may have your own KVM solution - this is okay, as well, as long as it has the ability to cycle system power. If you have VM's, you should be able to get console access through your Virtualisation host OS (such as Vmware ESXi). It's also okay to have physical access to the box in your data centre. Just so long as you can get to the local console.

Update the Other SA Hosts

Now it's time to update the remainder of your SA hosts. We do this in exactly the same way as we did for the SA host. Remember to check their Grub configs, as well!



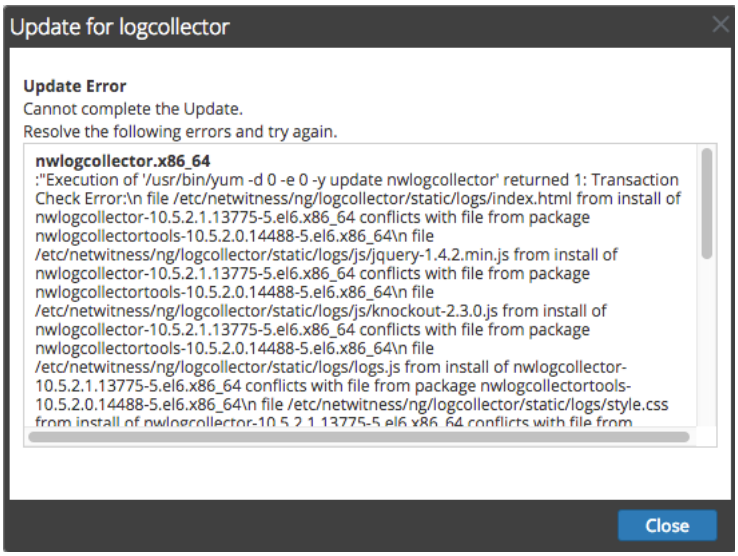
Upgrade Order: start at the top of the stack (Decoders and Log Decoders), then Archivers, then Concentrators, and then Brokers you may have, and finally, any Java services (ESA and Malware Analysis.

Upgrade Errors

Some hosts may throw errors whilst upgrading, which you can see in the Updates column.

By clicking on Update Error, you can see the specific error. For some errors (such as the below example), by simply re-launching the update, it will succeed the second time.

To re-launch an update, tick the box for the host with the error, then click on the button on the toolbar, and choose **Check Update**. Once the check completes, you can initiate the update again.



You'll likely get this error when upgrading your lab's Log Collector. I was able to find a solution quickly on the RSA Community site, so please ensure you're using this as a resource when you run into problems. To resolve it, use "**yum erase nwlogcollectortools**" from the command line, and re-run the update. You'll need to do the same for your Log Decoder.

I also ran into a problem on the SA head whereby puppet couldn't find the **keytool** command (shown in **/var/log/messages**). Again, the community site came to my aid. Let's see if you can work that one out on your own.

One last problem arose for me wherein RabbitMQ didn't start automatically after rebooting one of the hosts. It required running "**service rabbitmq-server restart**" and then re-running puppet with "**service puppet stop; puppet agent -t; service puppet start**"

See how much fun SA upgrades can be!

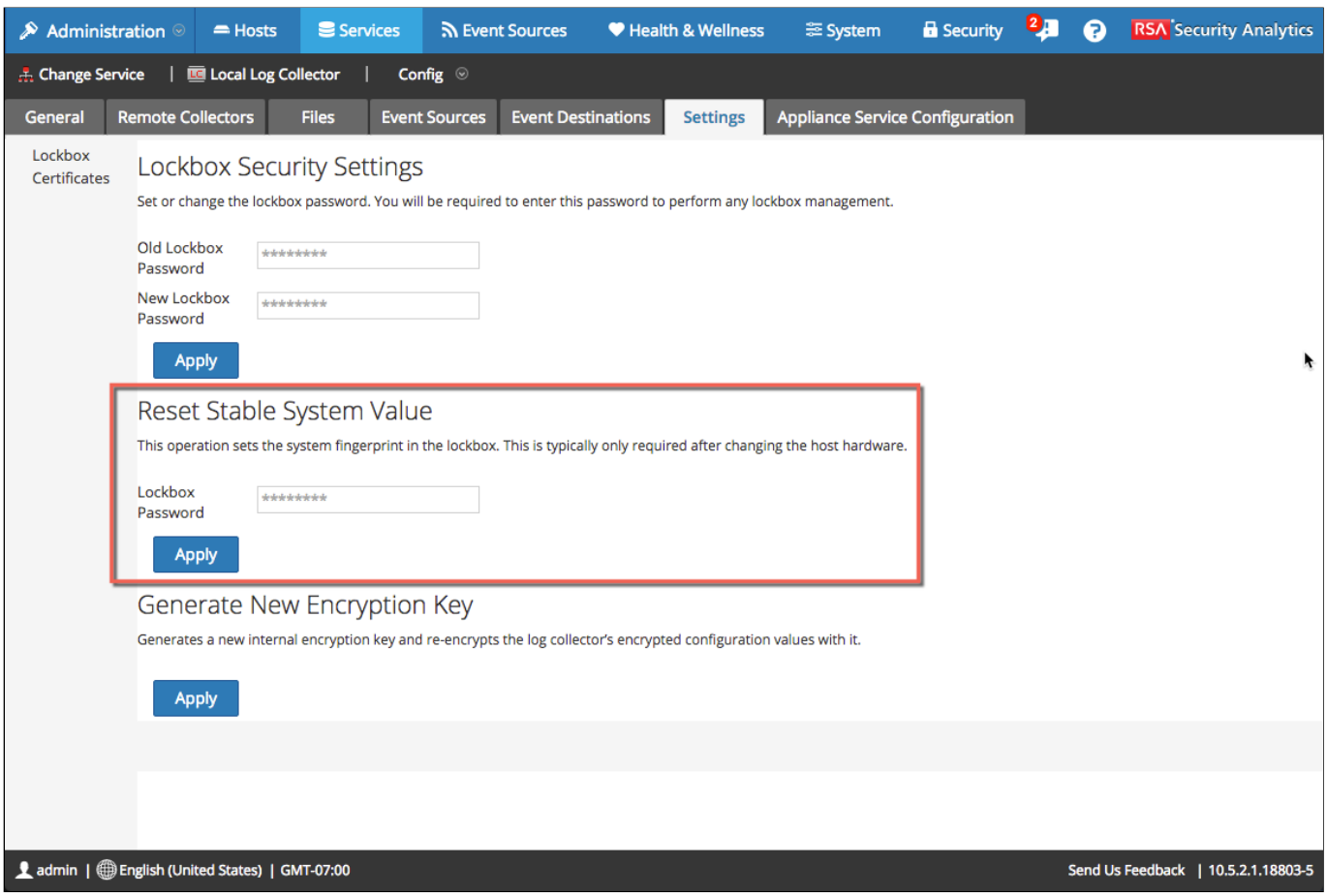
```
[root@logcollector ~]# yum erase nwlogcollectortools
Loaded plugins: fastestmirror
Setting up Remove Process
Resolving Dependencies
--> Running transaction check
---> Package nwlogcollectortools.x86_64 0:10.5.2.0.14488-5.el6 will be erased
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                    Arch                Version                  Repository                  Size
================================================================================
Removing:
 nwlogcollectortools        x86_64              10.5.2.0.14488-5.el6     @RSASoftware                70 M

Transaction Summary
================================================================================
Remove        1 Package(s)

Installed size: 70 M
Is this ok [y/N]: y
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Erasing    : nwlogcollectortools-10.5.2.0.14488-5.el6.x86_64                1/1
  Verifying  : nwlogcollectortools-10.5.2.0.14488-5.el6.x86_64                1/1

Removed:
  nwlogcollectortools.x86_64 0:10.5.2.0.14488-5.el6

Complete!
[root@logcollector ~]#
```

Top Tip: One common problem during the update process is that the puppet agent stops on the host and doesn't restart. You can check this with "**service puppet status**" from the command line. Start it again with "**puppet agent start**", though I probably didn't have to tell you that.

Another Top Tip: One can run the puppet agent directly from the command line by running "**puppet agent -t**".

Upgrading Log Collectors

After upgrading a Log Collector, it may be necessary to reset the Stable System Value of its lockbox in **Administration->Services-><LogCollector>->Config->Settings**. I hope you've vaulted your lockbox passwords!



System Firmware

For those running on RSA appliances, it's often a good idea, for performance and stability reasons, to update the system and RAID controller firmware. You can find this for download on https://community.rsa.com, or you can contact RSA Support. Be forewarned, this can sometimes go wrong, as well, which may require RAID firmware recovery.

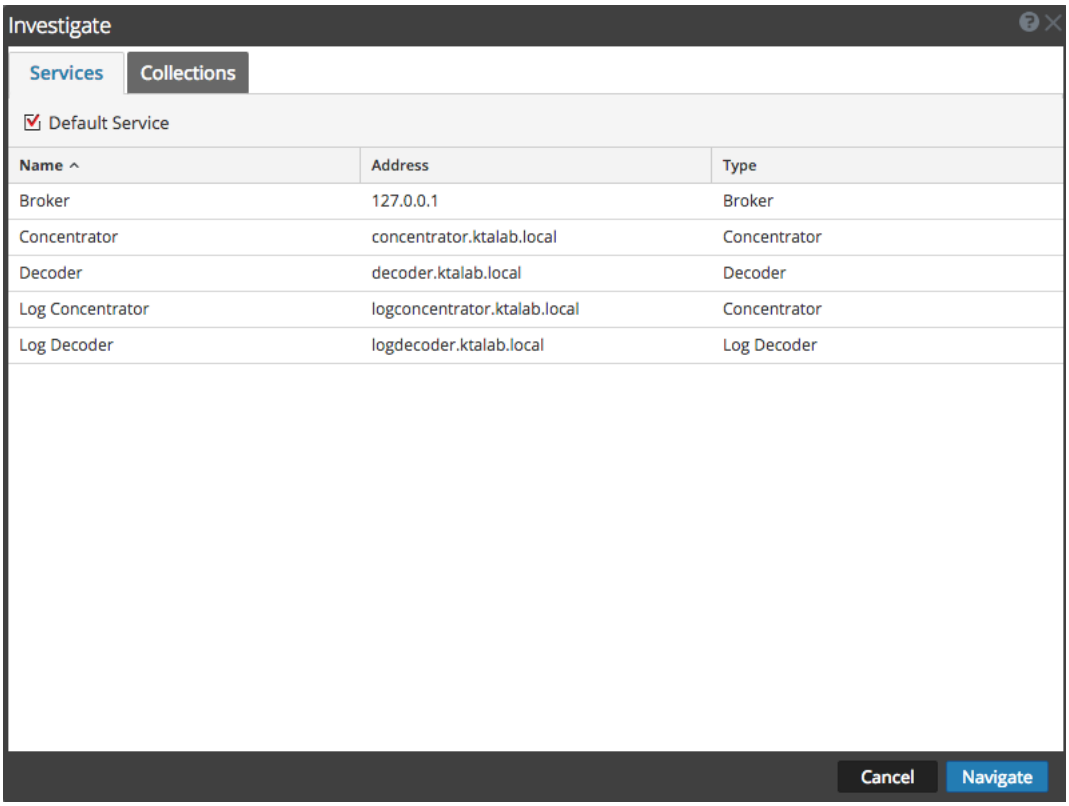
10. SA UI - Investigation

The Investigation UI module is the space where SA's meta-based model comes together. It is here that you can conduct investigations, viewing and drilling through generated meta, as well as by reconstructing captured packet sessions and viewing captured log data.

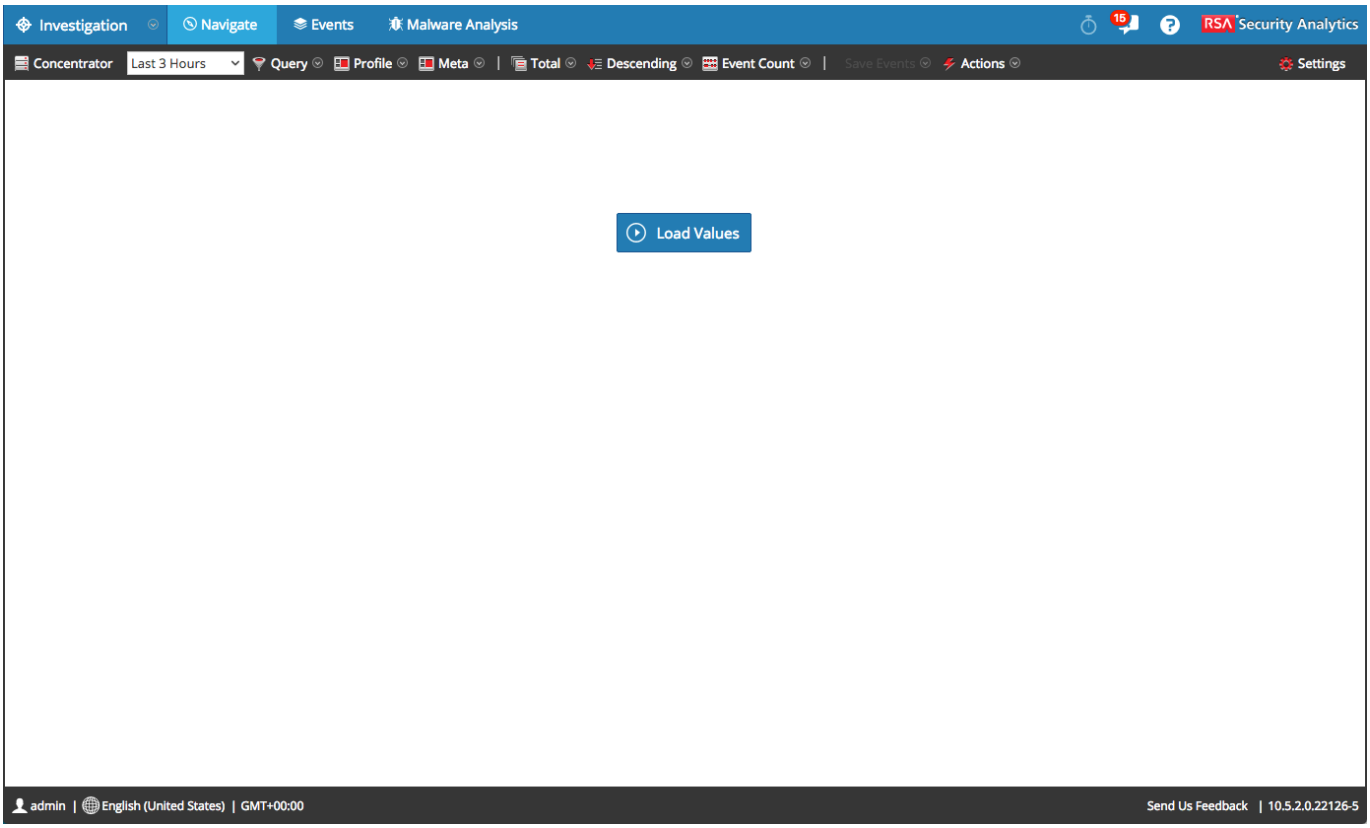
Navigate Tab

Investigation's Navigate tab allows you to view into the sea of SA's generated meta, drilling through and querying it. When you first come here, you'll be asked to select a service to connect to, typically either a Concentrator or a Broker. Which to choose is entirely dependent on your SA infrastructure and where you want to focus your investigation. For instance, if you have a Concentrator in San Francisco, and you know you only want to look at San Francisco meta and sessions, then you may want to connect to the San Francisco Concentrator. If you're unsure where to start your Investigation or you need to see meta across the entire organisation, the bottom (or top) level Broker would be the best place to start.

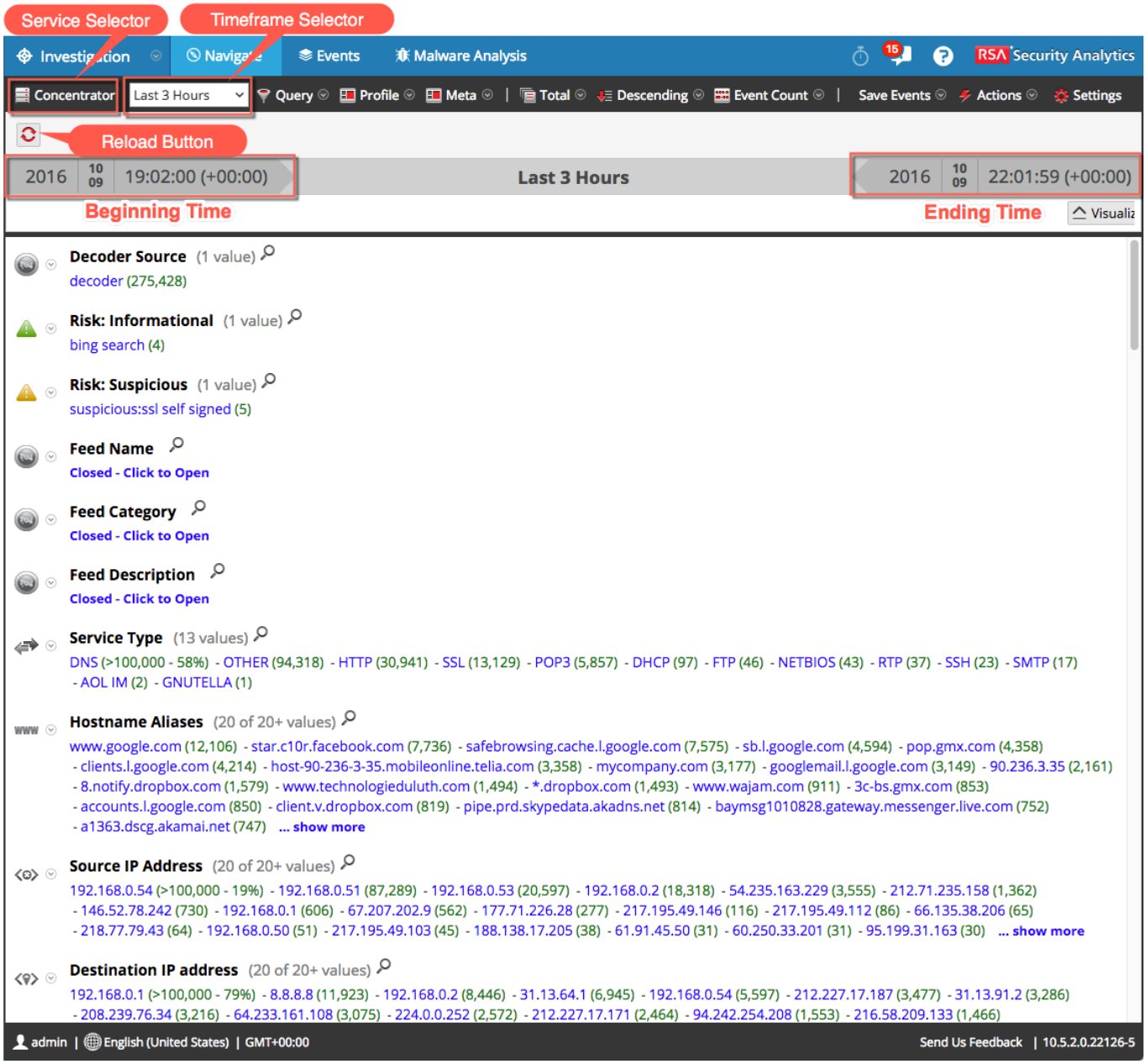
By default, SA will ask you which Core service you wish to connect to every time you start a new investigation. If you always wish to start your investigation with the same Core service, after clicking the service, click the **Default Service** button before clicking **Navigate**.



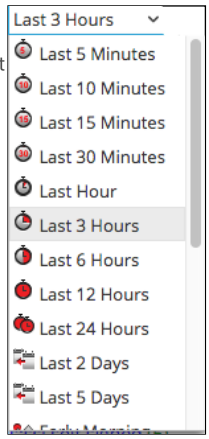
Once you've chosen a service to connect to, you will be taken to a nearly blank page with a Load Values button in the middle of the page. Click this to start loading meta keys and values. As will be noted later on, there's a setting you can change to no longer require the Analyst to hit the Load Values button.



You will see a graph near the top of the Investigation window. Click the Hide button for now, and we'll come back to that in a bit.



Timeframe Selector



The timeframe selector is one of the most important settings in the Investigation workflow. It controls the time window of data that is to be loaded into Investigation. As you can see from the screenshot, there are many options here, but you must keep the following things in mind:

1. This setting most impacts how much of the Concentrator's index gets loaded into memory. The bigger the timeframe, the more of the index that must be loaded into memory, and the slower query results will load. This also has the effect of increasing contention for Concentrator resources against other users and queries.

2. Because of this, it is best to keep your timeframe as narrow as possible, and then broaden it as needed

3. The worst of these time windows, performance-wise, is **All Data** (not pictured). This queries the entirety of the index.

4. You can use the **Custom** option to select a specific beginning and ending time to search on

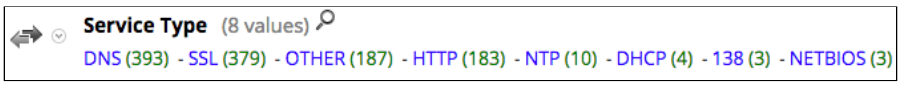
5. In the above screenshot, notice the beginning and ending times, shown below the toolbar. This tells you exactly what the beginning and ending times of meta for the meta that is currently displayed on the screen.

Notable Meta Keys

Decoder Source ('did' meta key) - The value stored under **Decoder Source** (or Decoder ID, hence '**did**') is the name of the Decoder which generated the session. When using a Broker or Concentrator which has visibility into multiple Decoders, you can drill into only those sessions which originated at a specific Decoder, without having to change the Investigation device to look at only a specific Concentrator.

Service Type ('service' meta key) - The **service** key tracks what type of traffic was captured in a given session, or its application-layer protocol. Its value corresponds to the most commonly-used port number for that protocol. For instance, HTTP sessions have a service meta value of **80**, SSL has **443**, SMTP has **25**, etc.

Yet when you look at the service meta key in Investigation, you don't see numbers; you see friendly names like HTTP, SMTP, DNS, etc. This is because an alias has been defined for the most common values.

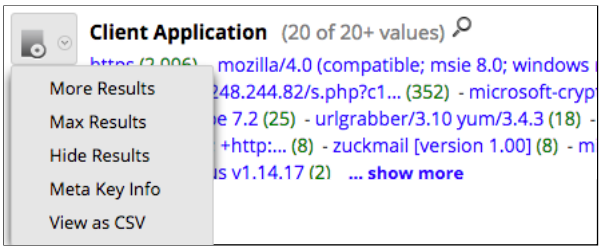


Keep in mind, though, that SA is port-agnostic, so even though the service number may look like a port, a session where '***service=80'*** may have **tcp.dstport** being something completely different.

The 'OTHER' Service Type - OTHER = unclassified sessions. When a Decoder cannot identify the protocol of a session, it sets the **service** meta key value to **0**, or **OTHER.**

Loading More Values

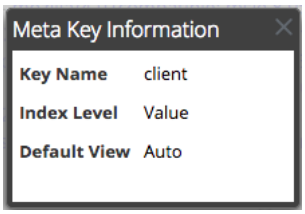
When there are more than twenty values under a given meta key, Investigation will show a **... show more** button. This prevents a single key from taking up too much screen real estate. Click this button to load twenty more values for the key.



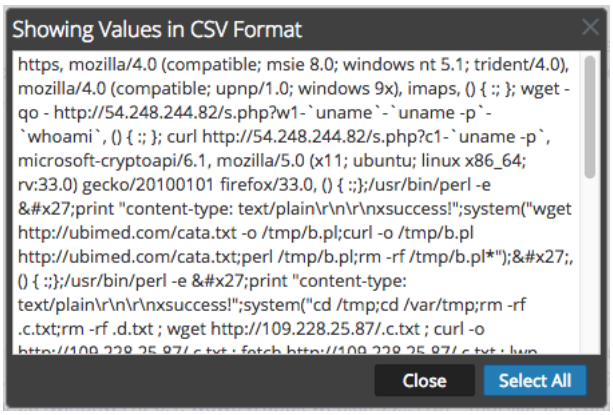
You may have noticed an icon next to each key shown in Investigation, with a dropdown. You can also use this to:

- **Load More Results** - Same as the **... show more** button.
- **Load Maximum Results** - Loads the configured maximum number of values. This is defined in the logged-in users Investigation settings. More on this to come.
- **Hide Results** - Collapses the selected key in the Navigate view.

- **Meta Key Info** - Shows the meta key's real name, index level, and default view setting.



- **View as CSV** - As the name implies, it shows all values for the key in CSV notation



Pause / Resume / Reload Buttons

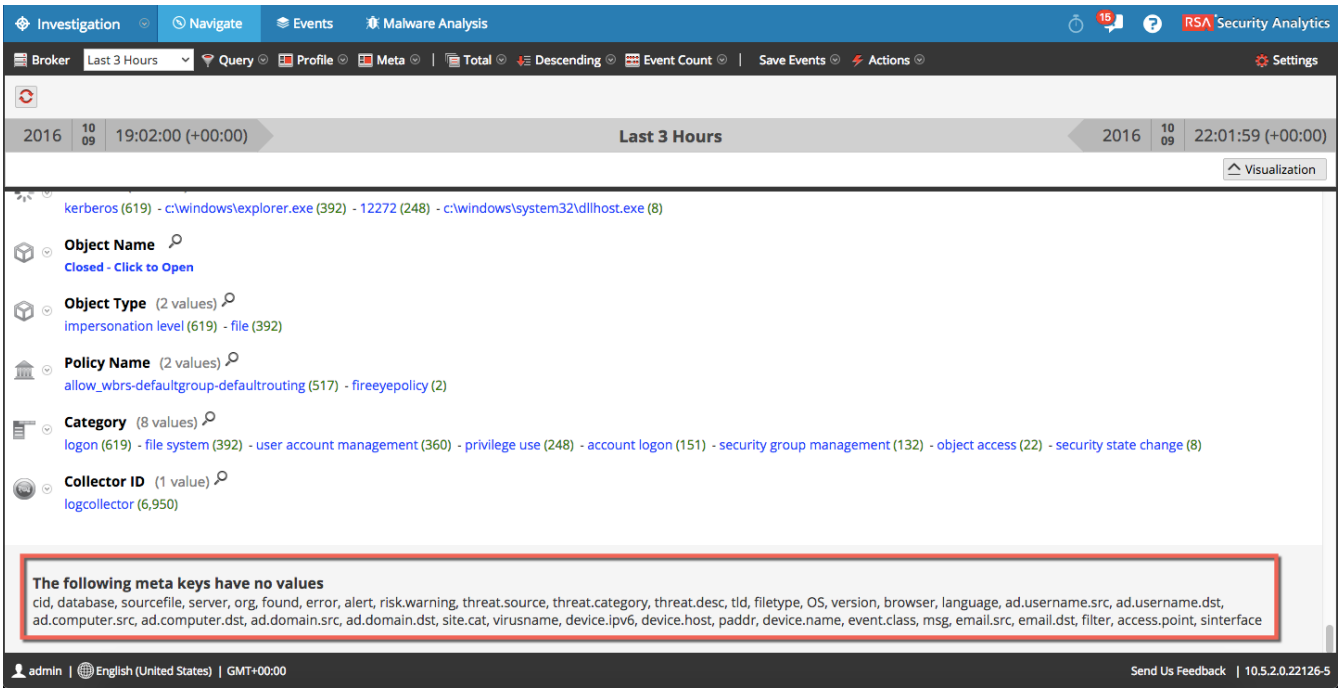
The reload button (see above illustration for its location on the screen) simply instructs Investigation to reload the view using the current time parameters.

The pause button pauses the loading of all meta in the current Investigation view. When this is pressed, the pause button changes to a resume button .

Top Tip: When using a timeframe selector of **Last *N* Minutes** or **Last *N* Hours**, the reload button will not actually move the beginning and ending time of the investigation forward to match the current time. Refresh the page using your web browser to do this.

Meta Keys With No Values

There will sometimes be meta keys that, for whatever reason, will display no meta. This could be cause a key is enabled in the index but not its associated parser(s), or that the Decoder isn't seeing the type of traffic that would cause that meta to be generated. When this occurs, keys that have no meta will display at the bottom of the Investigation screen.



Drilling

As you have already likely surmised, one of the most powerful features of SA is the ability to drill through meta. By clicking on a meta value (shown in blue), the Investigation page will be refreshed with only sessions that have the particular meta value that you clicked on. By drilling intelligently through the meta, one can find a proverbial needle in a haystack. Try this on your own and see what you find. The practice of drilling through this meta to find specific anomalies is called **hunting;** it's most commonly practiced by Analysts and Incident Responders.

Performing a drill on data really is issuing a new query on the data set with an additional WHERE statement. Each level of a drill becomes a new clause in the WHERE statement, whereby each drill clause gets logically AND'ed together. For instance, if you perform an initial drill with **service=80**, and then another with **alias.host ends 'bbc.co.uk'**, the SDK query against the Core service will look like:

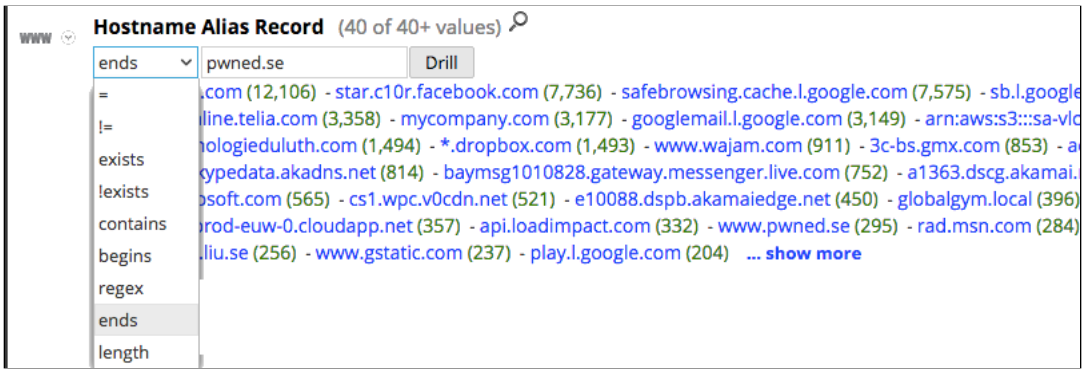
WHERE (service=80) && (alias.host ends 'bbc.co.uk')

Top Tip: The first view you see in Investigation with no drills applied is called the **Root Drill**.

Not all meta keys are populated for every session. For instance, not all sessions will have meta under the '**filename**' meta key. A simple way of drilling into only sessions which have populated the **filename** meta key, simply click on the '**Filename**' key name. The breadcrumb trail (which will soon be described) will then show "**filename exists**" as one of the applied Investigation queries.

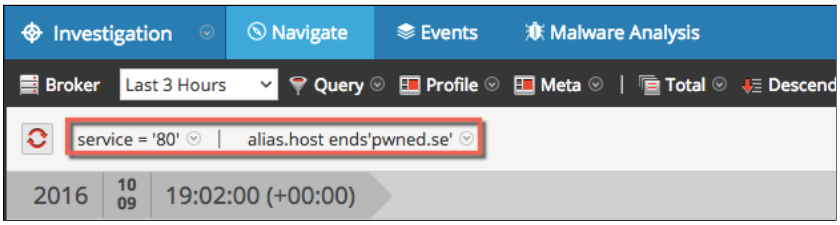
The specific methodology used to find bad things in this sea of data is beyond the scope of this course on Administration, but I would recommend taking an incident responder or analysis course if you wish to become a better hunter.

Rather than drilling via clicking on values, which uses an equals (=) conditional operator, it can also be useful to perform advanced drills against a meta key. For instance, I've noticed a value under **Hostname Alias Record** (**alias.host**) called www.pwned.se. This looks terribly suspicious to me, so I wish to drill not just on ***www***.pwned.se, but any domain under pwned.se. To do this, I can click on the magnifying glass button next to key name, and I will be presented with a list of all conditional operators that I can perform against the selected key.

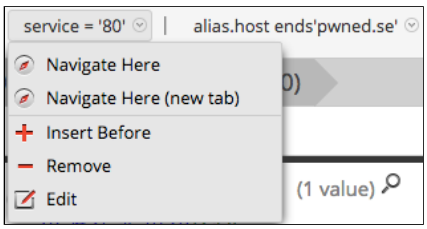


Breadcrumb Trail

As one drills through the meta over the course of an investigation, one can accumulate several different levels of queries. Each of these drill or query clauses is called a Breadcrumb, and is displayed across the top of the Navigate tab.



One can interact with each breadcrumb through the dropdown selector that's next to it.



Edit and **Remove** are self-explanatory. **Navigate Here** refocuses the current tab on only the selected breadcrumb, so that it is the first and only breadcrumb in the trail. **Insert Before** allows one to insert a new breadcrumb/query in front of the selected one.

Queries

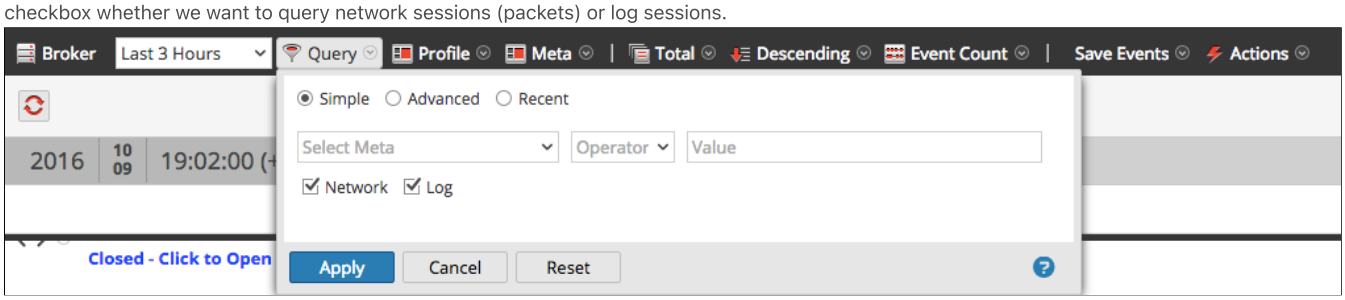
You've seen that **query** is really just another name for **drill**. SA gives us lots of ways that we can query our data set. Another is by use of the Query

button.

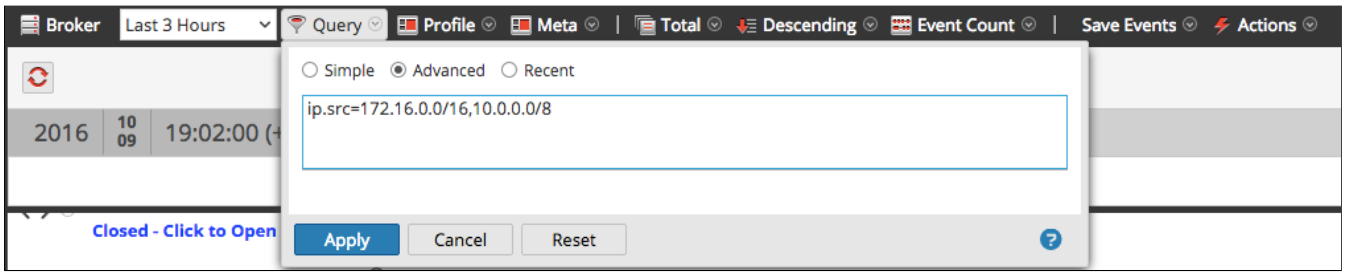
The Query button allows one to query the data set without having to look at the meta values below. It's mainly used when one already knows what one is looking for.

There are three types of queries:

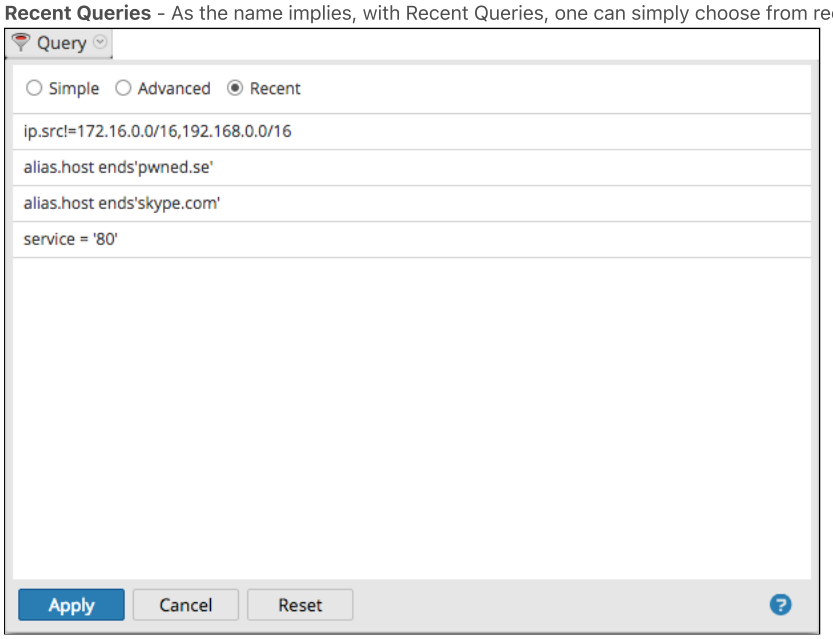
- **Simple Query** - This allows us to choose our meta key from the full list of meta keys, an operator, and a value or values. We can also specify via checkbox whether we want to query network sessions (packets) or log sessions.



- **Advanced Query** - Advanced queries are queries that are typed in manually. For instance, the query "ip.src=172.16.0.0/16,10.0.0.0/8"can be typed in here. You can also add additional clauses in a single query using logical AND (&&) and OR (||) operators. It will display red warning text if your query is invalid.



- **Recent Queries** - As the name implies, with Recent Queries, one can simply choose from recent queries here.



Comparison Operators

Queries support the following comparison operators. This is a standard which is used in many parts of SA.

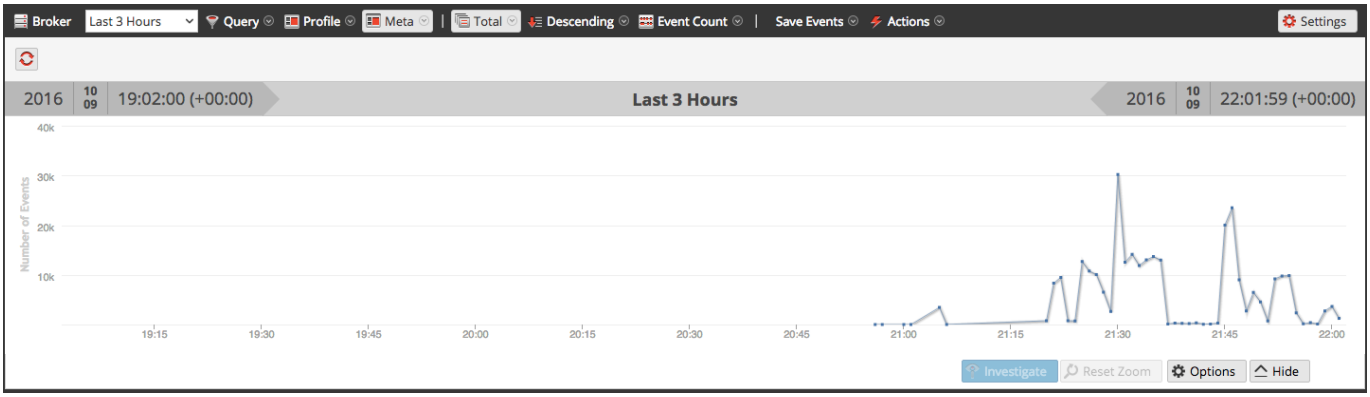
Operator	Description
=	Equality operator
!=	Inequality operator
&&	Logical AND operator
||	Logical OR operator
-u	Upper boundary, or greater-than operator. E.g. for selecting all TCP destination ports greater than 1024 (TCP high ports), use "tcp.dstport = 1024-u"
l-	Lower boundary, or less-than operator. E.g. for selecting all TCP ports below 1024 (TCP low ports), use "tcp.dstport = l-1024"
exists	Does the given meta key exist in the session (is it defined)? This is extremely fast and works well with keys indexed at the key level
!exists	Does the given meta key not exist in the session (is it not defined)? E.g. "attachment exists". This is extremely fast and works well with keys indexed at the key level
begins	Does a meta value start with a given string? E.g. "client begins 'mozilla/5.0'".
contains	Does a meta value contain a given string? E.g. "alias.host contains 'amazon'". This is computationally more expensive than =
ends	Does a meta value end with a given string? E.g. "filename ends '.exe'". This is computationally more expensive than =.
length	Is the length of a string equal to a given value? E.g. "tld length 3"
regex	Perform a regular expression match on the value. E.g. "email regex '^.+pwned.*+@.+\..+$'"
( )	Grouping operator. By enclosing an expression in parentheses, a new logical expression is created. These are usually used in conjunction with **&&** and **||** operators
- (hyphen)	Range operator. Only applies to numeric values. E.g. "udp.dstport=68-69"
, (comma)	List separator. Allows one to create a list of values. E.g. "ip.src=192.168.0.0/16,10.0.0.08"

IP addresses can be matched on individual IP's, a comma-separated list of IP's, or using CIDR notation.

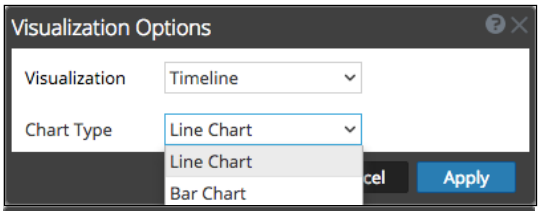
```
ip.src=10.1.2.3
ip.src!=10.2.3.4,10.5.6.7.10.8.7.6
ip.src=10.5.4.0/24
```

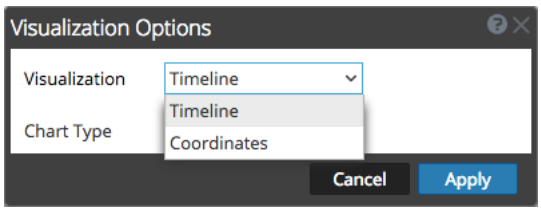
Visualizations

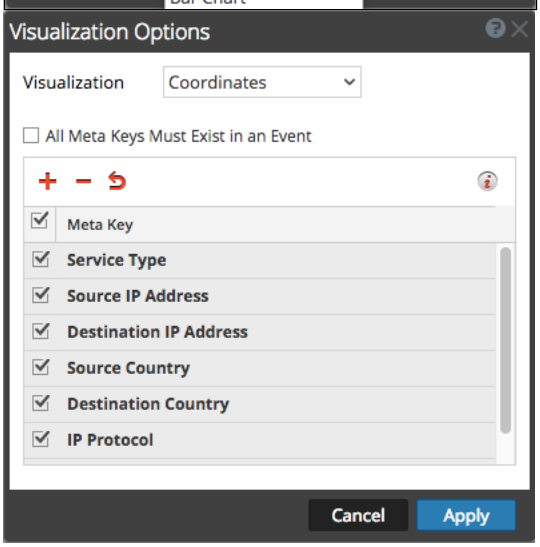
Remember that interesing-looking graphic we saw when we first got into Investigation? Let's have a look at that now. Click the Visualization Visualization button on the toolbar.



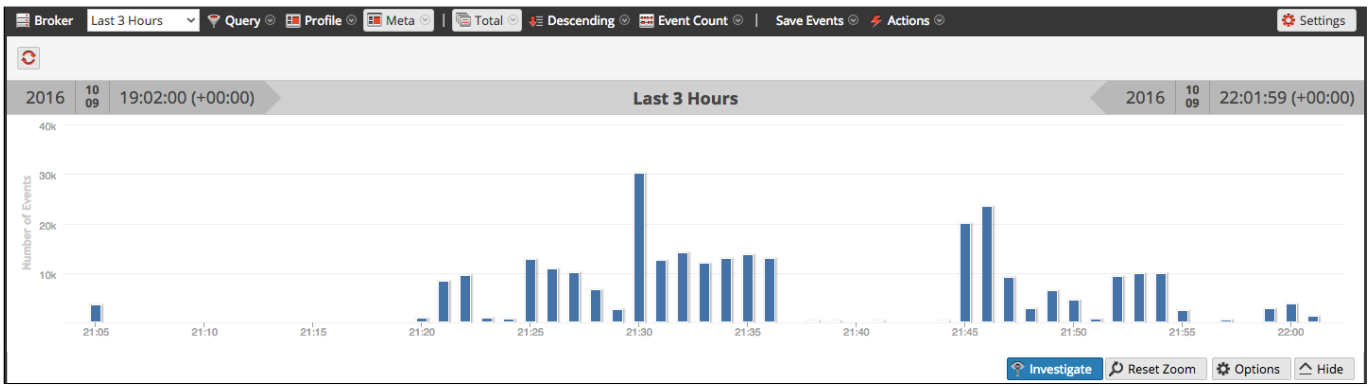
Timeline Charts - The default visualization is a **Timeline** chart. Timeline lets one see the volume of sessions in a given timeframe. This can help you focus your investigation on periods of peak traffic. Timeline charts can be displayed as a Bar Chart or a Line Chart. Click the options button to see your choices.





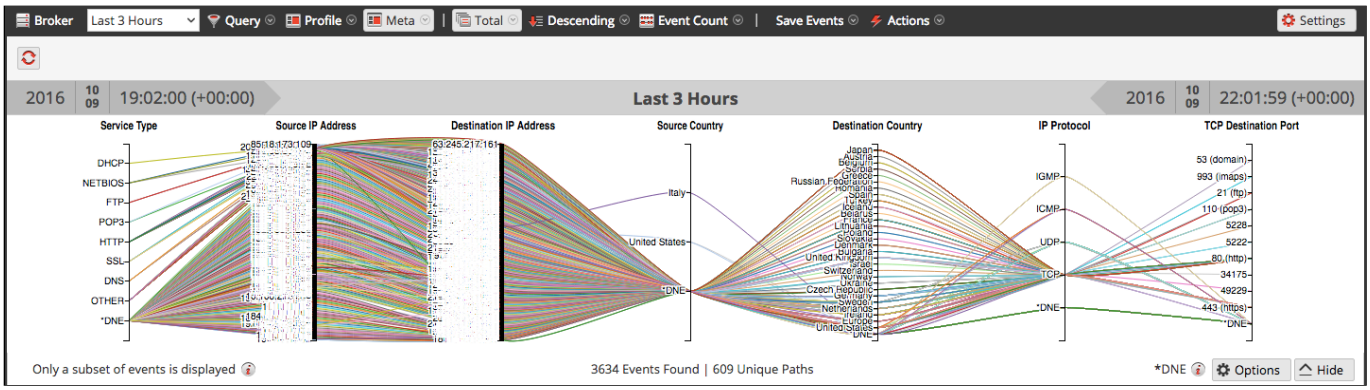


For Timeline charts, we can choose between a line chart or a bar chart.

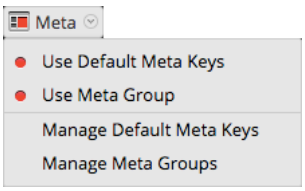


We can also refocus the chart by click-dragging over a portion of it. Give it a go.

Parallel Coordinates - We can also see that there is another type of visualization called **Coordinates**, or Parallel Coordinates. This aids hunters in finding patterns in the data which may not be immediately obvious. The methodology employed here is beyond the scope of this course.



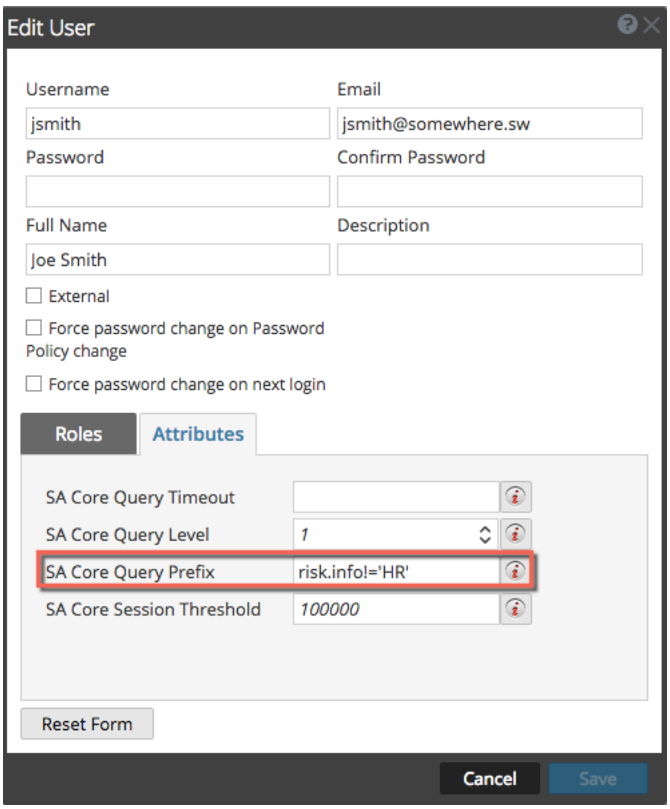
Query Prefixes

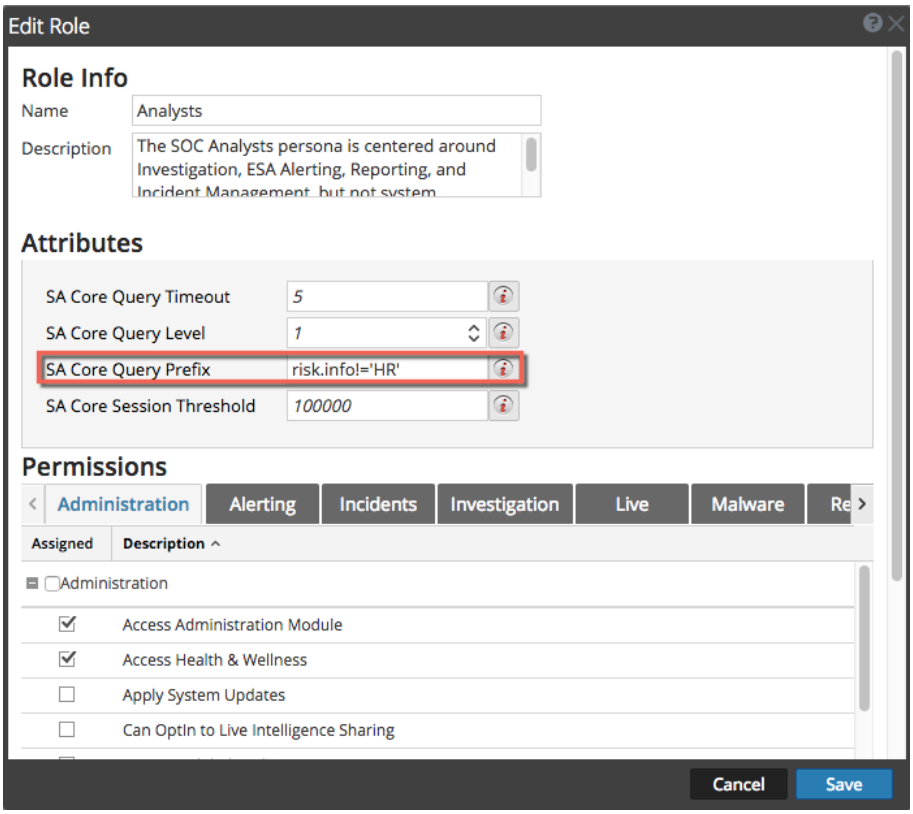


Sometimes it is necessary to apply pre-filters to queries on meta, for convenience or even security reasons. For instance, Company X has Analysts which are permitted to investigate all sessions EXCEPT those from the HR department, as those sessions may contain sensitive data (for instance, risk.info!='HR'). In this case, an Administrator may apply a Query Prefix to these Analysts which prevents them from seeing these sessions. A Query Prefix may be applied to individual SA users, or to the Analysts' SA Role.

A **Query Prefix** is a standard query/rule statement which gets logically AND'ed to all queries issued by a user. Query Prefixes are generally configured in **Administration -> Security -> Roles** or **Administration -> Security -> Users**.

When a **Query Prefix** is applied, the user will not even see it Investigation. The Investigation results will simply have the prefix applied to them, invisible to the user.



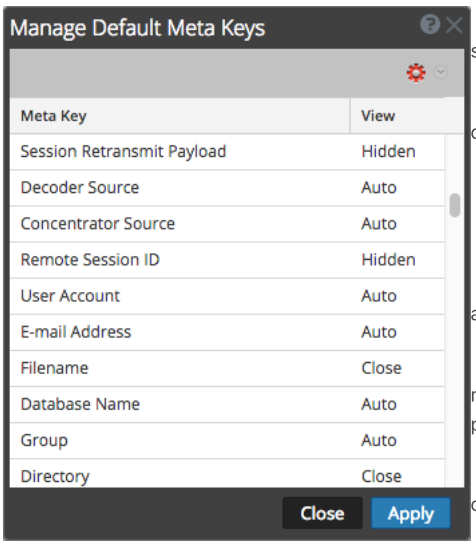


Meta Groups

Meta Groups are an Investigation feature which allow one to configure which meta keys are displayed and in what order. For instance, perhaps you have different investigation use-cases, like email investigations, or botnet investigations. Meta groups allow one to show only those meta keys that are relevant to that type of investigation, which can speed the Analyst's workflow.

To choose and/or Meta Groups, click the Meta link on the Investigation toolbar.

Here you can manage which keys to show by default in Investigation, and how they will open. Now click on **Meta -> Manage Default Meta Keys.**



Here you will see every meta key that is available to the Analyst. For each meta key, there is a **view** setting which controls the default display state in Investigation.

You can also control the order in which they are displayed by click-dragging a key upwards or downwards.

The view setting choices are:

Auto - Uses the value specified in the index configuration file of the service being investigated.

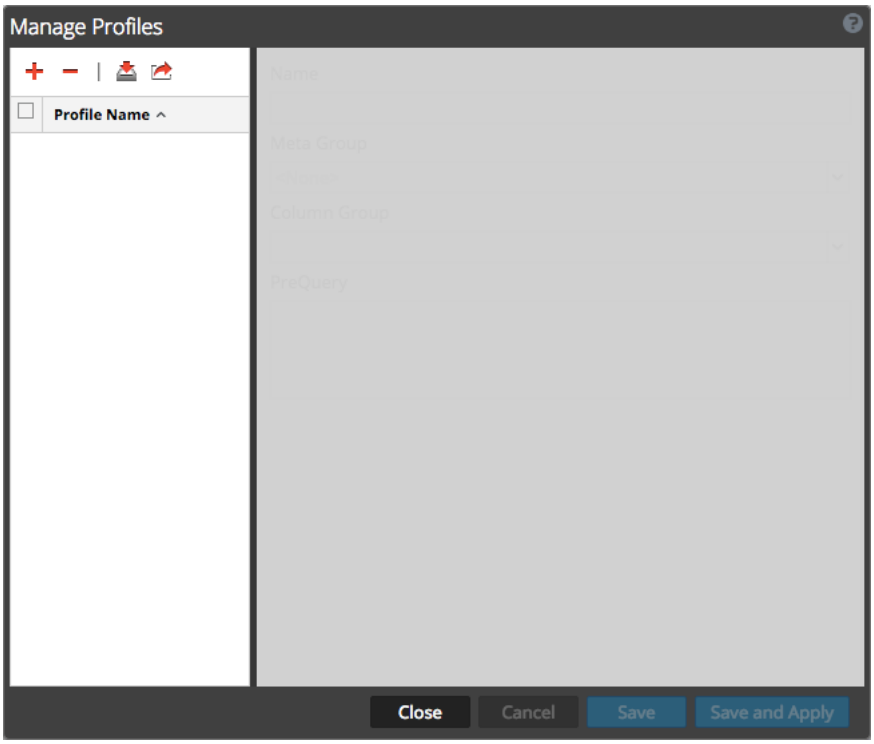
Open - The meta key will always appear and will always display its values by default. This can have a performance penalty if there are many values under the meta key.

Closed - The meta key will always appear but will not display its values by default. The Analyst must click the **Closed - Click to Open** button under the key to load the values. This can aid performance and prevent the Analyst becoming distracted by superfluous meta.

Hidden - Do not even display the meta key. This means the key is superfluous to the Investigation or isn't needed until we view the sessions as events.

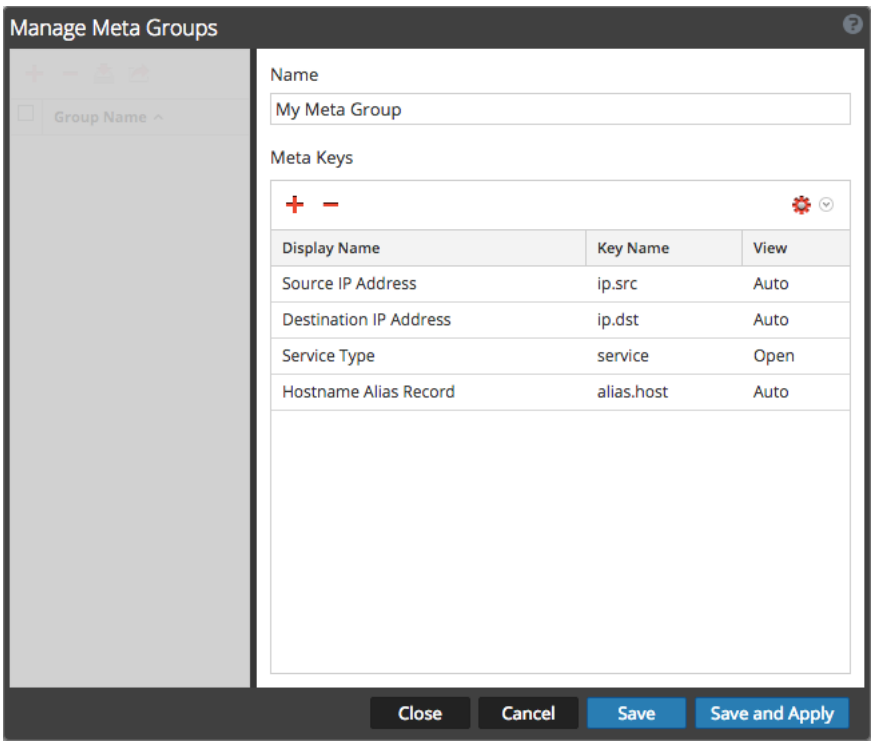
Creating a new Meta Group

1. Choose **Manage Meta Groups** under the **Meta** dropdown on the toolbar.

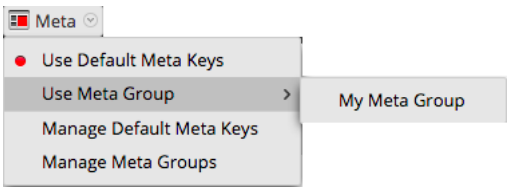


2. Click the + button.

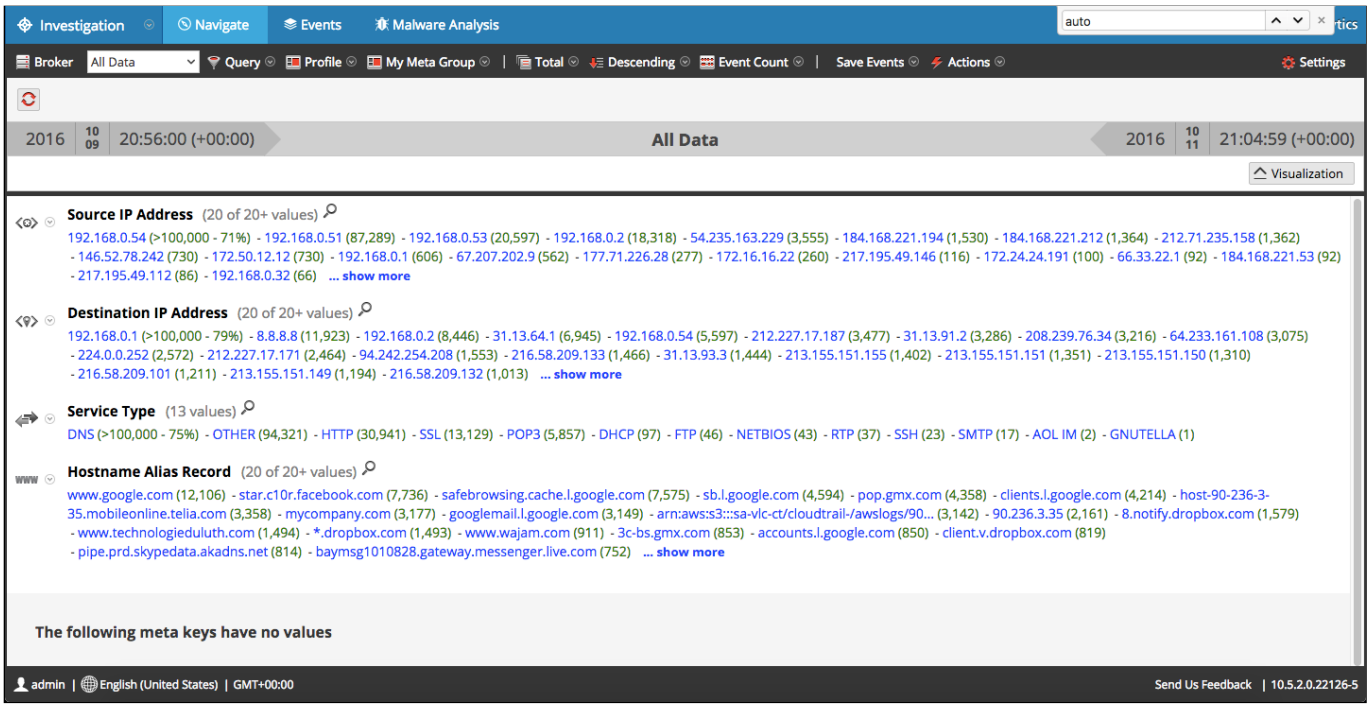
3. Give your Meta Group a name and use the controls to add keys to it, and click-drag to re-order them.



4. Click **Save** or **Save and Apply.** You will now see your meta group on the **Meta** dropdown under **Use Meta Group.**



5. Click on your new Meta Group to use it in the Investigation view. The toolbar will now display your **Meta Group** name instead of the Meta button.



6. Click on the **Meta** dropdown and **Use Default Meta Keys** to restore the default Investigation view.

Meta Profiles

Meta Profiles are an extension of **Meta Groups**. While Meta Groups provide the ability to show only meta keys which are relevant to an Analyst, Meta Profiles give an Analyst the ability to further isolate meta relevant to a particular investigation use-case, by ensuring that keys display only meta from sessions which have certain characteristics. Examples of these characteristics might be only HTTP sessions (**service=80**).

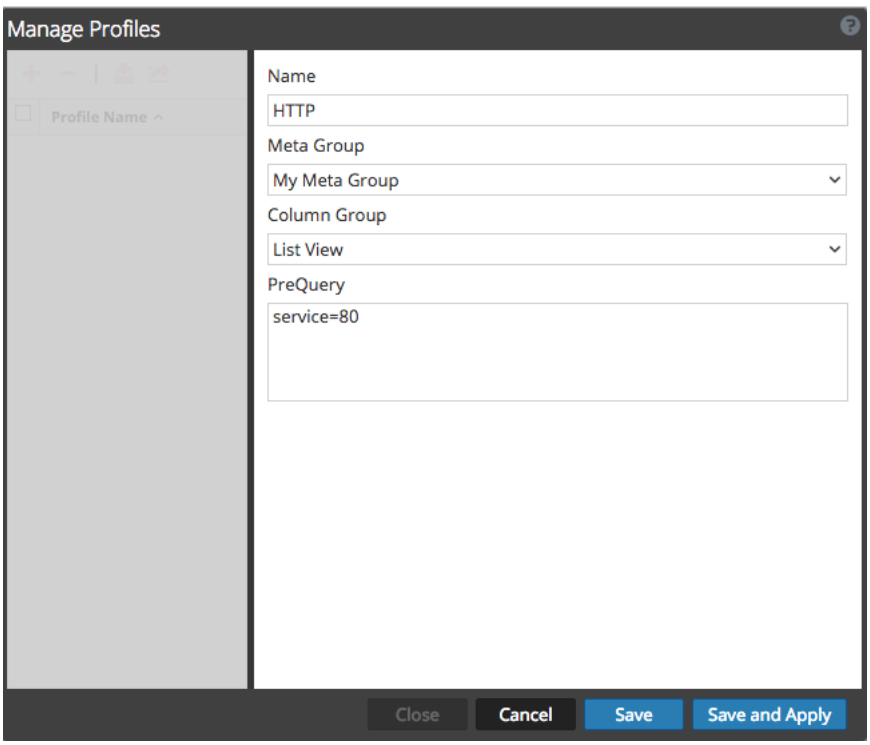
When a Meta Profile is selected in Investigation, its pre-filter statement is applied to the user's current Navigation view. This has the effect of working exactly like a **Query Prefix**, but instead of being a security restriction, is a filter for the workflow's sake.

Creating a Meta Profile is quite simple.

1. Choose the Profile button from the Investigation toolbar.

2. Choose **Manage Profiles** from the dropdown.

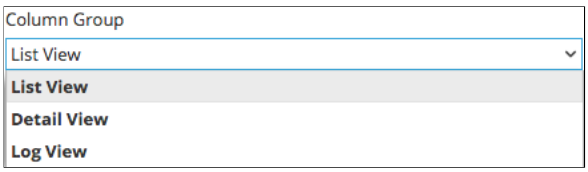
3. Click the + sign to add a new profile.



4. Define your Meta Profile parameters:

- **Name** - Self-explanatory

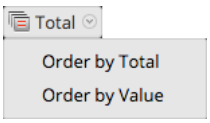
- **Meta Group** - Choose a Meta Group to apply to the Profile. Only meta keys defined by this **Meta Group** will be displayed when the **Meta Profile** is selected.
- **Column Group** - This controls how events are displayed in the **Event** view
- **PreQuery** - This is a query statement to apply to the Investigation when this Meta Profile is selected.



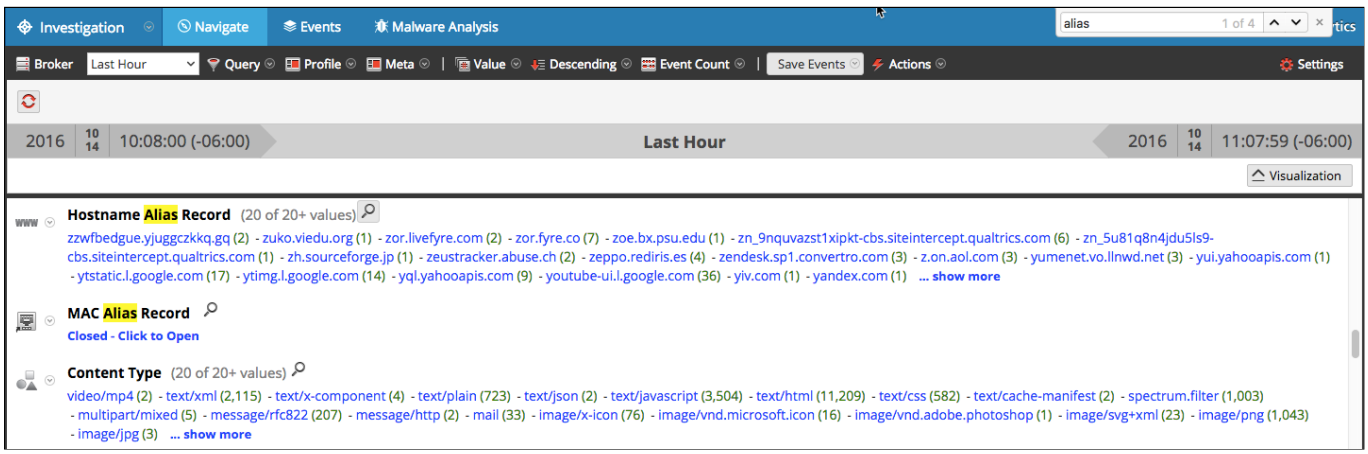
5. Click Save.
6. You can now choose your Meta Profile from the Profile dropdown. The dropdown will now assume the name of the **Meta Profile** you've selected.

Sorting

SA allows the Analyst to sort the values displayed under meta keys either by the number of sessions which contain a given value (the default), or also one can optionally sort by the value itself. Click the dropdown on the Total toolbar button to see your options.



Use the Descending / Ascending button to the right of the Total button to reverse the sort order.



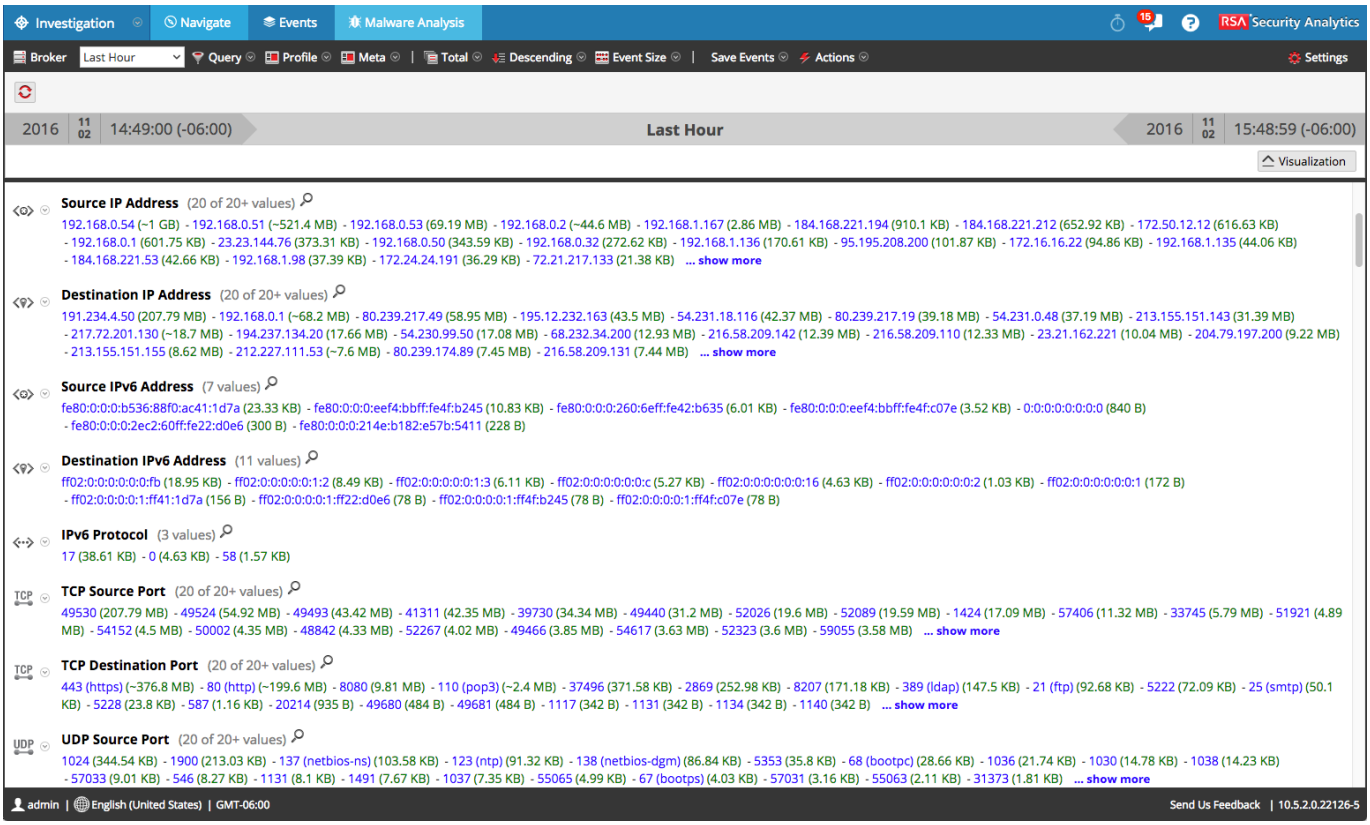
Meta Sorted by Value

Value Quantification

I'm sure by now you've noticed the green number, in parentheses, next to every meta value, which refers to a quantity associated with the value. There are three ways which the value quantity can be expressed:

Event Count - The default setting, this means that the number expressed to the right of every value refers to the number of sessions in the selected timeframe which contain the given meta value.

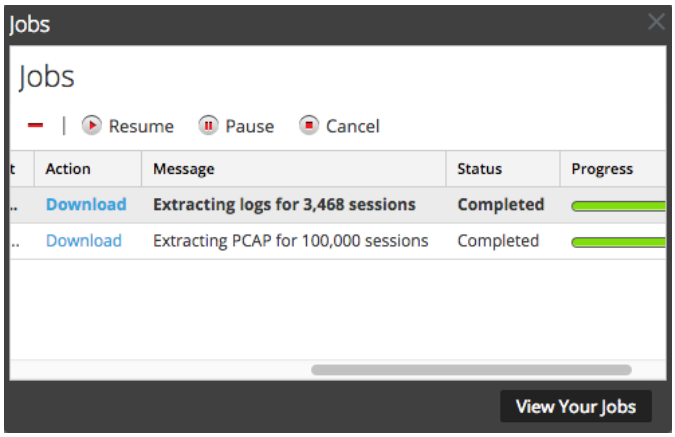
Event Size - The number expressed by a value refers to the aggregate size of packet data for all sessions containing the given meta value.



Packet Count - The aggregate number of packets contained in all sessions containing the given meta value.

Switching between the three settings means clicking on the Event Count button (by default) on the Investigation toolbar, and selecting one of the settings. The button's text will change, dependant on what setting is selected.

Saving Events



From the Save Events menu of Investigation, it is possible to save data from all of the sessions from which you've drilled into. You can save:

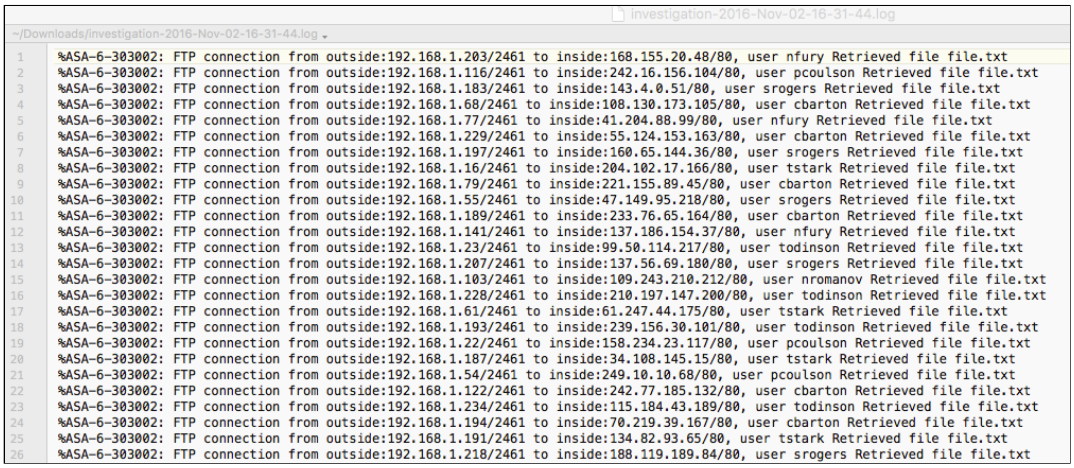
PCAPs - You can export all the PCAP data into a single PCAP file

Logs - Any logs in the current drill

Files - Any files detected within the current sessions

Be sure to drill in as far as you can before you save events - it drastically cuts down on system resources. Please don't try saving all events from the root drill! Every time you choose to save something, you'll be asked for a filename. Click Okay, and then click on the Jobs icon to see the extraction progress.

From the Jobs panel, click the **Download** link to download the finished file.



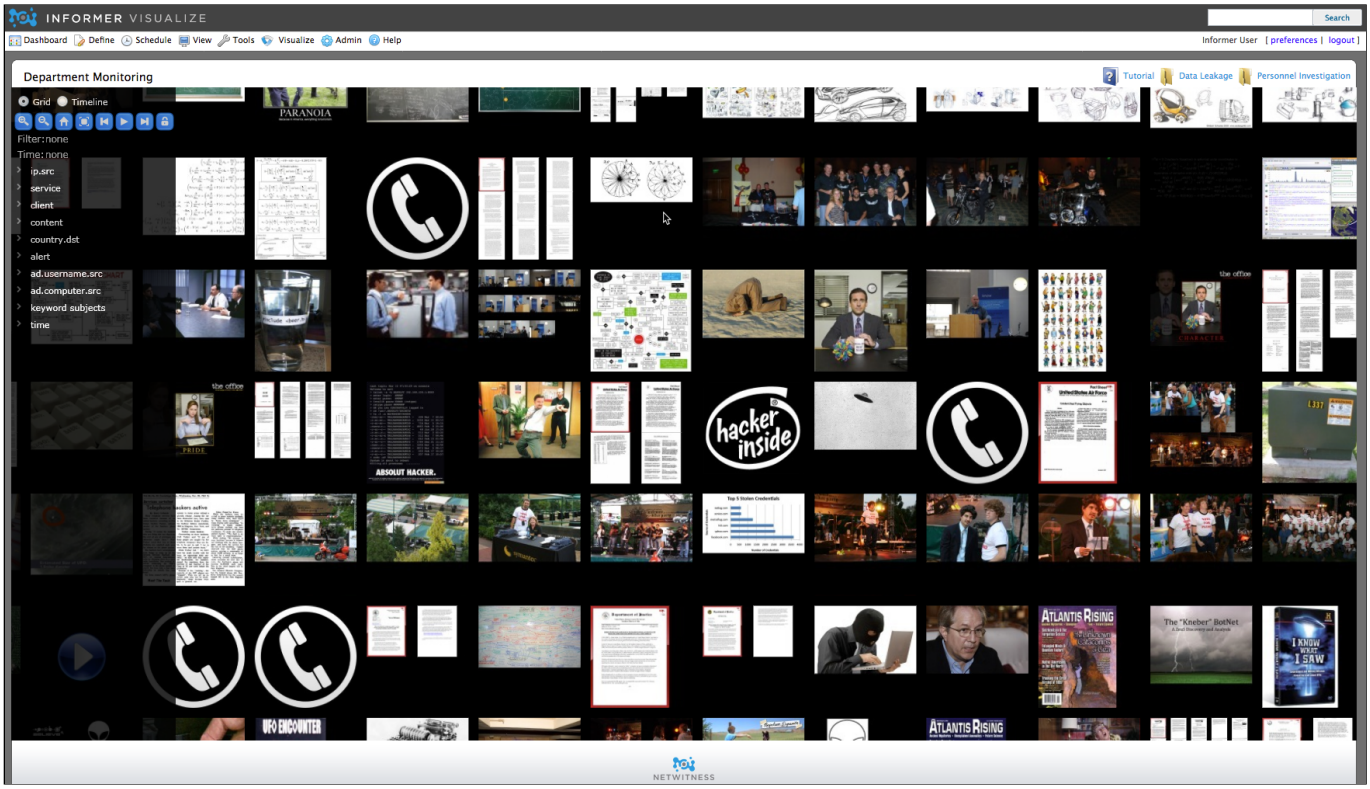
Actions

The Actions dropdown allows an Analyst to do three more things.

Go To Event - Use this if you have a session ID which you want to view in the Events view. E.g. an ESA alert or Malware Analysis have provided a session ID to look at. Remember that when using this, session ID's are specific to individual Core services, so ensure you've got the correct service selected in the service selector first, or you'll be liable to pull up an entirely different session.

Print - Provides a nice printable view of the current drill

Visualize - This pulls the current drillpoint up in a legacy NetWitness Informer instance, specifically in its Visualize feature. Visualize was a very cool NetWitness feature which gave analysts the ability to see documents and images, and listen to audio from captured sessions in a wonderful graph. This functionality unfortunately never made it into SA, but if you still have an old Informer appliance around, you can still access this functionality from within SA. To check out a demo of NetWitness Visualize, visit http://visualize.netwitness.com. It requires Sliverlight, so you won't be able to access it from your training laptops.

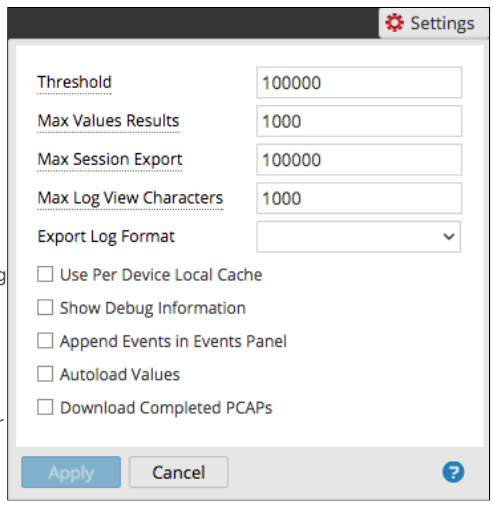


Cache Window

The Cache Window (/sdk/config/cache.window.minutes) is a setting which is configured on every individual SA Core service which Investigation queries. It sets a period of time, in minutes, in which the results of a given query will be cached. This means that during the cache window, the Core service will return cached results instead of re-querying. Once the configured time has elapsed, the query would be executed again. This setting is most germane to multi-user installations where multiple Analysts are querying SA simultaneously. This helps reduce load on the Concentrator, and potentially frees its resources to service other new queries. Sometimes whilst troubleshooting, this setting can be an annoyance as the Investigation results will not update during the cache window. Setting the value to 0 will disable the query cache and give a more real-time feel to Investigation, however it can also adversely affect query performance. Use a value of 0 either temporarily or give it a testing period to confirm it doesn't negatively affect your organisation's workflow.

Navigate Settings Panel

There's a nice little Settings panel in Investigation which will allow you to set:



- Session **Threshold** - This indicates the maximum number of sessions to load when calculating the quantity for a given value. A high value means more accurate quantity results, but longer load times.
- **Max Values Results** - This determines the maximum number of values to load when using the Max Results feature of a meta key.
- **Max Session Export** - Determines the maximum number of sessions which get extracted during an export operation.
- **Max Log View Characters** - Limits the number of characters of every log message to show in the Events tab (more on the Events Tab below).
- **Export Log Format** - Preselects the format to use when exporting logs from Investigation
- **Use Per Device Local Cache** - Use the local cache of the service chosen in the service selector rather than the UI's own cache.
- **Show Debug Information** - Adds text to the bottom of each key shown in the Navigate tab, showing how long it took to load the values. This can be useful for troubleshooting.

- **Append Events in Events Panel** - When loading more events by switching to a new page in the Events tab of Investigation, rather of switching pages, the new results will instead be added to the existing page, creating one long list of events.
- **Autoload Values** - Do not require Analyst to hit the Load button when entering the Navigate view of Investigation
- **Download Completed PCAPs** - Automatically download completed PCAP and Log files which have been exported in Investigation.

Top Tip: The Navigate settings panel can also be found in your user profile. Click **Start->Profile** and then the **Investigation** tab. You will find a few other settings here, as well.

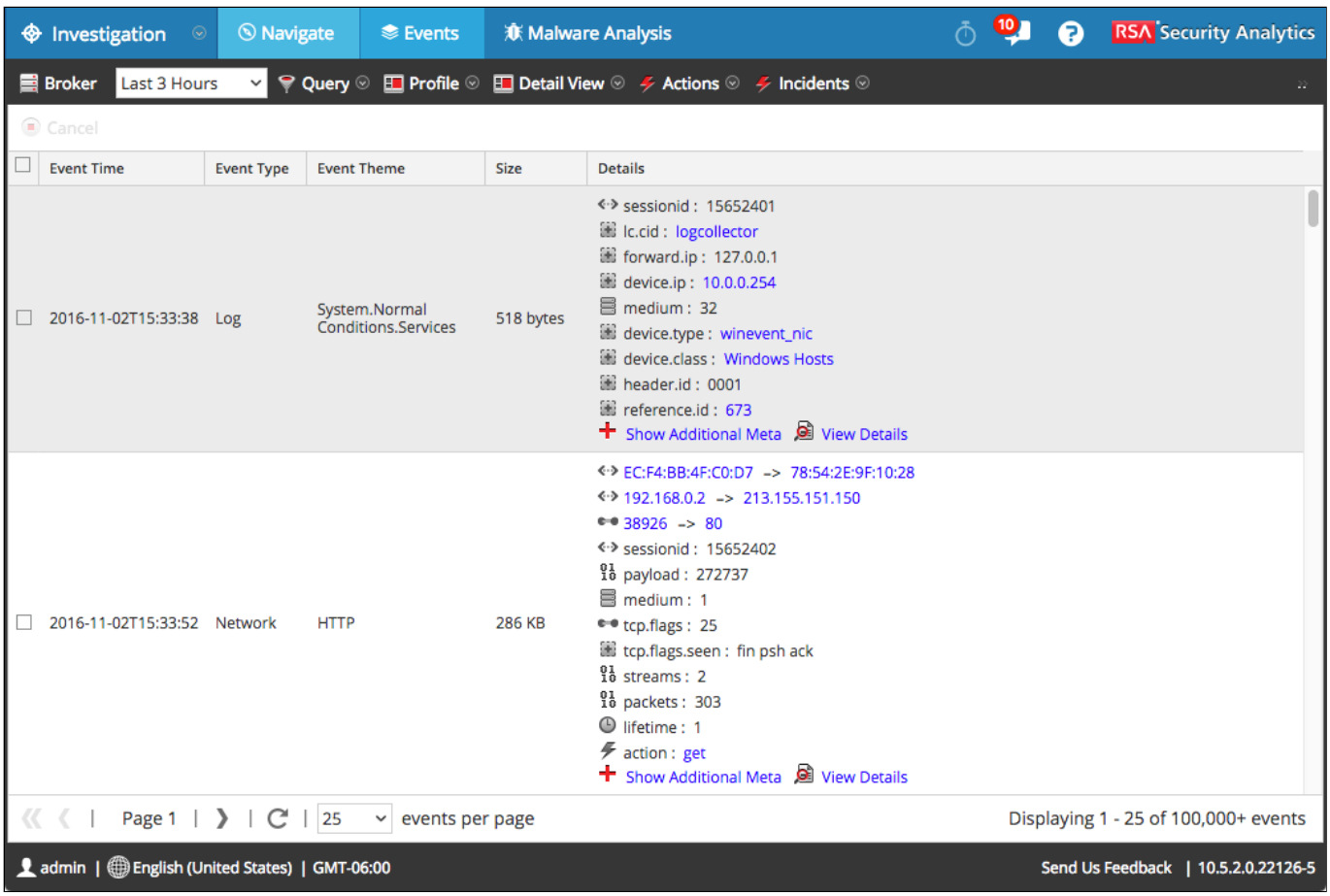
The Events Tab

The Events tab of Investigation provides an alternative view of sessions data than the Navigate tab. Rather than showing a meta-centric view of captured data, the Events tab instead shows a session-oriented view. Many of the concepts from the Navigate view also apply to the Events tab, such as the service selector, the timeframe selector, queries, the breadcrumb trail, meta profiles, and actions.

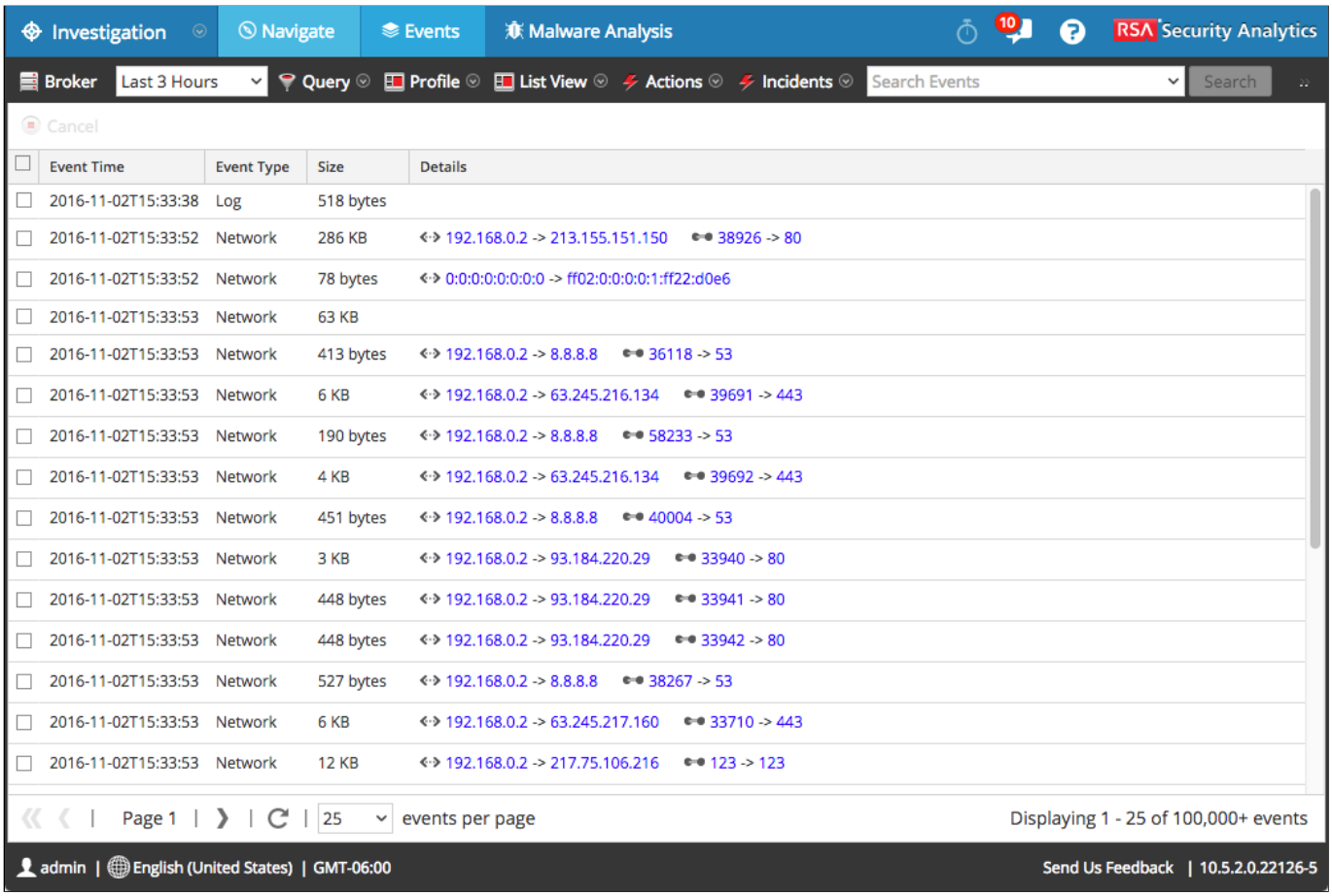
The Events view shows a top-down view of individual sessions, arranged into rows. Unlike the Navigate view, it is paginated, so it's sometimes necessary to click forwards and backwards to new pages.

The Events view provides three different views into SA's captured sessions:

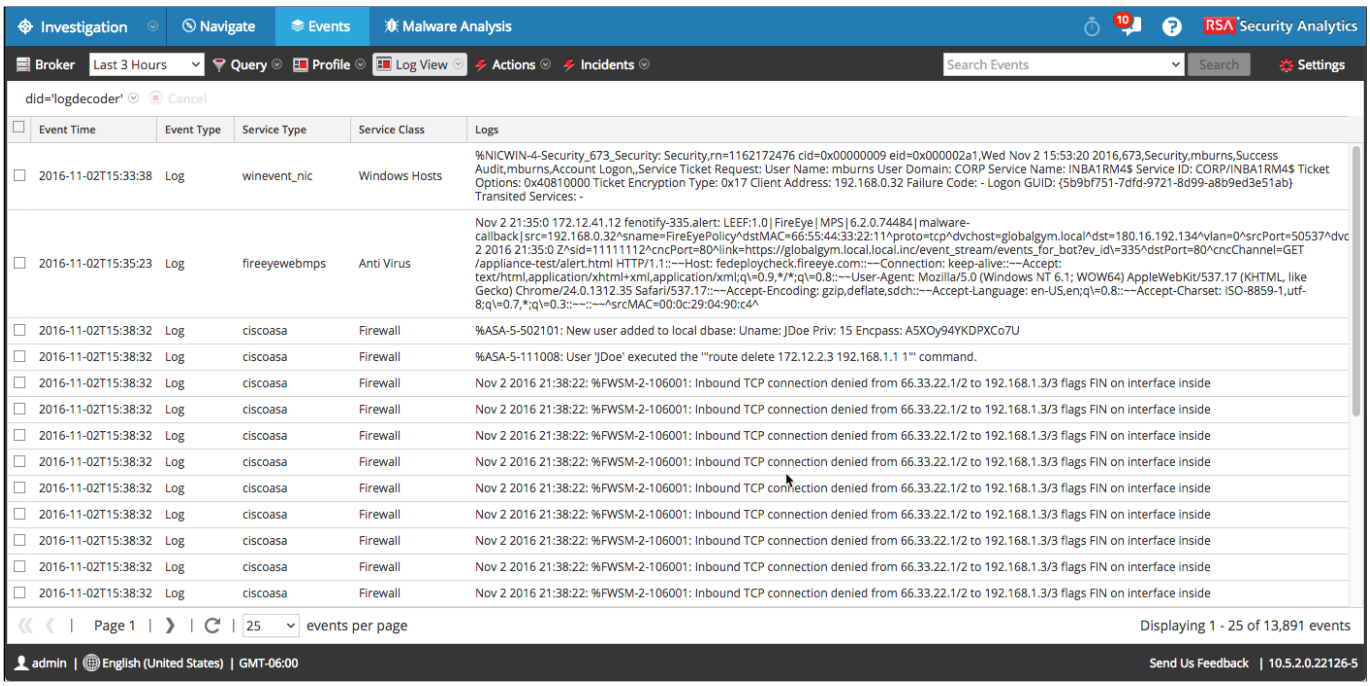
- **Detail View** - The detail view provides a way to see much pertinent meta for every session, displayed vertically in each session's row.



- **List View** - The list view is much more succinct than the details view, displaying meta in columns rather than vertically.



- **Logs View** - The logs view shows raw log messages captured by Log Decoders. This is not an in-motion, live streaming view of log data.



Drilling Into Event Details

One can always simply click the Events tab of Investigation, but a very common and useful way of moving into the Events tab is to click on one of the value totals in Investigation. A total is the parenthetical bit of green text next to every meta value.

When you click on a total, Investigation will switch to the Events view whilst preserve your existing drill from the Navigate view, and adds the value for the total you clicked on to the drill.

Investigation | Navigate | Events | Malware Analysis | RSA Security Analytics

Concentrator | All Data | Query | Profile | Meta | Total | Descending | Event Count | Save Events

service=80 | action = 'get' | client = 'mozilla/5.0'

2016 10 09 15:20:00 (-06:00) **All Data** 2016 11 02 17:45:59 (-06:00)

Visualization

Threat Category (5 values)

informational (>100,000 - 18%) - suspicious (2,512) - malware (560) - nonstandard (37) - cnc (37)

Threat Description (20 values)

unitedstates (>100,000 - 30%) - europeanunion (96,830) - germany (7,938) - france (7,229) - greatbritain (6,449) - canada (5,475) - denmark (5,236) - latvia (4,144) - netherlands (3,488) - china (3,183) - sweden (3,066) - switzerland (2,336) - dynamic dns domain (1,443) - russianfederation (1,113) - hongkong (666) - taiwan (605) - japan (555) - unspecified:2014-12-03 01_03_11: (338) - norway (324) - ukraine (315) **... show more**

Service Type (1 value)

HTTP (>100,000 - 8%)

Hostname Aliases (20 values)

Click Me!

90.236.3.35 (91,888) - host-90-236-3-35.mobileonline.telia.com (83,181) - www.google.com (57,251) - www.tomshardware.co.uk (45,687) - d.pzkysq.pink (40,686) - vidtech.cbsima.com (15,573) - safebrowsing-cache.google.com (13,637) - www.pwned.se (12,691) - pagead2.googlesyndication.com (6,805) - ad.zanox.com (4,951) - sourceforge.net (4,497) - image2.pubmatic.com (4,485) - www.google-analytics.com (4,078) - ads-grooveshark.com (3,822) - www.gstatic.com (3,813) - ib.adnxs.com (3,449) - www.wajam.com (3,379) - www.google.se (3,160) - b.scorecardresearch.com (3,061) - www.ly.com (3,032) **... show more**

Source IP Address (20 values)

192.168.0.51 (>100,000 - 13%) - 192.168.0.54 (>100,000 - 57%) - 54.235.163.229 (83,230) - 212.71.235.158 (61,887) - 192.168.0.2 (33,528) - 67.207.202.9 (27,538) - 217.195.49.146 (2,940) - 95.199.31.163 (1,470) - 217.195.49.112 (1,372) - 217.195.49.103 (1,176) - 184.75.214.163 (1,029) - 207.244.76.170 (980) - 37.48.77.227 (931) - 103.10.197.187 (833) - 178.238.237.230 (637) - 95.195.199.155 (535) - 123.59.33.26 (474) - 192.168.0.32 (419) - 91.217.90.49 (414) - 37.48.81.52 (392) **... show more**

Destination IP address (20 values)

192.168.0.2 (>100,000 - 32%) - 192.168.0.1 (>100,000 - 45%) - 80.239.178.178 (22,143) - 54.149.65.87 (20,245) - 80.239.178.187 (18,493) - 213.155.151.151 (13,087) - 54.200.221.84 (12,496) - 54.187.109.244 (12,446) - 80.239.178.200 (9,926) - 216.58.209.98 (8,965)

admin | English (United States) | GMT-06:00 Send Us Feedback | 10.5.2.0.22126-5

Investigation | Navigate | Events | Malware Analysis | RSA Security Analytics

Concentrator | Last Hour | Query | Profile | List View | Actions | Incidents

service=80 | action = 'get' | client = 'mozilla/5.0' | alias.host = 'www.tomshardware.co.uk' | Cancel

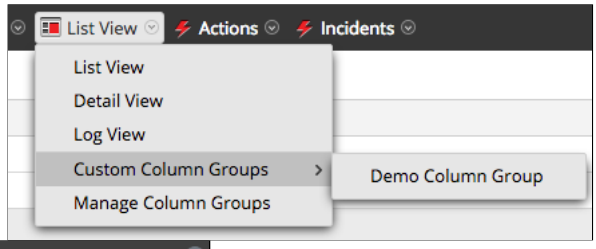
Event Time	Event Type	Size	Details	
2016-11-02T17:36:40	Network	135 KB	192.168.0.51 -> 54.200.221.84	50723 -> 80
2016-11-02T17:36:40	Network	19 KB	192.168.0.51 -> 54.200.221.84	50725 -> 80
2016-11-02T17:37:46	Network	131 KB	192.168.0.51 -> 54.200.221.84	51739 -> 80
2016-11-02T17:37:46	Network	27 KB	192.168.0.51 -> 54.200.221.84	51740 -> 80
2016-11-02T17:37:50	Network	5 KB	192.168.0.51 -> 54.149.65.87	48005 -> 80
2016-11-02T17:37:50	Network	5 KB	192.168.0.51 -> 54.149.65.87	48009 -> 80
2016-11-02T17:38:05	Network	5 KB	192.168.0.51 -> 54.149.65.87	48050 -> 80
2016-11-02T17:38:10	Network	5 KB	192.168.0.51 -> 54.149.65.87	48325 -> 80
2016-11-02T17:38:14	Network	5 KB	192.168.0.51 -> 54.149.65.87	48333 -> 80
2016-11-02T17:38:17	Network	5 KB	192.168.0.51 -> 54.149.65.87	48343 -> 80
2016-11-02T17:38:17	Network	5 KB	192.168.0.51 -> 54.149.65.87	48356 -> 80
2016-11-02T17:38:17	Network	5 KB	192.168.0.51 -> 54.149.65.87	48366 -> 80
2016-11-02T17:38:17	Network	5 KB	192.168.0.51 -> 54.149.65.87	48375 -> 80
2016-11-02T17:38:17	Network	5 KB	192.168.0.51 -> 54.149.65.87	48386 -> 80

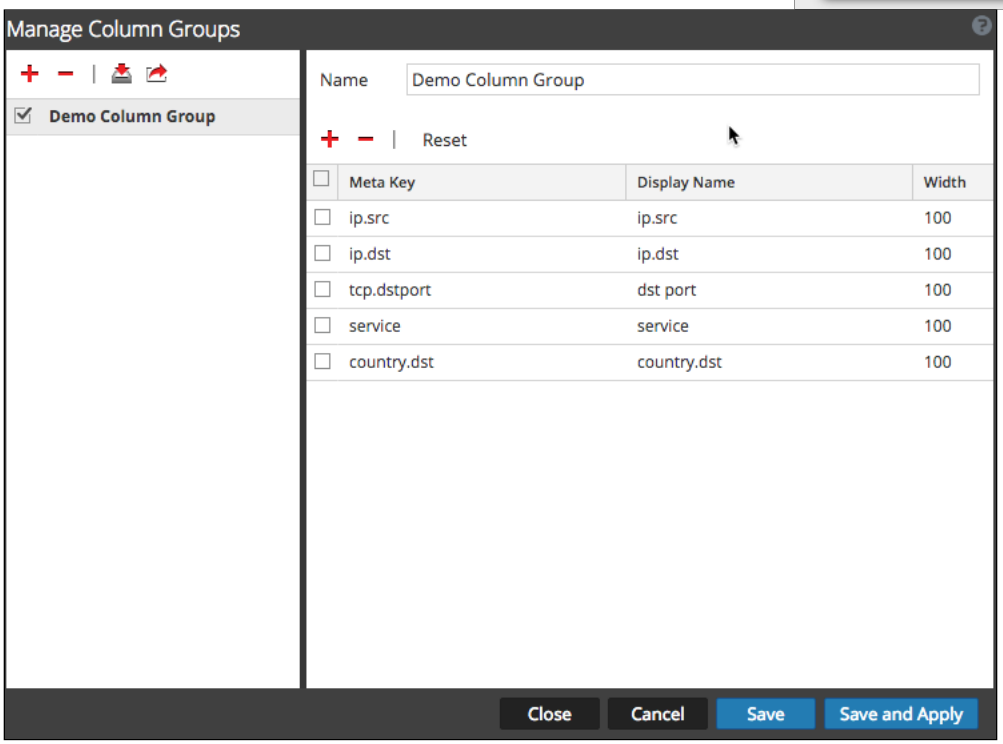
Page 1 | 25 events per page Displaying 1 - 25 of 904 events

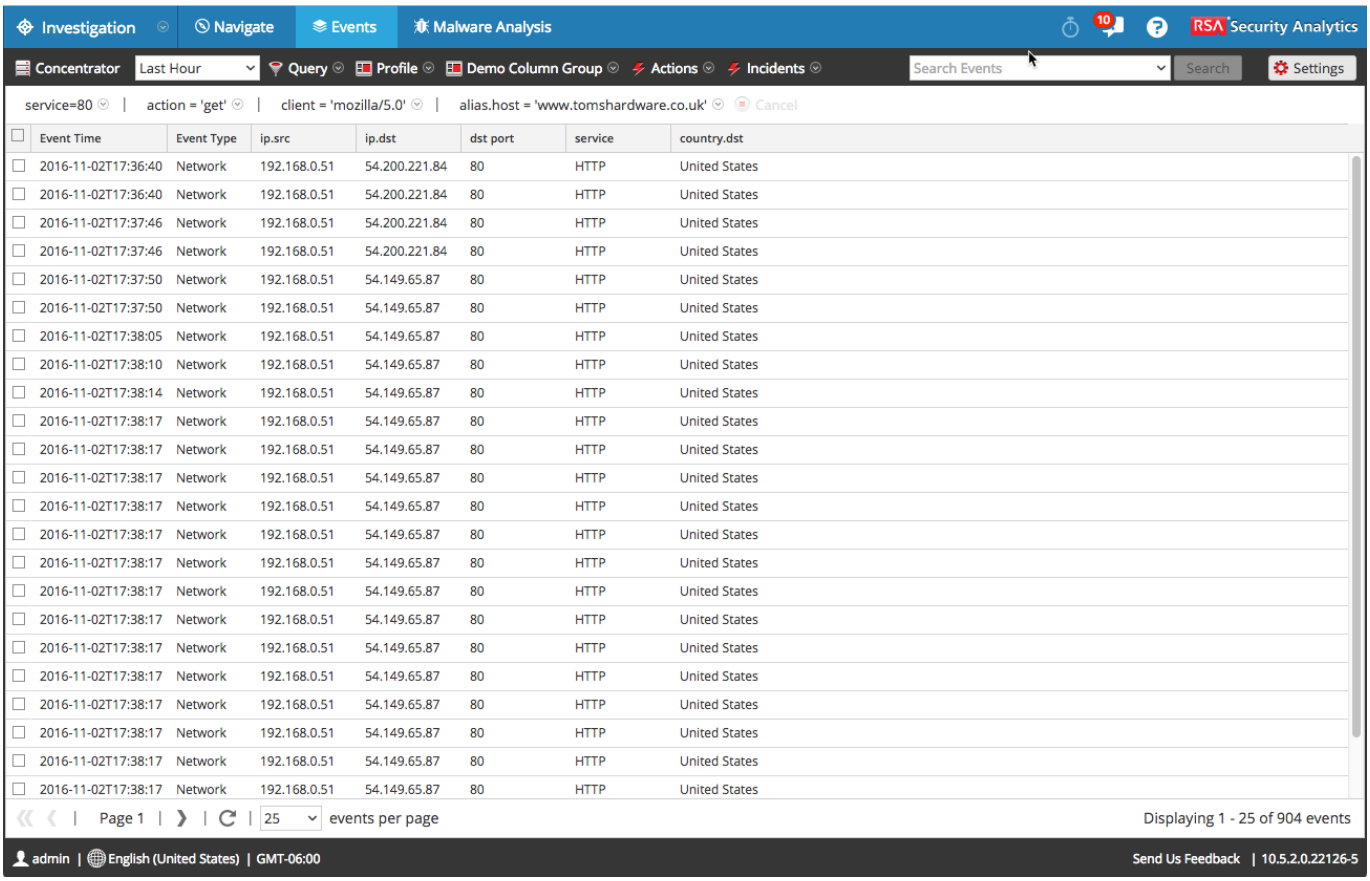
admin | English (United States) | GMT-06:00 Send Us Feedback | 10.5.2.0.22126-5

Column Groups

The Events view introduces the concept of Column Groups, where an Analyst can define which meta is shown in the Events view. Column groups only apply to the List view. Define and select them from the View button, where you select your Events view.





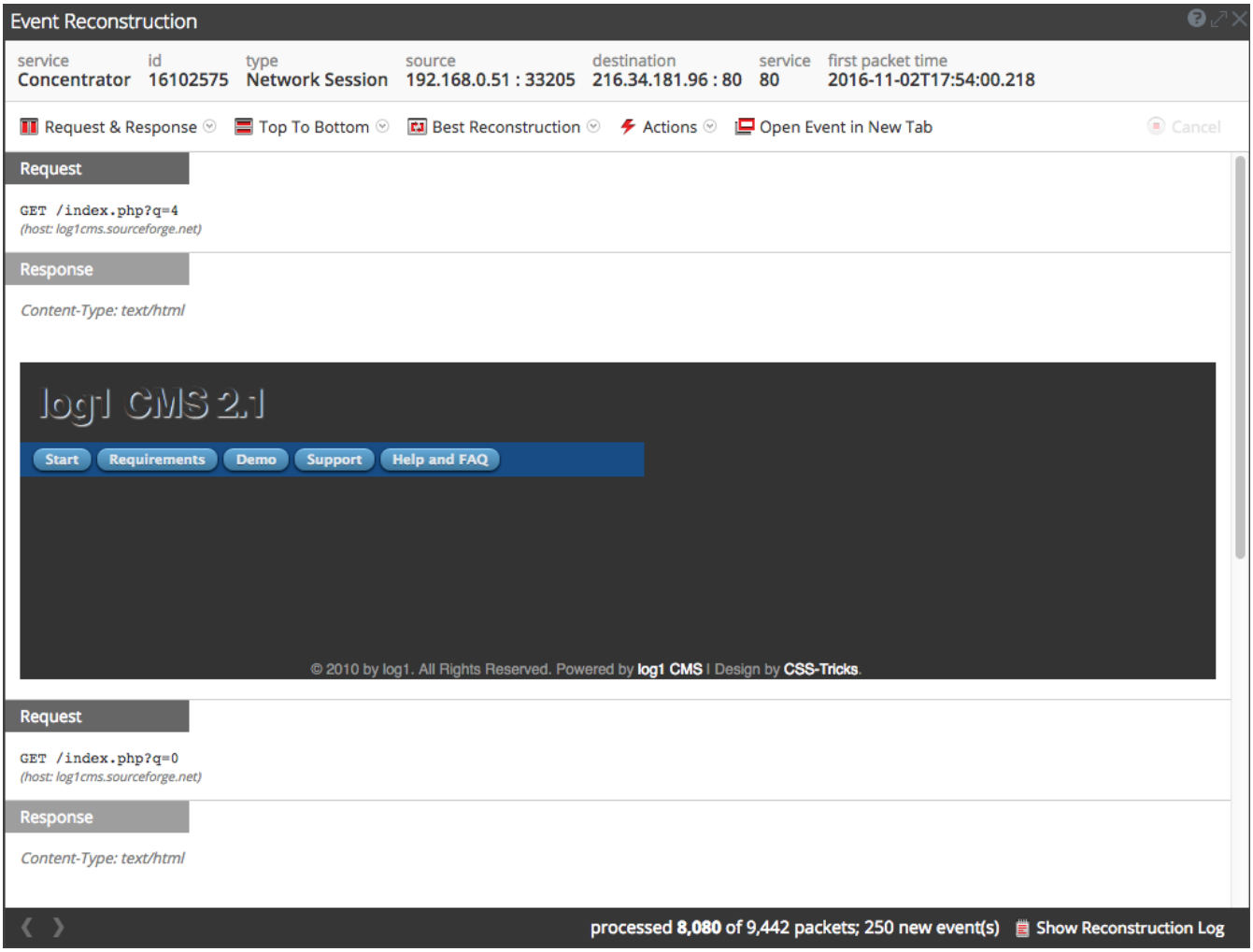


Event Reconstruction

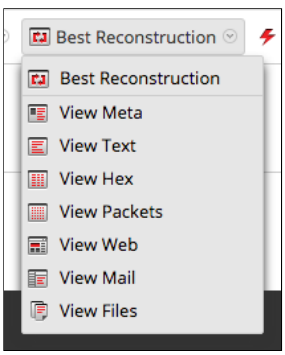
Perhaps most importantly, the Events view allows an Analyst to access Event Reconstruction, which provides a brilliant and comprehensive view into an individual session. Event Reconstruction allows one to view web pages, emails, and access files contained within sessions. You can view the raw packets of a session, view them in hex form, or as text. It also allows one to view all meta for a given session.

To reconstruct an event, simply double-click one of the sessions on the Events screen, or enter a session ID, if known, in **Actions->Go To Event**.

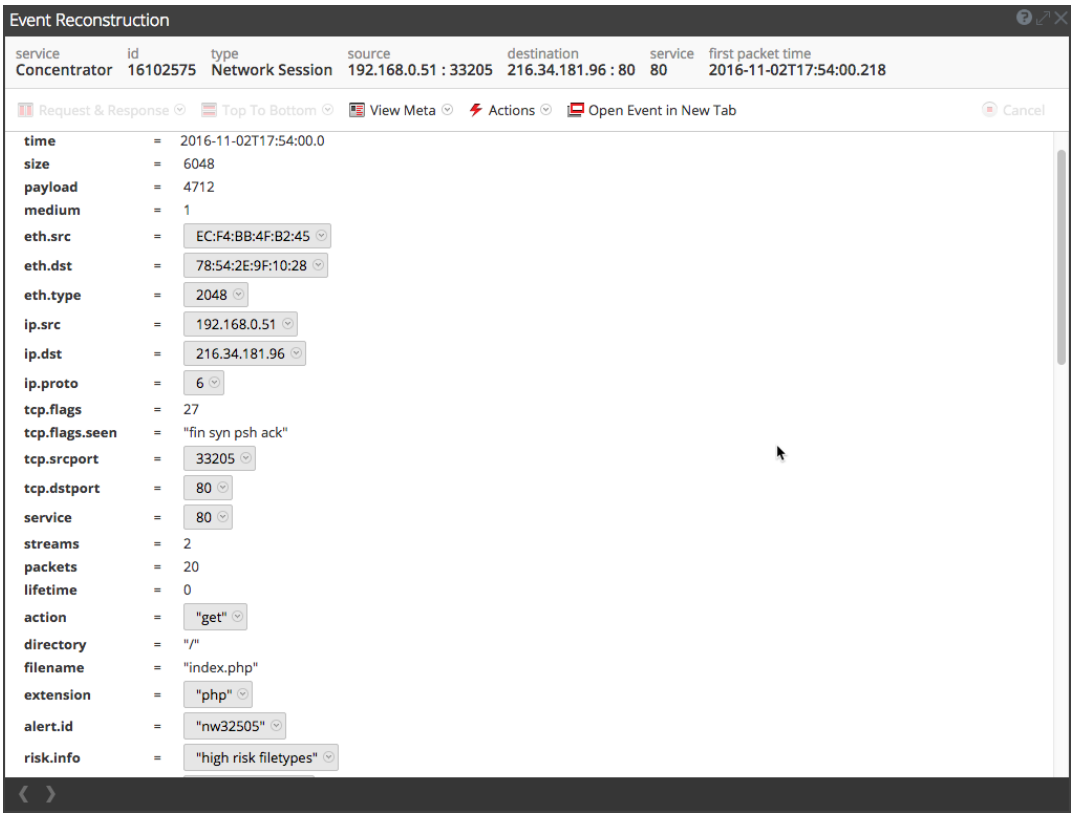
Best Reconstruction View: this is where you land when you first enter event *recon*. Based on the type of session, it will try to reconstruct it as either an email or a web page. Note that for web pages, **it will likely not be a faithful representation** of the original web page, as certain rendering technologies may be switched off for safety reasons, and parts of the page loaded through secondary sessions, such as XMLHttpRequests, won't be part of the loaded session.



Now that we've got event recon open, we can click on the Best Reconstruction button to see the other ways of viewing the session.

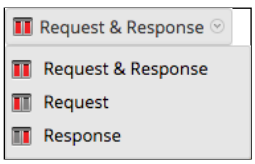


View Meta - This view allows one to see all the meta which belongs to a session. This can be extremely useful when conducting an investigation.

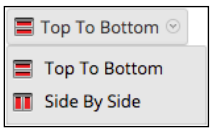


View Text - This view provides the ability to view captured session data as text. There are a view controls here, as well:

Request / Response Selector - This allows one to view only content associated with the request stream of a session (the source), or only data associated with the response stream of a session (the destination), or both.



Orientation Selector - This allows content data to be displayed in top-to-bottom (vertical) fashion, with content from request and response packets displayed alternately below each other. This is the default. It can also be displayed in a side-by-side manner, with data from request and response displayed next to each other.



Investigation | Navigate | Events | Malware Analysis | RSA Security Analytics

Event Reconstruction

service	id	type	source	destination	service	first packet time	last packet time	packet size	payload size	packet count
Concentrator	16102575	Network Session	192.168.0.51 : 33205	216.34.181.96 : 80	80	2016-11-02T17:54:00.218	2016-11-02T17:54:00.229	6,048 bytes	4,712 bytes	20

Request & Response | Side By Side | View Text | Actions | Cancel

Request

```
GET /index.php?q=4 HTTP/1.1
Host: loglcms.sourceforge.net
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64;
rv:33.0) Gecko/20100101 Firefox/33.0
Accept: text/html, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Referer: http://loglcms.sourceforge.net/
Connection: keep-alive
```

Response

```
HTTP/1.1 200 OK
Server: Apache/2.2.15 (CentOS)
Vary: Host, Accept-Encoding, User-Agent
Cache-Control: max-age=172800
Expires: Thu, 12 Mar 2015 11:28:31 GMT
Content-Type: text/html
Content-Length: 1278
Date: Tue, 10 Mar 2015 11:28:31 GMT
X-Varnish: 905608543
Age: 0
Via: 1.1 varnish
Connection: keep-alive
Content-Encoding: gzip

<!DOCTYPE html>


<html>
<head>
<head>
<title>log1 CMS 2.1</title>
<meta http-equiv='Content-Type' content='text/html;
charset=UTF-8' />
<meta http-equiv="Generator" content="log1_cms" />
<meta name="description" content="loglcms official page"/>
<meta name="keywords" content="log1, log 1, CMS, content
managment system" />
<link href="templates/dynamic/css/style.css"
rel="stylesheet" type="text/css" media="screen" />
<link href="templates/dynamic/css/picasa.css"
rel="stylesheet" type="text/css" />
<!--[if IE]>
<script
src="http://html5shiv.googlecode.com/svn/trunk/html5.js">
</script>
<![endif]-->
<script type='text/javascript'
src='http://ajax.googleapis.com/ajax/libs/jquery/1.4/jquery
.min.js'></script>
<script type='text/javascript'
src='templates/dynamic/js/jquery.ba-hashchange.min.js'>
</script>
<script type='text/javascript'
```

processed 20 packets; 1 new event(s) | Show Reconstruction Log

admin | English (United States) | GMT-06:00 | Send Us Feedback | 10.5.2.0.22126-5

Hex View - Get a hexidecimal view, similar to a hex editor, of session content.

Event Reconstruction

service	id	type	source	destination	service	first packet time
Concentrator	16102575	Network Session	192.168.0.51 : 33205	216.34.181.96 : 80	80	2016-11-02T17:54:00.218

Request & Response | Top To Bottom | View Hex | Actions | Open Event in New Tab | Cancel

Request

```
00000000 : 47 45 54 20 2f 69 6e 64  65 78 2e 70 68 70 3f 71   [ GET /ind ex.php?q ]
00000016 : 3d 34 20 48 54 54 50 2f  31 2e 31 0d 0a 48 6f 73   [ =4 HTTP/ 1.1..Hos ]
00000032 : 74 3a 20 6c 6f 67 31 63  6d 73 2e 73 6f 75 72 63   [ t: log1c ms.sourc ]
00000048 : 65 66 6f 72 67 65 2e 6e  65 74 0d 0a 55 73 65 72   [ eforge.n et..User ]
00000064 : 2d 41 67 65 6e 74 3a 20  4d 6f 7a 69 6c 6c 61 2f   [ -Agent:  Mozilla/ ]
00000080 : 35 2e 30 20 28 58 31 31  3b 20 55 62 75 6e 74 75   [ 5.0 (X11 ; Ubuntu ]
00000096 : 3b 20 4c 69 6e 75 78 20  78 38 36 5f 36 34 3b 20   [ ; Linux  x86_64;  ]
00000112 : 72 76 3a 33 33 2e 30 29  20 47 65 63 6b 6f 2f 32   [ rv:33.0)  Gecko/2 ]
00000128 : 30 31 30 30 31 30 31 20  46 69 72 65 66 6f 78 2f   [ 0100101  Firefox/ ]
00000144 : 33 33 2e 30 0d 0a 41 63  63 65 70 74 3a 20 74 65   [ 33.0..Ac cept: te ]
00000160 : 78 74 2f 68 74 6d 6c 2c  20 2a 2f 2a 3b 20 71 3d   [ xt/html,  */*; q= ]
00000176 : 30 2e 30 31 0d 0a 41 63  63 65 70 74 2d 4c 61 6e   [ 0.01..Ac cept-Lan ]
00000192 : 67 75 61 67 65 3a 20 65  6e 2d 55 53 2c 65 6e 3b   [ guage: e n-US,en; ]
00000208 : 71 3d 30 2e 35 0d 0a 41  63 63 65 70 74 2d 45 6e   [ q=0.5..A ccept-En ]
00000224 : 63 6f 64 69 6e 67 3a 20  67 7a 69 70 2c 20 64 65   [ coding:  gzip, de ]
00000240 : 66 6c 61 74 65 0d 0a 58  2d 52 65 71 75 65 73 74   [ flate..X -Request ]
00000256 : 65 64 2d 57 69 74 68 3a  20 58 4d 4c 48 74 74 70   [ ed-With:  XMLHttp ]
00000272 : 52 65 71 75 65 73 74 0d  0a 52 65 66 65 72 65 72   [ Request. .Referer ]
00000288 : 3a 20 68 74 74 70 3a 2f  2f 6c 6f 67 31 63 6d 73   [ : http:/ /loglcms ]
00000304 : 2e 73 6f 75 72 63 65 66  6f 72 67 65 2e 6e 65 74   [ .sourcef orge.net ]
00000320 : 2f 0d 0a 43 6f 6e 6e 65  63 74 69 6f 6e 3a 20 6b   [ /..Conne ction: k ]
00000336 : 65 65 70 2d 61 6c 69 76  65 0d 0a 0d 0a -- -- --   [ eep-aliv e....    ]
```

Response

```
00000000 : 48 54 54 50 2f 31 2e 31  20 32 30 30 20 4f 4b 0d   [ HTTP/1.1  200 OK. ]
00000016 : 0a 53 65 72 76 65 72 3a  20 41 70 61 63 68 65 2f   [ .Server:  Apache/ ]
00000032 : 32 2e 32 2e 31 35 20 28  43 65 6e 74 4f 53 29 0d   [ 2.2.15 ( CentOS). ]
00000048 : 0a 56 61 72 79 3a 20 48  6f 73 74 2c 20 41 63 63   [ .Vary: H ost, Acc ]
00000064 : 65 70 74 2d 45 6e 63 6f  64 69 6e 67 2c 20 55 73   [ ept-Enco ding, Us ]
00000080 : 65 72 2d 41 67 65 6e 74  0d 0a 43 61 63 68 65 2d   [ er-Agent ..Cache- ]
00000096 : 43 6f 6e 74 72 6f 6c 3a  20 6d 61 78 2d 61 67 65   [ Control:  max-age ]
00000112 : 3d 31 37 32 38 30 30 0d  0a 45 78 70 69 72 65 73   [ =172800. .Expires ]
00000128 : 3a 20 54 68 75 2c 20 31  32 20 4d 61 72 20 32 30   [ : Thu, 1 2 Mar 20 ]
00000144 : 31 35 20 31 31 3a 32 38  3a 33 31 20 47 4d 54 0d   [ 15 11:28 :31 GMT. ]
00000160 : 0a 43 6f 6e 74 65 6e 74  2d 54 79 70 65 3a 20 74   [ .Content -Type: t ]
00000176 : 65 78 74 2f 68 74 6d 6c  0d 0a 43 6f 6e 74 65 6e   [ ext/html ..Conten ]
```

processed 20 packets; 1 new event(s) | Show Reconstruction Log

Packets View - See packets in their raw form, similar to how they would be shown in WireShark.

Event Reconstruction

service	id	type	source	destination	service	first packet time
Concentrator	16102575	Network Session	192.168.0.51 : 33205	216.34.181.96 : 80	80	2016-11-02T17:54:00.218

Request & Response | Top To Bottom | View Packets | Actions | Open Event in New Tab | Cancel

Request

```
Packet 1   (id = 463518588  seq = 3369840175)   2016-11-02 23:54:00.218   (0 Payload Bytes)

00000000 : 78 54 2e 9f 10 28 ec f4  bb 4f b2 45 08 00 45 00   [ xT...(.. .O.E..E. ]
00000016 : 00 3c c9 c4 40 00 40 06  22 99 c0 a8 00 33 d8 22   [ .<..@.@. "....3." ]
00000032 : b5 60 81 b5 00 50 c8 db  ae 2f 00 00 00 00 a0 02   [ .`...P.. ./...... ]
00000048 : 72 10 25 3c 00 00 02 04  05 b4 04 02 08 0a 01 59   [ r.%<.... .......Y ]
00000064 : 67 eb 00 00 00 00 01 03  03 07 -- -- -- -- -- --   [ g....... .. ]
```

Response

```
Packet 2   (id = 463518597  seq = 3194932677)   2016-11-02 23:54:00.219   (0 Payload Bytes)

00000000 : ec f4 bb 4f b2 45 78 54  2e 9f 10 28 08 00 45 00   [ ...O.ExT ...(..E. ]
00000016 : 00 3c 2b 40 40 00 f0 06  11 1d d8 22 b5 60 c0 a8   [ .<+@@... ...".`.. ]
00000032 : 00 33 00 50 81 b5 be 6e  cd c5 c8 db ae 30 a0 12   [ .3.P...n .....0.. ]
00000048 : 10 68 7e f2 00 00 02 04  05 78 01 01 08 0a 03 71   [ .h~..... .x.....q ]
00000064 : 7b 80 01 59 67 eb 04 02  00 00 -- -- -- -- -- --   [ {..Yg... .. ]
```

Request

```
Packet 3   (id = 463518598  seq = 3369840176)   2016-11-02 23:54:00.219   (0 Payload Bytes)

00000000 : 78 54 2e 9f 10 28 ec f4  bb 4f b2 45 08 00 45 00   [ xT...(.. .O.E..E. ]
00000016 : 00 34 c9 c5 40 00 40 06  22 a0 c0 a8 00 33 d8 22   [ .4..@.@. "....3." ]
00000032 : b5 60 81 b5 00 50 c8 db  ae 30 be 6e cd c6 80 10   [ .`...P.. .0.n.... ]
00000048 : 72 10 48 9d 00 00 01 01  08 0a 01 59 68 1f 03 71   [ r.H..... ...Yh..q ]
00000064 : 7b 80 -- -- -- -- -- --  -- -- -- -- -- -- -- --   [ {. ]

Packet 4   (id = 463518599  seq = 3369840176)   2016-11-02 23:54:00.220   (349 Payload Bytes)

00000000 : 78 54 2e 9f 10 28 ec f4  bb 4f b2 45 08 00 45 00   [ xT...(.. .O.E..E. ]
00000016 : 01 91 c9 c6 40 00 40 06  21 42 c0 a8 00 33 d8 22   [ ....@.@. !B...3." ]
00000032 : b5 60 81 b5 00 50 c8 db  ae 30 be 6e cd c6 80 18   [ .`...P.. .0.n.... ]
00000048 : 72 10 c5 f1 00 00 01 01  08 0a 01 59 68 1f 03 71   [ r....... ...Yh..q ]
00000064 : 7b 80 47 45 54 20 2f 69  6e 64 65 78 2e 70 68 70   [ {.GET /i ndex.php ]
00000080 : 3f 71 3d 34 20 48 54 54  50 2f 31 2e 31 0d 0a 48   [ ?q=4 HTT P/1.1..H ]
```

processed 20 packets; 1 new event(s) | Show Reconstruction Log

View Web - Same as best reconstruction for HTTP sessions.

View Mail - Same as best reconstruction for email sessions.

Event Reconstruction

service	id	type	source	destination	service	first packet time
Concentrator	16883673	Network Session	192.168.0.53 : 2370	81.236.55.3 : 25	25	2016-11-03T11:55:41.319

Request & Response | Top To Bottom | Best Reconstruction | Actions | Open Event in New Tab | Cancel

Request

From: Password Ned <ned.pwned.se@gmx.com>
To: Krusty <krusty.pwned.se@gmail.com>, homer.pwned.se@gmx.com
Subject: Re: Krusty, unable to deliver your item, #00000529832
more

Sent on 2015-04-07 13:36:29.000

I tried opening the zipped document, but failed for some reason.

Blessings,
Ned
----- Original Message -----
From: Krusty
To: ned.pwned.se@gmx.com ; homer.pwned.se@gmx.com
Sent: Tuesday, April 07, 2015 3:28 PM
Subject: VB: Krusty, unable to deliver your item, #00000529832

Hey guys,

I've received an email from FedEx about a parcel has been shipped, but I haven't sent anything lately.

Have you guys sent anything with FedEx this week?

Warning: Attachments provided in this view contain the original raw unsecured content. It is not recommended to open them directly using the browser because they may contain malicious content.

processed 37 packets; 1 new event(s) | Show Reconstruction Log

View Files - This view allows an Analyst to view and download any files which were embedded within the session. It displays the file's MIME type, and SHA1 and MD5 hashes.

Event Reconstruction

service	id	type	source	destination	service	first packet time
Concentrator	16386478	Network Session	192.168.0.51 : 40038	2.23.144.54 : 80	80	2016-11-02T18:36:49.290

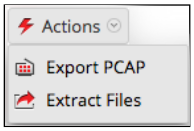
Request & Response | Top To Bottom | View Files | Actions | Open Event in New Tab | Cancel

Filename	Size	Info	File Hashes
16386478-107-0_1.Dell-Precision-M4800-Spec-Sheet.pdf	470,704 bytes	application/p...	MD5: f653f05b2a169edf5a448eda978d9a96 SHA1: fe40e584cce911b37a8a194c3b357f22d8e99c50

Warning: Files provided in this view contain the original raw unsecured content. It is not recommended to open them directly using the browser because they may contain malicious content.

1 file(s) found in event

Open Event in New Tab - If you opened event recon by double-clicking on a session in the Events view, as opposed to opening a session ID directly, you will have this button which will open the current event recon window in a new tab.



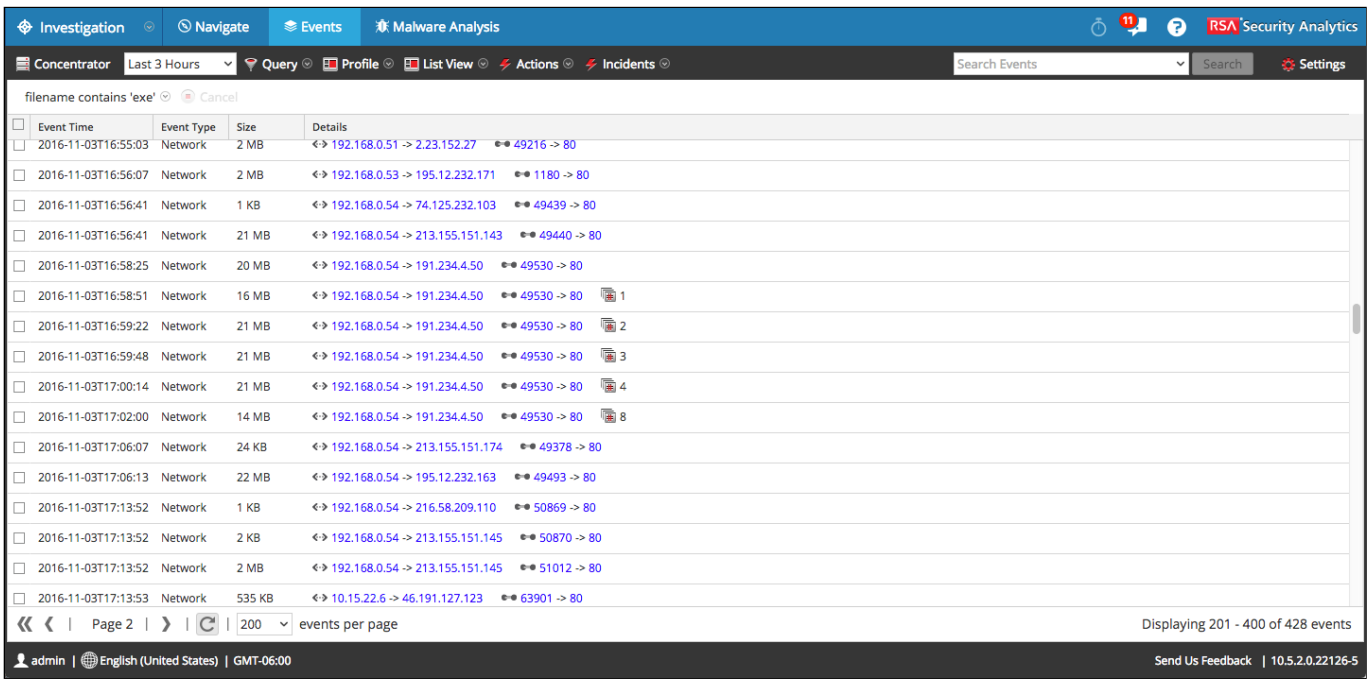
Export PCAP - Downloads a PCAP file of the current session

Extract Files - Extracts all files in the current session for download as a zip file.

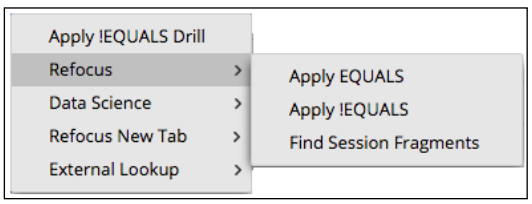
Fragmented Sessions

There is a maximum session size in SA, defined by the API node **/decoder/config/assembler.size.max**. When a given session exceeds this size, a new session is created, with certain meta values transferring from the old session to the new session, like source and destination IP address and service type. Each session part under this condition is called a **session fragment**. You know whether a session is fragmented in the Events view by the presence of a Fragmented Sessions icon next to the session, which look like this: 1.

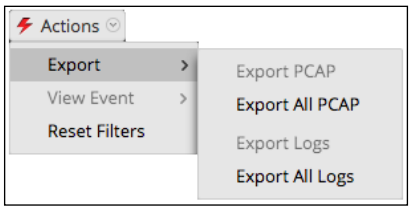
Each number next to the fragment icon indicates which fragment it is. **The first session fragment will not have a number.**



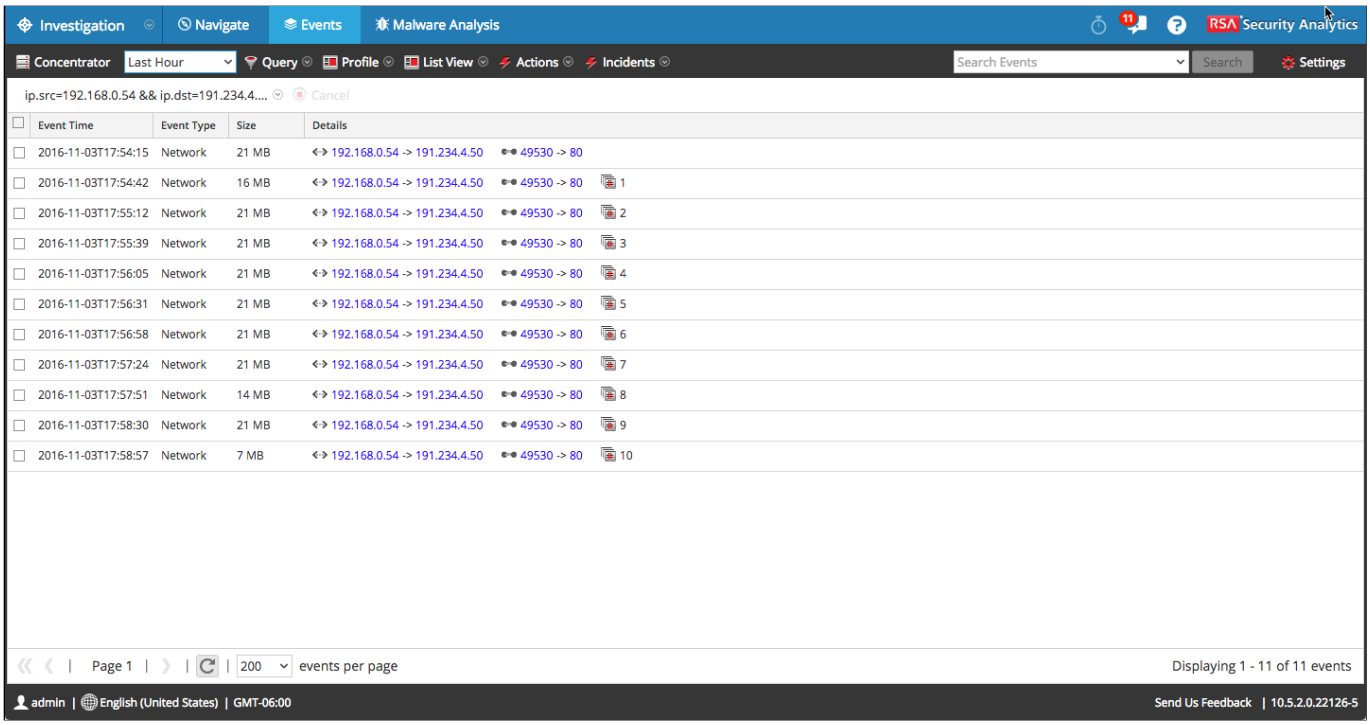
To export all fragments of a session as a single session PCAP file, first right-click on either of the IP addresses, then **Refocus -> Find Session Fragments.**



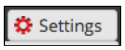
Now all fragments of the session you refocused on should be shown. Simply click **Actions-> Export -> Export All PCAP**, and all the session fragments will be saved as a single pcap file. The jobs panel will give you access to the file, or if you've chosen to auto-download completed extractions in your Investigation settings, this will happen automagically.



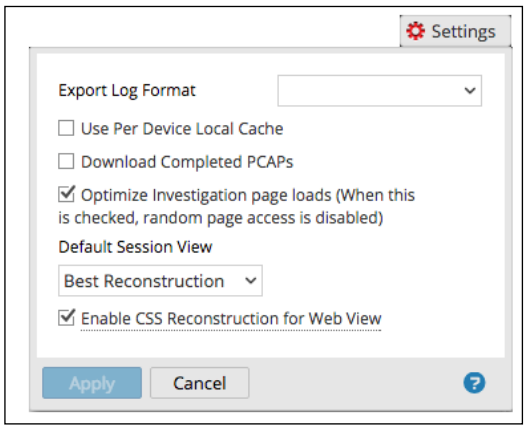
Top Tip: If refocusing doesn't reveal all session fragments, it may be necessary to adjust the time window to encompass all fragments of a session.



Events Settings Panel

Just as for the Navigate view, there is also a 

panel for the Events view. You can set:

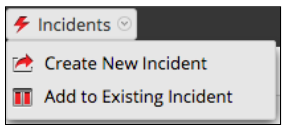


- **Use Per Device Local Cache** - Use the local cache of the service chosen in the service selector rather than the UI's own cache.
- **Download Completed PCAPs** - Automatically download completed PCAP and Log files which have been exported in Investigation.
- **Optimize Investigation Page Loads** - A pagination option, which disabling provides the ability to jump to a specific page by keying in the page number, but in so doing, speed in loading pages of sessions is sacrificed.
- **Default Session View** - Chooses which view to load when entering session reconstruction.
- **Enable CSS Reconstruction for Web View** - This option controls whether session reconstruction should utilise CSS styles and stylesheets when rendering a web page. Disable this if you have trouble reconstructing certain web pages.

Top Tip: The settings panel can also be found in your user profile. Click **Start->Profile** and then the **Investigation** tab. You will find a few other settings here, as well.

Creating/Adding to Incidents

Though this course doesn't cover SA's Incident Management (IM) component in much detail, the Events view allows one to create a new Incident from selected sessions, or add them to an existing Incident. Using the tickboxes to the left of each session, just select the sessions you want to create an incident with or add to an existing incident and click the appropriate button from the Incidents dropdown.



11. SA Core Internals

SA Core Databases

Decoder, Log Decoder, Concentrator, Broker, and Archiver all use three types of proprietary databases to store data captured and generated from captured data. Every SA Core database:

- Is row-based - Every record is stored in a new row
- Is sequential - Each row has an integer identifier, starting at 1, and increasing sequentially by 1 for every record. The row number is called the 'id'.
- Is first in, first out - When a database has reached its maximum capacity, the oldest records in the respective database will be rolled out to make room for new records. This is called **rollout**.
- Is stored in one or more directories on the host filesystem.
- Can be extended by adding additional storage at a given mount point on the host filesystem, and then passing that mount point to the SA Core service
- Is stored internally in UTC time. The OS timezone doesn't influence this, so long as the host clock is set correctly for its configured timezone.

Packet DB - The packet database stores all packets or logs captured by either type of Decoder. There is a log database, but it isn't used to store logs captured by Log Decoder. Each packet or log message comprises one row in the database.

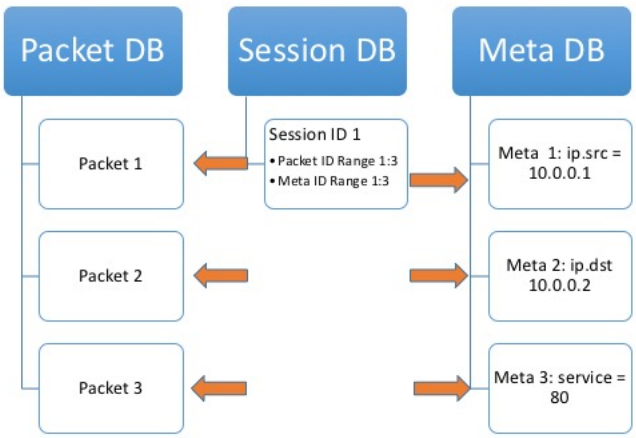
Meta DB - The meta DB stores all meta generated by Decoder's / Log Decoder's parsers, feeds, and rules. Each row stores one key/value pair, such as "ip.src = 10.1.2.3".

Session DB - The session database serves as the glue that keeps packet rows and meta rows associated with specific sessions. As you'll recall, a session is a collection of packets and meta that represent a conversation that flows over the network. In the case of logs, a single 'packet', or log message, is associated with every session, as every log message represents a single packet in the database. This means that the session DB of a Log Decoder will be significantly larger than the session DB of a packet Decoder.

Each session, or row, in the session DB contains the following columns:

- The beginning packet ID of the session
- The ending packet ID of the session
- The beginning meta ID of the session
- The ending meta ID of the session

By tracking these characteristics in the session DB, it makes it possible to reconstruct a session with true forensic likeness, and retrieve all meta associated with the session. **Session ID's are used throughout SA to identify sessions.**



A Broker's Databases

Technically, a Broker doesn't have any databases at all, except for an index of sorts, which serves to track the session ID's of its upstream Concentrators and Brokers. It doesn't actually store any meta, packets, or session DB files of its own, though.

Extraneous Databases

SA Core has two other types of databases, but they are not used to hold captured security data.

Logs DB

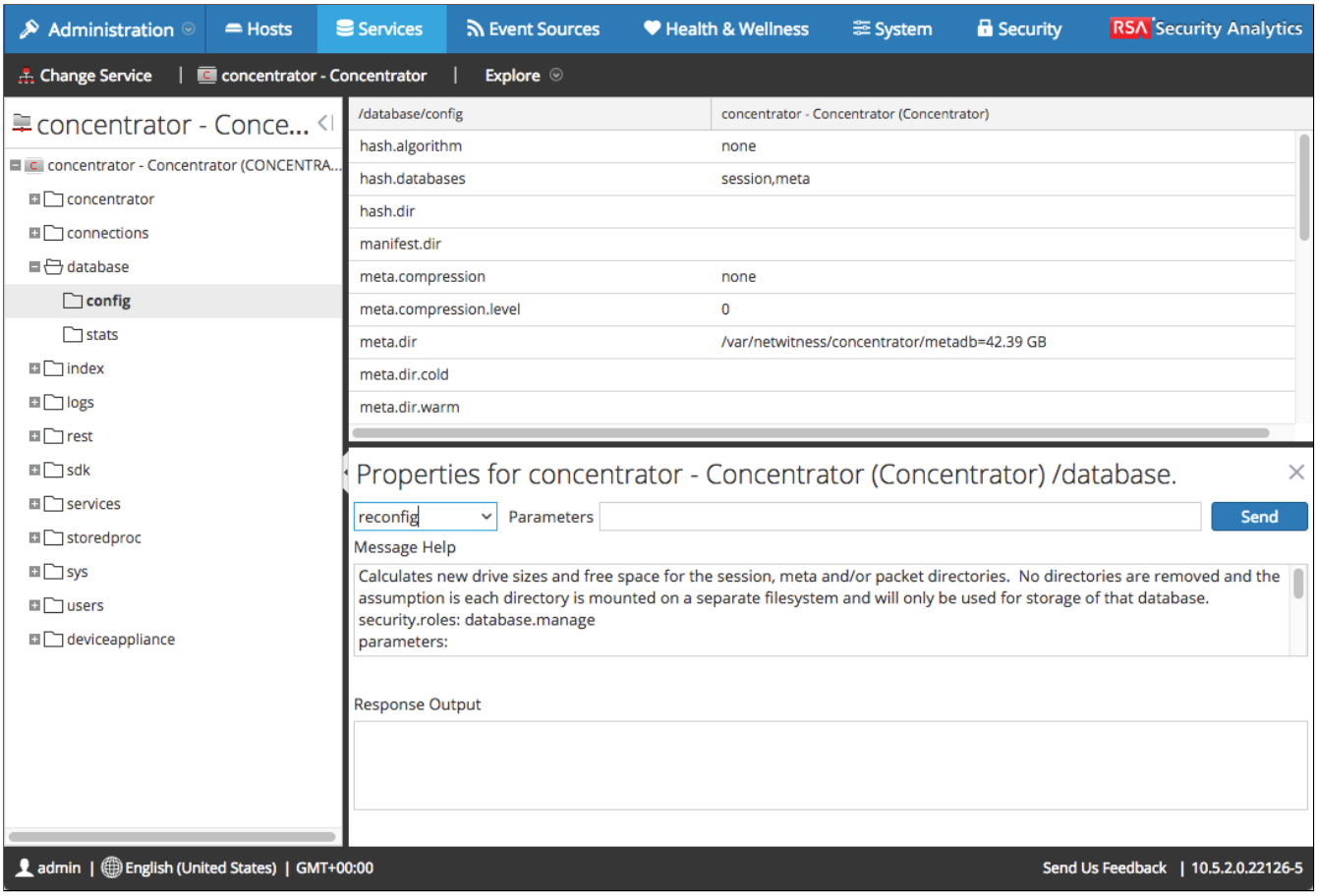
All logs generated by the SA core services are by default sent to two locations - the main system log, via syslog (which show up in /var/log/messages), and in a proprietary log database, located at **/var/log/netwitness/<service>**. For the most part, the logs are the same between them, but there are some logs which do show up in the log DB but do not show up via syslog.

Stats DB

SA Core services generate statistics which can be retrieved at a later time to help baseline or troubleshoot a system. These statistics are stored in the stats database in **/var/netwitness/<service>/statdb**.

The API / Explorer Tree

All



interactions with SA Core are in fact a series of API connections against the service. The API for SA Core is called the NetWitness API. The NetWitness API is organised into a hierarchical tree, with different branches (or nodes) of the tree representing specific realms of configuration and statistics. Nearly every type of SA Core service has branches specific to it, but there are some branches which are universal to all SA Core services. Examples of this would be /connections, /users, /sdk, /sys, and /log. Many Core services also have a /database node.

Many nodes will contain two sub-nodes: **config** and **stats**. 'config' nodes contain configuration items which can usually be changed by an administrator. 'stats' nodes contain statistics specific to that root node. For instance, /database/stats is where you would find statistics pertaining to database size and utilisation, whereas /database/config is where one can make changes to the database configuration.

Viewing the API tree for a particular Core service is as easy as going to the 'Explorer' view of the service in the SA UI.

Once you see the tree, you can also see that each node of the API tree can be interacted with via its 'properties'. To do this, right-click on the node and click properties. A new area will pop out from the bottom of the screen with a drop-down of API commands that can be executed on this node. Simply pick one of the items in the drop-down to see documentation on its usage. Admittedly, this documentation is sometimes lacking and may require some research. I would encourage exercising extreme caution when interacting with the API tree.

In addition to Explorer view, there are a few other ways of interacting with the API tree:

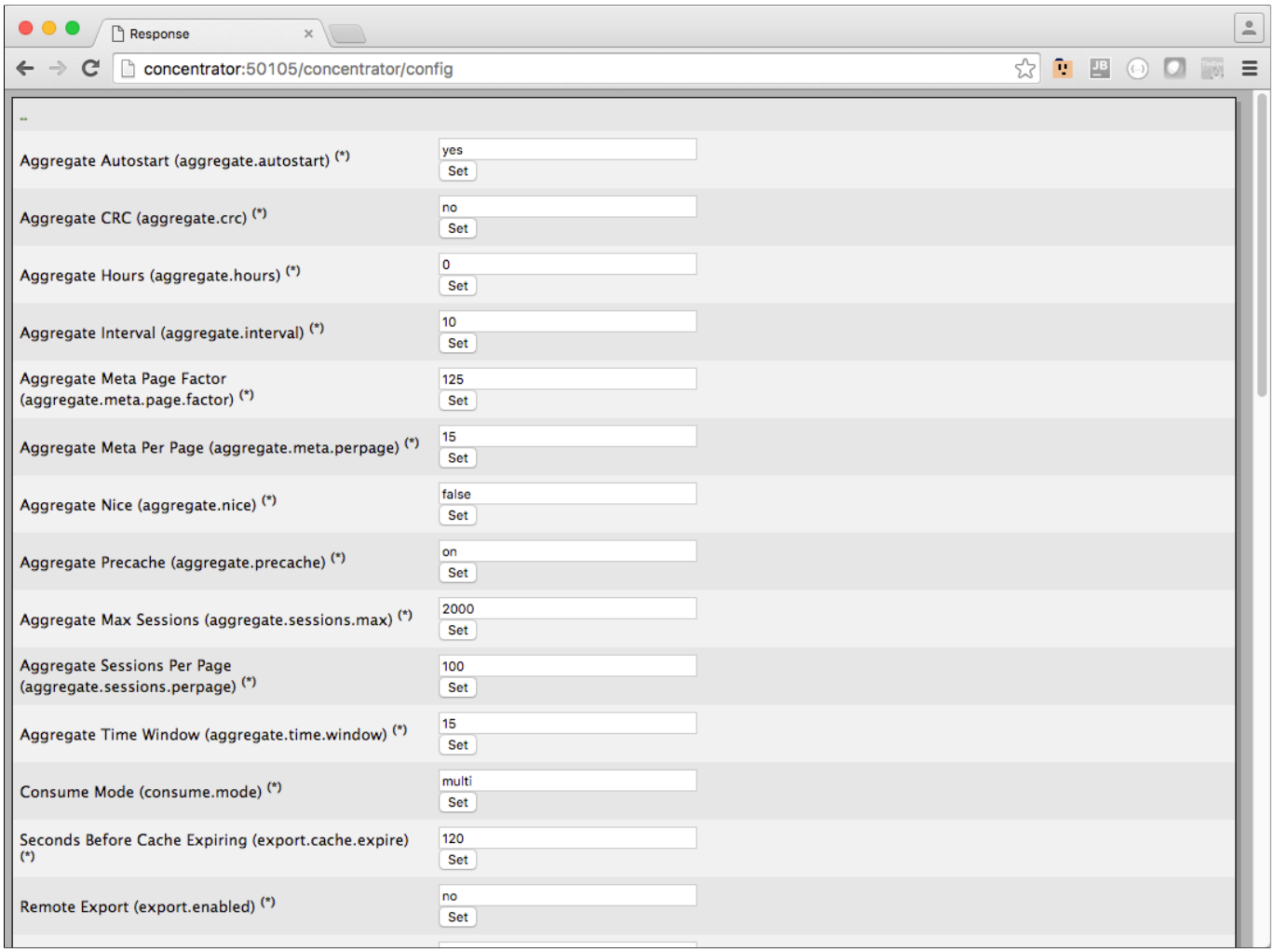
NwConsole - A command line utility which is installed on every SA Core host. Can be used to login to a Core service, navigate through the tree and run API commands through it. NwConsole is sometimes nice when you don't have UI access or convenient UI access, and can also be used within scripts.

```
[[root@concentrator ~]# NwConsole
RSA Security Analytics Console 10.5.2.0.7032
Copyright 2001-2016, RSA Security Inc.  All Rights Reserved.

[> login localhost:50005 admin netwitness
Successfully logged in as session 866
[[localhost:50005] /> cd /concentrator/stats
 [localhost:50005] /concentrator/stats
[[localhost:50005] /concentrator/stats> ls
correlation.results.created (Correlation Results Created) = 0
correlation.results.dropped (Correlation Results Dropped) = 0
export.cache.files (Session Export Cache Files) = 0
export.percent.usage (Export Cache Percent Usage) = 0
export.remote.status (Export Remote Status) = offline
meta.free.pages (Free Meta Pages) = 37500
meta.rate (Meta Rate (current)) = 0
meta.rate.max (Meta Rate (maximum)) = 0
pages.correlate (Correlation Pages Waiting) = 0
pages.evaluate (Evaluation Pages Waiting) = 0
pages.export (Export Meta Pages Waiting) = 0
pages.package (Packaged Meta Pages Waiting) = 0
pages.persist (Database Pages Waiting) = 0
pages.serialize (Serialize Pages Waiting) = 0
pages.stat (Statistic Pages Waiting) = 0
pages.stream (Stream Meta Pages Waiting) = 0
pages.update (Update Meta Pages Waiting) = 0
session.free.pages (Free Session Pages) = 330
session.rate (Session Rate (current)) = 0
session.rate.max (Session Rate (maximum)) = 0
status (Status) = started
[localhost:50005] /concentrator/stats>
```

NwConsole Utility

REST API - Every SA Core service implements a REST API where the NetWitness API tree can be interacted with with an HTTP or HTTPS interface. Also very useful for scripts. It also provides a nice Explorer-like interface you can use when you have no Explorer available interface.

Top Tip: To get the REST interface port number for a given-service, add 100 to the native port number. E.g. for a Decoder's native port of 50004, a Decoder's REST port would be 50104.



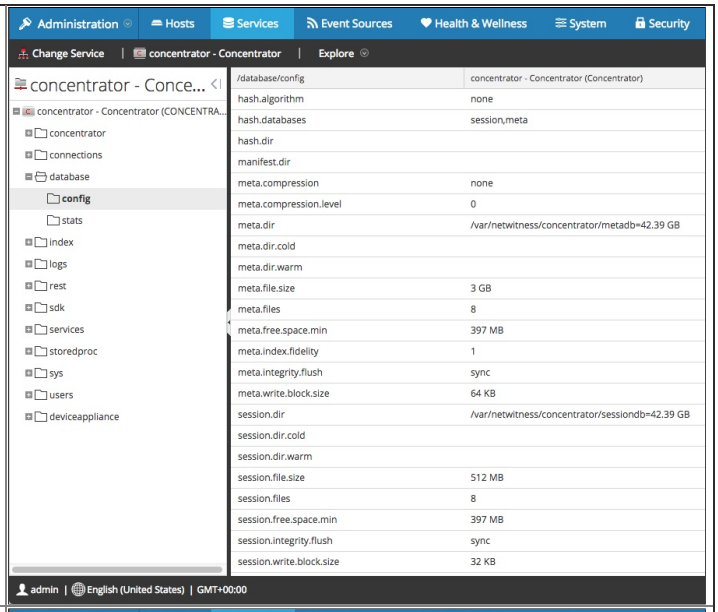
REST API Web Interface

C++ SDK - RSA provides a C++ library which can be used by C++ programs for interacting with SA Core services. Though this will always likely be supported, few customers will use it now in favour of the REST API, unless they have a very specific high-performance use case in mind.

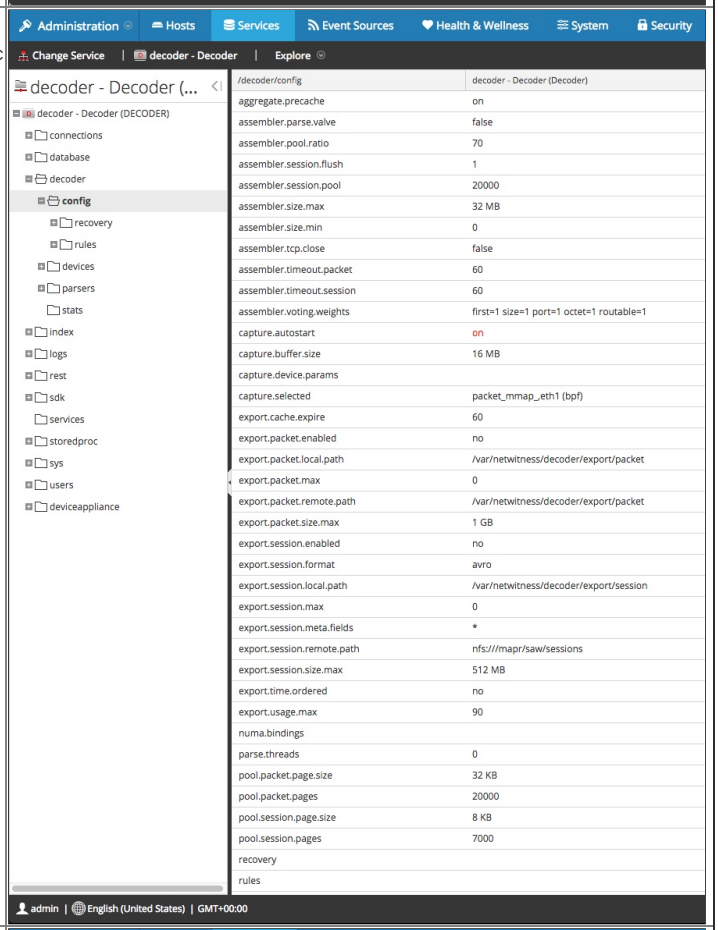
Key API Nodes

These are some of the key SA API nodes which you will see in various SA Core services.

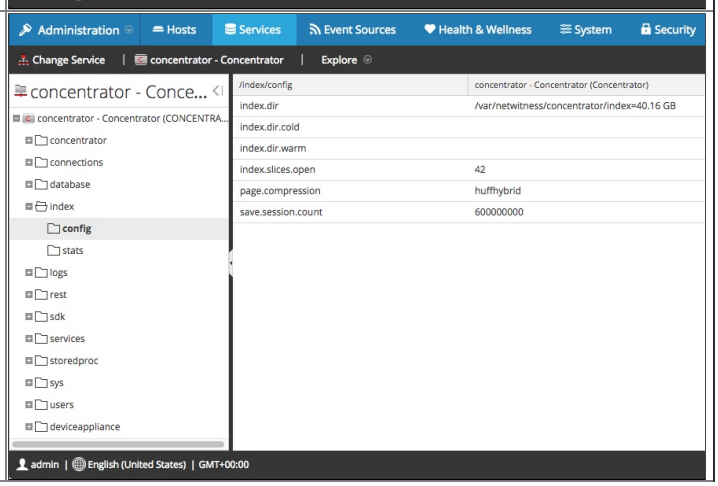
/database - Folder pertaining to NWDB configuration - database sizes and locations, statistics pertaining to NWDB.



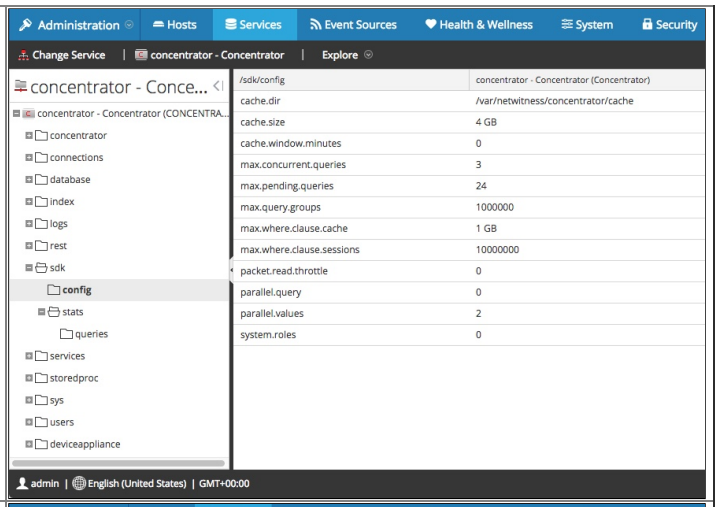
/<serviceType> -Where *<serviceType>* is decoder, concentrator, broker, logdecoder, archiver, etc. Contains configuration and stats nodes specific to the selected type of Core service. For Concentrators and Brokers, this might be nodes concerning aggregation, whereas for Decoders, it would pertain to capture parameters and performance



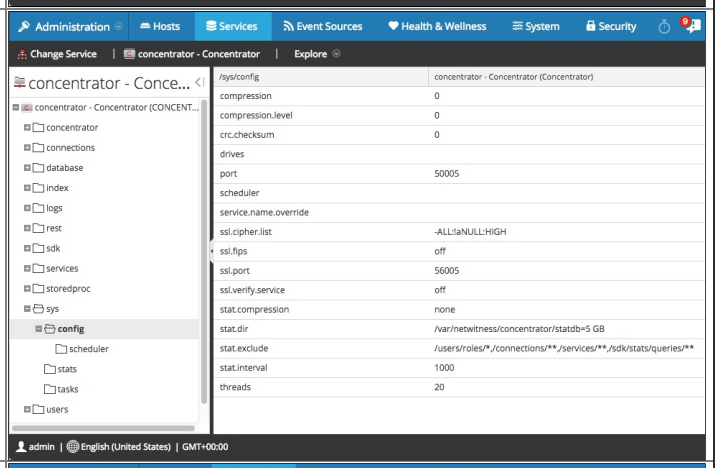
/index - Defines configuration and statistics around the Core index



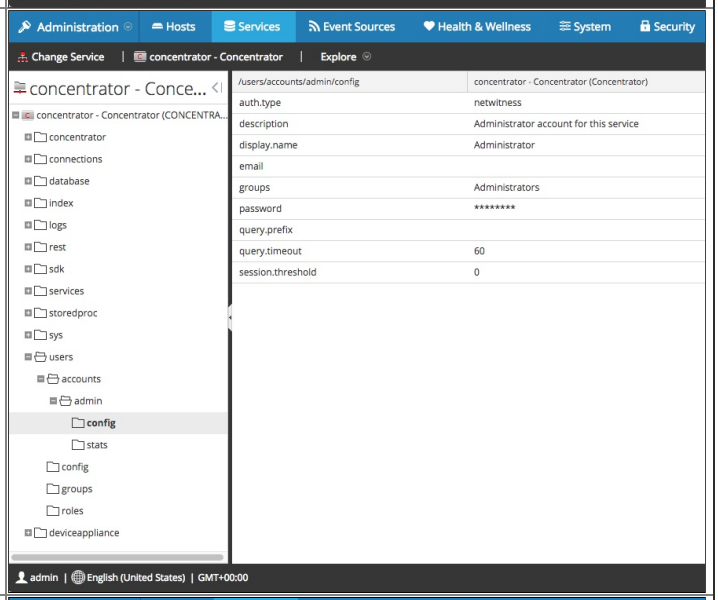
/sdk - The /sdk node controls queries against the service's session, packet, and meta databases, as well as the index. It also defines nodes specific to query performance and time limitations. It also controls certain parameters concerning security



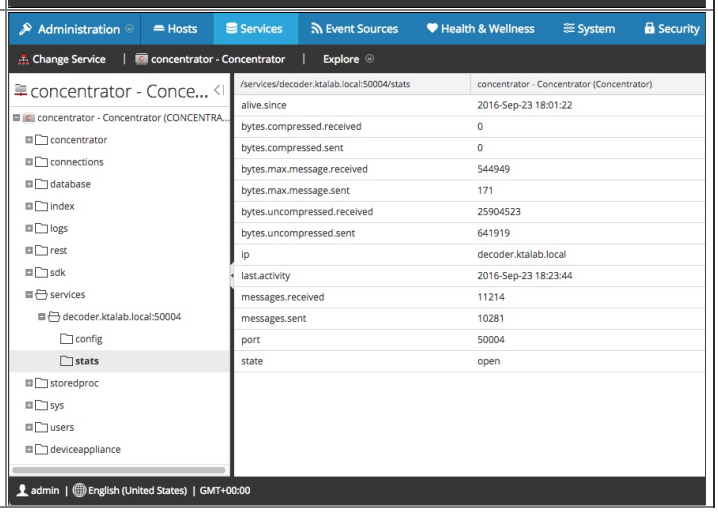
/sys - Defines nodes around low-level system parameters, uptime, kernel, CPU and memory usage, etc. **Rarely ever directly used**

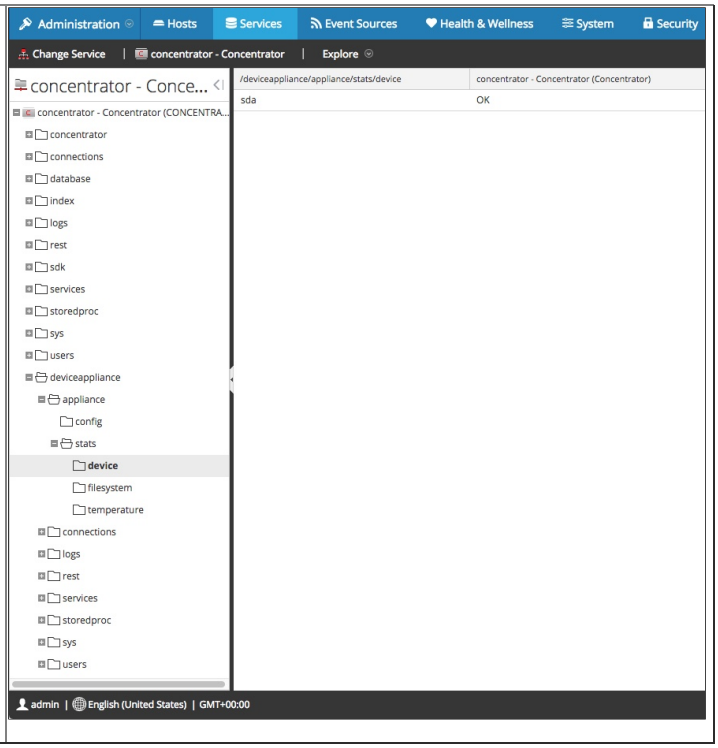


/users - As it should be - defines users and groups



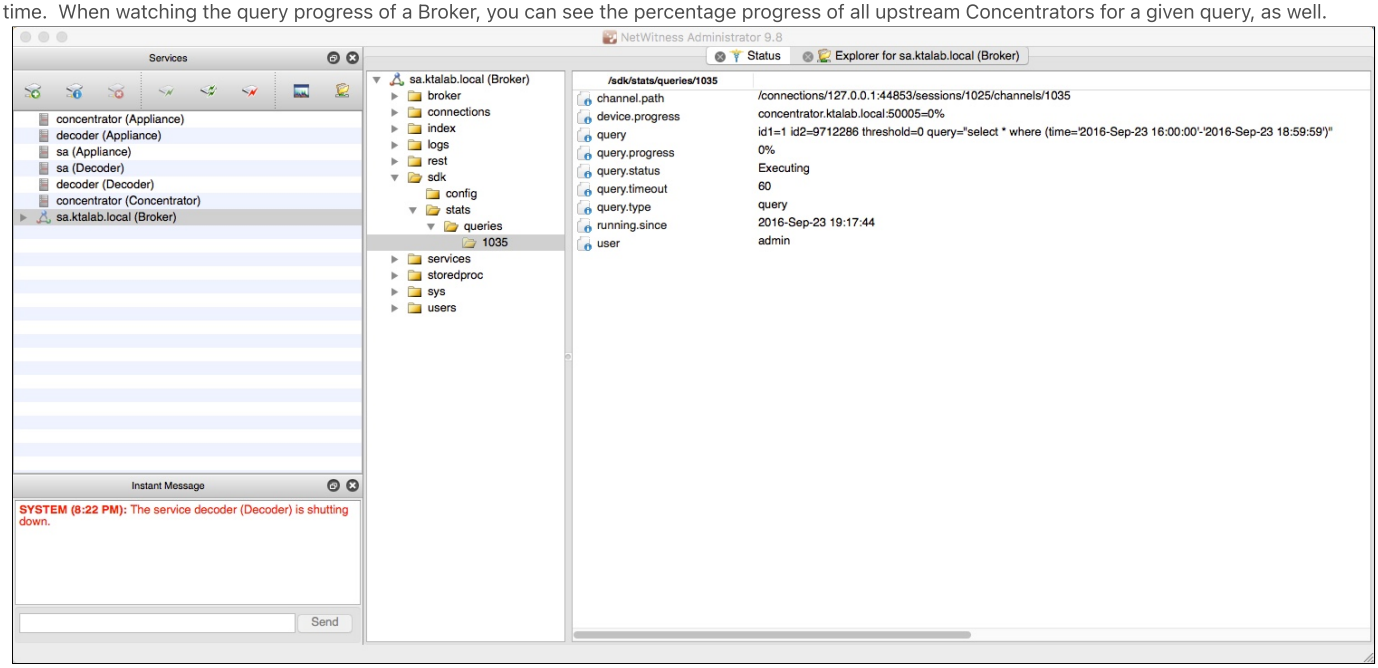
/services - This node represents upstream connected devices from which the current Core service is aggregating



/deviceappliance - An abstract node shown in the UI explorer view which allows the administrator to access the API tree of the legacy Appliance Service installed on the host. It is not shown in the legacy NwAdministrator thick client, in which one would instead connect directly to the legacy Appliance Service	

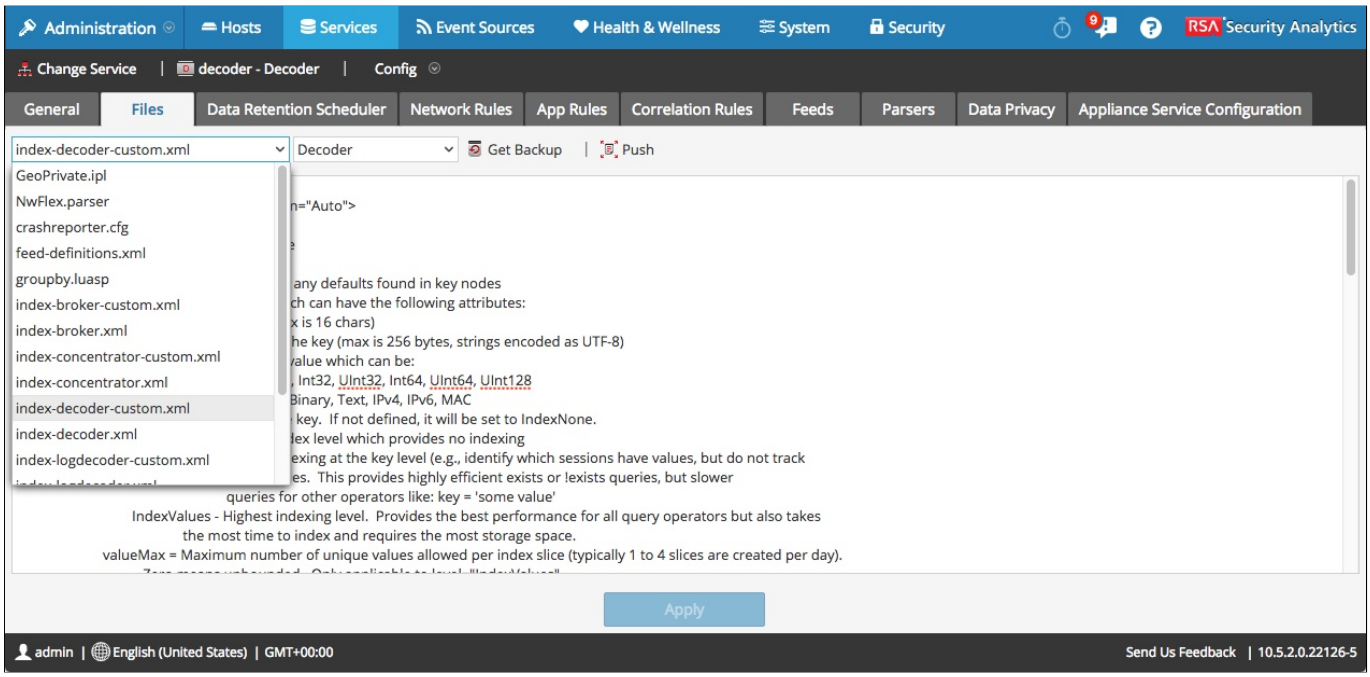
Query Performance

One can watch queries against SA Core in a 'live' fashion by observing the **/sdk/stats/queries** node of the API tree. You will want to use the legacy NwAdministrator thick client to do this, as the thin client does not auto-refresh the explorer tree, but the thick client will show query information in real-time. When watching the query progress of a Broker, you can see the percentage progress of all upstream Concentrators for a given query, as well.



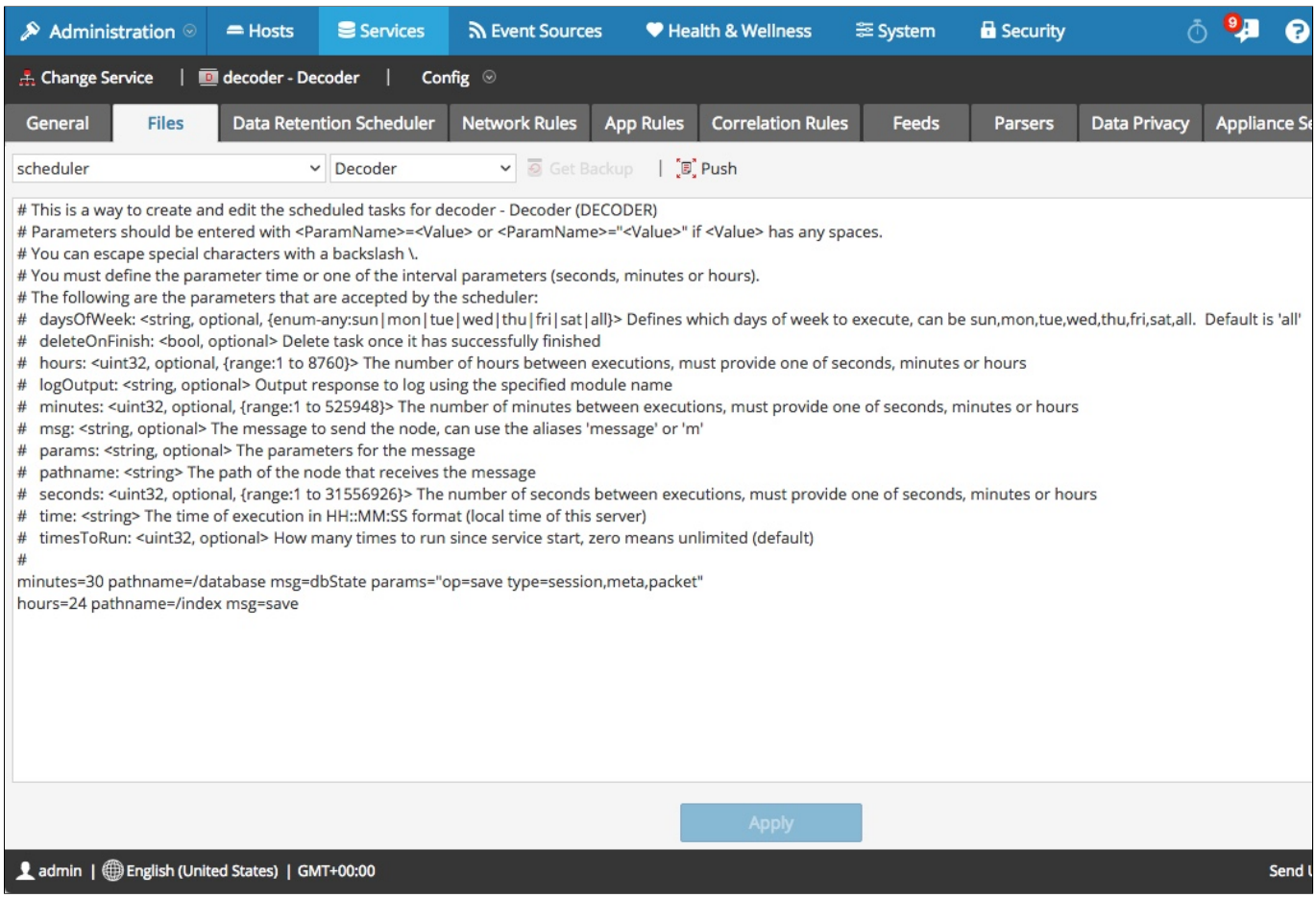
Files

The **Services-> <*ServiceType*> -> Config** page of the UI exposes a tab called '**Files**', in which one can view and edit configuration files associated with the selected service. Note that some of the files shown are not actual files on the filesystem but abstractions of the service's configuration file, such as *scheduler*.



Scheduler

The scheduler is cron-like job scheduling tool which is built-in to every Core service. You can access it from the aforementioned 'Files' section of the Core service config UI. The scheduler runs API commands at specified intervals. Instructions for scheduler parameters are listed in the scheduler text.

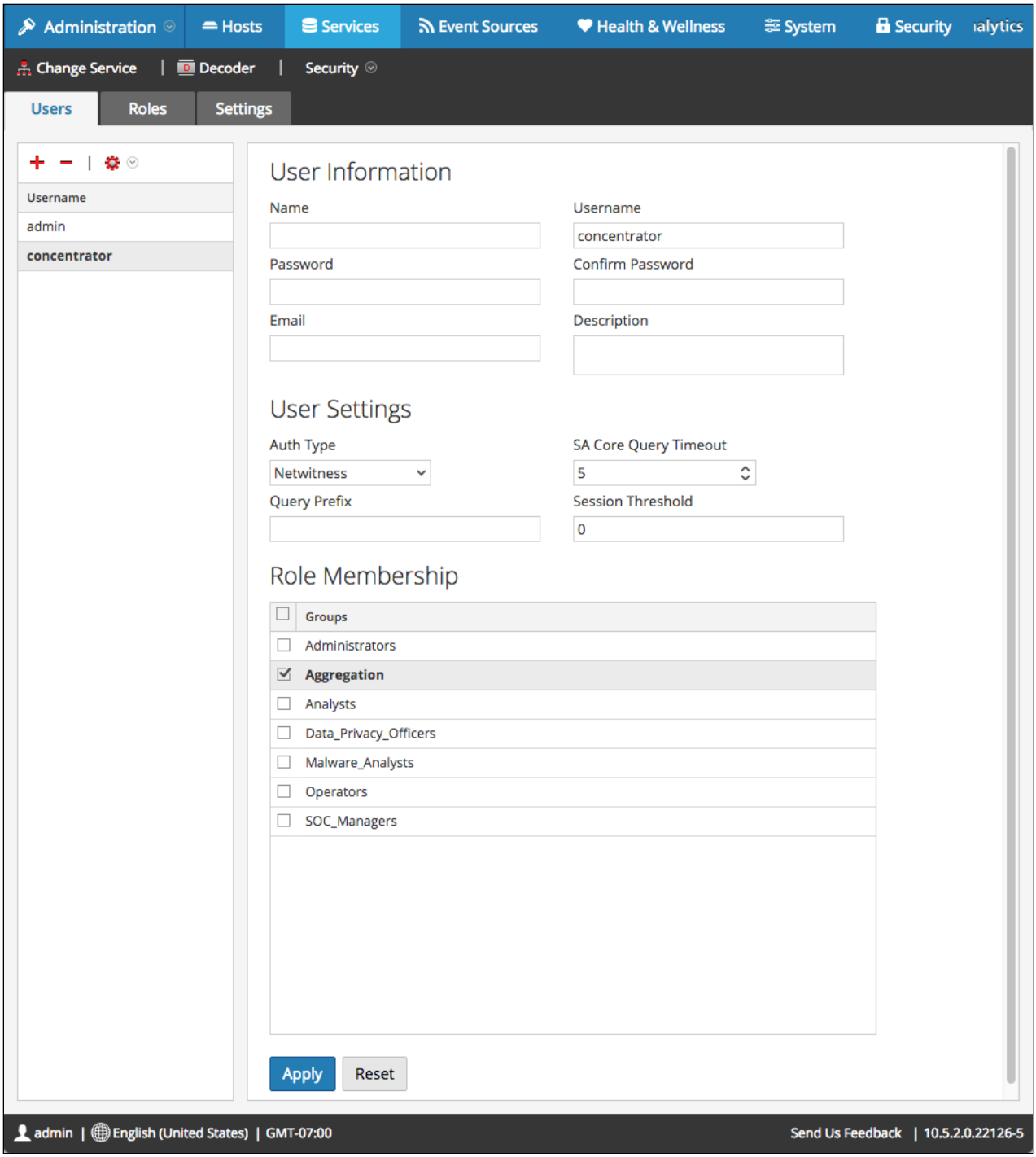


SA Core Security

Users

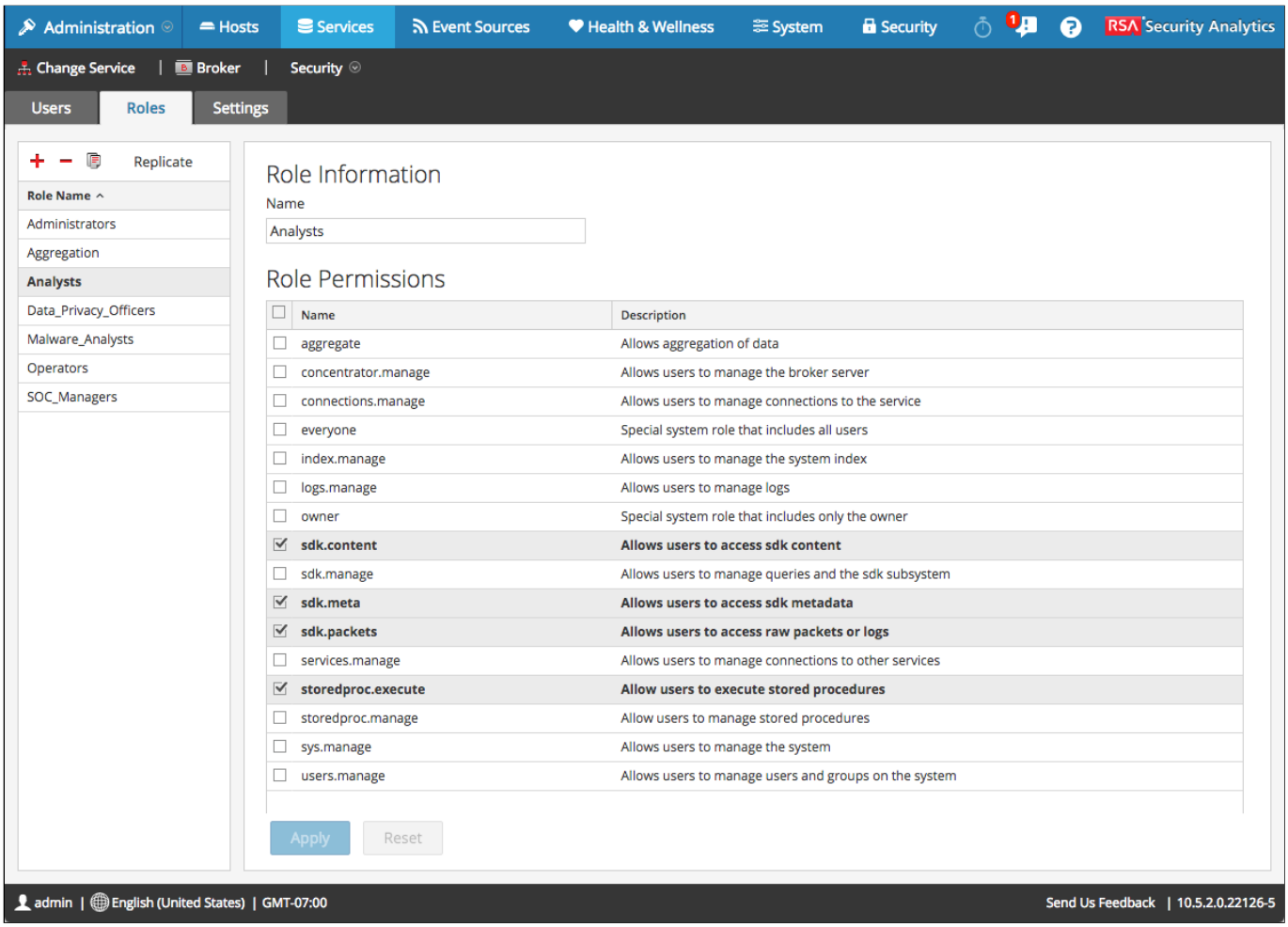
Every interaction with SA Core must be made through an authenticated connection. Each SA Core service manages its own 'database' of users and roles but will also be configured to trust users which are defined within the UI service. Each user has specific properties such as Roles, Authentication Type, and Query Timeout.

It is far less necessary to define users within SA Core than it was prior to SA 10.4. In SA 10.3 and earlier, every user defined in the UI also had to be defined on every individual Core service, with passwords kept manually in sync. As of 10.4, users of the UI no longer must have accounts defined on every Core service, though there are certain circumstances where a locally-defined account is still necessary, such as for aggregation and the reporting engine. This is called the **Trusted Connection Model**. This will be discussed in a later chapter of the training.



Roles

SA Core roles define what a given user is permitted to do during their connected session to the Core service. Each role has individual permissions defined which permit certain operations. For instance, the 'Analyst' role is permitted to perform queries and reconstruct sessions, but is not permitted to make configuration changes. For the 'Operator' role, the user can make configuration changes but not perform queries on customer data.



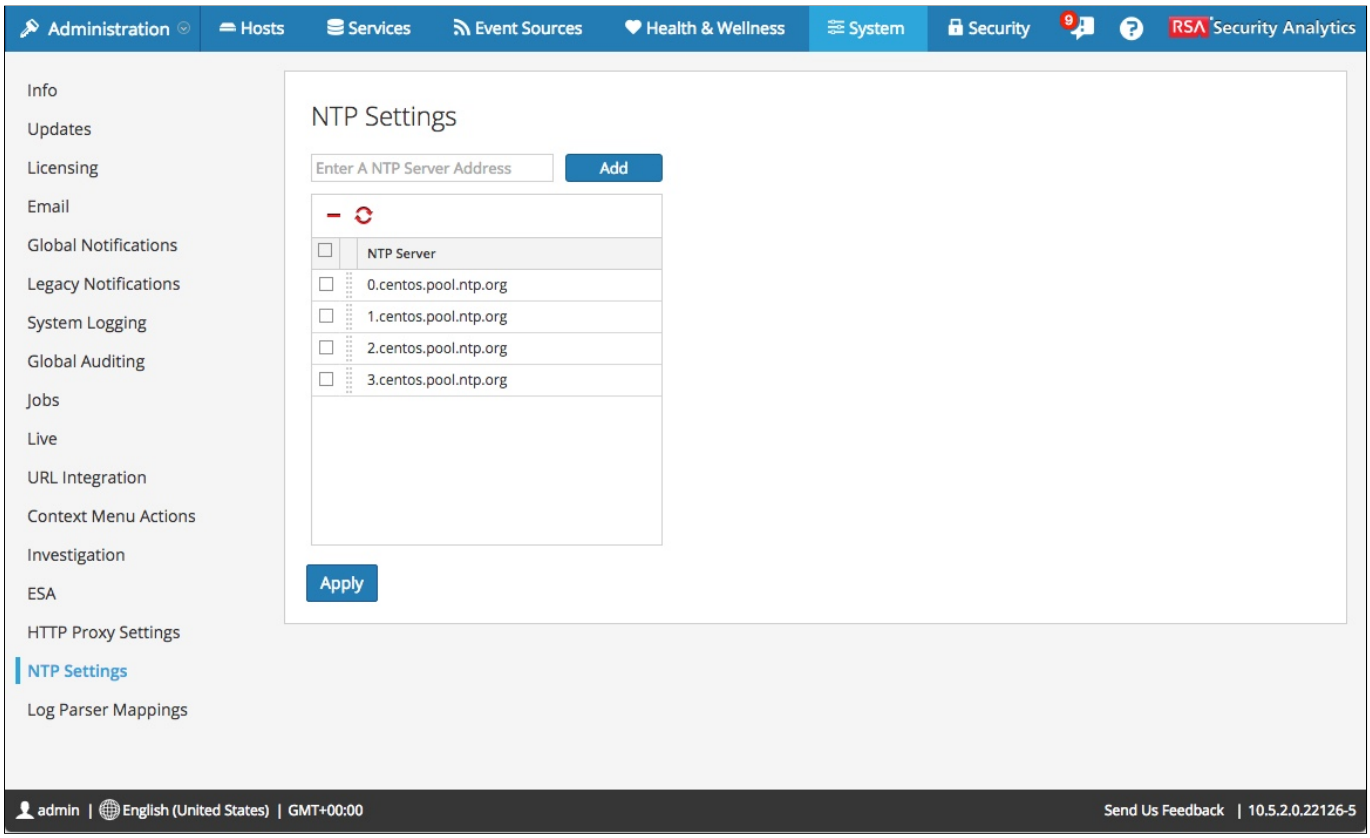
Authentication

SA Core supports two authentication modes for its users:

- **NetWitness** - User password is defined within SA Core, and authentication is performed within the Core service.
- **External** - User password is not defined within SA Core, and authentication of the user is performed by the PAM library, which is provided by the host OS. This can be useful for authenticating users against an external authentication provider, such as Active Directory, though under the trusted connection model it is in practice seldom used now, except for thick client connections or custom SDK connections (e.g. REST API), such those used by scripts.

Host Clocks and NTP

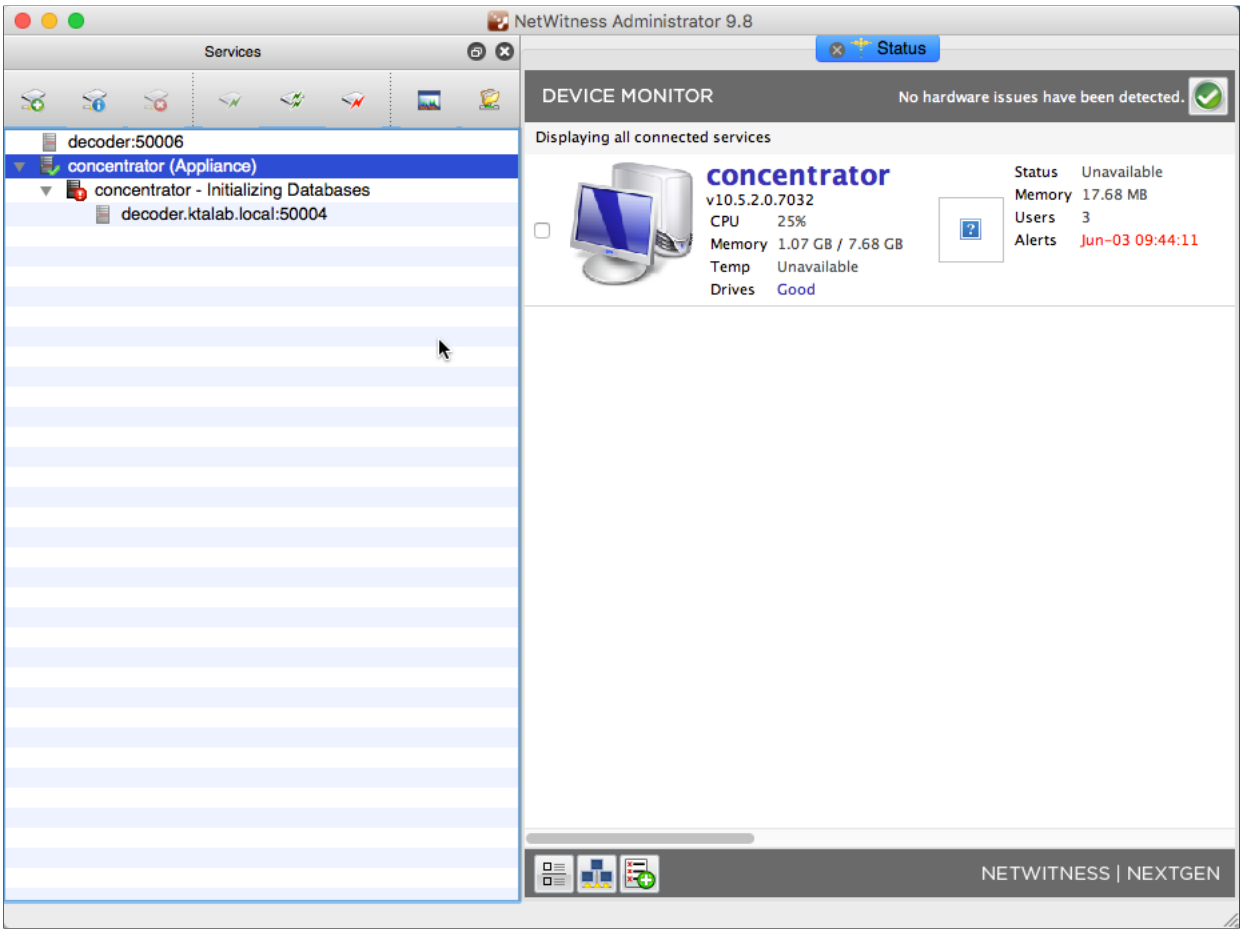
The SA product stack is extremely sensitive to clock synchronisation. In order for captured data to be fully synchronised across the SA stack, their host clocks MUST match. To achieve this, it has always been necessary to take care to configure NTP properly, so that all SA services synchronise to the same NTP time source. Beginning in version 10.5.1, NTP synchronisation for the entire SA deployment is configurable from the UI in **Administration->System->NTP Settings**.



Legacy Thick Clients - NwAdministrator and Investigator

Prior to Security Analytics 10, there was NetWitness 9.8. And it was good. One of the things which made the NetWitness platform great was its reliance on thick client applications to manage it.

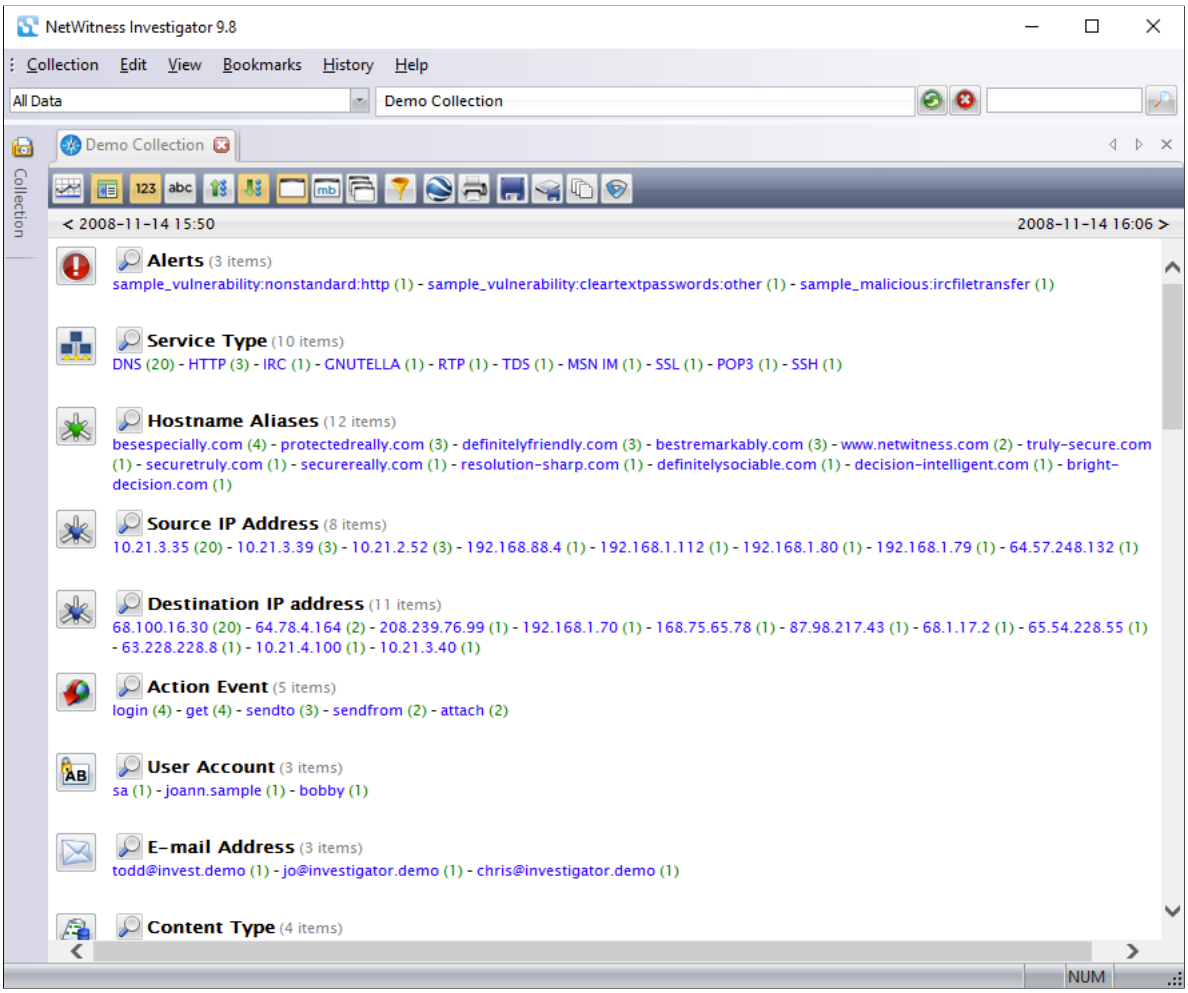
NwAdministrator was the thick client used to manage the NetWitness services. One significant benefit of NwAdministrator over the SA thin client is that explorer view of the thick client has live-update of its nodes and statistics, whereas the thin client requires constant reloading. This makes watching SDK queries in a live manner next to impossible on the thin client, but very easy on the thick client. Our recommendation is to always keep the latest NetWitness 9.8 NwAdministrator client installed on your workstation, in case you should ever need it.



NwAdministrator Thick Client

Investigator was the legacy equivalent to the Investigation module of SA. It allowed for the browsing of and drilling into meta, and the reconstruction of sessions. It was highly useful because it permitted the use of local collections, which were small databases of packet and meta data which were generated by and stored locally within Investigator, which it could generate by processing PCAP files or reading packets off the wire of the local workstation. **Essentially the thick Investigator client is a mini Decoder!** You can even install custom parsers, feeds, application rules, and load RSA Live content, just as you can for a packet Decoder. It was also very handy for custom parser development as it could be done on the local workstation without impacting a production Decoder.

These clients were fairly stable, and they were also lightweight and fast. Some people frankly preferred them and still prefer them for the tasks which they are equipped to do. It's your choice - I won't steer you one way or the other, but you need to know about them. The great thing about them, though, is that the **most recent versions of NwAdministrator and Investigator both work with Security Analytics!**



NetWitness Investigator Thick Client

Key Directories and Files for SA Core

- **/etc/netwitness/ng** - Where SA Core configuration files are stored
- **/etc/netwitness/ng/Nw<Service>.cfg** - Configuration file for SA Core services, where <service> is Decoder, LogDecoder, Concentrator, Broker, Archiver, LogCollector, WarehouseConnector, or Appliance.
- **/etc/netwitness/ng/index-<service>.cfg** -
- **/etc/netwitness/ng/index-<service>-custom.cfg** -
- **/etc/netwitness/ng/feeds** - Directory containing feeds used by Decoder and Log Decoder
- **/etc/netwitness/ng/parsers** - Directory containing LUA parsers used by Decoder. Can also contain legacy FLEX parsers. It is technically possible to use LUA parsers on a Log Decoder but in practice this is rarely, if ever, done.
- **/etc/netwitness/ng/envision** - Log Decoder-specific. Contains enVision parsers used for parsing log content
- **/etc/netwitness/ng/table-map.xml** - Log Decoder-specific. Responsible for mapping meta generated by enVision log parsers onto a SA index keys

- **/var/netwitness/<service>** - Contains non-configuration data of SA Core services, where <service> is decoder, logdecoder, concentrator, broker, archiver, logcollector, warehouseconnector, or appliance.
- **/var/netwitness/<service>/packetdb** - Where packet DB files are stored
- **/var/netwitness/<service>/packetdb<N>** - where N is a number, 0 or greater, representing additional mounted packet DB filesystems, such as those stored on additional DAC's
- **/var/netwitness/<service>/metadb** - Where meta DB files are stored
- **/var/netwitness/<service>/metadb<N>** - where N is a number, 0 or greater, representing additional mounted meta DB filesystems, such as those stored on additional DAC's
- **/var/netwitness/<service>/sessiondb** - Where session DB files are stored
- **/var/netwitness/<service>/sessiondb<N>** - where N is a number, 0 or greater, representing additional mounted session DB filesystems, such as those stored on additional DAC's
- **/var/netwitness/<service>/index** - Where index files are stored
- **/var/netwitness/<service>/cache** - Where SDK cache files are stored
- **/var/netwitness/<service>/rest** - Where REST API cache files are stored
- **/var/netwitness/<service>/statdb** - Where SA Core stat DB files are stored

- **/var/log/messages** - Main system log, where many admins view SA Core service logs
- **/var/log/netwitness/<service>** - Contains SA Core log database files

Starting and Stopping SA Core Services

For the most part, SA uses the Upstart system daemon that is packaged with CentOS 6. Occasionally some SA services will use the legacy System V init scripts but this does not apply to SA Core.

To start an SA Core service: ***start nw<service>***

To stop an SA Core service: ***stop nw<service>***

To restart an SA Core service: ***restart nw<service>***

To check the status of an SA Core service: ***status nw<service>***

Upstart configuration files are stored in **/etc/init**

```
[[root@concentrator ~]#
[[root@concentrator ~]# stop nwconcentrator
nwconcentrator stop/waiting
[[root@concentrator ~]#
[[root@concentrator ~]# start nwconcentrator
nwconcentrator start/running, process 29677
[[root@concentrator ~]#
[[root@concentrator ~]# status nwconcentrator
nwconcentrator start/running, process 29677
[[root@concentrator ~]#
[[root@concentrator ~]# restart nwconcentrator
nwconcentrator start/running, process 29720
[root@concentrator ~]#
```

Debugging a Core Service From the CLI

For whatever reason, sometimes running a Core service as a service can get in the way of troubleshooting. When this happens, you can stop the service with "stop nwdecoder" and run the service executable directly from the CLI with /usr/sbin/Nw<Service>, i.e. /usr/sbin/NwDecoder. All log output will go to the local console tty, and you can stop the service with Ctrl-C.

```
[[root@sa ~]# stop nwdecoder
nwdecoder stop/waiting
[[root@sa ~]# /usr/sbin/NwDecoder
(i) 2016-Oct-05 17:44:08 [Engine]  RSA Security Analytics Service Copyright 2001-2016, RSA Security Inc.  All Rights Reserved.
(i) 2016-Oct-05 17:44:08 [Engine]  Running decoder in console
(d) 2016-Oct-05 17:44:08 [Engine]  [decoder](7f87dc90f800): Entering ServiceBase::Initialize()
(d) 2016-Oct-05 17:44:08 [Engine]  [decoder](7f87dc90f800): ServiceBase::SetStatus(Stopped, Start Pending)
(i) 2016-Oct-05 17:44:08 [Engine]  RSA Security Analytics Service, Decoder 10.6.1.0 (Jul 12 2016) 64 bit Starting
(d) 2016-Oct-05 17:44:08 [Decoder]  Creating decoder module tree
(i) 2016-Oct-05 17:44:08 [Engine]  Configuration loaded from /etc/netwitness/ng/NwDecoder.cfg
(i) 2016-Oct-05 17:44:08 [Engine]  Initializing OpenSSL 1.0.0-fips 29 Mar 2010
(d) 2016-Oct-05 17:44:08 [Engine]  Creating SSL context
(d) 2016-Oct-05 17:44:08 [Engine]  Using certificate file /etc/netwitness/ng/decoder_cert.pem
(d) 2016-Oct-05 17:44:08 [Engine]  Using private key /etc/netwitness/ng/decoder_cert.pem
(d) 2016-Oct-05 17:44:08 [Engine]  Using dh file /etc/netwitness/ng/decoder_dh2048.pem
(d) 2016-Oct-05 17:44:08 [Engine]  Setting SSL cipher list to -ALL:!aNULL:HIGH
(d) 2016-Oct-05 17:44:08 [Database]  Setting index to use 32 bit offsets
(d) 2016-Oct-05 17:44:08 [NwServerLog]  Setting index to use 32 bit offsets
(d) 2016-Oct-05 17:44:08 [NwServerLog]  Adding directory "/var/log/netwitness/decoder" of max size 1,024 MBs
(i) 2016-Oct-05 17:44:08 [NwServerLog]  Found 15 files (3.88 MB) when loading /var/log/netwitness/decoder of max size 1 GB
(d) 2016-Oct-05 17:44:08 [NwServerLog]  Database Write Cache Size set to 0 B.  Max File Size is 250 MB.
(i) 2016-Oct-05 17:44:08 [NwServerLog]  Database is open with 15 file(s) containing 18,776 objects with ID range 1 to 18,776
(i) 2016-Oct-05 17:44:08 [Engine]  Creating a pool of 20 server threads
(i) 2016-Oct-05 17:44:08 [Thread]  Starting thread: Engine Stats  id: 21376
(i) 2016-Oct-05 17:44:08 [Engine]  Loading module 'decoder'
(i) 2016-Oct-05 17:44:08 [session]  Found 1 files (756 KB) when loading /var/netwitness/decoder/sessiondb of max size 28.47 GB
(i) 2016-Oct-05 17:44:08 [meta]  Found 1 files (32.4 MB) when loading /var/netwitness/decoder/metadb of max size 98.91 GB
(i) 2016-Oct-05 17:44:08 [packet]  Found 6 files (21.68 GB) when loading /var/netwitness/decoder/packetdb of max size 98.94 GB
(i) 2016-Oct-05 17:44:08 [session]  Database is open with 1 file(s) containing 22,767 objects with ID range 1 to 22,767
(i) 2016-Oct-05 17:44:08 [meta]  Database is open with 1 file(s) containing 1,645,274 objects with ID range 1 to 1,645,274
(i) 2016-Oct-05 17:44:08 [packet]  Database is open with 6 file(s) containing 28,582,547 objects with ID range 1 to 28,582,547
(i) 2016-Oct-05 17:44:08 [Database]  Checking session database for invalid packet references beyond trimmed packet id 28582547
(i) 2016-Oct-05 17:44:08 [Engine]  Security Analytics Decoder Server 'sa' is running and listening on port 50004 and SSL port 56004
(F) 2016-Oct-05 17:44:08 [Engine]  SNMP AgentX Master connection is DOWN due to: No such file or directory.  Likely cause: snmpd is disabled or misconfigured.
(i) 2016-Oct-05 17:44:08 [Index]  Using language file found at /etc/netwitness/ng/index-decoder.xml
(i) 2016-Oct-05 17:44:08 [Index]  Using custom language file found at /etc/netwitness/ng/index-decoder-custom.xml
(i) 2016-Oct-05 17:44:08 [Index]  Overriding level for alert from IndexNone to IndexValues
(i) 2016-Oct-05 17:44:08 [Index]  Overriding max values for alert from 0 to 100000
(i) 2016-Oct-05 17:44:08 [Index]  Overriding level for feed.name from IndexNone to IndexKeys
(i) 2016-Oct-05 17:44:08 [Index]  Overriding flags for feed.name from  to keyhidden,keyopen,keyclose
(i) 2016-Oct-05 17:44:08 [Index]  Overriding level for feed.category from IndexNone to IndexKeys
(i) 2016-Oct-05 17:44:08 [Index]  Overriding flags for feed.category from  to keyhidden,keyopen,keyclose
(i) 2016-Oct-05 17:44:08 [Index]  Overriding level for feed.desc from IndexNone to IndexKeys
(i) 2016-Oct-05 17:44:08 [Index]  Overriding flags for feed.desc from  to keyhidden,keyopen,keyclose
(i) 2016-Oct-05 17:44:08 [Index]  Overriding level for service from IndexNone to IndexValues
(i) 2016-Oct-05 17:44:08 [Index]  Overriding flags for service from singleton,aliases,keyopen,keyclose to keyopen,keyclose
(i) 2016-Oct-05 17:44:08 [Index]  Overriding level for alias.host from IndexNone to IndexValues
(i) 2016-Oct-05 17:44:08 [Index]  Overriding description for alias.host from Hostname Alias Record to Hostname Aliases
(i) 2016-Oct-05 17:44:08 [Index]  Overriding max values for alias.host from 0 to 2500000
(i) 2016-Oct-05 17:44:08 [Index]  Overriding flags for alias.host from  to keyauto
```

Removing All Data - Data Reset

Sometimes it is desirable to remove all captured data from an SA Core host. This is called a **data reset**. There are a few ways to go about this. The first is to use the **/<*service*>/reset** API call. E.g. **/concentrator/reset**, **/decoder/reset**, etc. This API call takes the following parameters (taken from the API call message help):

```
Reset data, index, manifests, stats, configuration, or logs for this service. Data automatically deletes index and stats. Service is
automatically restarted.

Example arguments:
data=1 config=1 log=1
This example will reset data, index, logs, and configuration

index=1
This example will reset the index only

manifest=1
This example will delete everything in the manifest directory

security.roles: concentrator.manage
parameters:
 data - <bool, optional> Reset data, automatically resets index (may not be applicable to all services)
 index - <bool, optional> Reset index, only valid if data is not reset (may not be applicable to all services)
 manifest - <bool, optional> Reset all manifests stored in the long term manifest directory (may not be applicable to all services)
 config - <bool, optional> Reset configuration settings to default
 stats - <bool, optional> Reset the stats database
 log - <bool, optional> Reset the log database
 cl - <string, optional> A (cl="name=value name2=value2") set of options for setting the command line on the next restart
```

So if you issue the API call "**/concentrator/reset data=1**", the response output of the call will say something like:

```
You have requested a data reset. This is an unrecoverable operation that will delete all session, meta and packets from this
service. Any services that aggregate from this device will be forced to invalidate all sessions, unless they are also data reset.
Please verify this operation by adding "verify=753155" to the parameters and resubmit your request.
```

Now you need to add the **verify** parameter it gives you (in this case "verify=753155") to your API call and reissue it, so that it now looks like: "**/concentrator/reset data=1 verify=753155**". The data reset will kick off, and should only take a few seconds. The service should restart after this.

Manual Data Reset

One can also perform a data reset from the command line by simply removing the database files:

1. **stop nw<*service*>** (i.e. nwdecoder)
2. **cd /var/netwitness/<*service*>** (i.e. **decoder**)
3. **rm -rf index/* metadb/* sessiondb/* packetdb/* statsdb/*** *(if you have additional storage, you will need to adapt this command to encompass it as well)*
4. **start nw<*service*>**

Index Reset

It is possible to initiate a re-index of a Core service's meta DB by performing an index reset. This is sometimes necessary if a Core service is not returning expected results from queries. In an index reset, all other databases will be preserved, but the index will be rebuilt from existing metadata. This can be done with the "**/<*service*>/reset index=1"**, or from the command line with:

1. **stop nw<*service*>** (i.e. nwconcentrator)
2. **cd /var/netwitness/concentrator**
3. **rm -rf index/***
4. **start nw<*service*>**

Warning: When performing an index reset, it's important to know that the Core service will **not be operational** (i.e. aggregating, servicing queries, or capturing) whilst the reindexing is in progress. If there are many terabytes of meta data, this can mea many hours of downtime. For Decoders, though, an index reset usually only takes a few minutes, as Decoders only index the **time** meta key.

12. The Index

The SA Index is so important to understand that it deserves its own topic.

The Index Purpose

The Security Analytics index is arguably the bit of magic that makes it all work. The ability to capture packets, then sessionise them, parse them, store them and their associated meta, would be almost useless without the ability to quickly dig through all of this data in order to find something useful. The index is what makes it possible to find things quickly in this sea of data. It is not a sequential database in the same sense that the packet, meta, and session DB's are. Any time you issue a query in Investigation or on an NWDB source in the reporting engine, that query is being serviced by the SA index.

The index is found at **/var/netwitness/<*service*>/index**. We will use a Concentrator for most of our examples.

When we issue an Investigation or report query, it is the index which the query is run on, not any of the NWDB databases.

How Does the Index Work?

Black magic.

What Does the Index Store?

For every meta key, the index stores either:

- **Meta values for a given meta key, and the session ID's associated with those values.** This method allows values in a key to be queried very quickly as the values themselves are stored in the index. This is referred to indexing at the value level, or **IndexValues** in the index configuration file. This type of indexing takes up more memory in memory and on disk.

or)

- **Session ID's only for a given meta key**. In this method, no meta values are stored in the index. This is referred to indexing at the key level, or **IndexKeys** in the index configuration file. In this method, any queries using the EXISTS operator will return extremely quickly, but queries using other operators will typically return very slowly, as meta will have to be looked up with two calls: one to the session DB to get the meta ID's for a session, and another call to the meta DB to return the meta. This type of indexing is advantageous for 1) infrequently searched meta keys or for 2) meta keys for which one only wants to know whether meta exists, not what its value is. It also takes less room in memory and on disk.

or)

- **No meta or session ID's for a given key**. With this type of indexing, the given meta key cannot be queried. It can only be displayed if retrieved from a direct call against the meta DB, such as in session reconstruction within Investigation. This is referred to as **IndexNone** or **NoIndex** (both are valid) in the index configuration file.

The index is probably the most memory-intensive part of the SA Core stack. It is also very random-IO in nature, so in most instances an index should be stored in solid state disks, or SSD's, which perform much better for random-IO. The only exception to this is Series 4 or Series 4S Hybrids, which use a spindle drive for the index and are thus considerably slower at servicing queries against the index.

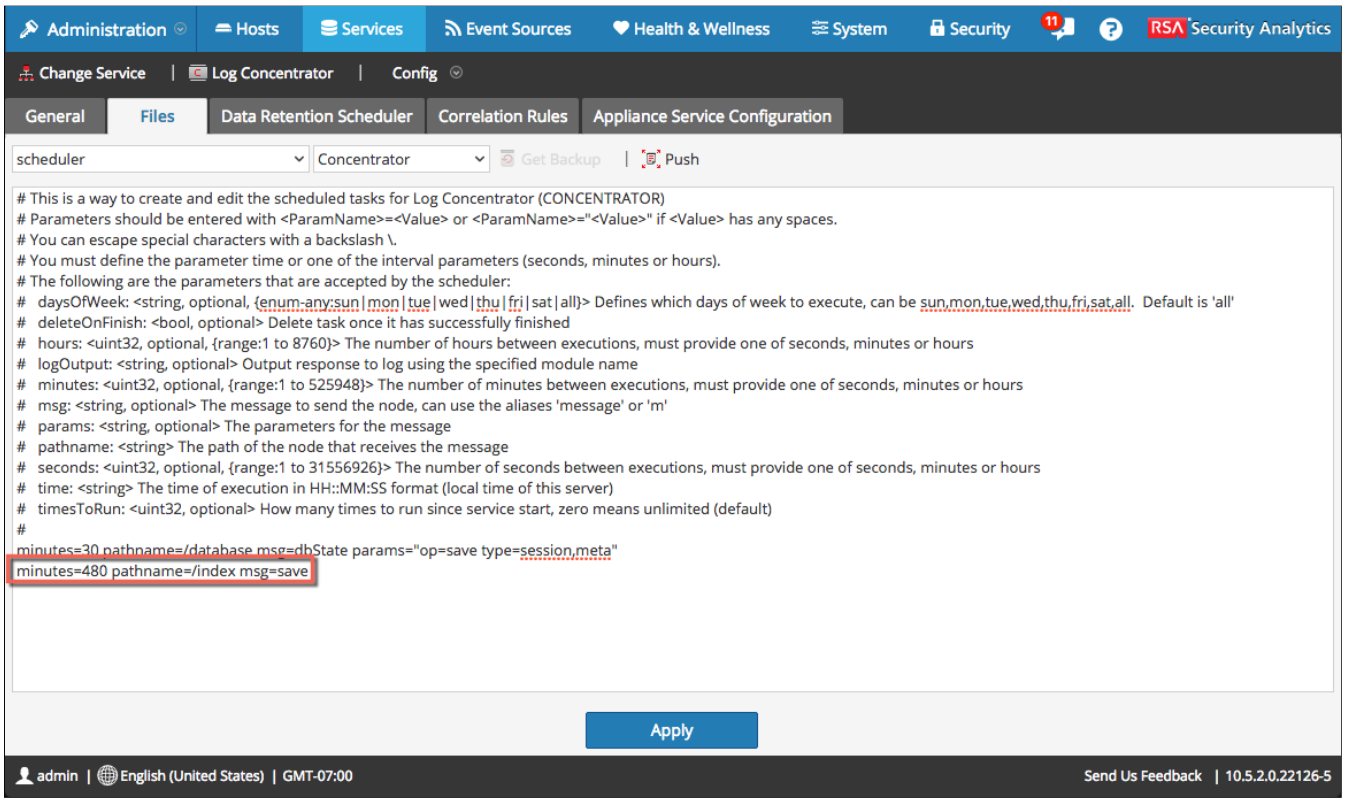
Index Slices

The index is organised into series of slices of time, with each slice representing meta that was generated during that time slice. The purpose of slices is to allow queries to focus on specific periods of time with only having to load into system memory the slice or slices associated with that timeframe, as opposed to having to keep the entire index in memory, which would likely be expensive, swappy, and slow.

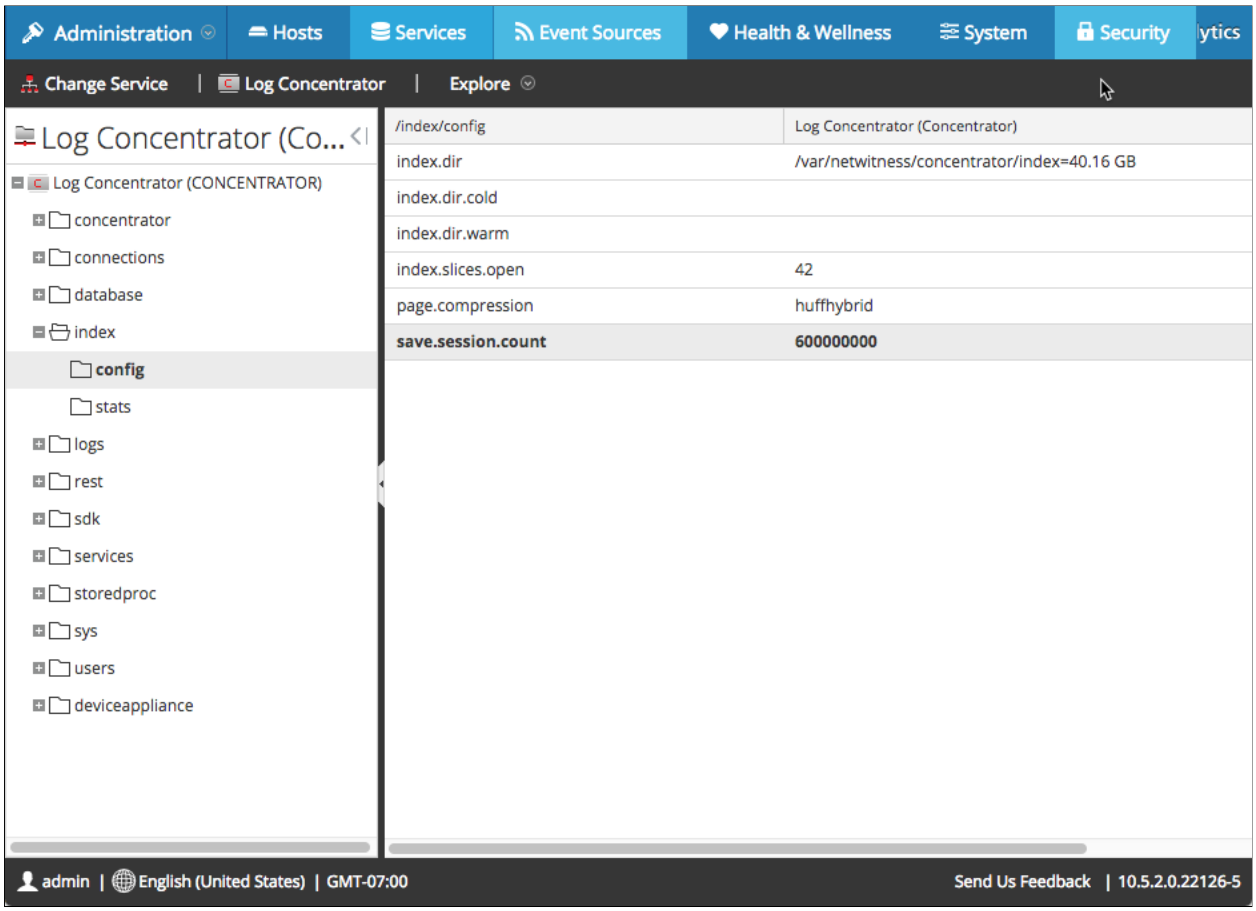
Each index slice is stored in a directory **named managed-values-*N*** under **/var/netwitness/<*service*>/index**, where N is a sequential number, beginning with zero.

A new index slice is created when one of several conditions are met:

- At a specified time interval, which is defined by an SA Core scheduler job, with the API call **/index/save.** In the below



- When the slice contains data for N number of sessions defined in **/index/config/save.session.count**, which defaults to a value of **600000000** (6 million).



Index Structure - Slice Files

```
[root@sa managed-values-0]# ls
access.point.nwindex      client.nwindex        emailfrom.nwindex       found.nwindex           mailfromdomain.nwindex     query.nwindex            summary.db
action.nwindex            content.nwindex       email.nwindex           fromname.nwindex        mailfrom.nwindex           rcpttodomain.nwindex     tcp.dstport.nwindex
ad.computer.dst.nwindex   country.dst.nwindex   email.src.nwindex       fullname.nwindex        managed-values-0.manifest  rcptto.nwindex           tcp.srcport.nwindex
ad.computer.src.nwindex   country.src.nwindex   emailto.nwindex         group.nwindex           match.nwindex              rdp.info.nwindex         threat.category.nwindex
ad.domain.dst.nwindex     crypto.nwindex        email.url.host.nwindex  http.request.nwindex    medium.nwindex             reference.id.nwindex     threat.desc.nwindex
ad.domain.src.nwindex     database.nwindex      emailxmailer.nwindex    http.response.nwindex   msg.id.nwindex             referer.nwindex          threat.source.nwindex
ad.username.dst.nwindex   device.class.nwindex   error.nwindex           ip.addr.nwindex         named.pipe.nwindex         req.uniq.nwindex         time.nwindex
ad.username.src.nwindex   device.group.nwindex   eth.dst.nwindex         ip.dst.nwindex          netname.dst.nwindex        result.code.nwindex      tld.nwindex
agent.ext.nwindex         device.host.nwindex   eth.src.nwindex         ip.dstport.nwindex      netname.src.nwindex        res.uniq.nwindex         udp.dstport.nwindex
alert.id.nwindex          device.ip.nwindex     eth.type.nwindex        ip.proto.nwindex        obj.name.nwindex           risk.info.nwindex        udp.srcport.nwindex
alert.nwindex             device.ipv6.nwindex   event.cat.name.nwindex  ip.src.nwindex          obj.type.nwindex           risk.suspicious.nwindex  user.dst.nwindex
alias.host.nwindex        device.name.nwindex   event.class.nwindex     ipv6.dst.nwindex        org.dst.nwindex            risk.warning.nwindex     username.nwindex
alias.ip.nwindex          device.type.nwindex   event.desc.nwindex      ipv6.proto.nwindex      org.nwindex                search.text.nwindex      user.src.nwindex
alias.ipv6.nwindex        direction.nwindex     event.source.nwindex    ipv6.src.nwindex        org.src.nwindex            server.nwindex           version.nwindex
alias.mac.nwindex         directory.nwindex     event.type.nwindex      ir.alert.nwindex        orig_ip.nwindex            service.nwindex          virusname.nwindex
attachment.nwindex        domain.dst.nwindex    extension.nwindex       ir.desc.nwindex         OS.nwindex                 sinterface.nwindex       vis.level.nwindex
browser.nwindex           domain.src.nwindex    feed.category.nwindex   ir.feed.nwindex         paddr.nwindex              site.cat.nwindex         wlan.channel.nwindex
buddy.nwindex             ec.activity.nwindex   feed.desc.nwindex       ir.general.nwindex      page.db                    sourcefile.nwindex       wlan.ssid.nwindex
bytes.ratio.nwindex       ec.outcome.nwindex    feed.name.nwindex       language.nwindex        parse.error.nwindex        sql.nwindex              word.nwindex
category.nwindex          ec.subject.nwindex    filename.nwindex        lc.cid.nwindex          password.nwindex           ssl.ca.nwindex
city.dst.nwindex          ec.theme.nwindex      filetype.nwindex        link.nwindex            policy.name.nwindex        ssl.subject.nwindex
city.src.nwindex          email.dst.nwindex     filter.nwindex          logon.type.nwindex      process.nwindex            subject.nwindex
```

<keyname>.nwindex - These files generally discrete meta values for a given key, if the key is indexed at the key level. If the key is defined as text-formatted, you can use the **strings** command to see what values are in the file.

```
[root@sa managed-values-0]# strings alias.host.nwindex | head
INDEX_PAGE_SERIALIZED
www.icloud.com.edgekey.net
gmail-imap.l.google.com
imap.gmail.com
imap-mail.outlook.com
imap.glbdns2.microsoft.com
glb3.glbdns2.microsoft.com
glb4.glbdns2.microsoft.com
*.hotmail.com
sb.l.google.com
```

managed-values-N.manifest - A small text file in JSON format which describes the slice name, the first and last session ID's in the slice, and the first and last timestamp in the slice.

```
[root@sa managed-values-0]# cat managed-values-0.manifest
{
    "filename" : "managed-values-0",
    "id1" : 1,
    "id2" : 65801,
    "time1" : 1475718840,
    "time2" : 1475793120
}[root@sa managed-values-0]#
```

summary.db - Stores index summarisations. It isn't really necessary to understand this file.

page.db - Stores index pages. Also not very important to understand; it's part of the index's black magic.

Swap Avoidance

A good point to know is that where a Concentrator is concerned, you will want to try to avoid the service swapping significant amounts of memory to disk at all costs. If this should ever happen, you have either a misconfiguration and/or deployment in need of re-sizing.

Frequently Used vs Infrequently Used Meta Keys

Since meta which is indexed at the value level, or **IndexValues**, generally returns results more quickly, one may be tempted to index all keys at the value level. Resist this temptation, and be judicious. The more meta you index at the value level, the bigger your index will become - it will consume more memory, more disk space, and thus will retain data for less time.

Meta keys which are queried very frequently by Analysts and reports using the **=**, **!=**, **begins**, **ends**, and **regex** comparison operators (i.e. where one is comparing against a given meta value) should probably index that key at the value level. These will take up more index space, but will return results faster.

Conversely, infrequently queried meta keys, especially those keys where the queries use the **exists** and **!exists** comparison operators, should probably index those keys at the key level. These work better with **exists** and **!exists** operators because they don't concern themselves with what value is under the key, only that meta for a given session was registered. It's like asking "give me all sessions that have registered some kind of meta

Controlling One's Index Size - ValueMax

Another control an Administrator has over the side of his index is the **valueMax** key attribute. This attribute is only used on meta keys indexed at the value level. It controls the maximum number of values an index slice will hold for the given key. If the number of values stored in the index slice hits this number, no new values will be indexed for the meta key until a new index slice is created, and there will be a message in the system logfile that looks similar to

```
Nov 6 10:58:36 myconcentrator nw[13345]: [Index] [warning] Index key alias.host has reached max capacity of 2500000 values and will ignore new values for this slice.
```

The Index Configuration File

To configure the index, edit the index-<service>-custom.xml file in **/etc/netwitness/ng**. It is an XML configuration file which defines keys, their indexing values, and also something called aliases. Do NOT edit the index-<service>.xml file directly as any changes will be overwritten by a future upgrade! As you can probably surmise, the index-<service>.xml file contains the default RSA-provided index configuration settings, and the index-<service>-custom.xml file is used for performing overrides and customisations against the default index settings. Specifics on the index configuration file will be covered in a later chapter.

Let's have a look at an example from a Concentrator's **index-concentrator.xml** file, which defines the **ip.src** meta key:

```
<key description="Source IP Address" level="IndexValues" name="ip.src" format="IPv4" valueMax="15000000" />
```

Here are the attributes for this meta key:

- **description**: This is the friendly name for the key that gets displayed in the UI's Investigation module
- **level**: How should this key be indexed? At the value level (**IndexValues**), at the key level (**IndexKeys**), or not at all (**IndexNone**)?
- **name**: This is the true name of the meta key, which will be referenced in rules, parsers, and feeds. For directional meta that refers to the source or destination side of a connection, use the suffix "**.src**" or "**.dst**"
- **format**: The type of meta which the key will store. Valid values here are **Int8**, **UInt8**, **Int16**, **UInt16**, **Int32**, **UInt32**, **Int64**, **UInt64**, **UInt128**, **Float32**, **Float64**, **TimeT**, **Binary**, **Text**, **IPv4**, **IPv6**, or **MAC**. Many and even most meta keys will be of type **Text,** but in our example the meta key stores a standard IPv4 address, so we set ' **format="IPv4"** '.
- **valueMax**: If the key is set to index at the value level (**IndexValues**), this refers to the maximum number of unique meta values which will be stored in a single given index slice. If the index should receive more meta values than **valueMax** in the active index slice, a warning message will be logged, and unique additional meta values will **not** be indexed for this key until a new slice is created. The purpose of this is to keep the index from bloating too badly, but this can possibly lead to queries against the meta key yielding incomplete results, as queries are run against the index.
- **defaultAction**: Not shown in this example, this attribute instructs the UI's Investigation module how to display the key when loading values. Valid values are **Open**, **Closed**, **Auto**, or **Hidden**. Explanations for these are found on the lesson on Investigation.

One good resource for configuring an index configuration file is the out-of-the-box files themselves. The top of the file provides some ad hoc documentation on each attribute and valid value.

index-concentrator.xml

Making Sure It All Matches

When dealing with a multi-tiered stack of services like SA, it's critical to keep the index configurations synchronised between services. This means that for every common meta key defined in your deployment should be configured with the same attributes. The most important attributes to keep in sync are **name** (obviously), **level** (**IndexKeys**, **IndexValues**), and **format** (**IPv4**, **Text**, etc). If these don't match, you will get unexpected results such as queries returning no data, etc.

The exception to this is on Decoders, where the **level** attribute is almost always going to be set to **IndexNone**.

Special Meta Keys - 'did', 'rid', and 'cid'

Here's a bit of a recap of some of what we learned in the Investigation lesson.

There are three meta keys which can aid in investigations, namely Decoder ID, or '**did**', Remote Session ID, or '**rid**', and Concentrator Source, or '**cid**'. This meta is generated by the Concentrator or Broker during aggregation rather than by the Decoder itself, and thus is stored only on Concentrators, Brokers, and Archivers.

'**did**' - The value stored under **Decoder Source** (or Decoder ID, hence '**did**') is the name of the Decoder which generated the session. When using a Broker or Concentrator which has visibility into multiple Decoders, you can drill into only those sessions which originated at a specific Decoder, without having to change the Investigation device to look at only a specific Concentrator.

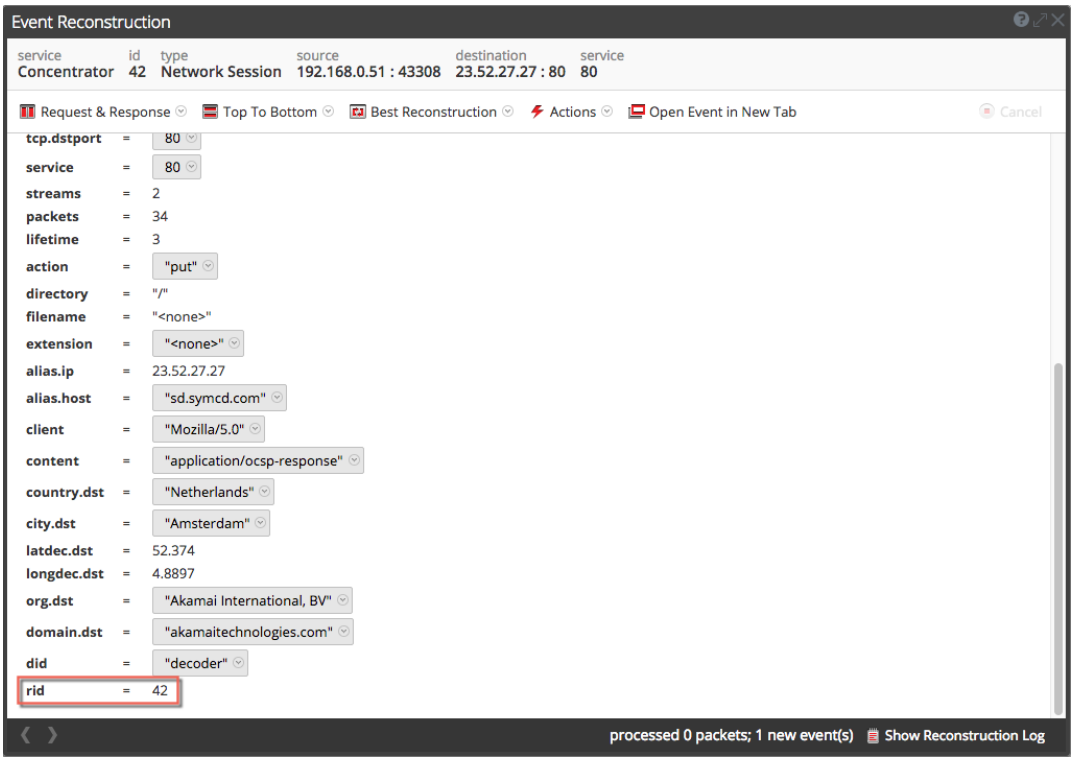
'**cid**' - Similar to the 'did' key, the Concentrator Source key stores the name of the Concentrator which stores a session's meta.

<insert cid screenshot from Investigate>

'**rid**' - The **Remote Session ID** key stores the upstream session ID of a given session. Because NWDB stores values sequentially beginning at 1, it is possible that a given session which is identified by one session ID, say 12345, might be have an entirely different Session ID on a downstream service, like 25488. This is especially true when one service aggregates from more than one upstream service, like when a Broker aggregates from 4 Concentrators, or when a Concentrator aggregates from two Decoders. See the following illustration.

<insert illustration showing the relationship of session ID's to each other>

When one conducts an Investigation through a Broker and is viewing the meta for a given session, if for some reason one wishes to directly pull up the session on the upstream Concentrator which holds the session's meta, one can't use the session ID from the Broker. Instead, the '**rid**' meta value stored for the session must be used to pull up the same session from the Concentrator. If you use the session ID from the Broker on the Concentrator, you'll get a completely different session!

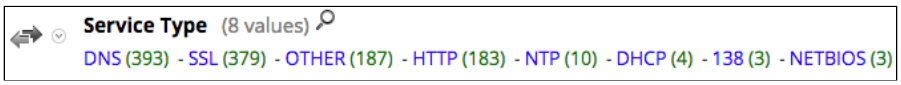


Value Aliases

Frequently used meta values can sometimes have friendly names assigned to them, so that when you see these values in Investigation, you see the friendly name rather than the indexed value. The best example of this is the **service** meta key.

The **service** key tracks what type of traffic was captured in a given session, or its application-layer protocol. Its value corresponds to the most commonly-used port number for that protocol. For instance, HTTP sessions have a service meta value of **80**, SSL has **443**, SMTP has **25**, etc.

Yet when you look at the service meta key in Investigation, you don't see numbers; you see friendly names like HTTP, SMTP, DNS, etc. This is because an alias has been defined for the most common values.



Keep in mind, though, that SA is port-agnostic, so even though the service number may look like a port, a session where '***service=80'*** may have **tcp.dstport** being something completely different.

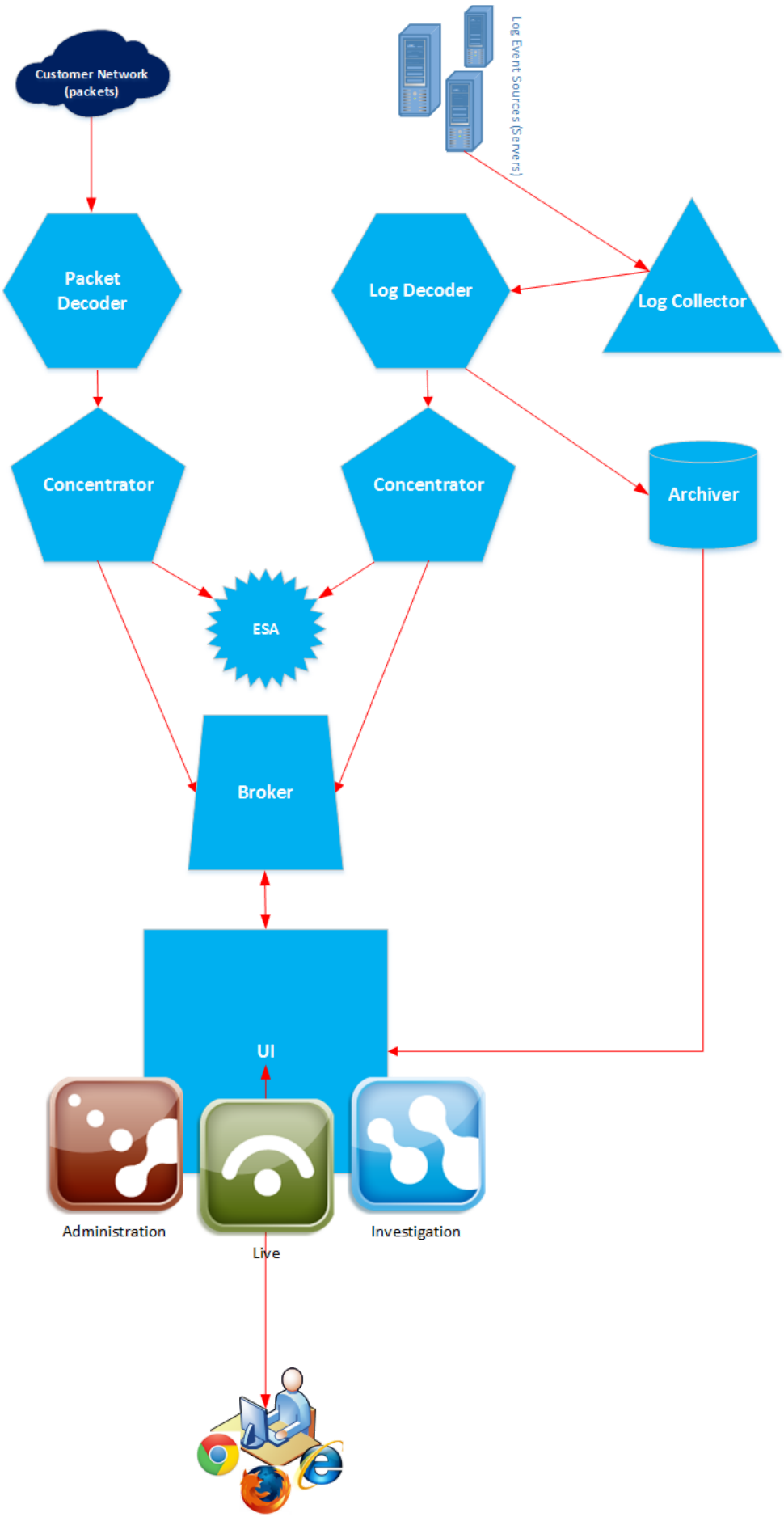
Aliases are defined as part of a meta key's XML definition. Before closing the **<key>** tag, you add an **<aliases>** subkey. I think you should should be able to work it out fairly easily from the following example from **index-concentrator.xml**.

```
<key description="Service Type" format="UInt32" level="IndexValues" name="service" valueMax="75" defaultAction="Open">
        <aliases>
                <alias format="$alias" value="0">OTHER</alias>
                <alias format="$alias" value="20">FTPD</alias>
                <alias format="$alias" value="21">FTP</alias>
                <alias format="$alias" value="22">SSH</alias>
                <alias format="$alias" value="23">TELNET</alias>
                <alias format="$alias" value="25">SMTP</alias>
                <alias format="$alias" value="53">DNS</alias>
                <alias format="$alias" value="67">DHCP</alias>
                <alias format="$alias" value="69">TFTP</alias>
                <alias format="$alias" value="80">HTTP</alias>
                <alias format="$alias" value="110">POP3</alias>
                <alias format="$alias" value="111">SUNRPC</alias>
                <alias format="$alias" value="119">NNTP</alias>
                <alias format="$alias" value="123">NTP</alias>
                <alias format="$alias" value="135">RPC</alias>
                <alias format="$alias" value="137">NETBIOS</alias>
                <alias format="$alias" value="139">SMB</alias>
                <alias format="$alias" value="143">IMAP</alias>
                <alias format="$alias" value="161">SNMP</alias>
                <alias format="$alias" value="179">BGP</alias>
                <alias format="$alias" value="443">SSL</alias>
                <alias format="$alias" value="502">MODBUS</alias>
                <alias format="$alias" value="520">RIP</alias>
                <alias format="$alias" value="1024">EXCHANGE</alias>
                <alias format="$alias" value="1080">SOCKS</alias>
                <alias format="$alias" value="1122">MSN IM</alias>
                <alias format="$alias" value="1344">ICAP</alias>
                <alias format="$alias" value="1352">NOTES</alias>
                <alias format="$alias" value="1433">TDS</alias>
                <alias format="$alias" value="1521">TNS</alias>
                <alias format="$alias" value="1533">SAMETIME</alias>
                <alias format="$alias" value="1719">H.323</alias>
                <alias format="$alias" value="1720">RTP</alias>
                <alias format="$alias" value="2000">SKINNY</alias>
                <alias format="$alias" value="2040">SOULSEEK</alias>
                <alias format="$alias" value="2049">NFS</alias>
                <alias format="$alias" value="3270">TN3270</alias>
                <alias format="$alias" value="3389">RDP</alias>
                <alias format="$alias" value="3700">DB2</alias>
                <alias format="$alias" value="5050">YAHOO IM</alias>
                <alias format="$alias" value="5060">SIP</alias>
                <alias format="$alias" value="5190">AOL IM</alias>
                <alias format="$alias" value="5222">Google Talk</alias>
                <alias format="$alias" value="5900">VNC</alias>
                <alias format="$alias" value="6346">GNUTELLA</alias>
                <alias format="$alias" value="6667">IRC</alias>
                <alias format="$alias" value="6801">Net2Phone</alias>
                <alias format="$alias" value="6881">BITTORRENT</alias>
                <alias format="$alias" value="8000">QQ</alias>
                <alias format="$alias" value="8002">YCHAT</alias>
                <alias format="$alias" value="8019">WEBMAIL</alias>
                <alias format="$alias" value="8082">FIX</alias>
                <alias format="$alias" value="20000">DNP3</alias>
                <alias format="$alias" value="1000000">KERNEL</alias>
                <alias format="$alias" value="1000001">USER</alias>
                <alias format="$alias" value="1000003">SYSTEM</alias>
                <alias format="$alias" value="1000004">AUTH</alias>
                <alias format="$alias" value="1000005">LOGGER</alias>
                <alias format="$alias" value="1000006">LPD</alias>
                <alias format="$alias" value="1000008">UUCP</alias>
                <alias format="$alias" value="1000009">SCHEDULE</alias>
                <alias format="$alias" value="1000010">SECURITY</alias>
                <alias format="$alias" value="1000013">AUDIT</alias>
                <alias format="$alias" value="1000014">ALERT</alias>
                <alias format="$alias" value="1000015">CLOCK</alias>
        </aliases>
</key>
```

13. Packet Decoder

The Decoder is arguably the great workhorse of the Security Analytics product stack. Without it, nothing else would be possible.

```
The Packet Decoder listens on port TCP 50004 / 56004 (SSL).
```



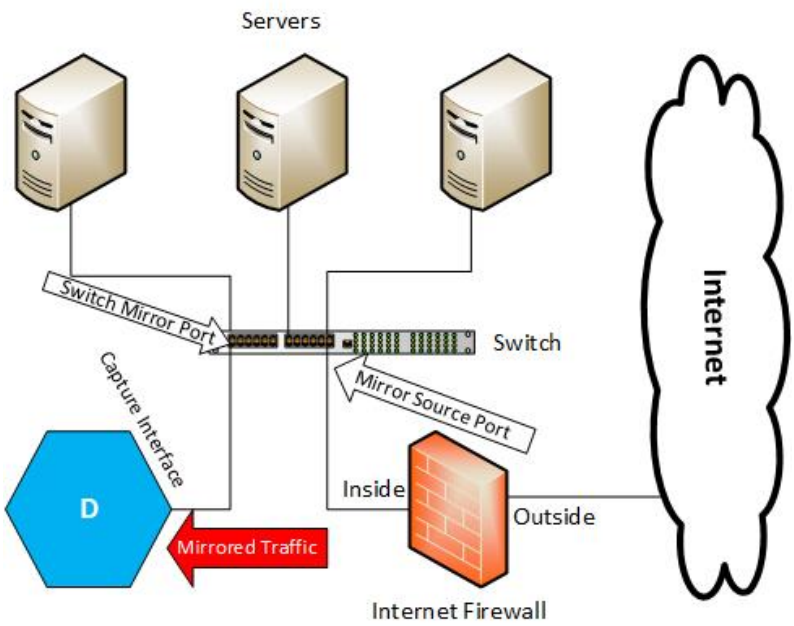
To review its functionality, the Decoder performs the following functionality:

1. **Capture** - is the process of reading packet or log data off of the wire and into memory
2. **Assembly** - the process of separating packet data and log data into individual sessions
3. **Parsing** - The process of generating meta from assembled packet or log streams
4. **Rules & feeds** - Generate additional meta on sessions based on other values, such as a rule or a feed from an external data source (e.g. a list of users, DHCP feed, etc). Can also be used to truncate or drop traffic based on certain characteristics.
5. **Database storage** - Save packet, log, meta, and session data to disk for future retrieval.
6. **Indexing** - Create a searchable index of metadata which can be used to quickly drill through and find sessions matching certain characteristics

How Do We Get Packets Into SA?

This is a fair question. SA is not an inline product, meaning it does not sit in the flow of network traffic for the purposes of disrupting sessions with suspected malicious traffic. The Packet Decoder is passive in nature, greedily gobbling up traffic that is sent to it from another network source.

There are three primary ways of sending network traffic to SA:



SPAN / Copy Port Scenario

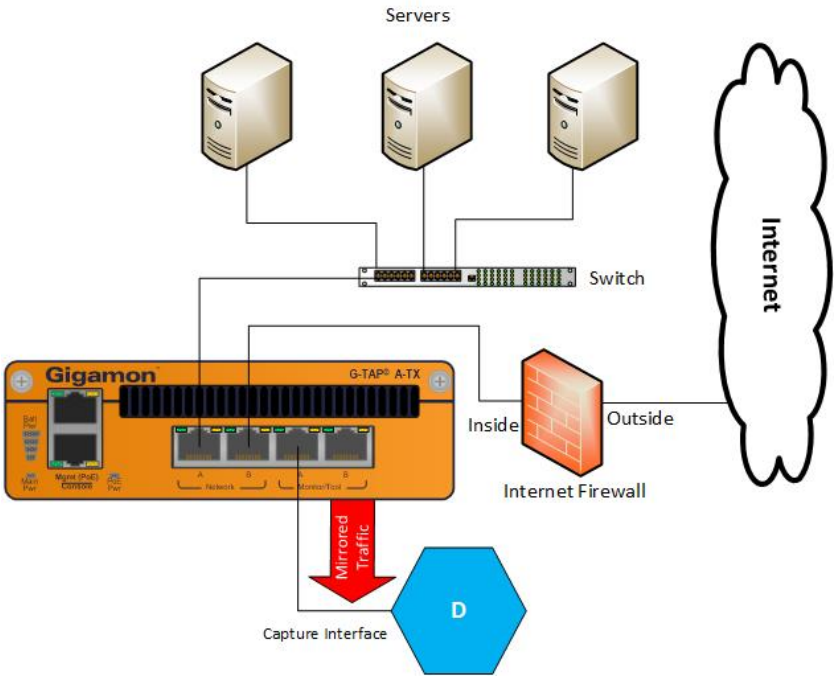
1. Switch mirror or SPAN ports

Nearly any high-quality managed network switch has the capability of copying all frames being received on any or all of its ports out of another network port. This is an inexpensive method enabling capture of traffic as it uses existing infrastructure, however it has one major drawback. A switch is not unlike any other computer, possessing resource constraints and periods of high utilisation as network conditions dictate. The port mirroring operation is essentially a software process on the switch which consumes processor time and memory. It will run with a lower CPU priority than the switch's main software, and consequently when a switch encounters a period of high utilisation, the mirror operation may drop frames in order to permit the switch to do its main job, which is forwarding frames between communicating ports. Dropped frames on the switch mean a loss of data fidelity at the capture point - the Decoder - as the sessions it receives at that time will be incomplete. This means that frames which are critical to detecting an incident may never be seen by the Decoder, this lessening your organisation's security posture. And unlike when a Decoder drops packets itself due to resource overload, the Decoder has no way of detecting whether it missed a frame of a session or not.

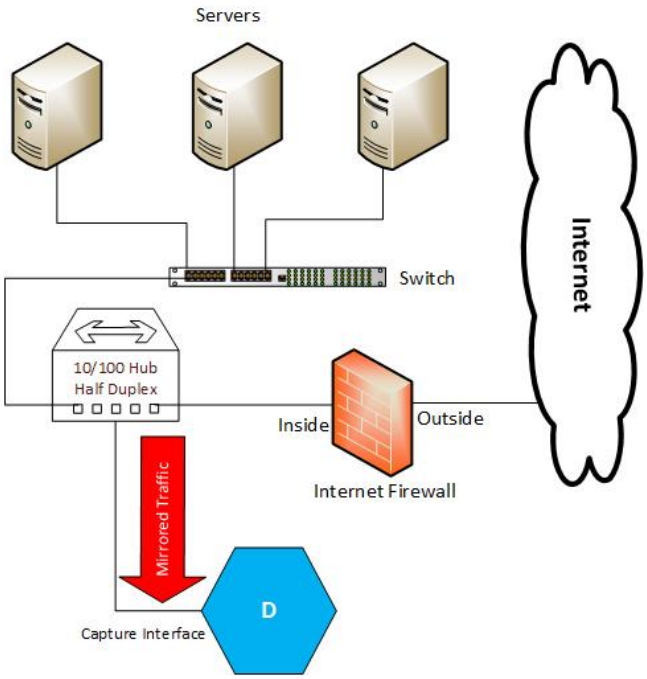
2. Network taps

The recommended traffic copy method for all SA customers is to use a network tap. A tap is a dedicated network device which itself sits inline, whose only job is to copy network frames to a traffic sniffing device out of one or more of its ports. Because the tap's only purpose in life is to copy traffic, it is not prone to the fidelity and frame drop problems which switches are susceptible to. VSS is a commonly-used vendor of network taps.

Some taps can copy traffic from more than one source port to one or more destination port, which can accommodate Decoders with multiple NIC's, or even copy the same source stream to multiple Decoders. Some can also load-balance mirrored traffic across multiple destination ports, and are smart enough to copy all packets of a given session to the same destination port, so that sessions don't get split between multiple capture ports.



Network Tap Scenario



Hub Scenario

3. Old-fashioned network hub

Lastly, it is possible to capture traffic using a dumb, old-fashioned network hub (not a switch). Hubs copy any frame received on any port out all of its other ports. There are three major drawbacks to this method: it is limited to 100Mbps, it is limited to half-duplex operation, and is susceptible to frame collisions. This method is not recommended except for the most low-bandwidth of capture points. It can also be difficult to source these old devices today.

Where do I Place the Capture Point?

Organisations running SA should take great care in selecting its capture point or points in their network. There are several deployment scenarios for SA for packets:

1. Capture point just inside network perimeter - The most common and typically high-value deployment scenario. The capture point is placed just to the inside of the internet perimeter firewall. This allows for capture of inbound traffic which has made it through the firewall, which is typically the most interesting traffic. It also allows visibility of traffic before it has been NAT'ed by the perimeter firewall, which makes investigation much easier, when you can see the original IP of a sesion.

2. Capture inside of internet proxy - One problem with placing the capture point just to the inside of the perimeter firewall arises if an organisation uses a proxy for outbound traffic prior to hitting the firewall. If the capture point is between the proxy and the firewall, then proxied traffic will always show with the source IP of the proxy rather than its original source IP, which can make investigations very frustrating. If an organisation uses a web proxy, it often makes sense to place a capture point just to the inside of the web proxy. This allows the Decoder to see the original source IP of the session. One potential workaround to placing the capture point after the proxy is to activate **X-Forwarded-For** (XFF) on the proxy, which adds a field to the HTTP header of the session which contains the original source IP of the session. Using XFF has a drawback, though - if an organisation is not careful, the XFF field can expose your private network topology to the outside world, as it sends the original source IP in the HTTP header. To mitigate against this, some firewall products can strip the XFF field from the HTTP header before it leaves the network.

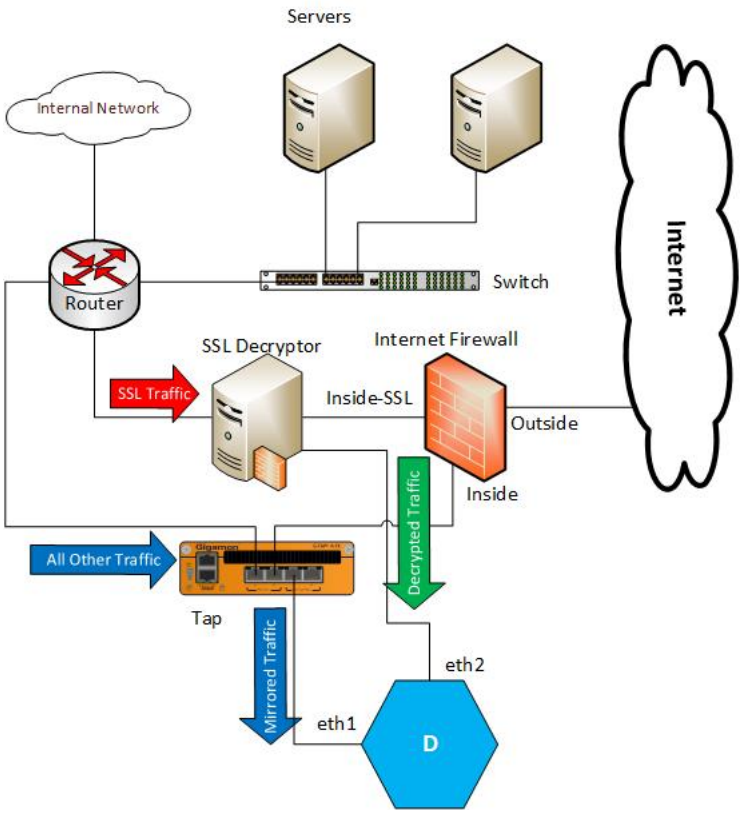
3. Capture in network core - Capture points can be placed in the core of the organisation's network (i.e. in its datacenter), so that internal-only traffic is captured. The value in this is that it captures traffic between internal network hosts, which can gain visibility into lateral attacks inside the network. The downside is that the capture rate is much higher than capturing at the perimeter, therefore it can require more Decoders and much more storage to capture all the traffic.

4. Capture outside network perimeter - The capture point can be just outside the perimeter firewall so that all internet traffic is seen by the Decoder, but this is not recommended. The traffic which should be of forensic use will only be the traffic which makes it through the perimeter. The rest is just noise, and is of no forensic value. You also will only see the NAT'ed source IP of sessions rather than the original internal IP's, making investigation very difficult. It also will take up more storage and cause more sessions than necessary to hang out in the capture pipeline. It may also send your analysts on wild goose chases. **Don't do this.**

Lastly, it is possible to deploy your SA services so that they service more than one deployment scenario. E.g. you may wish to capture both at the perimeter and at the core simultaneously.

What About Encrypted Traffic?

Many customers also want to see inside encrypted sessions so that they can see hidden malicious traffic. SA is not intrinsically capable of decrypting SSL sessions as they go over the wire on its own. This does not, however, mean that it can't be done. There are third-party products (Bluecoat Netronome is one example) which are capable of decrypting SSL traffic by proxying the SSL sessions - a man-in-the-middle (MITM) proxy. The decrypted traffic is then sent out an ethernet port which is then captured by a Decoder, which can then be forensically analysed and reconstructed. These devices can come with some administrative overhead as they break the default trust that PKI provides by substituting its own SSL certificates for the public certificates which were signed by a trusted CA. Thus the organisation must configure their network hosts to trust certificates issued by the MITM SSL proxy, though this is outside the scope of this course.



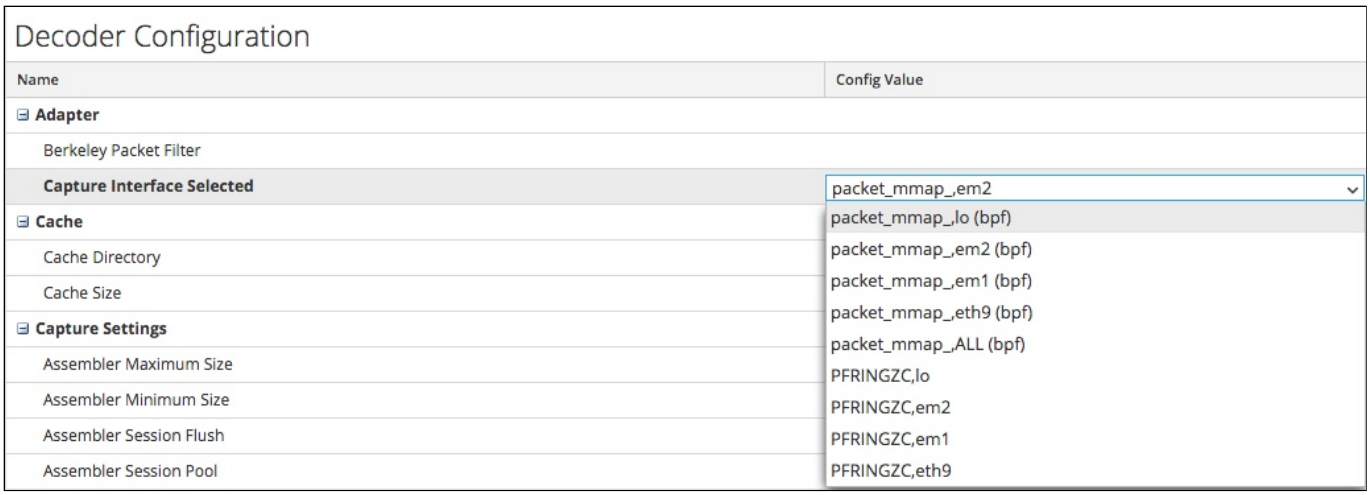
Capturing Traffic Using an SSL Decryptor

Storage Requirements

Decoders typically have the greatest storage requirements of any SA service. Many organisations are under directives to store captured data for a certain number of days. When dealing with the volume of data which is typical of SA, especially raw packets, the storage requirements can be very large. With the exception of Hybrids, Decoders are paired with one or more DAC's or a SAN. It is not unusual to see five or more DAC's daisy chained together.

Choosing a Capture Interface

In order for the Decoder to capture packets off the wire, you must first tell it which NIC to listen for packets on. An interface is not autoselected at first run, so ensure you do this bit. In general, you may choose either one interface (e.g. em2) or all interfaces. When choosing all interfaces, this will also include Decoder management traffic. This can be done from the UI in **Administration -> Services -> <*Decoder*> -> Config**. Be sure to hit 'apply' after selecting an interface.



Technical Note: On some Decoders, you may notice the presence of interfaces beginning with 'PFRINGZ'. These interfaces are provided by the pf_ring driver, which is a commercial driver for certain 10G NIC's. This driver binds the Decoder service more closely with 10G NIC's, allowing for much accelerated capture rates (typically ~8 Gbps). For Decoders without 10G NIC's, these interfaces should be ignored.

Another Technical Note: 10G operation requires special Decoder configuration, but this is outside the scope of this course. For more information on configuring 10G, please consult the SA documentation.

Setting Snapshot Length

There is a hidden Decoder parameter which can greatly impact the effectiveness and performance of SA, and it is called snapshot length, or **snaplen** for short. To put it simply, snaplen is a configuration parameter which controls how many bytes of each frame should be captured and parsed. The default snaplen for a Decoder is 1500 bytes, which is less than the standard ethernet frame size of 1514. This means that that by default, the last 14 bytes of every frame captured will be missing in many customer environments - this will lead to loss of data fidelity, as well as missed meta. Snaplen must be customised for every customer environment. Some environments might have jumbo frames, and others may not.

To set snaplen, set "snaplen=<*value*>" under API tree node **/decoder/config/capture.device.params**, where <value> is the snapshot length in bytes. E.g. "snaplen=9000".

Please keep in mind that by increasing the snaplen value, you are placing your Decoder under additional load, because there will be more for the Decoder to capture and parse. This could potentially have adverse effects, especially in high-bandwidth environments.

To determine the correct snapshot length value for your environment, paste this one-liner script into a shell on your Decoder, substituting the correct interface name in the tcpdump command, i.e. eth1, eth2, etc. It will output numbers which increase in value, with each number representing a frame length received on the selected interface. Allow it to run for a little while, and once the numbers have stopped incrementing, the largest and last number should indicate the largest frame received by the Decoder.

```
tcpdump -enni emX | perl -n -e '$maxlen=0 if ! defined $maxlen;s/length (\d+)\://;my $len=$1;if ($len > $maxlen) {$maxlen=$len;print "$maxlen\n";}'
```

Here's some sample output from running the command:

```
[root@decoder ~]# tcpdump -enni em1 | perl -n -e '$maxlen=0 if ! defined $maxlen;s/length (\d+)\://;my $len=$1;if ($len > $maxlen) {$maxlen=$len;print "$maxlen\n";}'
tcpdump: WARNING: em1: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
103
144
1479
1514
```

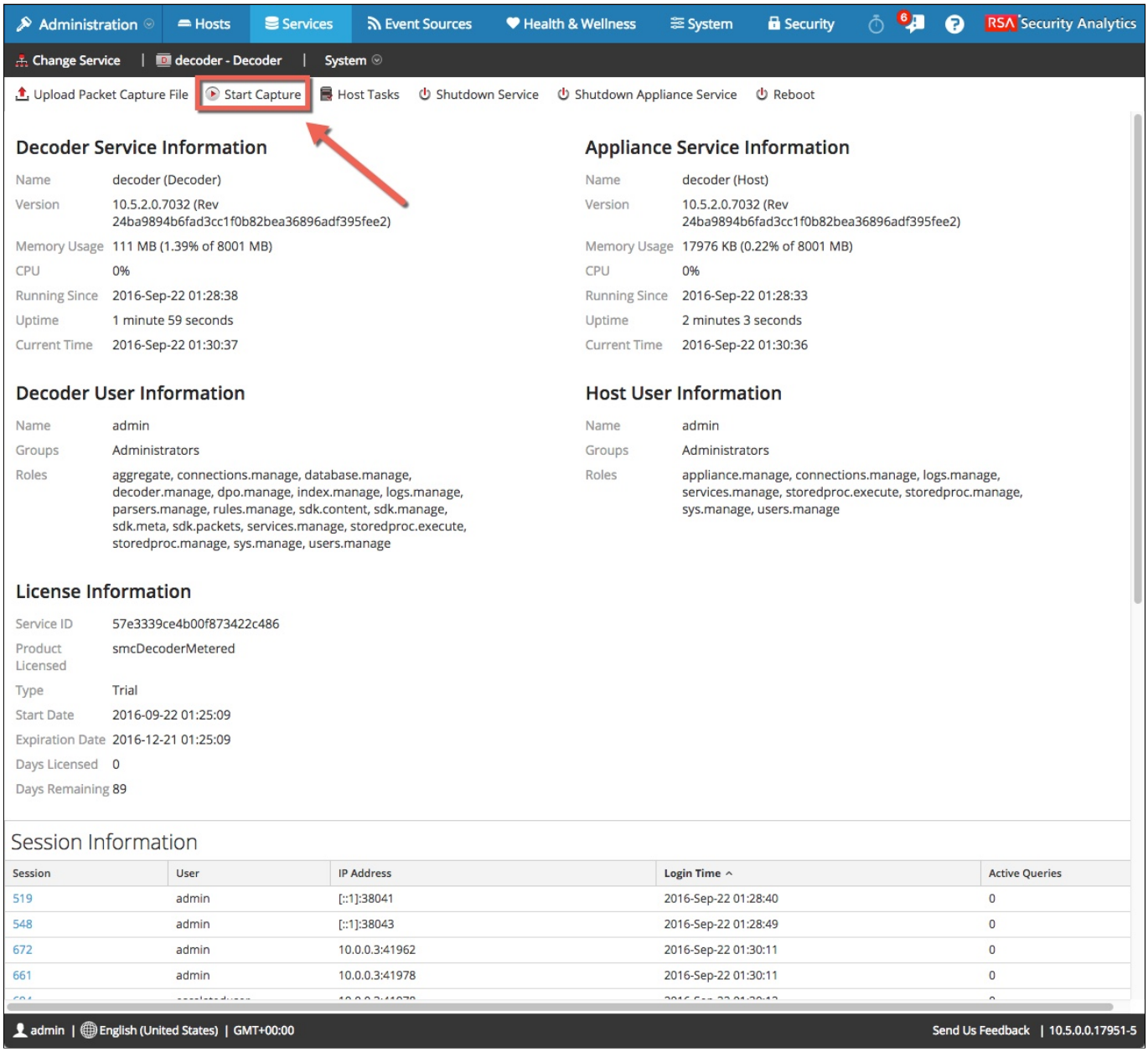
In this example, the correct snaplen for this Decoder is 1514 bytes.

Illustration of why snaplen should be configured

Starting and Stopping Capture

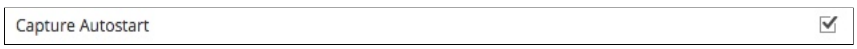
Decoders by default do not start capturing packets automatically. Capture must be started by an Administrator from **Administration -> Services -> *<Decoder>* -> System**. You can tell whether capture is started by whether the button says "**Start Capture**" or "**Stop Capture**".

Once capture is started, the Decoder will place the configured capture NIC or NIC's into promiscuous mode, allowing it to see all traffic, not just traffic that is destined for the Decoder. On a Packet Decoder, with the exception of the management interface, capture interfaces will not have an IP address. It is not necessary to enable capture interfaces in the OS, either. The Decoder will do this for us.



Capture Autostart

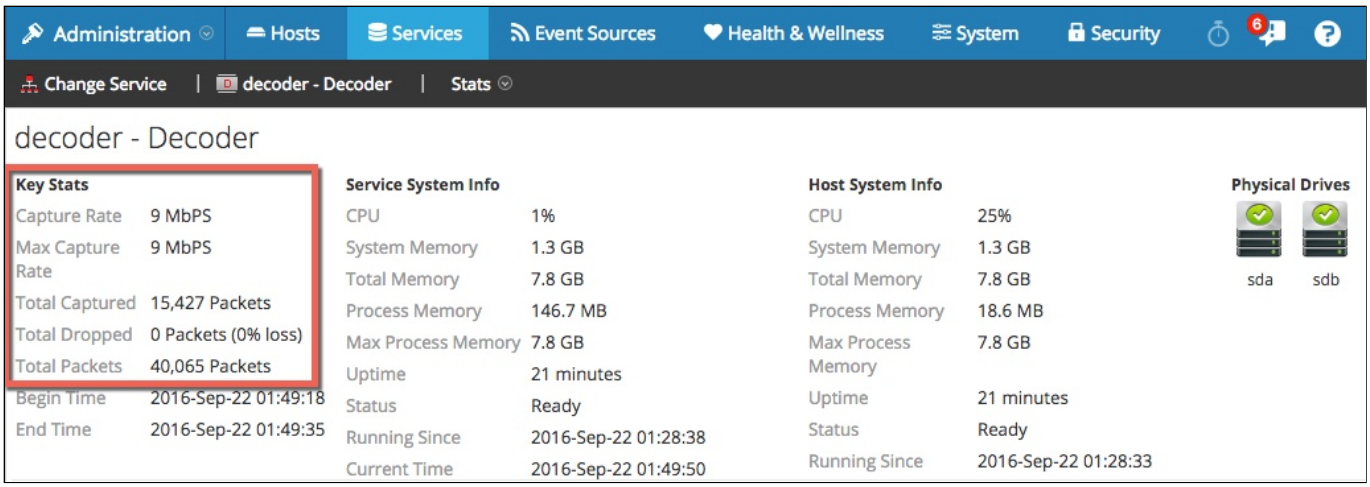
As noted, capture does not by default start automatically when the Decoder process starts. This must be done from **Administration -> Services -> *<Decoder>* -> Config.** Tick the box next to 'Capture Autostart' and click **Apply**. The next time you restart the Decoder process, capture should start automatically.



Ensuring That Capture is Working

After starting capture, it's always a good idea to visit the Stats page of the Decoder to ensure that capture is functioning correctly. One can look at the capture rate and max capture rate statistics to ensure that traffic is being received by the Decoder. Another thing to make sure of is that the Decoder is not dropping any packets, either.

Most statistics displayed here are only for the current capture session. Max Capture Rate, Total Dropped, and Total Packets stats are only calculated since the last time capture was started.



Capture Rates & Dropped Packets

Every Decoder has a maximum packet rate at which it is capable of ingesting before it begins **dropping packets**. Packet drops occur when the Decoder cannot process incoming packets quickly enough due to exhaustion of resources - either CPU, memory, or disk queue, or a combination thereof. When a drop occurs, the associated packet is not saved to disk and any meta which could be parsed from the packet will be lost. It is therefore important to avoid a dropping packet scenario.

There are many factors which can contribute to packet drops, but chiefly among them is the capture rate. RSA avoids publishing official capture rates for SA because there are so many variables which can impact its maximum. A good rule of thumb is 2 Gbps for Decoders not equipped with 10G capture cards. For Decoders that use 10G capture cards, the theoretical maximum is considerably higher, around 8 Gbps. The reason for this disparity is that certain Decoder optimisations take advantage of the special PF_RING 10G capture driver features, which are not available with the 1G NIC drivers.

Pre-series 5 RSA Decoders are equipped with 1G copper NIC's by default. These can be swapped for 1G or 10G fibre NIC's.

Series 5 Decoders have two 10G NIC ports built-in.

These are the main variables which impact capture performance:

- SA code
- Capture rate
- Packet size and snapshot length
- Type of traffic
- Enabled parsers (different parsers have different performance implications)
- Application rules
- Network rules
- BPF filters
- Number and size of feeds
- CPU cores & speed
- Memory size & speed
- Disk performance & RAID firmware
- OS kernel

Assembly

Assembly is the process of creating sessions and streams from captured packets. When packets first make their way into the capture NIC or NICs, they first go into the *capture buffer*, a slice of memory where packets live until the Decoder process gets around to processing them. Once the Decoder reads the packets from the capture buffer, they go into the *assembler*. The assembler takes each individual packet and adds it to a session.

The Decoder is aware of the inner workings of the TCP/IP protocol, and uses characteristics of each packet to identify which session it should belong to. These are things such as protocol (TCP, UDP, ICMP, etc), and source port and destination port. Assembly also is able to identify which IP of a session is the source side and which IP is the destination side. It does this by applying settings called *voting weights*, which are attributes such as TCP flags (syn, ack, syn-ack, fin), and number of bytes transferred per stream (the destination side is presumed to transfer more bytes than the source), and these are applied in a hierarchy (i.e. a syn packet trumps all other settings, since the source side of a session always sends a syn packet).

The packets of a session sent by the source IP is one 'stream', in SA parlance, and the packets sent by the destination IP is another stream. The Decoder will generate a meta value under the *streams* key of either 1 or 2, indicating how many streams are in a session. The only sessions which should have a single stream should be certain one-sided UDP sessions (as UDP is connectionless and can happily blast traffic in one direction all day long without an acknowledgement), and TCP sessions which are dropped by a firewall after the first packet.

Parsers

Parsers are responsible for the first stage of meta generation at capture time. Parsers read the packet streams which are being captured and watch for 'tokens', which are byte array patterns in the packets. Once the necessary tokens have been seen and processed by the parser, the parser then registers one more more meta value for the given session, which in turn usually also gets stored in the metadb and often in the index.

Parsers are not the only things that can generate meta: so can application rules and feeds (though feeds technically run inside of a parser, too, called the *feed parser*.

- **Binary** - Binary parsers are built in to the compiled code of the Decoder service itself. They can be enabled or disabled like any other parser, but no customisation is possible. They are not represented by files anywhere on the filesystem, other than the /sbin/NwDecoder executable itself. Examples of binary parsers are the **network** parser, which generates ip.src and ip.dst (among other things), or the DHCP parser. There are many more binary parsers.

- **LUA** - LUA parsers are written in a standardised programming language which the Decoder understands. All modern parsers which RSA currently develops are written in LUA.

- **FLEX** - FLEX is an XML-based programming language, and in Decoder terms was the predecessor to LUA. Version 10.x Decoders can still handle FLEX parsers but they do not perform as well as LUA. Some believe they are considerably more difficult to write.

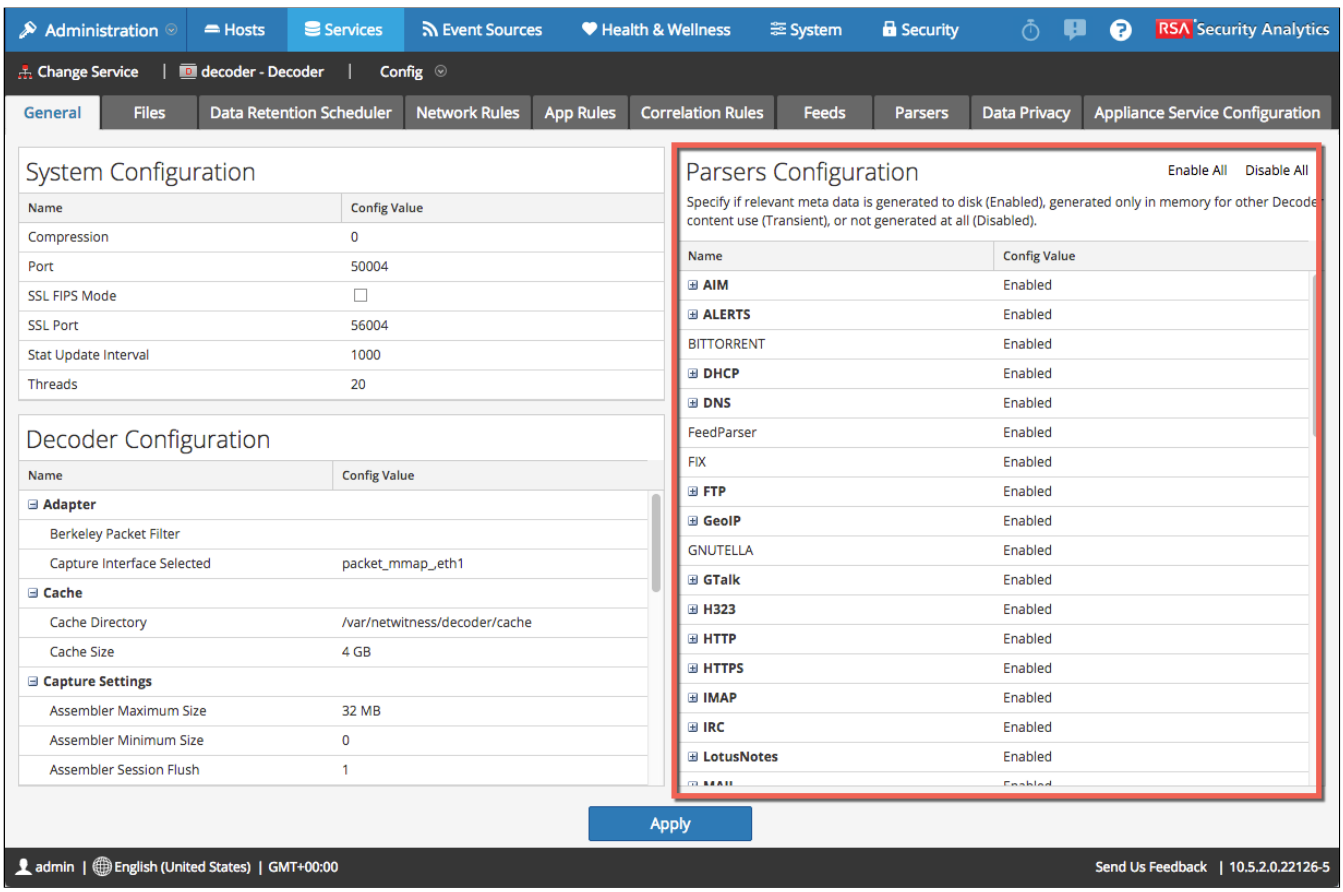
Note: The specifics of parser syntax and development is beyond the scope of this course.

One can see which parsers are installed on a Decoder in a few different ways:

- By looking in the **/etc/netwitness/ng/parsers** directory on the Decoder. This will only show LUA and FLEX parsers, but not binary parsers.

```
[[root@sa parsers]# pwd
/etc/netwitness/ng/parsers
[[root@sa parsers]# ls
aim.luax                          exif.luax-attr.xml                ldap.luax                 phishing.luax                 smtp.luax-attr.xml
aim.luax-attr.xml                 fix.luax                          ldap.luax-attr.xml        phishing.luax-attr.xml        socks.luax
creditcard_detection.luax         fix.luax-attr.xml                 luax.tokens               poison_ivy.luax               socks.luax-attr.xml
creditcard_detection.luax-attr.xml formdata.luax                    mail.luax                 poison_ivy.luax-attr.xml      soulseek.luax
dcerpc.luax                       formdata.luax-attr.xml            mail.luax-attr.xml        pop.luax                      soulseek.luax-attr.xml
dcerpc.luax-attr.xml              ftp.luax                          mitozhan.luax             pop.luax-attr.xml             ssh.luax
derusbi_server.luax               ftp.luax-attr.xml                 mitozhan.luax-attr.xml    pwdump.luax                   ssh.luax-attr.xml
derusbi_server.luax-attr.xml      ghost.luax                        MSU_rat.luax              pwdump.luax-attr.xml          tftp.luax
dhcp.luax                         ghost.luax-attr.xml               MSU_rat.luax-attr.xml     rdp.luax                      tftp.luax-attr.xml
dhcp.luax-attr.xml                glass_rat.luax                    NetBIOS.luax              rdp.luax-attr.xml             tld.luax
dns_verbose.luax                  glass_rat.luax-attr.xml           NetBIOS.luax-attr.xml     rlogin.luax                   tld.luax-attr.xml
dns_verbose.luax-attr.xml         HTML_threat.luax                  nfs.luax                  rlogin.luax-attr.xml          tls.luax
duqu.luax                         HTML_threat.luax-attr.xml         nfs.luax-attr.xml         rsync.luax                    tls.luax-attr.xml
duqu.luax-attr.xml                htran.luax                        ntp.luax                  rsync.luax-attr.xml           types.xsd
dyndns.luax                       htran.luax-attr.xml               ntp.luax-attr.xml         rtmp.luax                     types.xsd.rpm_new
dyndns.luax-attr.xml              http.luax                         nwll.luax                 rtmp.luax-attr.xml            vnc.luax
ein.luax                          http.luax-attr.xml                nwll.luax-attr.xml        rtsp.luax                     vnc.luax-attr.xml
ein.luax-attr.xml                 http_sql_injection.luax           ocsp.luax                 rtsp.luax-attr.xml            xor_executable.luax
ethernet_oui.luax                 http_sql_injection.luax-attr.xml  ocsp.luax-attr.xml        search_engines.luax           xor_executable.luax-attr.xml
ethernet_oui.luax-attr.xml        imap.luax                         packers.luax              search_engines.luax-attr.xml
evilgrab.luax                     imap.luax-attr.xml                packers.luax-attr.xml     shadyrat.luax
evilgrab.luax-attr.xml            isakmp.luax                       parsers.xsd               shadyrat.luax-attr.xml
exif.luax                         isakmp.luax-attr.xml              parsers.xsd.rpm_new       smtp.luax
[root@sa parsers]#
```

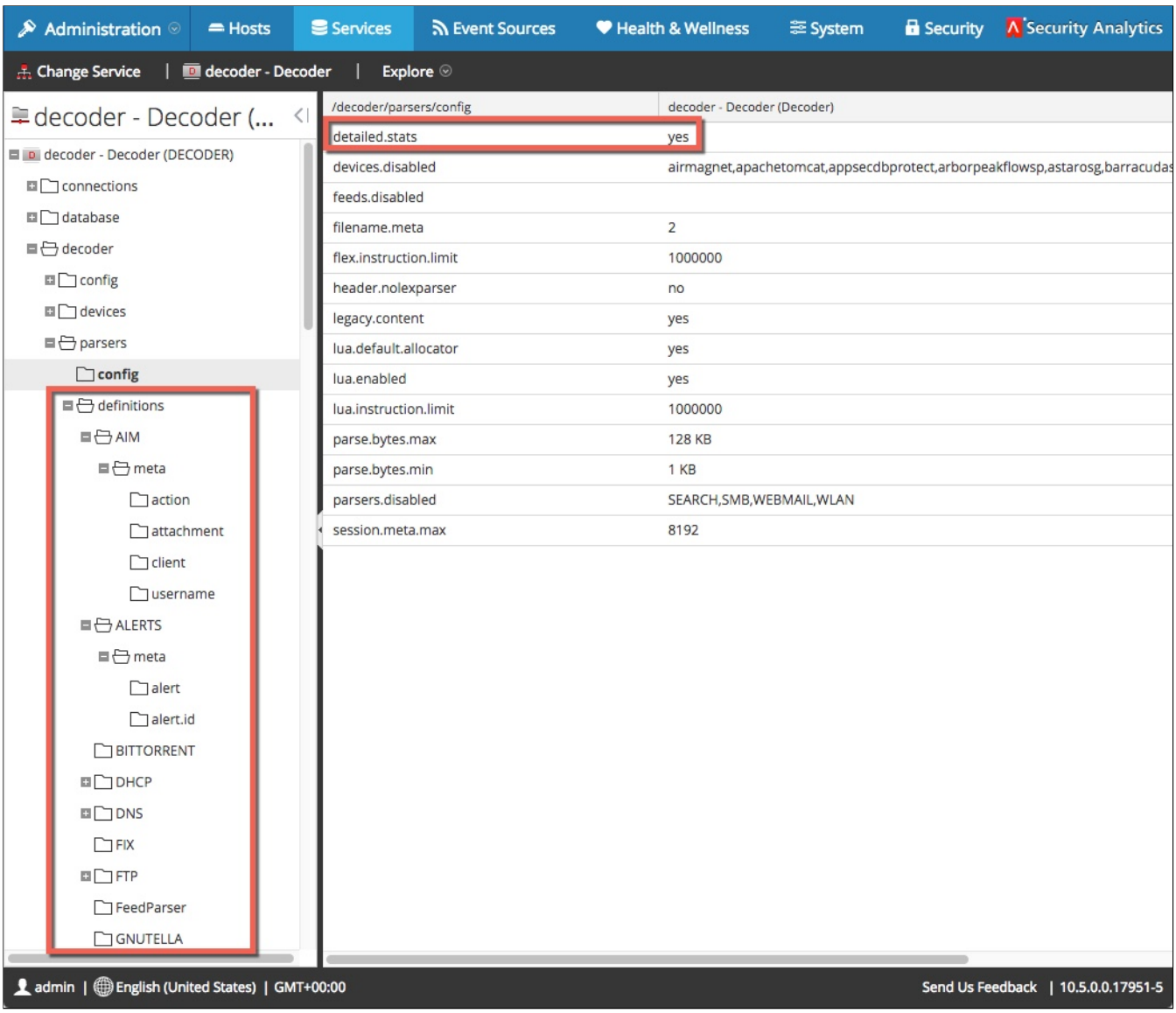
- In the UI, by looking on the **Administration -> Services -> *<Decoder>* -> Config** page, under **Parsers Configuration.** This will show all types of parsers



Under the parsers configuration, you can enable and disable which parsers will be used by the Decoder. By expanding the **+** symbols, you can even

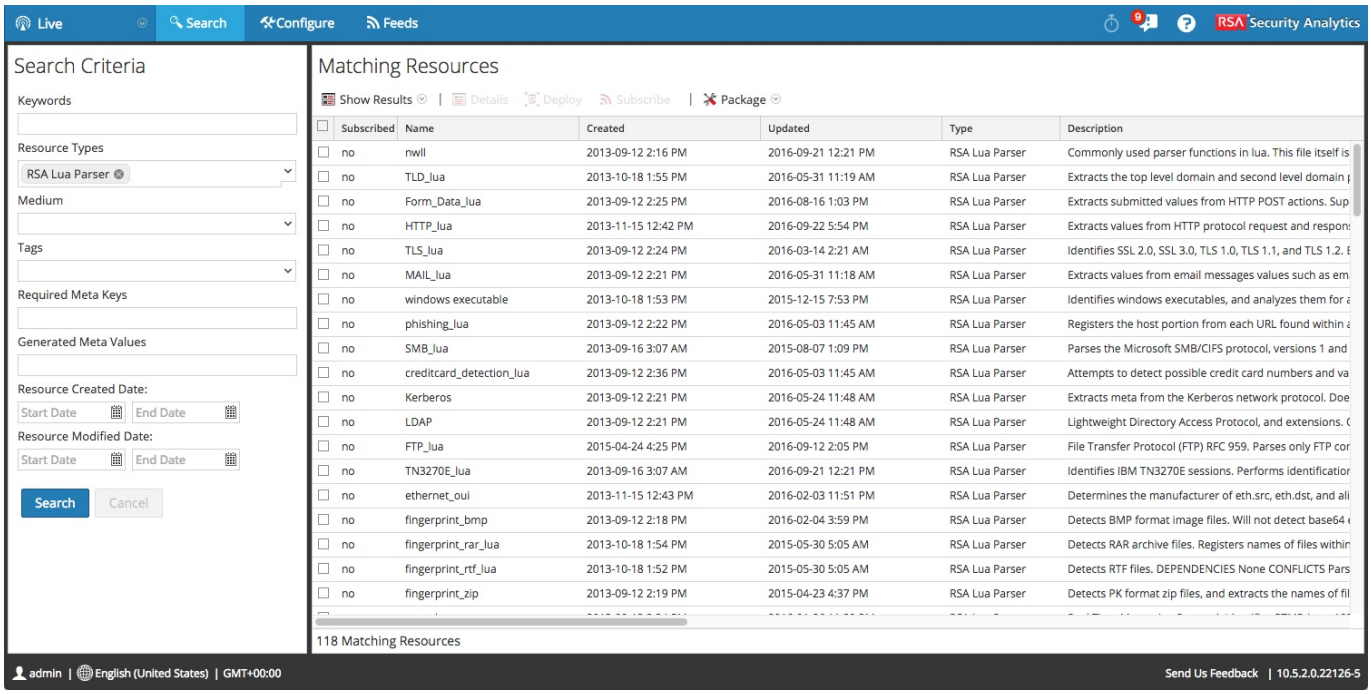
see which meta keys a given parser generates meta under, and you can granularly configure the generation of meta to occur only under specific keys

- In the UI, by **Administration -> Services -> <Decoder> -> Explore** page, under node **/decoder/parsers/definitions**. Note that this node does not show up without first changing the value of **/decoder/config/detailed.stats** from '*no*' to '*yes*'. These detailed stats can also be very useful in troubleshooting Decoder performance. This will show all types of parsers. In the below example, notice how every parser also has a node called '*meta*', which shows which meta keys the parser will populate.



Installing Additional Parsers

Additional parsers can be added to a Decoder via **Live -> Search** in the UI, using the Resource Type of '*RSA LUA Parser*'. Simply select the parsers you want to install, then click '**Subscribe**', then '**Deploy**'.



Special Parsers

- **The SEARCH parser** - The SEARCH parser is a regex (Regular Expression) and keyword-based parser with its own configuration file - /etc/netwitness/ng/search.ini. This parser is disabled by default, as it can incur massive performance penalties. It is recommended for use only on low-bandwidth Decoders. The **search.ini** file contains instructions for configuring it (click the link to see the file) - which sessions and services it should apply to
- **The SMB parser** - As the name implies, the built-in SMB parser parses the Microsoft Server Message Block protocol. It has been known to cause performance problems, and is disabled by default.
- **The WLAN parser** - Disabled by default, the WLAN parser is capable of decrypting some 802.11 wireless LAN frames (though not all types). It is configured by the file /etc/netwitness/ng/wlan-config.xml. Please see this file for instructions on configuring it.

Decoder Databases

- The Session DB - Stored in **/var/netwitness/decoder/sesssiondb**
- The Packet DB - Stored in **/var/netwitness/decoder/packetdb**
- The Meta DB - Stored in **/var/netwitness/decoder/metadb**

Index Configuration File

The index configuration files are **index-decoder.xml** and **index-decoder-custom.xml** under **/etc/netwitness/ng**.

The Decoder Index

```
[root@decoder ng]# grep level /etc/netwitness/ng/index-decoder.xml
<!-- time needs to always be indexed at value level -->
<key description="Time" level="IndexValues" name="time" format="TimeT" valueMax="0"/>
<key description="Service Type" level="IndexNone" name="service" valueMax="75" format="UInt32" defaultAction="Open">
<key description="TCP Source Port" level="IndexNone" name="tcp.srcport" format="UInt16">
<key description="TCP Destination Port" level="IndexNone" name="tcp.dstport" format="UInt16" valueMax="65536">
<key description="UDP Source Port" level="IndexNone" name="udp.srcport" format="UInt16">
<key description="UDP Target Port" level="IndexNone" name="udp.dstport" format="UInt16" valueMax="65536">
<key description="Medium" level="IndexNone" name="medium" format="UInt8" defaultAction="Hidden">
```

In more than 99% of scenarios, Decoders have no index to speak of, with one exception: the '**time**' meta key. This key is used to track the time of each captured session. Because Decoders are not intended to be queried directly, they do not index any other key. This does not mean that Decoders are incapable of indexing - they have the same indexing code as Concentrators, and can in fact run a full index and be queried directly, if they are configured to do so.

At one point in time, an appliance called an Eagle Decoder was sold by NetWitness. An Eagle was a compact appliance intended to be temporarily deployed in datacentres to help with ad hoc incident response in an environment which didn't possess a full NetWitness product stack. Software-wise, the Eagle was no different than any other Decoder, except for its index configuration file, which was mostly that of a Concentrator's.

A Decoder's index configuration historically also served another purpose - to inform the Decoder of which meta keys it would be storing in the meta DB. Its index configuration file still contained entries for all the various meta keys, but had their level set to 'NoIndex' or 'IndexNone', which are the same thing. The Decoder would not record meta in the meta DB for a key which did not have an entry in the index configuration file.

These days, Decoders are more versatile, and no longer require explicit entries in their index configuration files. They can learn their meta key settings from configured feeds, parsers, and rules. You still need to understand the index configuration files for Concentrator and Broker configuration.

You will see in implementations, though, that some people still explicitly declare their meta keys in the index-decoder-custom.xml file. There is nothing wrong in doing this.

Decoder-Specific API Tree Nodes

- **/decoder/config** - Defines configuration parameters pertaining to capture and assembly.
- **/decoder/stats** - Defines parameters pertaining to Decoder performance
- **/decoder/parsers** - Container pertaining to LUA, FLEX, and binary parsers.
- **/decoder/parsers/config** - Defines configuration parameters pertaining to parsers
- **/decoder/parsers/definitions** - Shows statistics for individual parsers, which can be useful for analysing which parsers are generating the most meta. Its existence depends on value of **/decoder/parsers/config/detailed.stats**.

Importing PCAP's - Direct Import vs tcpreplay

If you have need to import PCAP files directly into your Decoder to be parsed, you have two options: you can use the UI to import a PCAP file, or you can use the **tcpreplay** CLI utility to import it.

With a direct import, there is one main drawback: you must first stop capture to perform the import. Packets imported have their original timestamps contained in the PCAP.

With tcpreplay, there is no need to stop capture on the Decoder, but PCAP file packets will have current timestamps, rather than their original.

Note: **tcpreplay** is not installed in Decoders. If this approach is desired, it is necessary to obtain the requisite RPM package from a third-party source and install it manually on the Decoder.

To import a PCAP file from the UI:

1. Navigate to **Administration -> Services -> <Decoder> -> System.**
2. Click the **Stop Capture** button, if capture is already started.
3. Click the "**Upload Packet Capture File**" button and choose the PCAP file to import. The file will be uploaded and processed by the Decoder.
4. If capture was started when you started, click the **Start Capture** button. Don't forget to do this!

14. Application Rules, BPF Filters, & Feeds

Custom Feeds

One of the most difficult SA concepts to grasp, feeds are the cornerstone of adding network or threat intelligence to SA. By injecting feeds into a Decoder, we can add information from third-party sources to create additional meta for captured sessions. Feeds can either be bespoke (i.e. user-created) or they can come from subscription services such as RSA Live.

Feeds operate upon meta which is in the process of being generated from a given session, during the process of being parsed by the Decoder. Feeds watch a feed-defined meta key (such as **ip.dst** or **alias.host**) and match it against list of values which are also defined in the feed, and if a match is made between the ip.dst value and the feed-defined value, a new feed-defined meta value or values will be created under one or more defined meta keys.

Feeds are essentially "If, Then" logic, from programming.

E.g. **IF** ip.src =~ 10.8.8.0/24, **THEN** campus.src = 'Home Office' **AND** city.src = 'London' **AND** 'country.src' = 'United Kingdom' **AND** asset = 'Human Resources'

Feeds are defined by a combination of two separate files: an XML definition file and a CSV file.

Once a feed is installed on a Decoder, they are processed by the ***feed parser***, which is a binary parser that can be enabled or disabled like any other parser.

Creating a Custom Feed, Example 1 - IP Address Range Low and High Index Fields

Let's look at a basic example of a feed, which watches the ip.src and ip.dst meta keys for specific values, and if a match is found, generate meta values under keys 'location' and 'netsegtype'. The first thing we'll look at is the feed CSV file, and from there we'll work our way back to its definition.

```
# Low IP, High IP, Office, Department
10.0.9.0,10.0.9.255,London,HR
10.0.12.0,10.0.12.255,London,Sales
10.0.15.0,10.0.15.255,Reading,TechSupport
```

See here that there are four columns in the CSV file. Since we're matching against ranges of IP addresses, we actually have two index fields. An index field is a CSV column that the Decoder is seeking to match in order to generate the meta contained in the value column(s). The first column represents the low IP address to search for, and the second column the high address.

The third and fourth columns are called value fields, in which the Decoder will register the individual values in a session that matches on the index field(s). It will register the value(s) under the meta keys which are defined in the feed XML definition.

Now let's have a look at the feed definition XML file:

```
<FDF>

  <FlatFileFeed name="Example Feed Definition" path="feed-ex1.csv" separator="," comment="#">

    <LanguageKeys>

      <LanguageKey name="office"
                   valuetype="Text"
                   srcname="office.src"
                   destname="office.dst" />

      <LanguageKey name="department"
                   valuetype="Text"
                   srcname="department.src"
                   destname="department.dst" />

    </LanguageKeys>

    <Fields>

      <Field index="1" type="index" range="low"/>

      <Field index="2" type="index" range="high"/>

      <Field index="3" type="value" key="office"/>

      <Field index="4" type="value" key="department"/>

    </Fields>

  </FlatFileFeed>

</FDF>
```

From this file, we can see that the feed definition (defined by <FlatFileFeed/>) is organised into two basic sections:

- The meta keys, defined within the <LanguageKeys> tag. These <LanguageKey> tags are used to tell the Decoder under which meta keys the feed will generate new meta. It tells the Decoder:

- Which basic meta key to register the new meta under (***name*** attribute)
- Which index format to store it under (***valuetype*** attribute). From the index configuration file, is one of these types: *Int8, UInt8, Int16, UInt16, Int32, UInt32, Int64, UInt64, UInt128, Float32, Float64, TimeT, Binary, Text, IPv4, IPv6, MAC*
- Because the Decoder is direction-aware and can differentiate between the source and destination sides of a session, if the index field is matched on the source side of a session, then register the new meta under a corresponding source-side key (***srcname*** attribute), to indicate that the generated value was on the source side of the session
- The ***destname*** attribute does likewise if the index field is matched under the destination side of a connection.
- Note that the ***srcname*** and ***destname*** attributes are optional. If they are omitted, newly-generated meta will be registered under the key defined by the ***name*** attribute.

- The CSV fields section, defined under the <Fields> XML tag. This section defines every column in the feed's CSV file, and instructs the Decoder which fields or columns in the CSV file are what. Each CSV column should get a separate **<Field/>** tag in this section. It tells the Decoder:
  - Which column number the entry is for (the index attribute)
  - Which type of column the feed is (the type attribute: either index or value)
  - If the column type is ***index*** and if the feed index fields use ranges of IP addresses, whether the column represents the low or high side of the range (the ***range*** attribute, with a value of either ***high*** or ***low***)

Now let's bring it all together. Using this example:

IF a session has **ip.src = 10.0.9.122**, **THEN** (register value '***London***' under meta key **office.src &&** register value '***HR***' under meta key **department.src**)

IF a session has **ip.dst = 10.0.12.45**, **THEN** (register value '***London***' under meta key **office.dst &&** register value '***Sales***' under meta key **department.dst**)

IF a session has **ip.src = 10.0.15.200**, **THEN** (register value '***Reading***' under meta key **office.src &&** register value '***TechSupport***' under meta key **department.src**)

We can see that the purpose of all of this is to tag sessions with additional meta which we can then drill and search on, which can give us an idea of where an asset is and the type of asset it is, which can help us prioritise incidents.

Creating a Custom Feed, Example 2 - MetaCallback Index Field

Rather than having our index field working on IP addresses, our second feed example utilises what's called a MetaCallback. This simply means that the feed's index field operates on non-IPv4 meta keys, such as a domain name. In this example, we willy classify websites by tagging sessions where meta key **alias.host** matches certain websites. Here is our feed CSV:

```
#Index on alias.host,feed.name
telegraph.co.uk,news
amazon.com,shopping
google.com,search engine
```

Other than the fact that we aren't using IP addresses and there is only a single index field, this works the same as the first example. Here's our feed definition file:

```
<FDF>

  <FlatFileFeed name="TargetWebsites" path="feed-ex2.csv" separator="," comment="#">

    <MetaCallback name="alias.host" valuetype="Text" apptype="80" truncdomain="true"/>

    <LanguageKeys>

      <LanguageKey name="feed.name" valuetype="Text" />

    </LanguageKeys>

    <Fields>

      <Field index="1" type="index" key="alias.host"/>

      <Field index="2" type="value" key="feed.name"/>

    </Fields>

  </FlatFileFeed>

</FDF>
```

Our feed definition is also very much the same as our first example, with the exception of a new **<MetaCallback>** tag. The presence of the **MetaCallback** tag tells the Decoder that this feed will search against a non-IPv4 meta key, in this instance the ***alias.host*** meta key. The attributes in the <Metacallback> tag are:

- ***name*** - This is the meta key you want your index field to match against

- ***valuetype*** - The type of meta key format, as defined in the index configuration files. This should be '***Text***' if dealing with hostnames
- ***apptype*** - This defines which service type the feed will solely operate on. This refers to the ***service*** meta key. In this case, for the feed to tag a session, meta key ***service*** must have a value of ***80*** (http).
- ***truncdomain*** - If set to ***true***, this attribute causes the Decoder to strip all subdomains from the session's index key value before matching it against the CSV's index field. E.g. If ***alias.host*** **= 'www.somedomain.com'**, then the feed will strip the **'www.'** from the session's index value, and the feed will match the feed's index column (i.e. ***alias.host***) against **'somedomain.com'** rather than **'www.somedomain.com'**.

The rest of the XML is the same as the first example, except that in the first index column (i.e. ***<Field index="1"***), you must add ***key="alias.host"*** to enable the metacallback, there is no ***range*** attribute on the index field, and there is no second index column. You must still define a **<LanguageKeys>** section to tell the field where to register the value columns, and also a **<Fields>** section to define the CSV columns.

Bringing it all together again. In this example 2:

IF a session has **alias.host** = **www.telegraph.co.uk**, **THEN** (register value '***news***' under meta key feed.name)

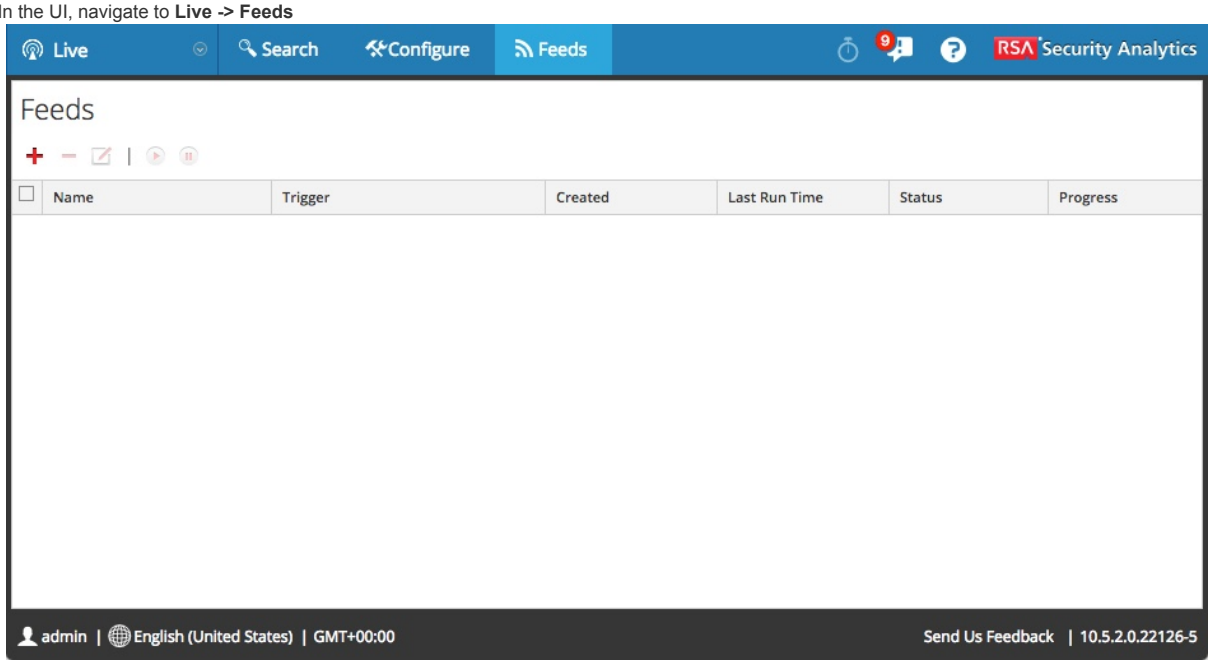
IF a session has **alias.host** = **amazon.com**, **THEN** (register value '***news***' under meta key feed.name)

IF a session has **alias.host** = **www.google.com**, **THEN** (register value '***search***' under meta key feed.name)

Installing a Custom Feed

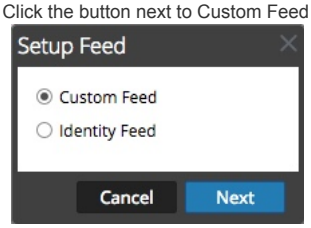
Once you've created your feed and feed definition, it must then be installed on your Decoder(s). You can most easily accomplish this through the Feed Wizard of SA.

1. In the UI, navigate to **Live -> Feeds**



2. Now hit the **+** button, and the Custom Feed Wizard will come up

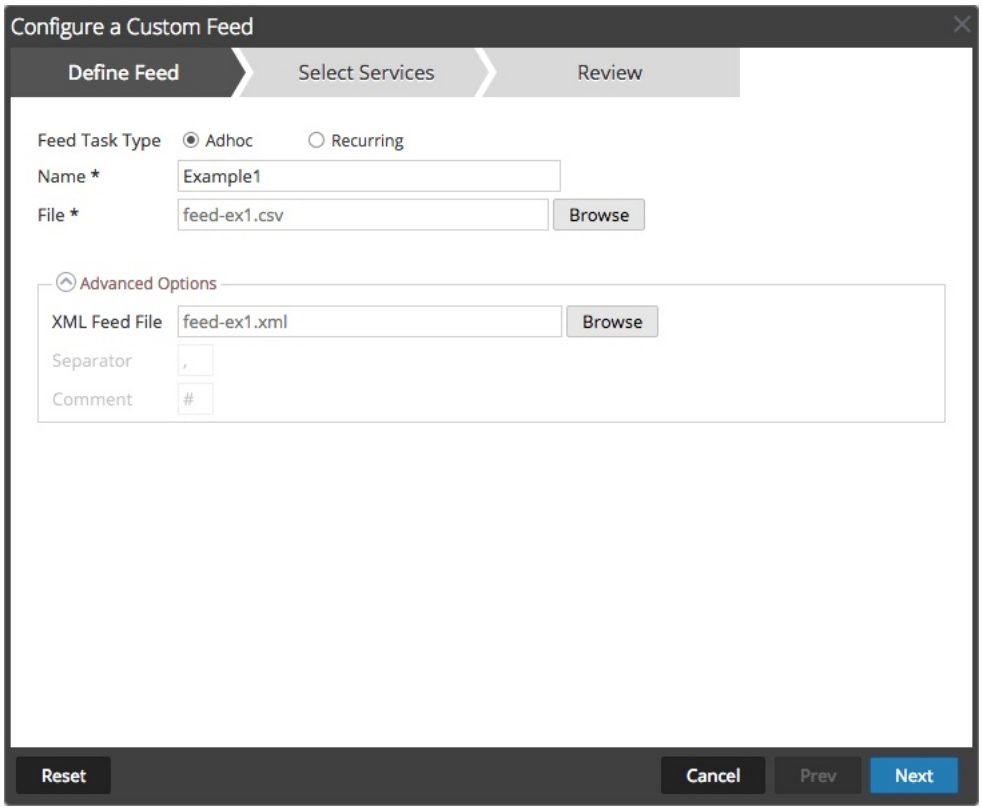
3. Click the button next to Custom Feed



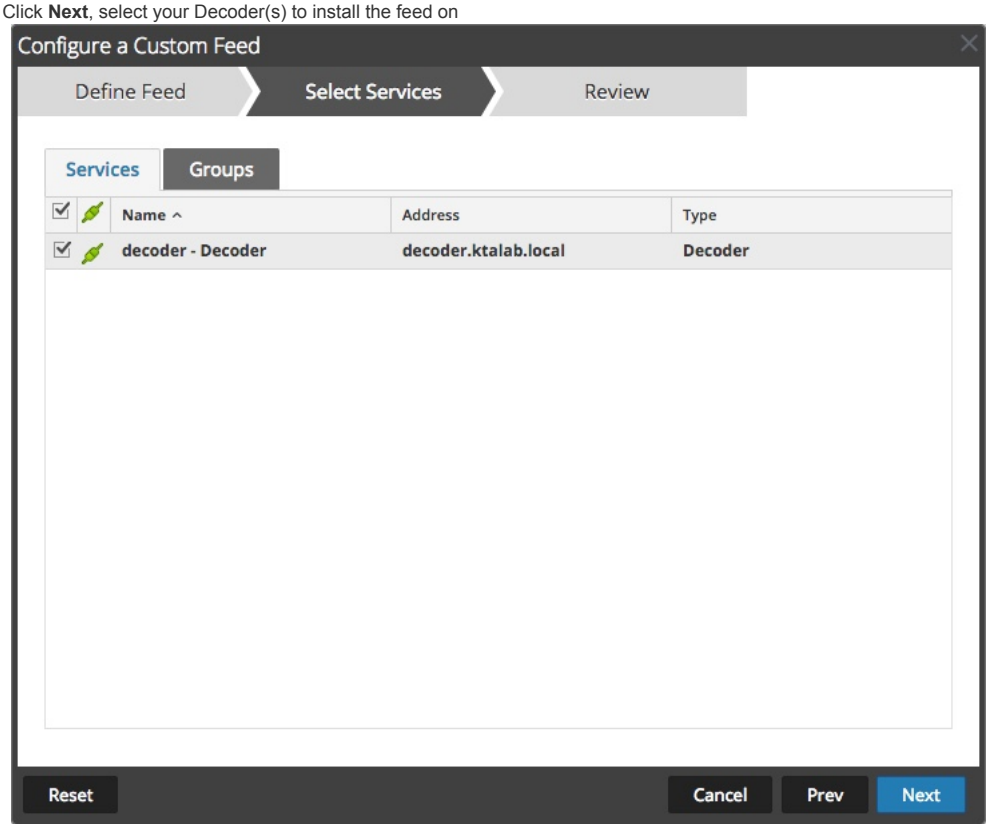
4. In the **Name** text box, enter a friendly name for your custom feed

5. In the file input box, select your CSV file

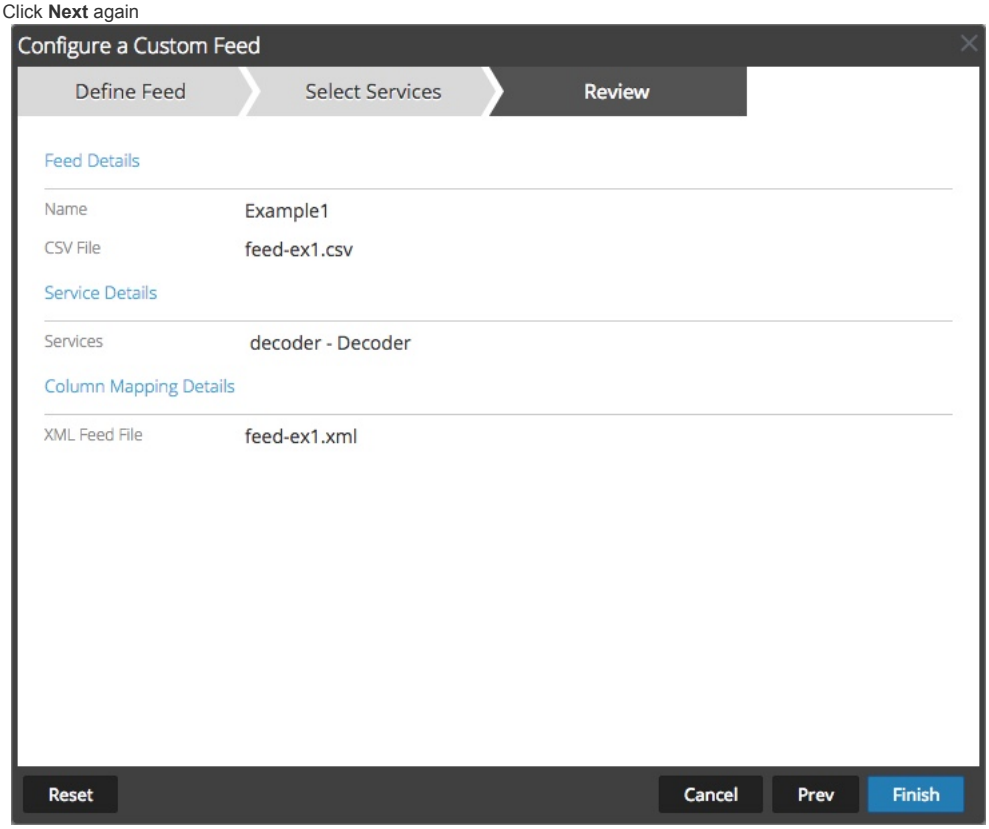
6. Under **Advanced Options**, select your feed definition XML file.



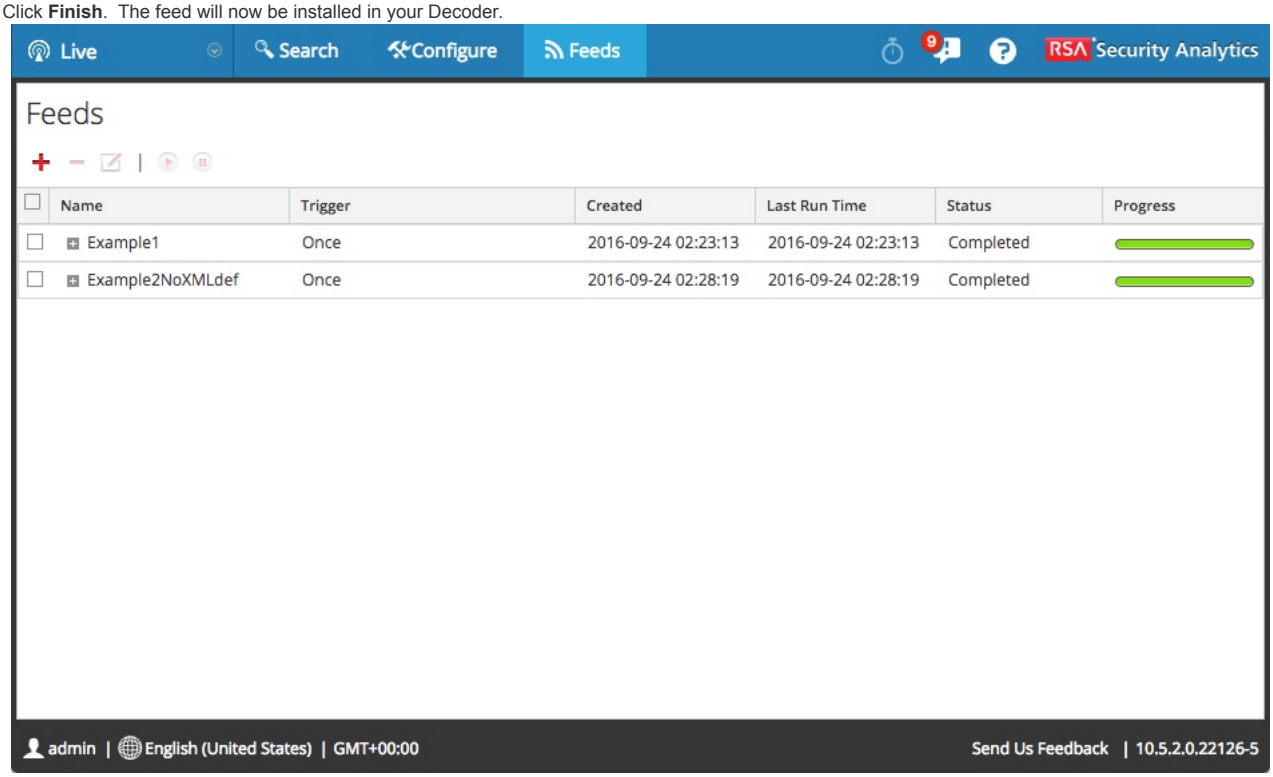
7. Click **Next**, select your Decoder(s) to install the feed on



8. Click **Next** again



9. Click **Finish**. The feed will now be installed in your Decoder.

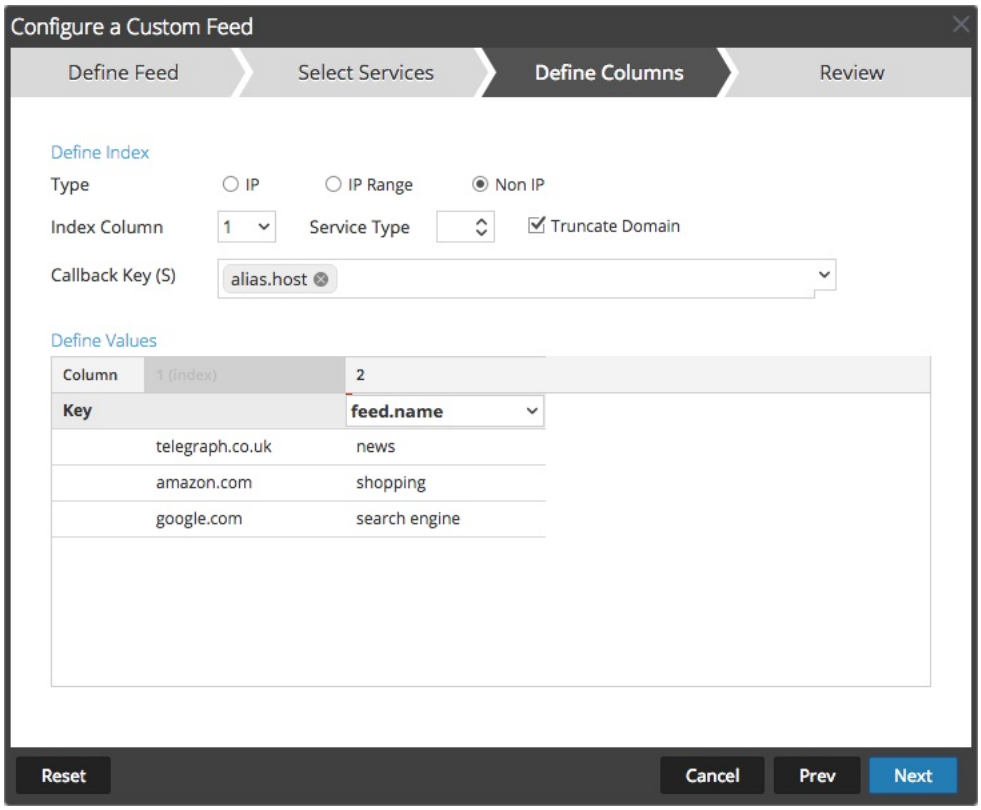


Actually, one can actually do this without even creating a feed definition file. The feed wizard enables one to map CSV columns to SA meta keys. To do this, don't add your feed definition XML file to the advanced options in the feed wizard. After you select your services to deploy the feed to, you will be taken to the **Define Columns** tab, where you can define all the options we defined in the feed XML file.

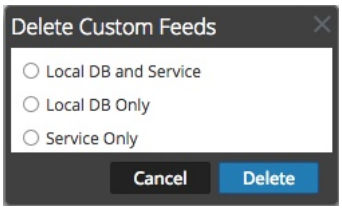
Wait, There's a Feed Wizard?!!!

That's right, ladies and gentlemen. Everything we just went through can be accomplished much more easily through use of the Custom Feed Wizard. But it's important to learn the basic concepts of feeds before we take a shortcut through the wizard.

Here is the **Define Columns** tab set up exactly the same way as our custom XML definition file.



Removing a Custom Feed



Custom feeds can easily be removed from both the UI and Live by selecting the a custom feed in Live -> Feeds, and clicking the **-** button. You will be presented with an option to:

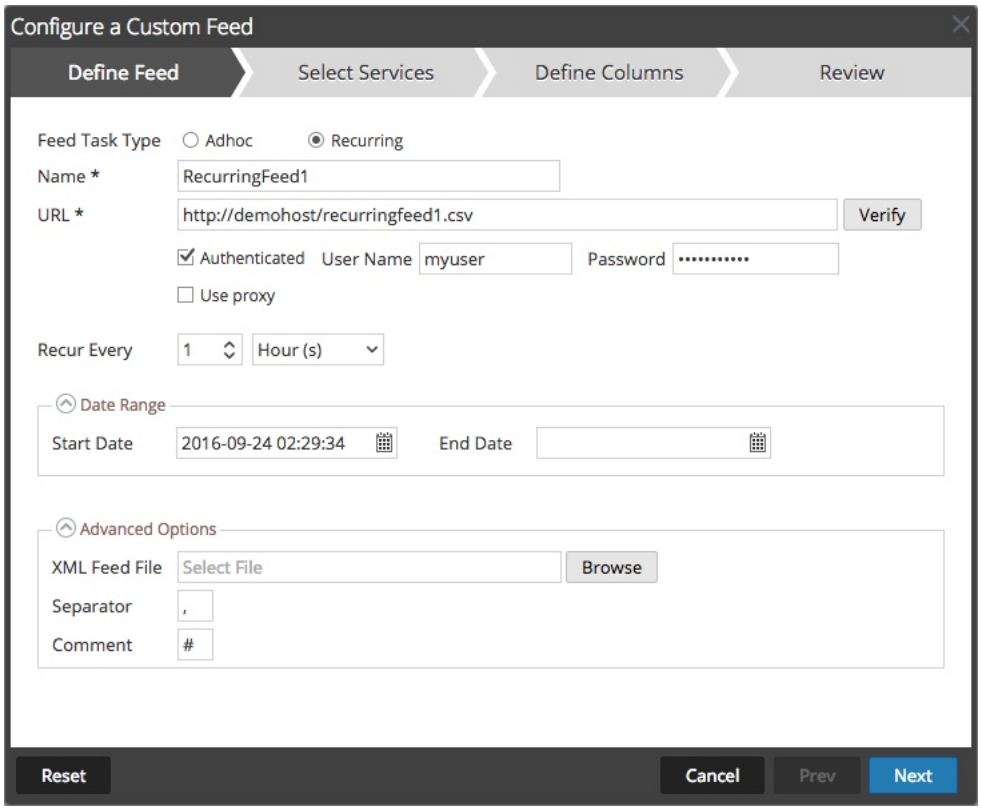
- Remove the feed from both the Decoder and UI (Local DB and Service)
- Remove the feed from the UI (Local DB Only)
- Remove the feed from the Decoder (Service Only)

Recurring Feeds

You may have noticed a radio button called '**Recurring**' on the first tab of the Feed Wizard. This means that updated versions of feeds can be pushed at predefined intervals.

Take this use case: your business is an Active Directory environment with over 1,000 users spread over at least as many PC's. You want to know what the hostname of every PC in your environment is, and this information isn't typically present in most network sessions. So you create a bespoke script to generate a CSV of the contents of your environment's DNS and/or DHCP servers. Because DNS and DHCP changes frequently, you must also update your feed regularly, so that SA doesn't have outdated host information. Thus, you run your script on a schedule.

SA supports this use-case through Recurring Feeds. On the first tab of the Feed Wizard, simply click the Recurring radio button and you will see that SA will pull your updated feed from an HTTP URL. The option are fairly self-explanatory, and the rest of the wizard works the same. You can still use a feed definition XML file or you can define your fields with the Feed Wizard.



Application Rules

Application (or ***app***) rules are a powerful order-based mechanism through which additional meta can be created or suppressed based on definable conditions, and sessions either ***truncated*** or completely ***filtered*** by the Decoder. The meta it creates can also be ***transient***. App rules employ a fairly simple logic to perform actions based on conditions which the Administrator specifies. They operate from a conditional string, specifying **AND/OR** syntax and logical grouping with parentheses, using simple comparison operators.

Comparison Operators

Application rules support the following comparison operators. They are exactly the same as for Investigation queries.

Operator	Description
=	Equality operator
!=	Inequality operator
&&	Logical AND operator
||	Logical OR operator
-u	Upper boundary, or greater-than operator. E.g. for selecting all TCP destination ports greater than 1024 (TCP high ports), use "tcp.dstport = 1024-u"
l-	Lower boundary, or less-than operator. E.g. for selecting all TCP ports below 1024 (TCP low ports), use "tcp.dstport = l-1024"
exists	Does the given meta key exist in the session (is it defined)? This is extremely fast and works well with keys indexed at the key level
!exists	Does the given meta key not exist in the session (is it not defined)? E.g. "attachment exists". This is extremely fast and works well with keys indexed at the key level
begins	Does a meta value start with a given string? E.g. "client begins 'mozilla/5.0'".
contains	Does a meta value contain a given string? E.g. "alias.host contains 'amazon'". This is computationally more expensive than =
ends	Does a meta value end with a given string? E.g. "filename ends '.exe'". This is computationally more expensive than =.
length	Is the length of a string equal to a given value? E.g. "tld length 3"
regex	Perform a regular expression match on the value. E.g. "email regex '^.+pwned.*+@.+\..+$'"
( )	Grouping operator. By enclosing an expression in parentheses, a new logical expression is created. These are usually used in conjunction with **&&** and **||** operators
- (hyphen)	Range operator. Only applies to numeric values. E.g. "udp.dstport=68-69"
, (comma)	List separator. Allows one to create a list of values. E.g. "ip.src=192.168.0.0/16,10.0.0.08"

IP addresses can be matched on individual IP's, a comma-separated list of IP's, or using CIDR notation.

```
ip.src=10.1.2.3
ip.src!=10.2.3.4,10.5.6.7.10.8.7.6
ip.src=10.5.4.0/24
```

As mentioned previously, app rules are order-based, meaning they are processed from top-to-bottom, one by one. An app rule can be configured to stop the processing of more rules for the given session, meaning that lower rules will not be applied.

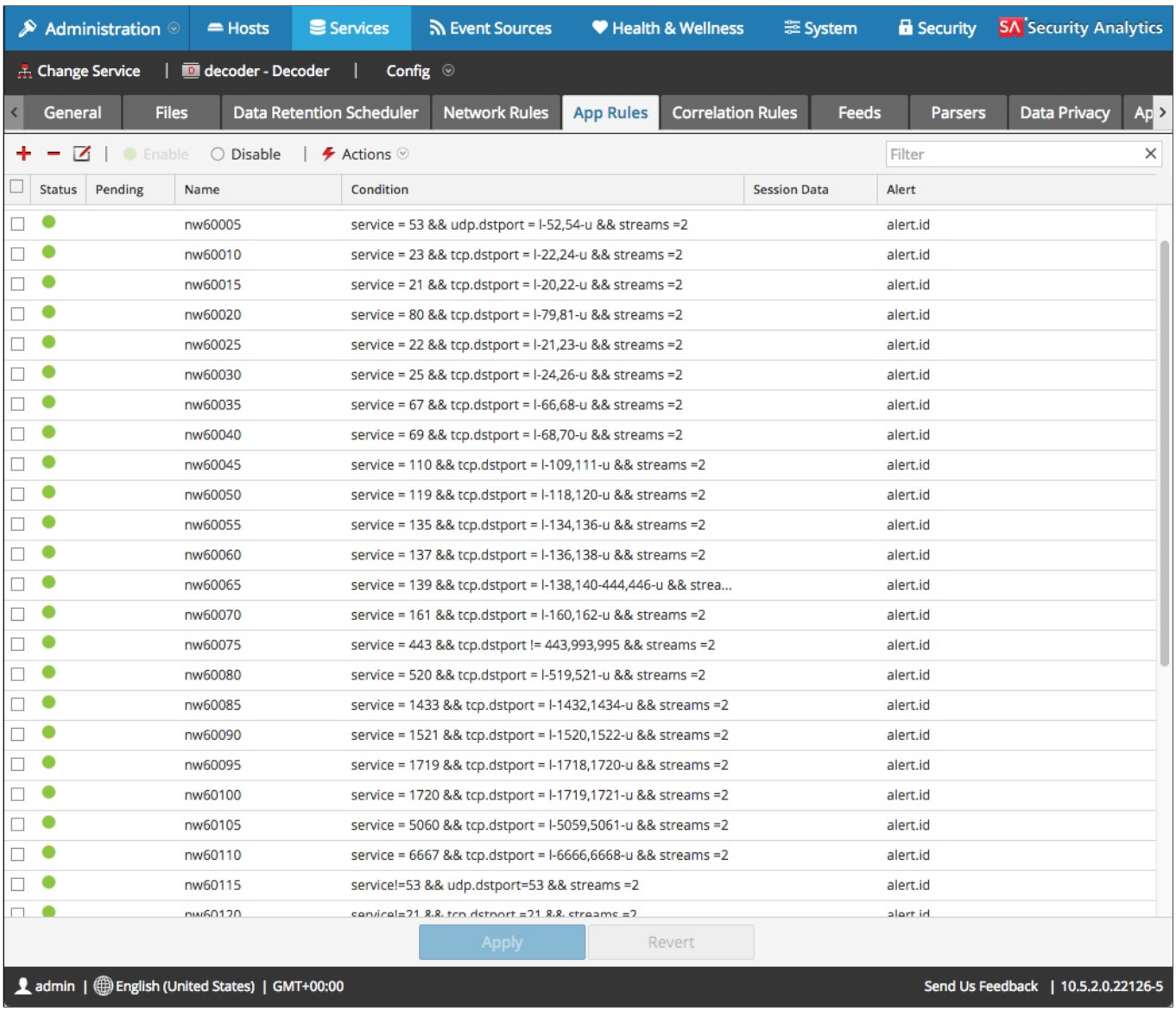
Transient meta is meta which is created only in the Decoder's RAM, but never saved to disk. This useful when you wish to create meta to be used in a lower app rule's conditional statement, but do not need to save that value in the index and meta DB.

Filtered means that a given session is completely dropped, both meta and packets. You don't even care whether it happened.

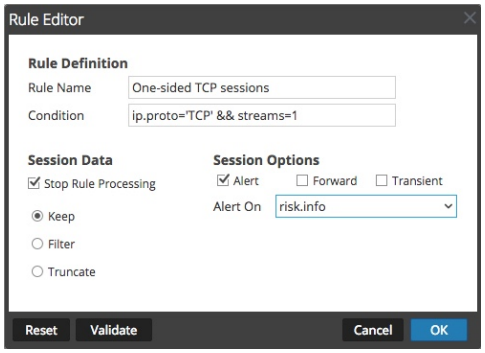
Truncated means a session's meta is retained, but its packets are not. This is desirable when you have traffic that you wish to know about, but is of limited or no forensic value. Such an example might be SSL traffic which cannot be decrypted (though it's arguable whether the packets are of no forensic value). Maybe you've decided that because a certain type of session is of limited utility, you don't want its packets taking up precious storage on your Decoder's filesystem, which reduces retention.

App rules are often used to generate new meta. This is the session option called ***alert***. Alert in this context does not mean an email or syslog is sent to someone, just that a meta value is registered under a selected meta key. The meta value created is actually the name of the app rule itself.

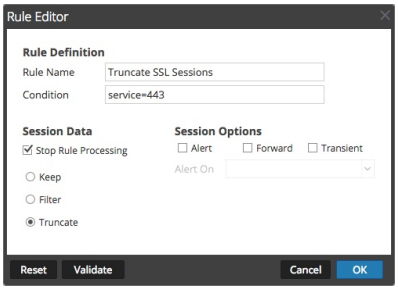
In the Decoder's order of operations, app rules are processed before feeds. Therefore, meta you create with an app rule can also be used by a feed to generate additional meta. You can see this approach used with app rules distributed via RSA live, where the app rules have cryptic names like 'nw60070', which is meaningless to most humans. These values then pass through an RSA Live feed which generates human-readable names for the rules which are then recorded under one of three keys: **risk.info, risk.warning,** or **risk.suspicious**, case depending.



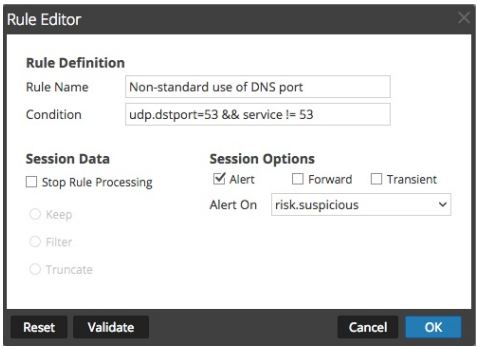
App Rule Page



Example 1 - Generate new meta under key risk.info when on-sided TCP sessions are found



Example 2 - Keep meta but drop packet data for all SSL sessions



Example 3 - Generate meta under risk.suspicious key if non-DNS traffic over UDP port 53

App Rule Examples

Here are a few common uses for app rules:

- Generate a meta value when a non-indexed key exists or matches a certain value. App rules can use either indexed, transient, or non-indexed meta. For instance, single-sided packet sessions, which could indicate traffic to a non-existent host, a firewall drop, or possibly a UDP session. The **streams** meta key indicates how many streams are in a session (either 1 or 2), but **streams** isn't an indexed key. Rather than indexing this key, you can create an app rule called "Single-sided session" where the condition is "**streams=1**", and alert on indexed meta key **content**. Now you can write more rules or investigate on query "**content="Single-sided session"**", rather than indexing the whole streams key. This is generally more efficient.

- App rules can be combined to produce a desired result. For instance, RSA has Live content for Malware Analysis, which includes an app rule named **spectrum.consume** with a condition of "**content='spectrum.analyze' && content!='spectrum.filter'**", which alerts on the **content** meta key. In order to find sessions which it should inspect, Malware Analysis then performs a query for "**content='spectrum.consume'**"

  Did you notice the part of the rule which says "**content!='spectrum.filter'**" ? By tagging sessions with **content = 'spectrum.filter'**, you can whitelist sessions so that they are not analysed by Malware Analysis. To do this, you can create an app rule or feed which alert on the **content** key, with a name of **spectrum.filter**.

  For example, if you know that traffic between specific hosts and with a specific filename is benign, then create an app rule called 'spectrum.filter', that alerts on the **content** key, with a condition of "**ip.src=1.2.3.4 && ip.dst=4.5.6.7 && filename='bigUpdate.exe'**". Make sure to place this new rule above the **spectrum.consume** rule, or else it won't have the desired effect. Malware Analysis will no longer inspect sessions between these two hosts where the filename is **bigUpdate.exe**.

BPF Filters

BPF stands for Berkeley Packet Filter. BPF filters are packet filters that get applied at the NIC level, which allow the Decoder to filter out traffic from capture that we don't find interesting, and that we don't want consuming SA resources. These filters typically incur little to no performance penalty as they are applied at the NIC rather than the CPU. BPF filters use standard libpcap syntax, which is the same syntax used by tcpdump. BPF syntax is beyond the scope of this course, but a few examples are provided. BPF filters can be used to whitelist or blacklist traffic.

BPF filters can be accessed from **Administration -> Services -> <Decoder> -> Config -> General** under **Adapter,** next to **Berkeley Packet Filter**. Simply paste the filter string into the field and hit **Apply**. Be advised that there is no syntax checking when applying BPF filters. To test your filter string, use it with tcpdump.

Please keep in mind that libpcap filters are applied to every packet coming through the capture interface(s), so be careful when specifying source and destination in filter syntax. For instance, if you use "***not dst port 22***", then you will only filter out SSH traffic from the source to the destination, but not traffic sent in the other direction. Also, multiple libpcap filters can be combined, but when doing so, it may be necessary to group the separate filters logically using parentheses.

```
not net 10.15.8.0/24
net 192.168.19.0/24 and port 53
not broadcast and not multicast
```

```
[[root@decoder ~]# tcpdump -nni eth0 port 22
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
17:30:19.953646 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 939617035:939617231, ack 831425751, win 239, options [nop,nop,TS val 5577513 ecr 798075672]
, length 196
17:30:19.955558 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 196:408, ack 1, win 239, options [nop,nop,TS val 5577515 ecr 798075672], length 212
17:30:19.956463 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 408:604, ack 1, win 239, options [nop,nop,TS val 5577516 ecr 798075672], length 196
17:30:19.957465 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 604:800, ack 1, win 239, options [nop,nop,TS val 5577517 ecr 798075672], length 196
17:30:19.958436 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 800:996, ack 1, win 239, options [nop,nop,TS val 5577518 ecr 798075672], length 196
17:30:19.959409 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 996:1192, ack 1, win 239, options [nop,nop,TS val 5577519 ecr 798075672], length 196
17:30:19.960673 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 1192:1388, ack 1, win 239, options [nop,nop,TS val 5577520 ecr 798075672], length 196
17:30:19.962506 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 1388:1584, ack 1, win 239, options [nop,nop,TS val 5577522 ecr 798075672], length 196
17:30:19.963826 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 1584:1780, ack 1, win 239, options [nop,nop,TS val 5577523 ecr 798075672], length 196
17:30:19.964829 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 1780:1976, ack 1, win 239, options [nop,nop,TS val 5577524 ecr 798075672], length 196
17:30:19.965383 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 1976:2172, ack 1, win 239, options [nop,nop,TS val 5577525 ecr 798075672], length 196
17:30:20.060176 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 196, win 4089, options [nop,nop,TS val 798075778 ecr 5577513], length 0
17:30:20.060386 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 2172:2368, ack 1, win 239, options [nop,nop,TS val 5577619 ecr 798075778], length 196
17:30:20.060510 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 408, win 4089, options [nop,nop,TS val 798075782 ecr 5577515], length 0
17:30:20.062044 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 2368:2852, ack 1, win 239, options [nop,nop,TS val 5577621 ecr 798075782], length 484
17:30:20.063443 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 2852:3048, ack 1, win 239, options [nop,nop,TS val 5577623 ecr 798075782], length 196
17:30:20.064574 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 3048:3244, ack 1, win 239, options [nop,nop,TS val 5577624 ecr 798075782], length 196
17:30:20.065951 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 604, win 4089, options [nop,nop,TS val 798075782 ecr 5577516], length 0
17:30:20.066054 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 3244:3440, ack 1, win 239, options [nop,nop,TS val 5577625 ecr 798075782], length 196
17:30:20.066123 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 800, win 4083, options [nop,nop,TS val 798075782 ecr 5577517], length 0
17:30:20.066165 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 996, win 4089, options [nop,nop,TS val 798075782 ecr 5577518], length 0
17:30:20.066188 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 1192, win 4089, options [nop,nop,TS val 798075783 ecr 5577519], length 0
17:30:20.066209 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 1388, win 4089, options [nop,nop,TS val 798075783 ecr 5577520], length 0
17:30:20.066231 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 1584, win 4089, options [nop,nop,TS val 798075785 ecr 5577522], length 0
17:30:20.066247 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 1780, win 4089, options [nop,nop,TS val 798075785 ecr 5577523], length 0
17:30:20.066488 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 1976, win 4089, options [nop,nop,TS val 798075786 ecr 5577524], length 0
17:30:20.066711 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 2172, win 4089, options [nop,nop,TS val 798075786 ecr 5577525], length 0
17:30:20.067535 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [.], seq 3440:4780, ack 1, win 239, options [nop,nop,TS val 5577627 ecr 798075786], length 1340
17:30:20.067723 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 4780:4916, ack 1, win 239, options [nop,nop,TS val 5577627 ecr 798075786], length 136
17:30:20.068458 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 4916:5272, ack 1, win 239, options [nop,nop,TS val 5577628 ecr 798075786], length 356
17:30:20.069405 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 5272:5468, ack 1, win 239, options [nop,nop,TS val 5577629 ecr 798075786], length 196
17:30:20.070468 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 5468:5664, ack 1, win 239, options [nop,nop,TS val 5577630 ecr 798075786], length 196
17:30:20.071602 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 5664:5860, ack 1, win 239, options [nop,nop,TS val 5577631 ecr 798075786], length 196
17:30:20.072419 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 5860:6056, ack 1, win 239, options [nop,nop,TS val 5577632 ecr 798075786], length 196
17:30:20.073403 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 6056:6252, ack 1, win 239, options [nop,nop,TS val 5577633 ecr 798075786], length 196
17:30:20.074538 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 6252:6448, ack 1, win 239, options [nop,nop,TS val 5577634 ecr 798075786], length 196
17:30:20.074801 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 6448:6644, ack 1, win 239, options [nop,nop,TS val 5577634 ecr 798075786], length 196
17:30:20.075542 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 6644:6840, ack 1, win 239, options [nop,nop,TS val 5577635 ecr 798075786], length 196
17:30:20.076388 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 6840:7036, ack 1, win 239, options [nop,nop,TS val 5577636 ecr 798075786], length 196
17:30:20.077383 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 7036:7232, ack 1, win 239, options [nop,nop,TS val 5577637 ecr 798075786], length 196
17:30:20.159275 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 2368, win 4089, options [nop,nop,TS val 798075882 ecr 5577619], length 0
17:30:20.159476 IP 10.0.0.4.22 > 10.255.254.2.56190: Flags [P.], seq 7232:7428, ack 1, win 239, options [nop,nop,TS val 5577718 ecr 798075882], length 196
17:30:20.159575 IP 10.255.254.2.56190 > 10.0.0.4.22: Flags [.], ack 2852, win 4080, options [nop,nop,TS val 798075883 ecr 5577621], length 0
```

Example - Using tcpdump with libpcap filter

Network Rules

You may at some point notice something in the Decoder configuration screens referring to Network Rules, or **net rules** for short. **DO NOT USE THESE!!!** There is nothing that a network rule can do that an app rule or BPF filter cannot, but a net rule is applied in the CPU to every packet that is captured by the Decoder, causing a major performance hit.

15. Log Decoder

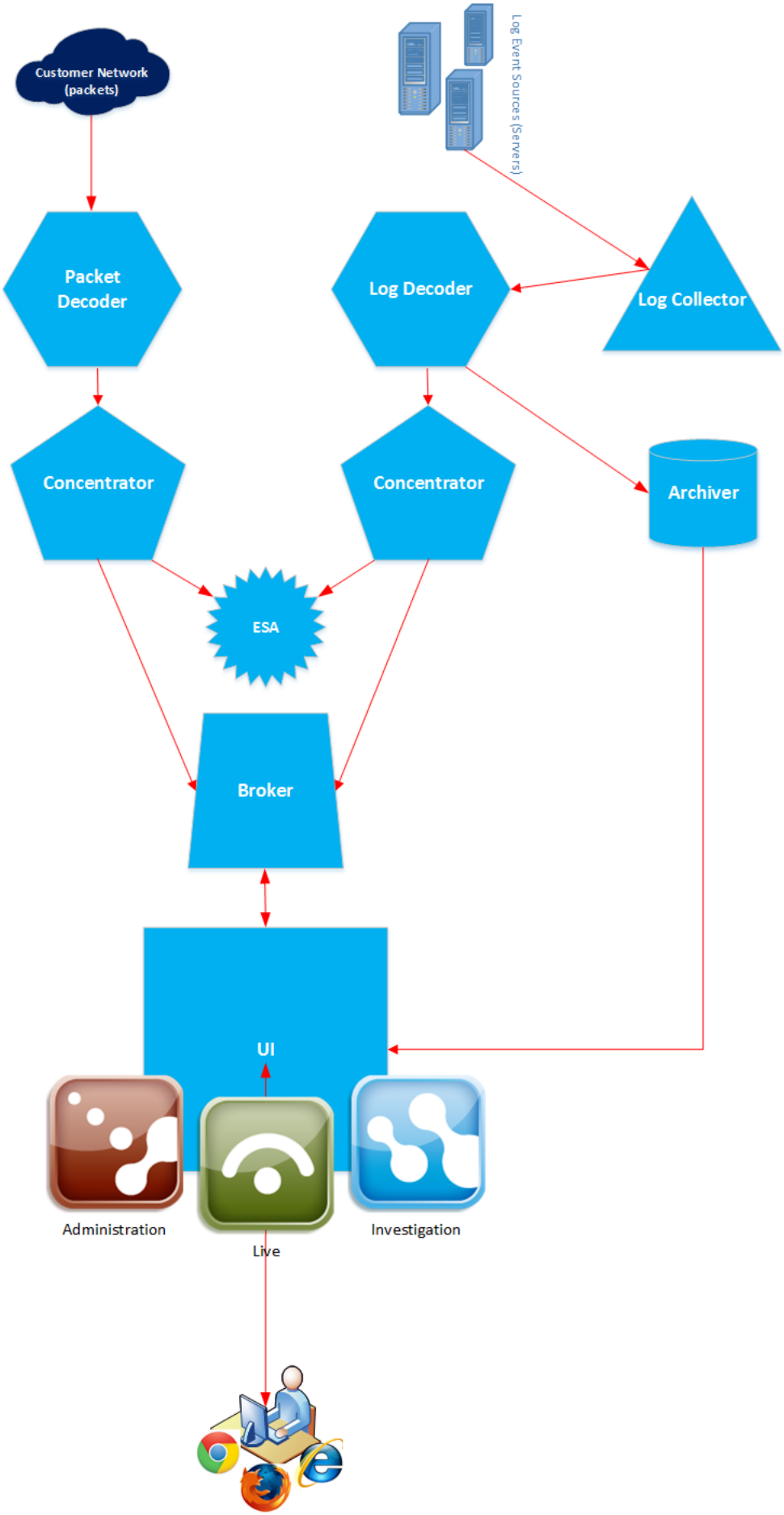
As one might guess, the job of the Log Decoder is to parse and store log data. From a broad perspective, it works in almost exactly the same manner as a Packet Decoder, but with two exceptions:

- Rather than using a NIC in promiscuous mode to capture packets, it listens on UDP and TCP ports 514 for log traffic.
- It uses RSA enVision-style log parsers rather than LUA parsers.

Top Tip: The original name of the Log Decoder was NetWitness Panorama, and then it was NetWitness for Logs (NWFL). You may still sometimes see both terms in use around documentation and discussions.

Note: Much information in this lesson will be exactly the same as for the Packet Decoder and will appear in this lesson verbatim, so if we've already covered a concept, then we will likely skip over it.

```
The Log Decoder listens on port TCP 50002 / 56002 (SSL).
```



SA Stack Diagram

The Log Decoder's job is to do the following:

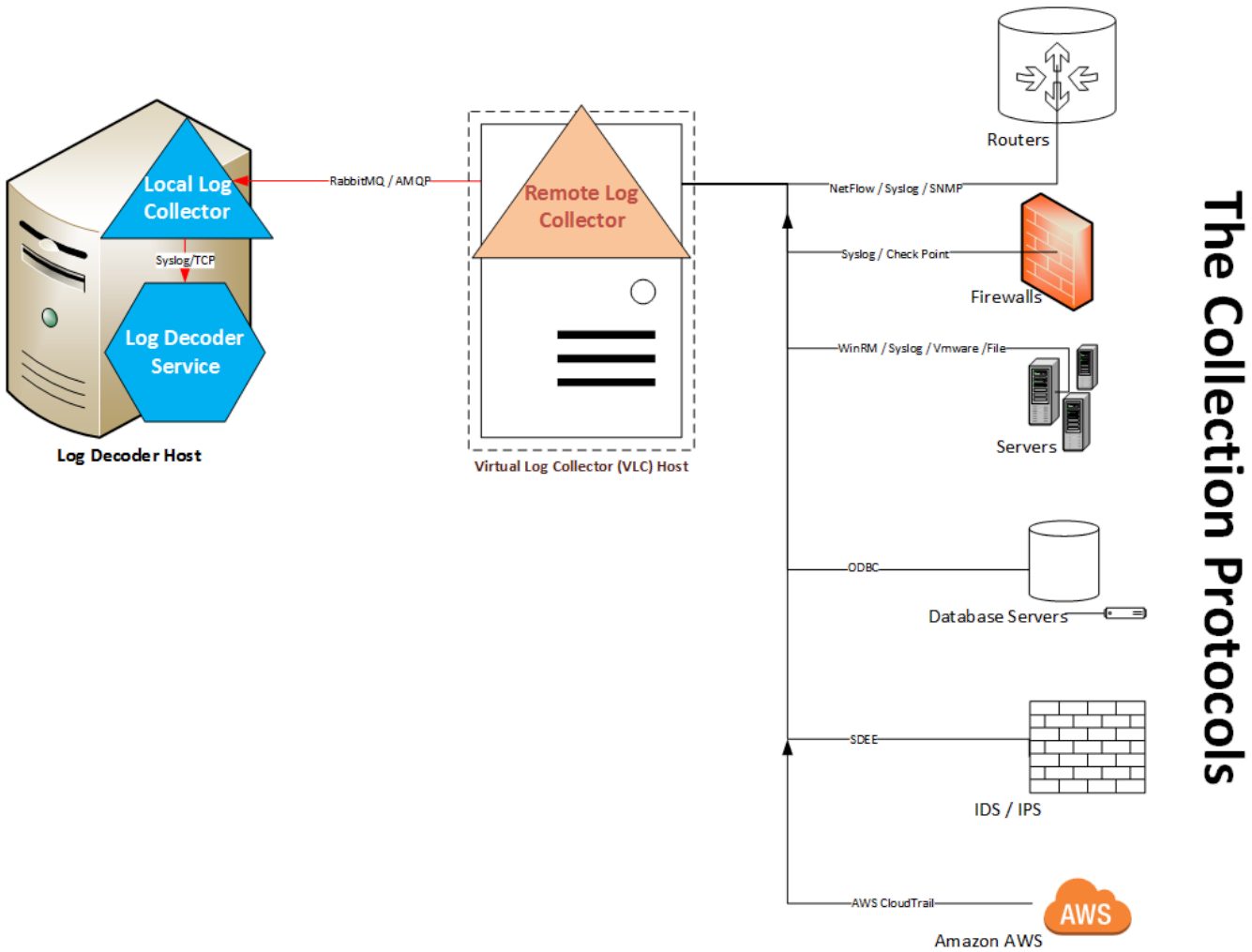
1. **Capture** - is the process of reading packet or log data off of the wire and into memory

2. **Assembly** - the process of separating packet data and log data into individual sessions
3. **Parsing** - The process of generating meta from assembled packet or log streams
4. **Rules & feeds** - Generate additional meta on sessions based on other values, such as a rule or a feed from an external data source (e.g. a list of users, DHCP feed, etc). Can also be used to truncate or drop traffic based on certain characteristics.
5. **Database storage** - Save packet, log, meta, and session data to disk for future retrieval.
6. **Indexing** - Create a searchable index of metadata which can be used to quickly drill through and find sessions matching certain characteristics

Getting Logs into SA With the Log Decoder's Helper - The Log Collector

Whilst the Packet Decoder functions happily on its own, requiring only a single piece of hardware to feed it, getting logs into SA is usually far more cumbersome. The Log Decoder in almost all cases requires the help of another service called a Log Collector. Even though the Log Decoder can listen for syslog traffic, not all log **event sources** are capable of sending their logs in syslog format. Enter the Log Collector. We'll cover specifics of the Log Collector in a separate lesson, but in order to understand the Log Decoder, one must understand that it is closely bound with Log Collectors.

An **event source** is a device or service whose logs are ingested by SA. Generally speaking, every log source will require its own configuration to push its log data out to SA, and/or configuration within SA to collect logs from that source. As mentioned, the administrative overhead is many times higher for onboarding log sources than it is for packets.



Log Collection Flow

The Three Collectors

There are three types of Log Collectors:

- **Local Log Collectors** - These live on the same host as the Log Decoder. Their job is to accept logs from Remote Log Collectors (or VLC's) and inject them into the Log Decoder via the syslog protocol.

- **Virtual Log Collectors (VLC's) or Remote Log Collectors** - These devices are standalone hosts running only a Log Decoder service and are almost always always run from a virtual machine. Thus the 'virtual' moniker. VLC's do most of the heavy lifting when it comes to collecting logs, and run all log collection services. VLC's send their logs on to Local Collectors via a glue service called **RabbitMQ**. Technically there is little difference between a Local Collector and a VLC, except that VLC's can collect syslog traffic, whereas Local Collectors cannot, as the Log Decoder is already listening for syslog traffic.

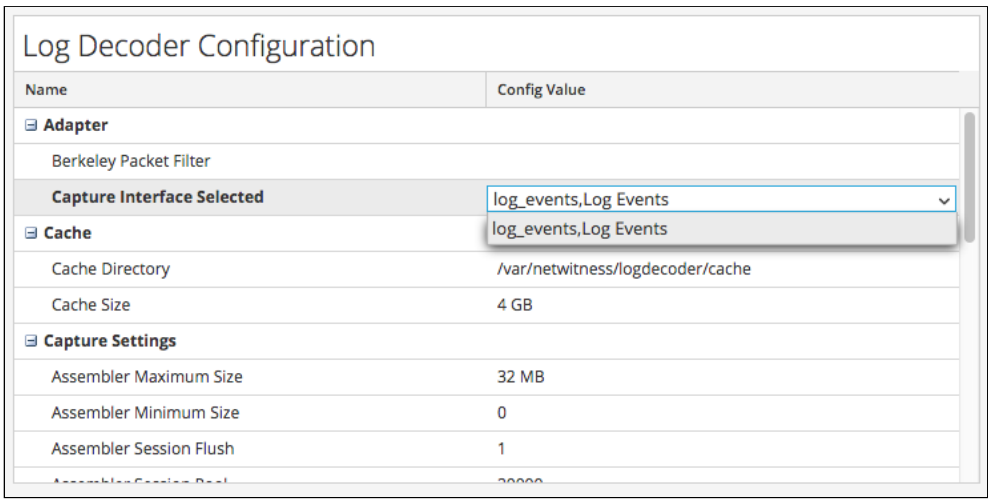
- **Legacy Windows Collectors** - These collectors are required to collect logs from pre-Vista Windows hosts. They run on a Windows 2008 64-bit system, and forward their logs to another Log Collector.

Storage Requirements

Decoders typically have the greatest storage requirements of any SA service. Many organisations are under directives to store captured data for a certain number of days, especially log data, as most regulatory requirements presently address logs rather than packets. When dealing with the volume of data which is typical of SA, the storage requirements can be very large. With the exception of Hybrids, Log Decoders are paired with one or more DAC's or a SAN. It is not unusual to see five or more DAC's daisy chained together.

Choosing a Capture Interface

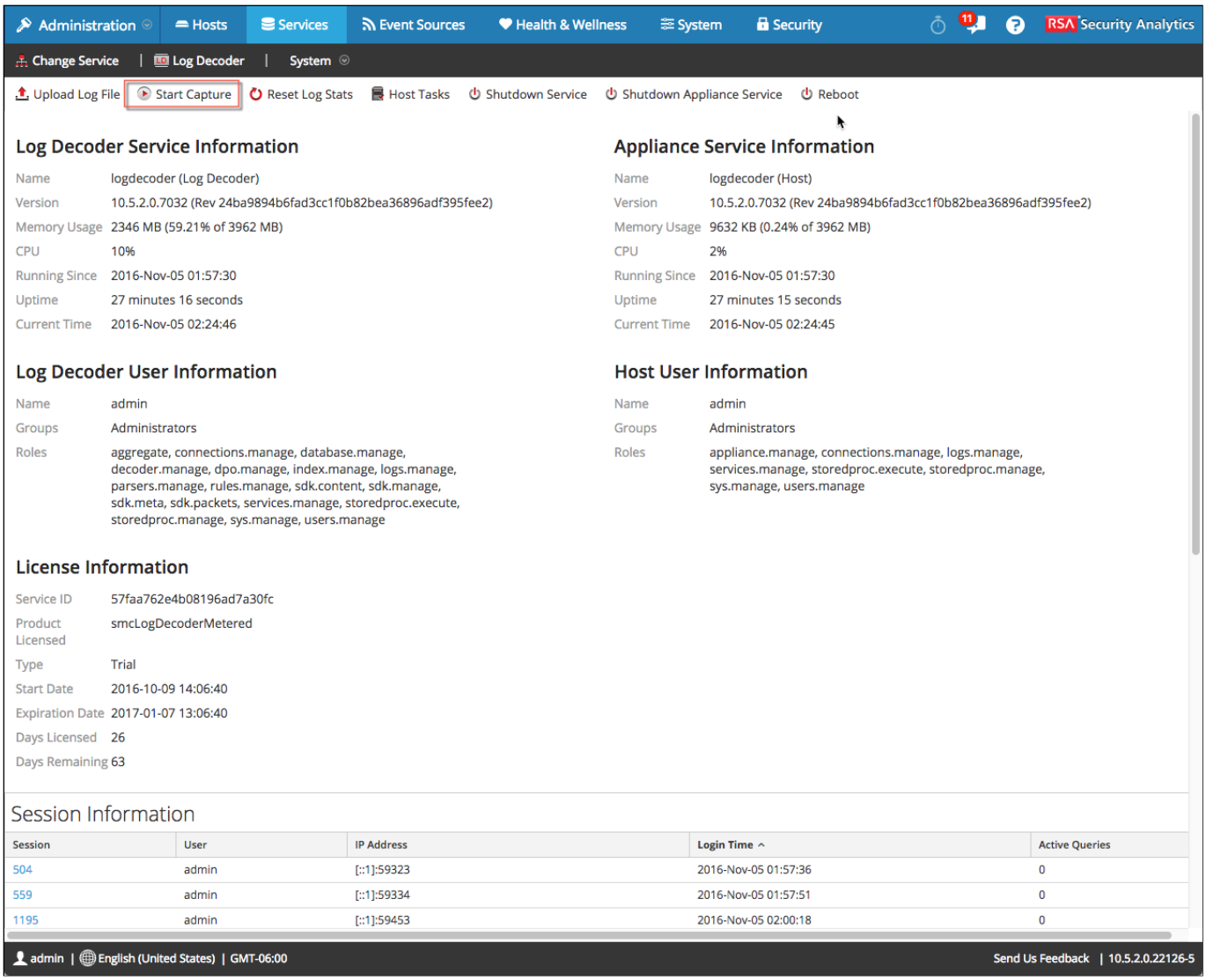
As there is no dedicated interface for capture in a Log Decoder, the Log Decoder simply listens on port 514 on its managements. It might seem a bit silly, then, that it still requires you to select a capture interface, even though there's only one choice, which is called "log_events, Log Events". You must do this prior to starting capture, or else capture won't start. This can be done from the UI in **Administration -> Services -> <Log*Decoder*> -> Config**. Be sure to hit 'apply' after selecting an interface.



Starting and Stopping Capture

Log Decoders by default do not start capturing logs automatically. Capture must be started by an Administrator from **Administration -> Services -> <Log*Decoder*> -> System**. You can tell whether capture is started by whether the button says "Start Capture" or "Stop Capture".

Once capture is started, the Log Decoder will start listening on TCP and UDP ports 514, promiscuously receiving any syslog traffic which is sent to it. On a Log Decoder, the secondary NIC's will not be used.



Capture Autostart

As noted, capture does not by default start automatically when the Log Decoder process starts. This must be done from **Administration -> Services -> <Log*Decoder*> -> Config.** Tick the box next to 'Capture Autostart' and click **Apply**. The next time you restart the Log Decoder process, capture should start automatically.

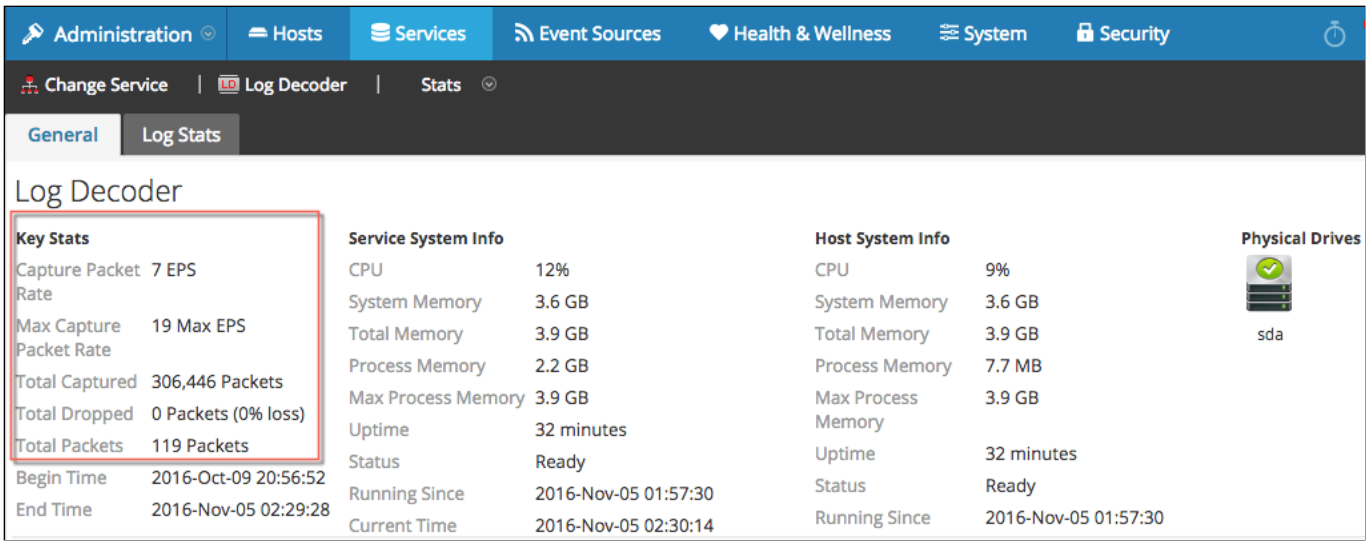
Ensuring That Capture is Working

After starting capture, it's always a good idea to visit the Stats page of the Log Decoder to ensure that capture is functioning correctly. One can look at the capture rate and max capture rate statistics to ensure that traffic is being received by the Decoder. Another thing to make sure of is that the Log Decoder is not dropping any packets, either.

Log ingestion is measured in Events Per Second, or **EPS**.

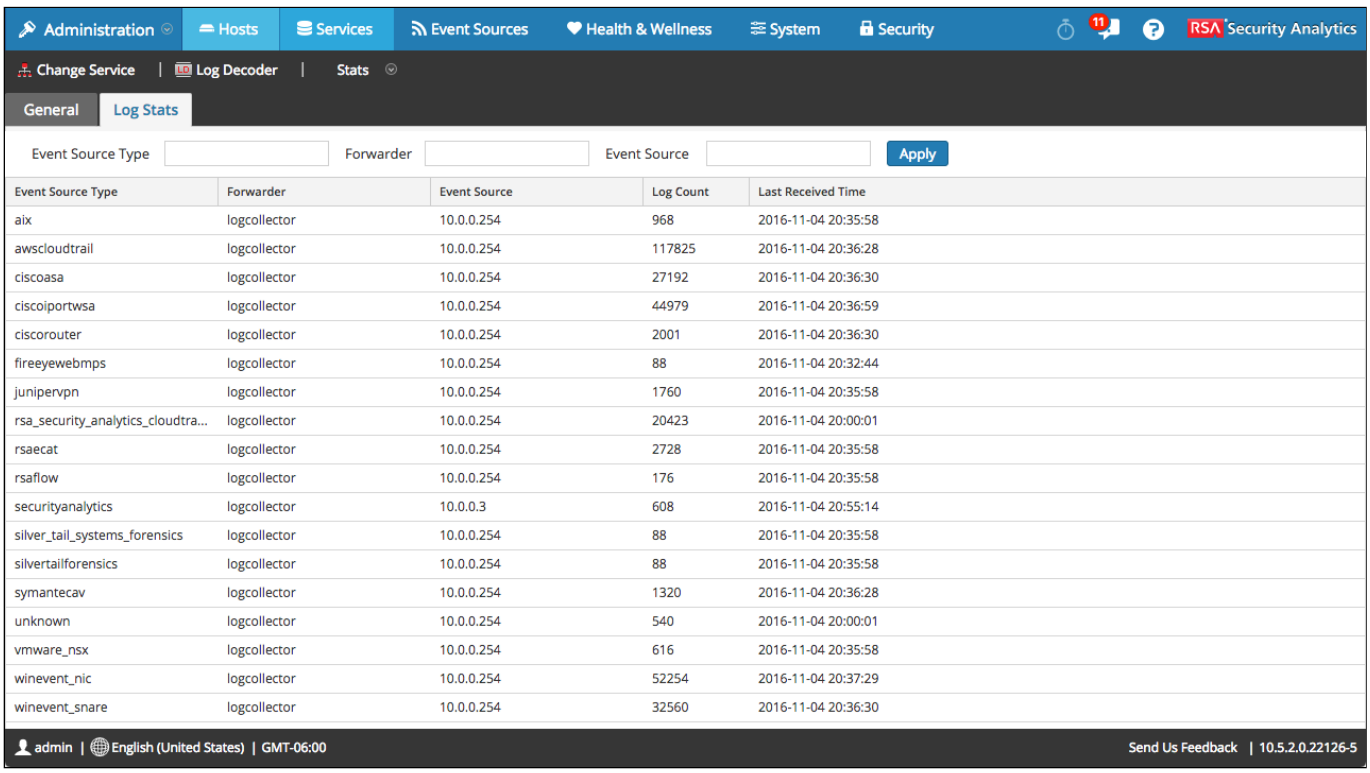
Top Tip: When you see the term 'packet' used in conjunction with a Log Decoder, it's referring to a single log message. The Log Decoder was based on the Packet Decoder's code, so they share a lot of the same terminology and internals.

Most statistics displayed here are only for the current capture session. Max Capture Rate, Total Dropped, and Total Packets stats are only calculated since the last time capture was started.



Log Stats

The Log Stats tab of the Log Decoder's Stats view gives an administrator an idea of what type of log data is being captured, from where it's originating, and how many logs from each event source have been received.



Event Source Type	Forwarder	Event Source	Log Count	Last Received Time
aix	logcollector	10.0.0.254	968	2016-11-04 20:35:58
awscloudtrail	logcollector	10.0.0.254	117825	2016-11-04 20:36:28
ciscoasa	logcollector	10.0.0.254	27192	2016-11-04 20:36:30
ciscoiportwsa	logcollector	10.0.0.254	44979	2016-11-04 20:36:59
ciscorouter	logcollector	10.0.0.254	2001	2016-11-04 20:36:30
fireeyewebmps	logcollector	10.0.0.254	88	2016-11-04 20:32:44
junipervpn	logcollector	10.0.0.254	1760	2016-11-04 20:35:58
rsa_security_analytics_cloudtra...	logcollector	10.0.0.254	20423	2016-11-04 20:00:01
rsaecat	logcollector	10.0.0.254	2728	2016-11-04 20:35:58
rsaflow	logcollector	10.0.0.254	176	2016-11-04 20:35:58
securityanalytics	logcollector	10.0.0.3	608	2016-11-04 20:55:14
silver_tail_systems_forensics	logcollector	10.0.0.254	88	2016-11-04 20:35:58
silvertailforensics	logcollector	10.0.0.254	88	2016-11-04 20:35:58
symantecav	logcollector	10.0.0.254	1320	2016-11-04 20:36:28
unknown	logcollector	10.0.0.254	540	2016-11-04 20:00:01
vmware_nsx	logcollector	10.0.0.254	616	2016-11-04 20:35:58
winevent_nic	logcollector	10.0.0.254	52254	2016-11-04 20:37:29
winevent_snare	logcollector	10.0.0.254	32560	2016-11-04 20:36:30

Capture Rates & Dropped Packets

If capturing directly off the NIC, every Log Decoder will have a maximum rate of logs which it is capable of ingesting before it begins **dropping packets**. Packet drops occur when the Log Decoder cannot process incoming packets quickly enough due to exhaustion of resources - either CPU, memory, or disk queue, or a combination thereof. When a drop occurs, the associated log is not saved to disk and any meta which could be parsed from the log will be lost. It is therefore important to avoid a dropping packet scenario.

There are many factors which can contribute to packet drops, but chiefly among them is the capture rate. RSA avoids publishing official capture rates for SA because there are so many variables which can impact its maximum.

These are the main variables which impact capture performance:

- SA code
- Capture rate
- Log size
- Type of logs
- Enabled parsers (different parsers have different performance implications)
- Application rules
- Network rules

- Number and size of feeds
- CPU cores & speed
- Memory size & speed
- Disk performance & RAID firmware
- OS kernel

Assembly

Assembly is the process of creating sessions and streams from captured packets. When packets first make their way into the capture NIC or NICs, they first go into the *capture buffer*, a slice of memory where packets live until the Decoder process gets around to processing them. Once the Decoder reads the packets from the capture buffer, they go into the *assembler*. The assembler takes each individual packet and adds it to a session.

Unlike packets, logs do not possess the concept of directionality, as every log identifies only one source: that which sent it. Granted some logs, like firewall logs, may show source and destination addresses in its message, but these do not translate directly into source and destination sides of an SA session, but rather other meta.

The process of assembly is incredibly simplified in a Log Decoder vs. a Packet Decoder. In a Packet Decoder, sessions are usually streams of data with a source and destination side. On a Log Decoder, every log message is its own session, therefore there is only one packet to assemble for every session as opposed to potentially many thousands.

This means that on a Log Decoder, the session count will equal the captured logs count, which also has the effect of usually making the Log Decoder's Session DB much bigger than a Packet Decoder's, as the session count is much higher. Because of this, Log Decoders will have more space allocated to its Session DB than a Packet Decoder, in order to avoid orphaning packets.

Top Tip: **Orphaning** packets means that a session's packets exist in the Packet DB, but its corresponding Session DB data is gone. In this case, even though the packets are there, there is no way of accessing them, as the Session DB is the cement between an Analyst and the meta and packet data he or she is trying to access.

Log (or enVision) Parsers

Log Parsers are responsible for the first stage of meta generation at capture time. The parsers used by Log Decoders are distinct from the LUA type of parser used by a Packet Decoder, though technically LUA parsers can still be used with a Log Decocder. Log Decoder uses the same type of parser which was used by RSA enVision, and many parsers have simply been moved over from enVision into the Log Decoder. We'll simply call them log parsers. A log parser reads a log message which has been captured, and watches for patterns in the log string. Once a match has been made by a log parser, the parser then registers one more more meta values from that log, which in turn usually also gets stored in the metadb and often in the index.

Only a single log parser is used on a given log message.

Parsers are not the only things that can generate meta: so can application rules and feeds (though feeds technically run inside of a parser, too, called the *feed parser*.

- **Binary** - Binary parsers are baked in to the compiled code of the Decoder service itself. They can be enabled or disabled like any other parser, but no customisation is possible. They are not represented by files anywhere on the filesystem, other than the /sbin/NwDecoder executable itself. Examples of binary parsers are the IMAP parser, the BITTORRENT parser, and the GeoIP parser. There are many more binary parsers. These generally do not have to be messed with on a Log Decoder.

- **LUA** - LUA parsers are written in a standardised programming language which the Decoder understands. All modern parsers which RSA currently develops are written in LUA.

- **FLEX** - FLEX is an XML-based programming language, and in Decoder terms was the predecessor to LUA. Version 10.x Decoders can still handle FLEX parsers but they do not perform as well as LUA. Some believe they are considerably more difficult to write.

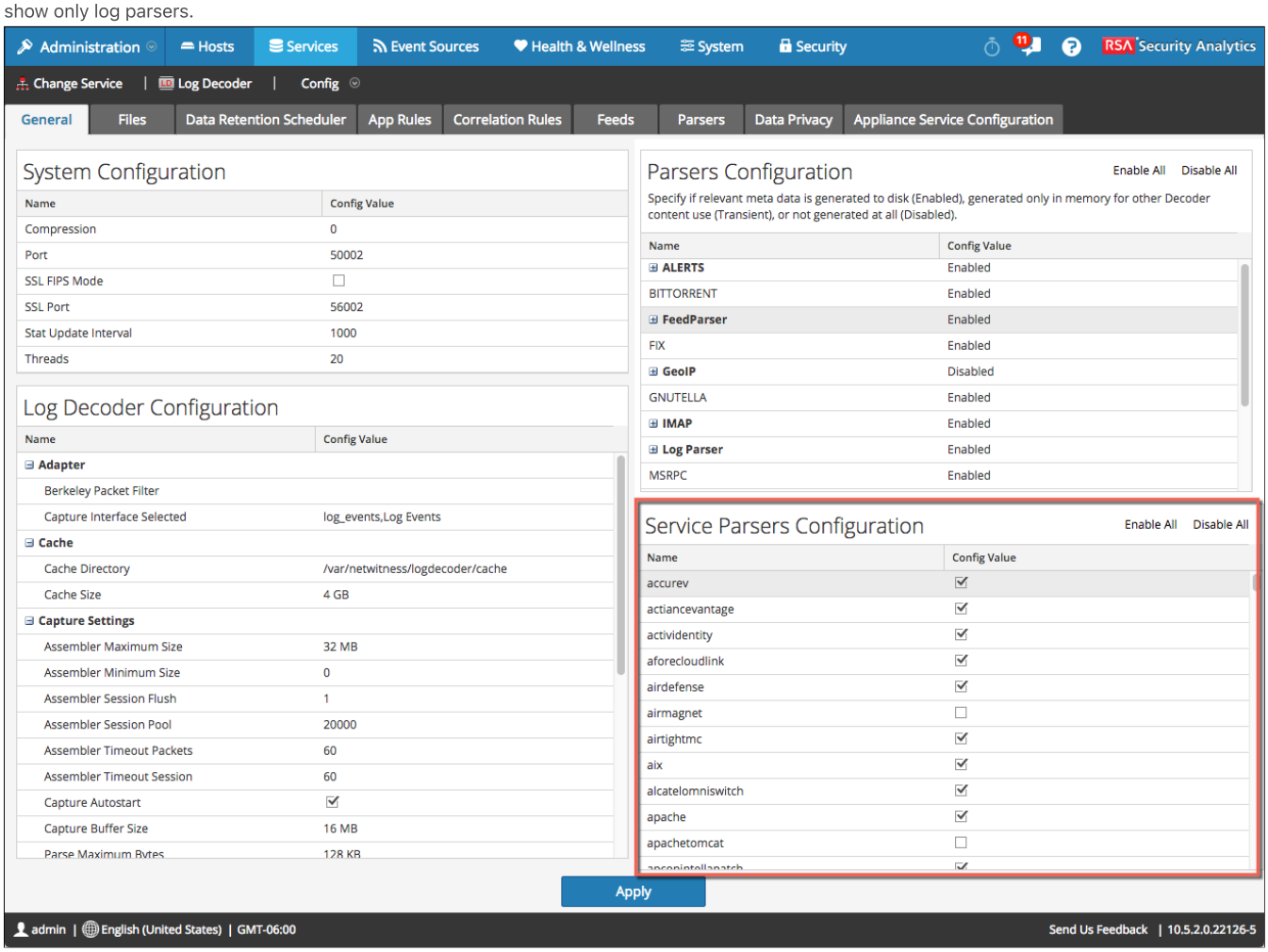
Note: The specifics of parser syntax and development is beyond the scope of this course.

One can see which log parsers are installed on a Log Decoder in a few different ways:

- By looking in the **/etc/netwitness/ng/envision/etc/devices** directory on the Log Decoder. This will only show LUA and FLEX parsers, but not binary parsers.

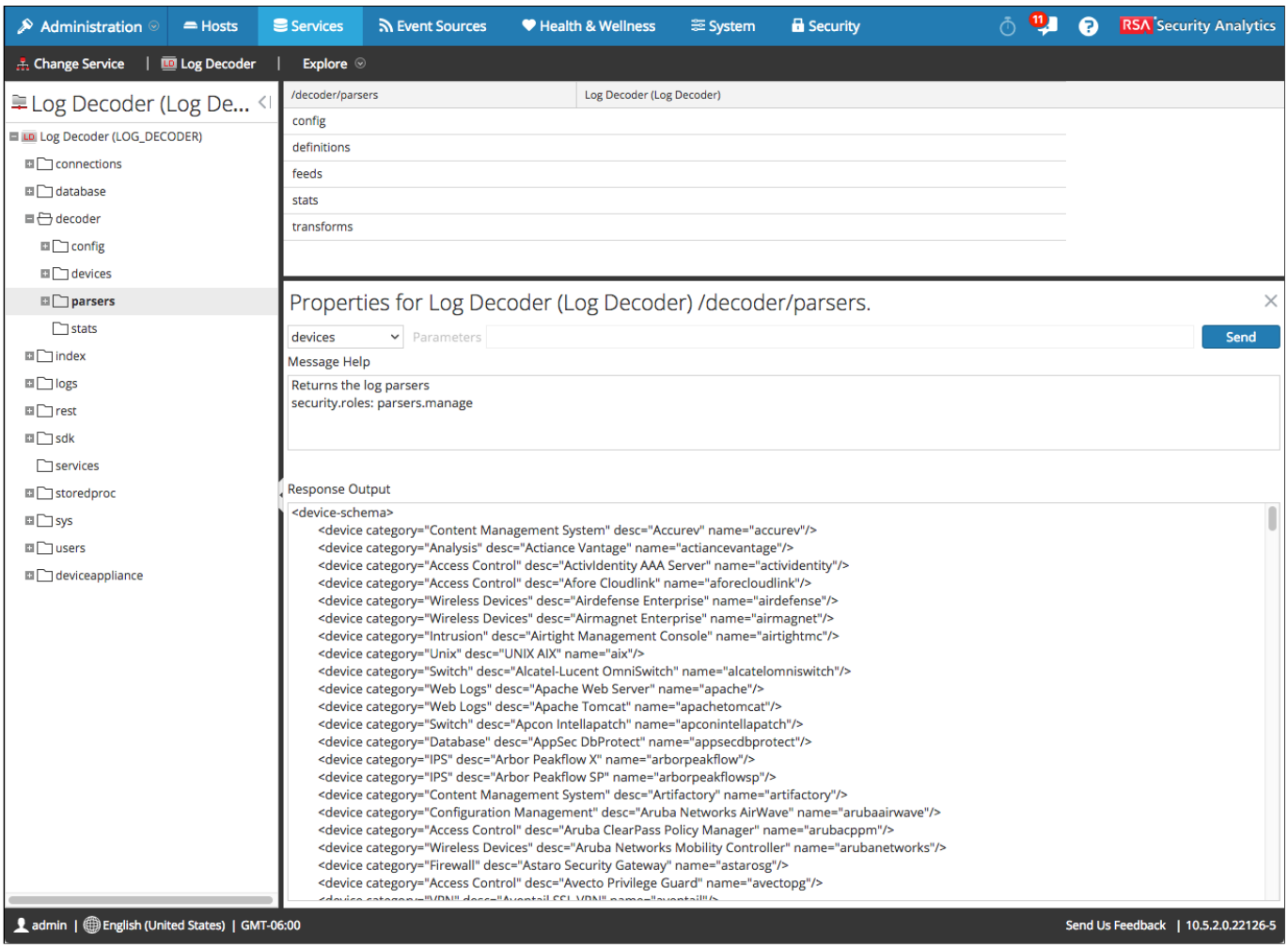
```
[[root@logdecoder ~]# cd /etc/netwitness/ng/envision/etc/devices/
[[root@logdecoder devices]# ls
accurev            ciscoiportwsa      fabricos           junipervpn         nfrnids            sonicwallgms
actiancevantage    ciscolms           fairwarningpm      juniperwlc         nokiaipso          sophos
actividentity      ciscomars          fireeyewebmps      junosrouter        nortelvpn          squid
aforecloudlink     ciscomeraki        firepass           kasperskyav        nortelwebos        stash
airdefense         ciscomse           forescoutcounteract kvm               novelledirectory   stealthwatch
airmagnet          cisconac           forticlientendpoint landesk            nsm                sunoneldap
airtightmc         cisconcm           fortinet           linux_snare        openvms            sybasease
aix                cisconxos          fortinetfortimail  lotusdomino        oracle             symantecav
alcatelomniswitch  ciscopix           fortinetmgr        lumensionemss      oracleam           symantecbrightmail
apache             ciscorouter        fsecureav          manageenginenetflow oracleav          symanteccsp
apachetomcat       ciscosecagent      gecea              mazuprofiler       oracledv           symantecdlp
apconintellapatch  ciscosecureacs     gepacs             mcafeedlp          oracleid           symantecintruder
appsecdbprotect    ciscoucs           git                mcafeeds           oracleim           symmetrix
arborpeakflow      ciscowcs           greenplum          mcafeeendpoint     oracleiplanetweb   teradata
arborpeakflowsp    ciscowlc           greenplumhd        mcafeefoundscan    oracleweblogic     tippingpoint
artifactory        ciscoworks         gseftserver        mcafeeic           paloaltonetworks   trendmicro
arubaairwave       citrixag           guardium           mcafeenac          perforce           trendmicrods
arubacppm          citrixns           hpnonstopserver    mcafeepa           postgresql         trendmicrodsa
arubanetworks      citrixxa           hpprocurvesw       mcafeereconnex     proofpoint         trendmicroimss
astarosg           clariion           hpux               mcafeevirusscan    radiator           trendmicroiwss
avectopg           clouderanavigator  huaweivrp          mcafeewg           radwaredp          trendmicroossec
aventail           courionpc          hytrust            mckessonhpf        rhlinux            trendmicroscanmail
avocentkvm         crossbeamc         ibmacf2            microdasys_xsg     riverbedsteelhead  trendmicrosp
barracudasf        cyberark           ibmdb2             microsoftiis       rsaaah             tripwire
barracudawaf       cyberguardclassic  ibmicsf            mom                rsaaaop            tufinsecuretrack
beewarewaf         cyberoamutm        ibmidms            msacs              rsaaccessmanager   unboundidids
bigfix             damballa           ibmims             msdhcp             rsaacesrv          varonisprobe
bigip              delldrac           ibmmainframeipsec  msexchange         rsaarcher          vmware_esx_esxi
bigipafm           dellswitch         ibmmfzossyslog     msforefrontcs      rsaaveksa          vmware_nsx
bigipapm           detectit           ibmracf            msfuag             rsacm              vmware_vc
bigipasm           dragonids          ibmtamesso         msias              rsadlp             vmware_vcac
bit9               eeyeblink          ibmtamws           msisa              rsaecat            vmware_vcloud
blackberryes       eeyerem            ibmtim             msnap              rsafim             vmware_vco
bluecat            eeyeretina         ibmtopsecret       mssccm             rsaflow            vmware_vcops
bluecoatdirector   emcavamar          ibmwebsphere       mssharepoint       rsakeymanager      vmware_view
bluecoatproxyav    emcdatadomain      ibmwebspheredp     mssql              rsaviaaccess       vmware_vshield
bmcremedyitsm      emcdocumentum      ibmwebspheremq     msurlscan          rsavlr             voltagesecuredata
brocadeswitch      emcdpa             impervawaf         mswsus             safendprotector    voyence
cacheflowelff      emcionixuim        infobloxnios       mysql              safenethsm         vssmonitoring
caitm              emcisilon          intrushield        ncirclecm          sap                websense
casiteminder       emcnetworker       invincea           nessusvs           secudesi           websenseds
cef                emcvplex           ironmail           netapp             sidewinder         whatsupgold
celerra            enterasysswitch    iseries            netasqutm          silverpeakwan      winevent_er
checkpointfw1      entercept          iss                netscreen          silvertailforensics winevent_nic
ciscoace           enterpriseitsfne   j4carehcc          netscreenidp       snort              winevent_snare
ciscoacsxp         entrustig          jboss              netwitness         solaris            zenprisemdm
ciscoasa           epolicy            jenkins            netwitnessspectrum solarisbsm         zscalernss
ciscoidsxml        esrs               juniperic          nexpose            sonicwall
ciscoiportesa      evidian            junipersbr         nfdump             sonicwallemail
```

- In the UI, by looking on the **Administration -> Services -> <Log*Decoder*> -> Config** page, under **Service Parsers Configuration.** This pane will show only log parsers.



Under the parsers configuration, you can enable and disable which parsers will be used by the Log Decoder.

- By issuing an API call to the **/decoder/parsers/devices** property



Top Tip: If the correct parser for an event source type isn't installed, it the **device.type** meta will usually be registered as 'unknown', but sometimes it may still get detected by another parser, albeit the wrong one. **device.type** meta is registered for every captured log to help identify which log parser parsed the log message.

Log Parser Structure

Every parser is contained in its own directory under **/etc/netwitness/ng/envision/etc**. Each parser consists of two files:

- *<parsername>*.ini file. E.g. '**trendmicro.ini**'. This defines some variable names and values in use by the log parser and read by SA.

- A [v20_]*<parsername>*msg.xml file. E.g. either **trendmicromsg.xml** or **v20_trendmicromsg.xml**. This defines the log headers and the log messages. Newer parsers now use filenames beginning with *v20_*. If using the v20_ prefix, be sure to include '***device="2.0"***' in the <VERSION> section of the xml. The easiest way to understand how the parser works is to load it in the ESI tool.

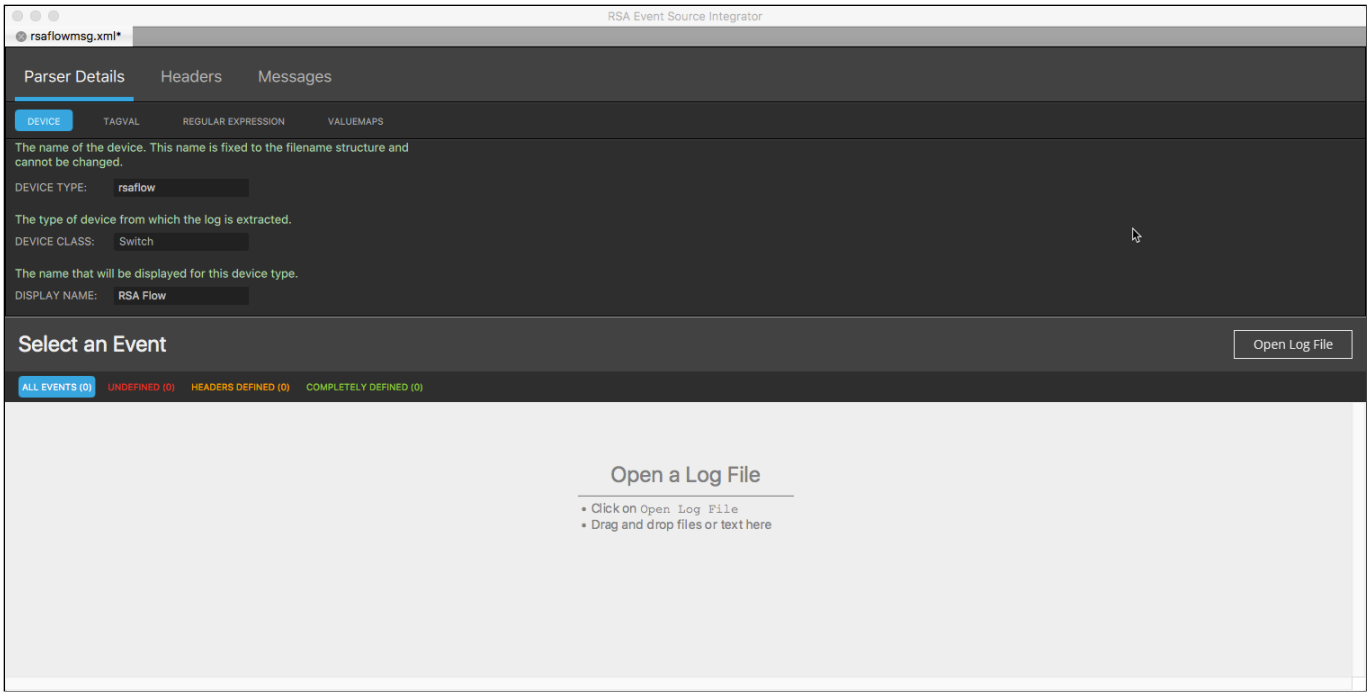
Creating Your Own Log Parsers - The ESI Tool

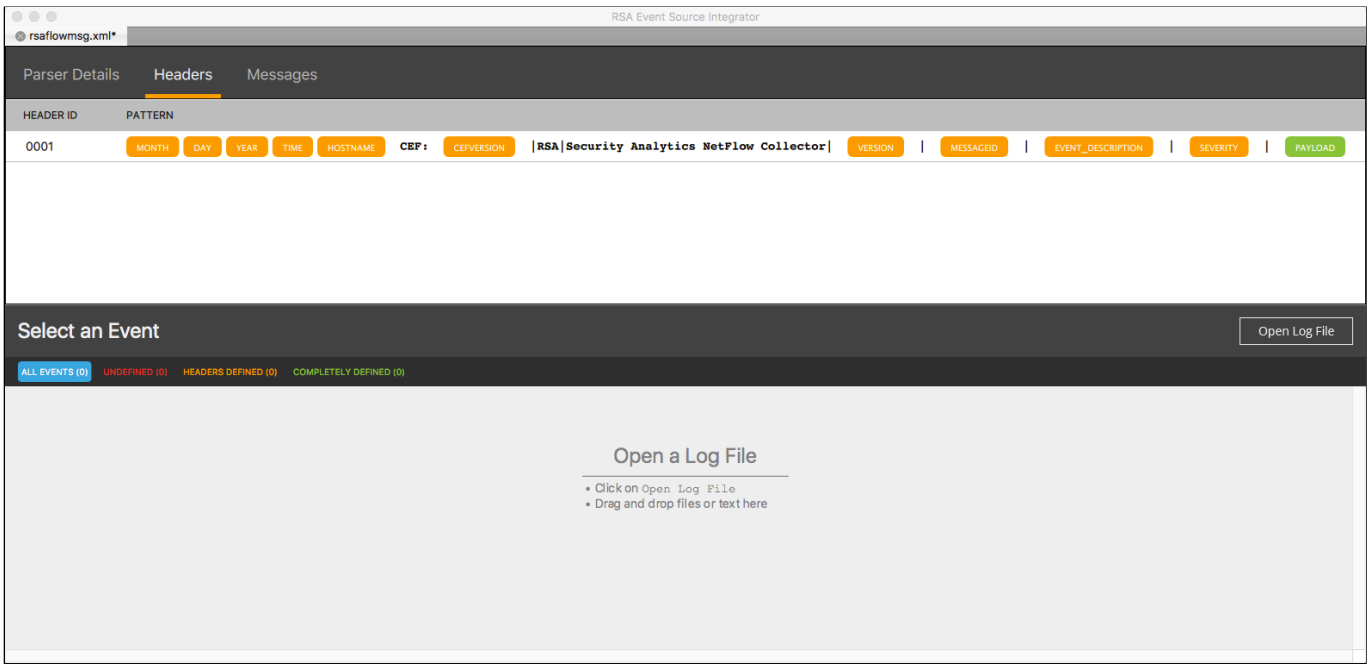
Sometimes you have a log event source for which RSA has no log parser. It's not strictly necessary to hire a Consultant write one for you. RSA provides a tool called the Event Source Integrator, or ESI tool, which you can download from https://community.rsa.com. It's currently still in beta, but it is still highly useful, if a bit unstable.

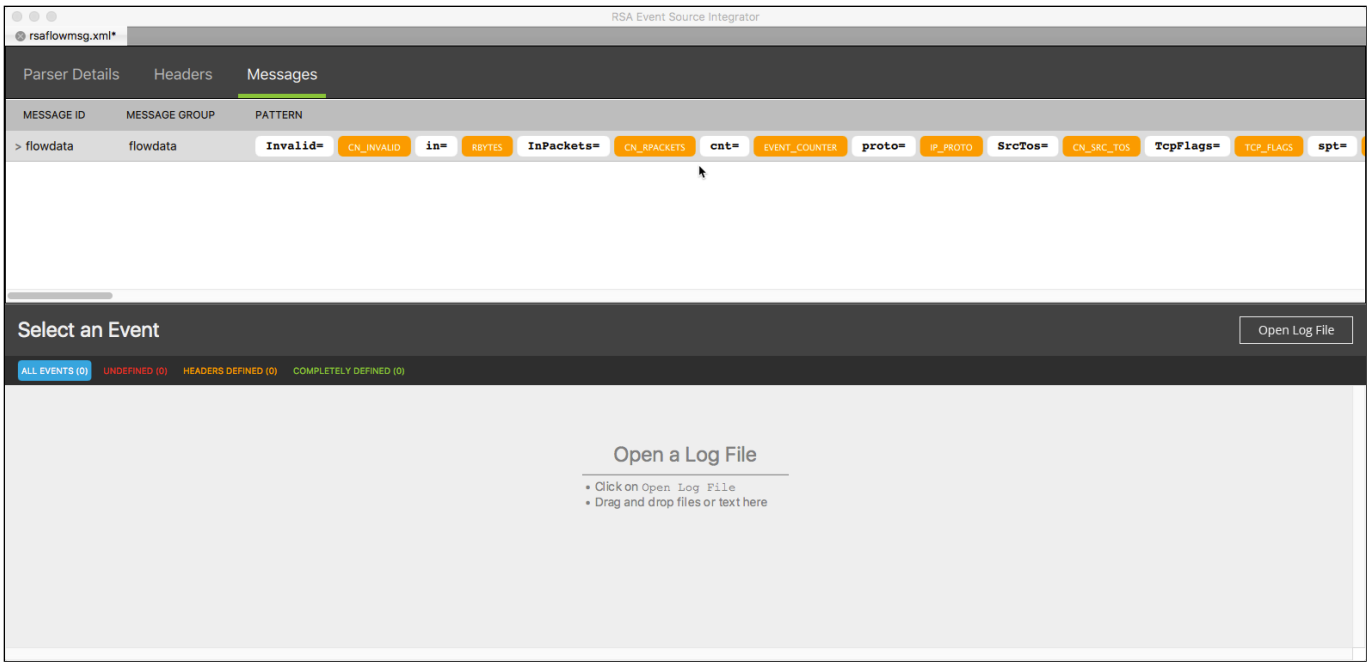
We shan't go into the specifics of its use, but the workflow is like this:

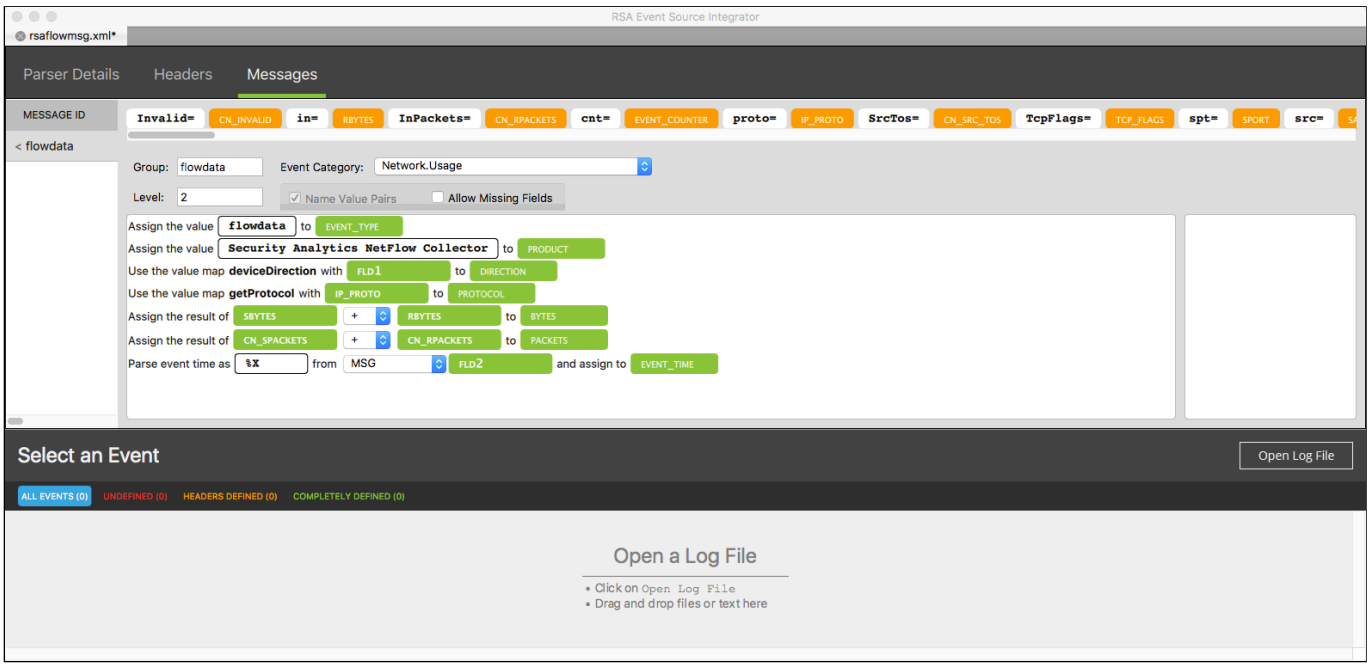
1. Create or edit a log parser
2. Import a log file from the application you're attempting to create a parser for
3. Define the headers that identify the device under key device.type
4. Define message types from the logfile.
5. Save and deploy parser files

You can even use this to modify existing RSA parsers. Here are some screenshots from the tool which are analysing the '**rsaflow**' parser, which parses NetFlow traffic collected by SA Log Collectors.


RSA Event Source Integrator
rsaflowmsg.xml*
Parser Details
Headers
Messages
DEVICE
TAGVAL
REGULAR EXPRESSION
VALUEMAPS
The name of the device. This name is fixed to the filename structure and cannot be changed.
DEVICE TYPE:
rsaflow
The type of device from which the log is extracted.
DEVICE CLASS:
Switch
The name that will be displayed for this device type.
DISPLAY NAME:
RSA Flow
Select an Event
Open Log File
ALL EVENTS (0)
UNDEFINED (0)
HEADERS DEFINED (0)
COMPLETELY DEFINED (0)
Open a Log File
• Click on Open Log File
• Drag and drop files or text here


RSA Event Source Integrator
rsaflowmsg.xml*
Parser Details
Headers
Messages
HEADER ID
PATTERN
0001
MONTH
DAY
YEAR
TIME
HOSTNAME
CEF:
CEFVERSION
|RSA|Security Analytics NetFlow Collector|
VERSION
|
MESSAGEID
|
EVENT_DESCRIPTION
|
SEVERITY
|
PAYLOAD
Select an Event
Open Log File
ALL EVENTS (0)
UNDEFINED (0)
HEADERS DEFINED (0)
COMPLETELY DEFINED (0)
Open a Log File
• Click on Open Log File
• Drag and drop files or text here





Misdetected Log Devices

On occasion, even with the right log parsers installed, the Log Decoder still misidentifies which parser should be used on a particular event source, usually when logs are similar between disparate event sources. This can be corrected by forcing the Log Decoder to parse all logs from a given IP address to parse with a particular log parser.

To do this, issue the **/decoder/parsers/ipdevice** API call, using your API interface of choice (explore view, REST API, NwConsole, etc). We'll demonstrate with the explore view.

- To add a new mapping, use the following syntax. The "reload=true" parameter reloads the log parsers, thus causing the new mapping to come into effect.

```
op=edit entries="+/-<ipAddress>=<parser>" reload=true
```

For example:

```
op=edit entries="+10.8.231.4=apache" reload=true
```

Properties for Log Decoder (Log Decoder) /decoder/parsers.

ipdevice | Parameters op=edit entries="+10.8.231.4=apache" reload=true | Send

Message Help

```
Map IP to Device type in log parsing. Takes effect after parser reload.
security.roles: parsers.manage
parameters:
  op - <string, {enum-one:edit|describe}> The operation to perform (edit|describe).
```

Response Output

```
IP2Device entry edited
```

- To remove a mapping, use the same API call, but change the + to a - in the "entries" string:

Properties for Log Decoder (Log Decoder) /decoder/parsers.

ipdevice | Parameters op=edit entries="-10.8.231.4=apache" reload=true | Send

Message Help

```
Map IP to Device type in log parsing. Takes effect after parser reload.
security.roles: parsers.manage
parameters:
  op - <string, {enum-one:edit|describe}> The operation to perform (edit|describe).
```

Response Output

```
IP2Device entry edited
```

- To show existing IP address to device type mappings, issue the **/decoder/parsers/ipdevice** call with parameter "op=describe"

Properties for Log Decoder (Log Decoder) /decoder/parsers.

ipdevice | Parameters op=describe | Send

Message Help

```
Map IP to Device type in log parsing. Takes effect after parser reload.
security.roles: parsers.manage
parameters:
  op - <string, {enum-one:edit|describe}> The operation to perform (edit|describe).
```

Response Output

```
<addressmap-schema>
    <DeviceEntry device="apache" ipv4="10.8.231.4"/>
</addressmap-schema>
```

Top Tip: Parsers can be reloaded manually with the **/decoder/parsers/reload** API call.

The Table Map

The table map is a file unique to the Log Decoder which is required to join the disparate enVision and and SA models contained within the Log Decoder. Its purpose is to map the meta generated by log parsers to SA keys, as the key names stored within the log parsers' old model don't sync well with SA's convention, which was developed initially by NetWitness. Its location is **/etc/netwitness/ng/envision/etc/table-map.xml**.

Let's have a look at the relationship between enVision parsers and SA meta.

enVision Log Parser Key	=>	Security Analytics Meta Key	Flags
device.class	=>	device.class	None
event_cat	=>	Transient	Transient
dhost	=>	host.dst	None
t_context	=>	context.target	Transient

Excerpts from the XML:

```
<mappings>
  <mapping envisionName="device.class" nwName="device.class" flags="None" envisionDisplayName="DeviceClassName"/>
  <mapping envisionName="event_cat" nwName="event.cat" flags="Transient" envisionDisplayName="EventCategory" format="UInt32"/>
  <mapping envisionName="dhost" nwName="host.dst" flags="None" envisionDisplayName="DestinationHostName"/>
  <mapping envisionName="t_context" nwName="context.target" flags="Transient"/>
</mappings>
```

Mapping Attributes

Every mapping entry in the table-map has certain attributes, some of which are optional.

- **envisionName** - This is the enVision key name that the log parser generates
- **nwName** - This is the name of the SA meta key to send the log parser's meta to. This key should exist in SA's index configuration file.
- **flags** - Either "*None*" or "*Transient*". This controls whether meta generated by the key will be saved by SA or not. Perhaps counter-intuitively, "None" means DO save the data, and "Transient" means DO NOT save the data. Transient meta can still be used by feeds and app rules, however.
- **envisionDisplayName** - Not really important unless you're using the IPDB extractor, in which it's a performance enhancement.
- **format** - If you'll recall our lesson on the SA index, this field controls what format the meta should be take. Most of these will be **Text**. Valid values here are **Int8**, **UInt8**, **Int16**, **UInt16**, **Int32**, **UInt32**, **Int64**, **UInt64**, **UInt128**, **Float32**, **Float64**, **TimeT**, **Binary**, **Text**, **IPv4**, **IPv6**, or **MAC**. This field should match SA's index configuration file.
- **nullTokens** - Do not populate the key if the meta value equals any of the values defined in this attribute are present, using **|** (the pipe symbol) as a logical OR. I.e. if **nulltokens='-|(null)'** means that if the meta value is either '-' or '(null)', then don't save this key for this log message.

Top Tip: Should you desire to make changes to the table-map configuration, to either change a name, add or remove an entry, or change its flags, **DO NOT** modify **table-map.xml,** as they will be overwritten when you upgrade the Log Decoder. Instead, create a file called **table-map-custom.xml** in /etc/netwitness/ng/envision/etc, and make your changes to this file. Make sure you use valid XML, with proper XML declaration and a parent <mappings> key.

Installing Additional Log Parsers via Live

Additional log parsers can be added to a Log Decoder via **Live -> Search** in the UI, using the Resource Type of '*RSA Log Device*'. Simply select the parsers you want to install, then click '**Subscribe**', then '**Deploy**'. See the lesson on Live for more info on deploying Live content.

Top Tip: When collecting logs through a Log Collector, in many instances when the collection method is not syslog, it may also be necessary to deploy "*RSA Log Collector*" content to your Log Collector from Live. That's in addition to deploying needed log parsers to your Log Decoder.

Envision Content Files

When looking at RSA Log Device content in Live, there are two special content items: **Envision Config File** and **Envision Content File**. RSA periodically updates key log content files and publishes

Envision Config File - This contains updates to **table-map.xml**, **ipaddr.tab**, and the **etc.ini** files.

Envision Content File - This is a package of base log parsers, identified by ESU (Event Source Update) version. Because applications that create logs change frequently, the content which parses these logs must frequently be updated to support these changes. Each ESU release contains updates to base log parsers. This also contains **table-map.xml**, **ipaddr.tab**, and **ecat.ini**.

Top Tip: If you have made changes to any of the base log parsers contained within an ESU release, then these changes will be overwritten when you deploy an ESU update. If you need to make changes to one of these parsers, DO NOT modify the original log parser. Instead, duplicate it, rename the parser so that it has a different name, and then modify this. Then when you deploy an ESU, it won't touch your modified parser.

Key Keys

The SA logs solution tends to use different meta keys than the packets solution. Here are some of the most common logs meta keys you will use. You may see quite a bit of overlap between they keys in their data.

Key Name	Friendly Name	Description
lc.cid	Collector ID	The Log Collector, if any, that collected the event.
device.ip	Device IP	The IP address that sent the log to SA. This doesn't necessary map to the original event source that generated the log, if the message went through a proxy or intermediary.
device.type	Device Type	This key identifies the type of event source. E.g. **awscloudtrail, rsaecat, ciscorouter, aix, vmware_nsx**. This generally corresponds with the name of the parser which parsed the message.
device.class	Device Class	The class of device which sent the log. E.g. **cloud, router, vpn, switch, firewall, windows hosts**
alias.host	Hostname Aliases	Device hostnames extracted from the sessions.
ip.src	Source IP Address	The source IP address of the subject of the log message.

ip.dst	Destination IP Address	The destination IP address included in the message. Many logs won't contain this. The most common log type which might contain ip.dst would be a firewall log.
device.group	Event Source Group	This refers to which SA Event Source Group the event source belongs to. This is definable by a system Administrator. E.g. **all windows event sources, all unix event sources.** See the lesson on Event Source Monitoring for more info.
event.type	Event Type	The raw type of event text taken from the log message. E.g. **awsapicall, success audit, data_access, authentication, failure audit.** Because this is raw, you may not see much consistency here.
event.desc	Event Description	The raw description from the log message of what happened. E.g. **An account was successfully logged on., a user account was changed., account was deleted., system audit policy was changed., connection denied**
ec.outcome	Event Outcome	Indicates whether the event was successful. E.g. **success** or **failure**
category	Category	What did the event pertain to? E.g. **logon, file system, user account management, privelege use, privilege use, account logon, security group management, security state change**
process	Process	The process name that generated the log. E.g. **kerberos, sa_server, c:\window\explorer.exe, nwconcentrator**
user.src	Source User Account	The source username the log pertains to. E.g. **cbarton, jschmoe, jdoe**
user.dst	Destination User Account	The username the log pertains to. Unfortunately, log parsers don't implement user.src and user.dst consistently, so one message may use user.src and another user.dst, both to describe the same subject user.
msg.id	Message ID	The message ID matched within the log parser for the message. E.g. **security_4663_microsoft-windows-security-auditing, get, new, security_675_security**

Log Decoder Databases

- The Session DB - Stored in **/var/netwitness/logdecoder/sesssiondb**
- The Packet DB (stores raw laws) - Stored in **/var/netwitness/logdecoder/packetdb**
- The Meta DB - Stored in **/var/netwitness/logdecoder/metadb**

Index Configuration File

The index configuration files are **index-logdecoder.xml** and **index-logdecoder-custom.xml** under **/etc/netwitness/ng**.

The Log Decoder Index

```
[root@decoder ng]# grep level /etc/netwitness/ng/index-logdecoder.xml

<language level="IndexNone" defaultAction="Auto">
     <!-- time needs to always be indexed at value level -->
     <key description="Time" level="IndexValues" name="time" format="TimeT" valueMax="0"/>
     <key description="Service Type" level="IndexNone" name="service" format="UInt32" valueMax="75" defaultAction="Open">
     <key description="TCP Source Port" level="IndexNone" name="tcp.srcport" format="UInt16">
     <key description="TCP Destination Port" level="IndexNone" name="tcp.dstport" format="UInt16" valueMax="65536">
     <key description="UDP Source Port" level="IndexNone" name="udp.srcport" format="UInt16">
     <key description="UDP Target Port" level="IndexNone" name="udp.dstport" format="UInt16" valueMax="65536">
     <key description="Medium" level="IndexNone" name="medium" format="UInt8" defaultAction="Hidden">
```

In more than 99% of scenarios, Log Decoders have no index to speak of, with one exception: the '**time**' meta key. This key is used to track the time of each captured session. Because Decoders are not intended to be queried directly, they do not index any other key. This does not mean that Decoders are incapable of indexing - they have the same indexing code as Concentrators, and can in fact run a full index and be queried directly, if they are configured to do so.

A Decoder's index configuration historically also served another purpose - to inform the Decoder of which meta keys it would be storing in the meta DB. Its index configuration file still contained entries for all the various meta keys, but had their level set to 'NoIndex' or 'IndexNone', which have the same effect. The Decoder would not record meta in the meta DB for a key which did not have an entry in the index configuration file.

These days, Decoders are more versatile, and no longer require explicit entries in their index configuration files. They can learn their meta key settings from configured feeds, parsers, and rules. You still need to understand the index configuration files for Concentrator and Broker configuration

You will see in implementations, though, that some people still explicitly declare their meta keys in the index-decoder-custom.xml file. There is nothing wrong in doing so.

Log Decoder-Specific API Tree Nodes

Both the Packet and Log Decoders use the **/decoder** node in the API tree for properties specific to the service.

/decoder/config

/decoder/parsers

/decoder/parsers/definitions - Shows statistics for individual parsers, which can be useful for analysing which parsers are generating the most meta. Its existence depends on value of **/decoder/parsers/config/detailed.stats**.

Importing Logs - Direct Import vs NwLogPlayer

If you have need to import log files directly into your Log Decoder to be parsed, you have two options: you can use the UI to import a log file, or you can use the **NwLogPlayer** CLI utility to import it. **NwLogPlayer** is an RSA-created utility for injecting logs into a Log Decoder, with installers available for Linux, Windows, and Mac.

With a direct import, there is one main drawback: you must first stop capture to perform the import. Packets imported have their original timestamps contained in the PCAP.

With NwLogPlayer, there is no need to stop capture on the Decoder, but sessions generated from its data will have current timestamps, rather than their original.

Top Tips:

- To obtain the Windows and Mac versions of NwLogPlayer, download it from the NetWitness 9.8 downloads page of the RSA Link site at https://community.rsa.com. The functionality should be the same as the SA 10.x versions.
- **NwLogPlayer** is not by default installed on SA systems, but it is included in the SA update package and yum repository. If this approach is desired, it is necessary to install the RPM with "**yum install nwlogplayer**".

NwLogPlayer Usage:

```
[root@logdecoder ~]# NwLogPlayer --help
NwLogPlayer
  --priority arg                  set log priority value
  -h [ --help ]                   show this message
  -f [ --file ] arg (=stdin)      input file
  -d [ --dir ] arg                input directory
  -s [ --server ] arg (=localhost) remote server
  -p [ --port ] arg (=514)        remote port
  -r [ --raw ] arg (=0)           Determines raw mode. 1= File contents will
                                  be copied line by line to the server. 0 =
                                  add priority mark. 3 = auto detect. 4 =
                                  envision stream. 5 = binary object, 6 =
                                  protobuf stream
  -m [ --memory ] arg             Speed test mode.  Reads up to MB of messages
                                  from the file contents and replays.
  --rate arg                      Number of events per second. No effect if
                                  rate > eps which program can achieve at
                                  continuous mode
  --maxcnt arg                    max number of messages to be sent
  -c [ --multiconn ]              multiple connection
  -t [ --time ] arg               simulate time stamp time.  Format as
                                  yyyy-m-d-hh:mm:ss
  -v [ --verbose ]                if true will verbose output
  --ip arg                        simulate ip tag.
  --ssl                           connect with SSL
  --certdir arg                   OpenSSL certificate authority directory.
  --clientcert arg                use this PEM-encoded SSL client certificate
  --clientkey arg                 use this PEM-encoded private key file.  If
                                  not specified, the clientcert path is used.
  --udp                           send in udp
  -g [ --gzip ]                   treat input stream as compressed gzip
  --version                       output the version of this program
```

To import a log file from the UI:

1. Navigate to **Administration -> Services -> <LogDecoder> -> System.**
2. Click the Stop Capture button, if capture is already started.
3. Click the "**Upload Log File**" button and choose the log file to import. The file will be uploaded and processed by the Log Decoder.
4. If capture was started when you started, click the **Start Capture** button. Don't forget to do this!

16. Log Collection

Log Collection is the process of injecting logs from external log sources, or event sources, into the SA stack. Arguably, log collection is the most difficult part of SA to implement, as it involves many components and technologies. The experience of administering the Log Collector can be frustrating, but have patience, and you will get through it.

Top Tip: The Log Collector listens on port TCP 50001 / 56001 (SSL).

Scope

This lesson aims to teach fundamentals of log collection. Unfortunately, we aren't yet able to implement every log collection type in our lab, so the log event data we collect in our lab all uses syslog. Because there are literally hundreds of differing procedures for onboarding different event source types, we can't cover them all. We will only configure syslog collection as part of this lesson, but we hope to give you enough of a foundation where you can begin to configure other collection types.

How Does One Stop and Start the Log Collector?

```
# stop nwlogcollector
# start nwlogcollector
```

Constituent Parts of SA Log Collection

Log Decoder - the object of log collection is ultimately to get your logs here. It is capable of consuming only syslog-formatted logs.

Local Log Collector - a service that runs on the same host as a Log Decoder. Its job is to receive logs from other Log Collectors, called either Remote Log Collectors (RLC) or Virtual Log Collectors (VLC), and shovel them into the Log Decoder process. The acronyms VLC and RLC are the same thing. Though the local collector can collect logs just like a VLC, you ideally should not be doing so. Local collectors cannot collect syslog protocol, as they are

Remote Log Collector - A standalone log collector service, almost always running in a virtual machine. Its job is to collect logs from log event sources. It does so through the use of **collection services** and/or **collection protocols**.

Collection Protocols - Not all log data comes cleanly over the network as syslog traffic. Log event data can come from a variety of event source types, and over many different protocols. Some protocols are akin to promiscuous mode capture (I'll call these 'promiscuosu', where the collector simply listens for any log traffic that's pushed to it from event sources. Other types require the Log Collector to connect out to event sources and pull the log data. We'll call these 'ad hoc'. For each type of event source, a customised procedure must be used to enable collection of its log data. Log Collector supports these protocols for log collection:

- **ODBC** - Ad hoc. For collecting from databases. Many data types are supported
- **SDEE** - Ad hoc. This is a Cisco protocol, implemented mostly by IDS and IPS-type devices, for describing security events.
- **SNMP Trap** - Promiscuous. For transmitting health or service event data from theoretically any type of device or application. Most commonly used by network devices like servers, routers, firewalls, and switches.
- **Syslog** - Promiscuous. Arguably the most used collection protocol, syslog devices can transmit almost any type of log data.
- **NetFlow** - Promiscuous. NetFlow data can be enabled in routers and switches to transmit information about network connections flowing through them. NetFlow communicates things like source and destination IP, protocol, port, and amount of data transmitted. It does not transmit packet data for the flows. NetFlow can be somewhat useful where budget does not allow for packet Decoders at remote sites.
- **Check Point** - Ad hoc. Used to fetch logs from Check Point firewalls.
- **Windows WinRM** - Ad hoc. Windows Remote Management (WinRM) protocol collection. Used to pull data from the event logs of Windows systems, either servers or workstations. These can be Windows event logs, or for any application which logs to the Windows event log.
- **File Collection** - Partially ad hoc and partially promiscuous. Uses SFTP or FTP to collect logs from remote log sources, but rather than a fetching style of collection, file event sources must be configured to periodically connect to a Log Collector and transfer their logs. Each event source must be individually configured in the Log Collector.
- **VMware** - Ad hoc. Collects log data from VMware ESX / ESXi servers.
- **Plugins** - Ad hoc. This really means Amazon Cloudtrail protocol. Collects security data from Amazon AWS instances.

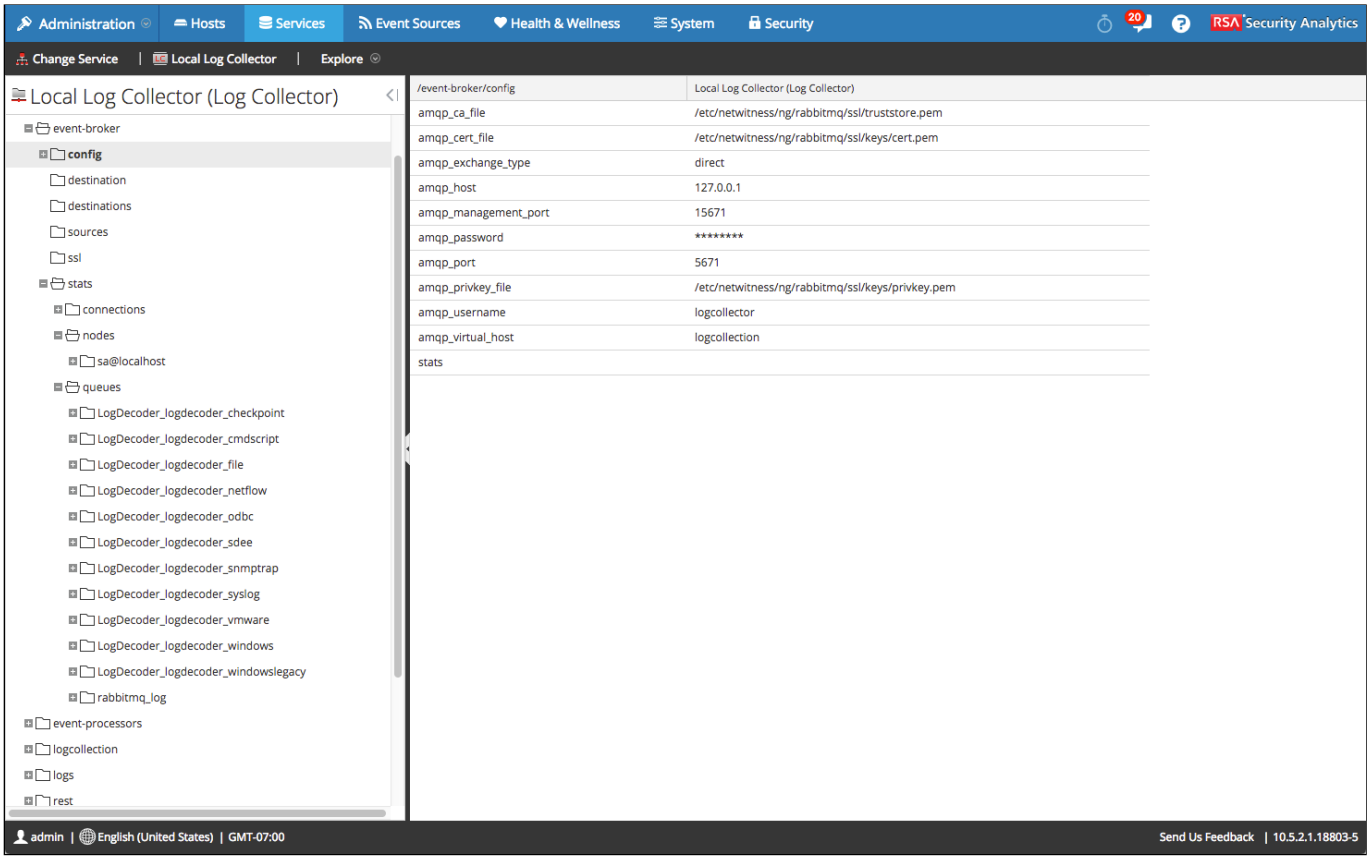
Collection Services - These are individual sub-processes which are part of the Log Collector service, which are responsible for collection each **collection protocol**. Like capture on a Decoder, each one of these processes can be started and stopped, and are also capable of being auto-started. There is one collection service per protocol type.

Log Collector Content - This is content which gets installed from Live onto the collecting Log Collector. Its purpose is to tell the Log Collector how to transform non-syslog-formatted data into syslog-formatted logs which can be parsed by a Log Decoder.

Log Parsers - These are parsers which instruct the Log Decoder how to process syslog data. These parsers are actually RSA enVision parsers. enVision was RSA's legacy SIEM product, now discontinued. Log parsers can be obtained from Live, or they can be written by SA users to parse custom logs.

RabbitMQ - SA's message bus service. It is responsible for transporting logs from VLC's to local collectors. It transmits data over the wire using **AMQP** (advanced message queueing protocol).

Event Broker - This is essentially RabbitMQ. If you look at any Log Collector in Explorer view, you will an **event-broker** node, which describe connection configuration parameters for RabbitMQ, and statistics describing the RabbitMQ node, its queues, and connections from the Collector into RabbitMQ.

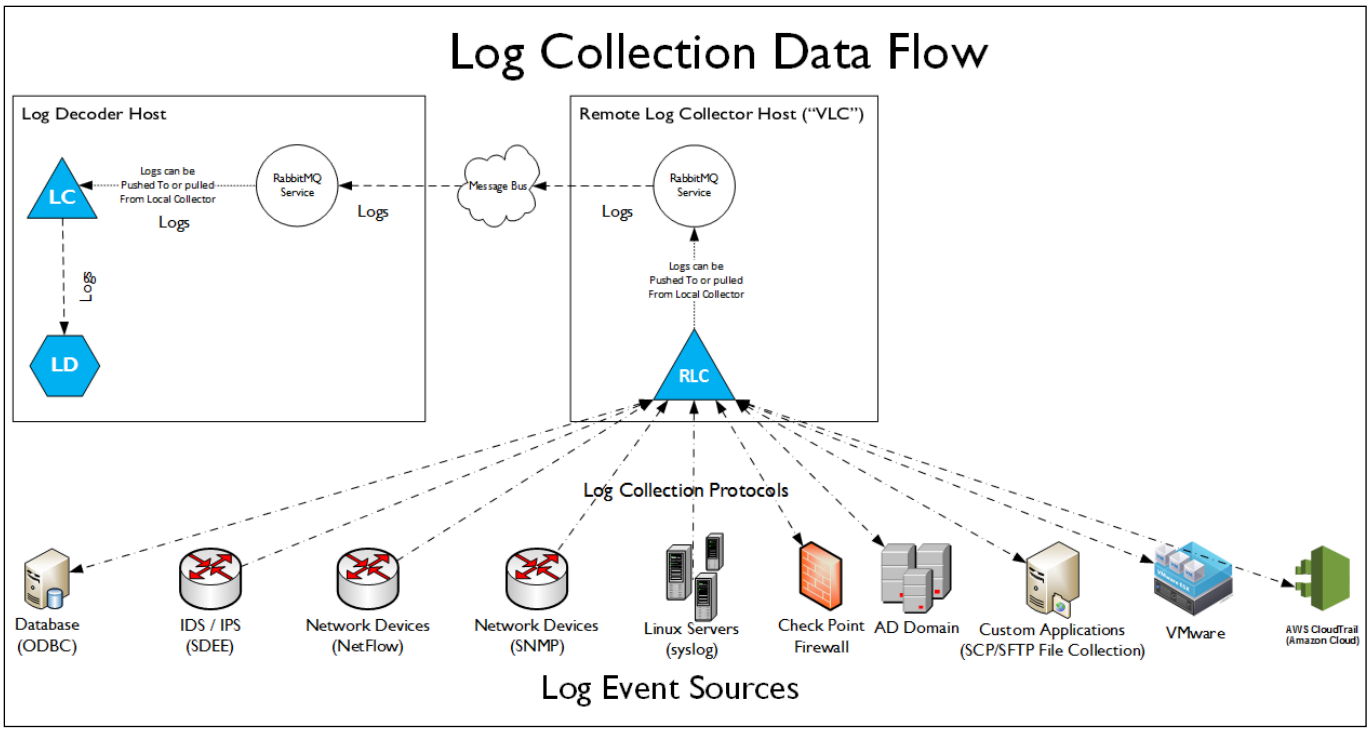


Event Destinations - These are the ultimate destination for logs. E.g. a Log Decoder listening on TCP 514. In addition to this, an event destination can also be an **Identity Feed** (see below).

Event Processors - An **Event Processor** is a under-the-bonnet Local Collector-specific subprocess which is responsible for pulling log events from the **Event Broker** (RabbitMQ), and then forwarding them to an **Event Destination** (either a Log Decoder or an Identity Feed).

Identify Feeds - This is a Decoder feed which the Log Collector is capable of generating, where some fields from collected WinRM events are transformed into a feed to be injected back into Decoders. This can add additional meta context to captured sessions. These work in conjunction with SA's **Live** component.

Architecture



The process of collecting logs through to parsing looks something like this:

1. VLC: Collect log message using log collection service
2. VLC: If not in a syslog-like format, apply collector Live content to the collected message
3. VLC: pass the log message to RabbitMQ service
4. VLC: RabbitMQ transmits message over the network using AMQP protocol to Local Collector
5. Log Decoder: RabbitMQ service receives log message
6. Log Decoder: Local Collector pulls message from RabbitMQ
7. Log Decoder: Log Collector process transmits log message to Log Decoder process via syslog on 127:0.0.1:514 (TCP)

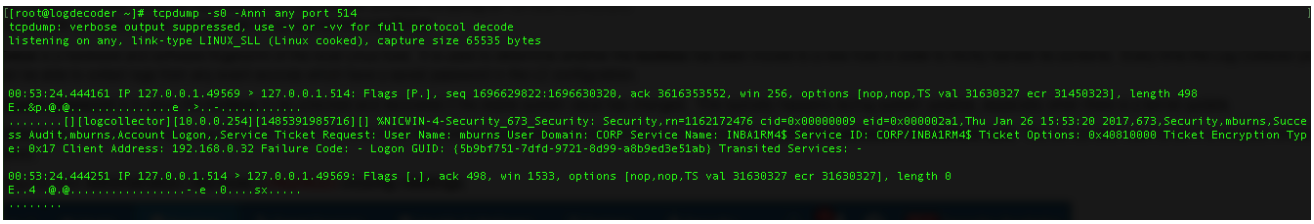


Illustration of Syslog Traffic Being Sent from Local Collector to Log Decoder Process

8. Log Decoder: Log Decoder process parses log message using appropriate Log Parser

Limitations

SA does not properly decode multi-line logs, such as what you might see in logfiles of Java applications. Every log transmitted by each event source be contained on a single line.

LC Lockbox

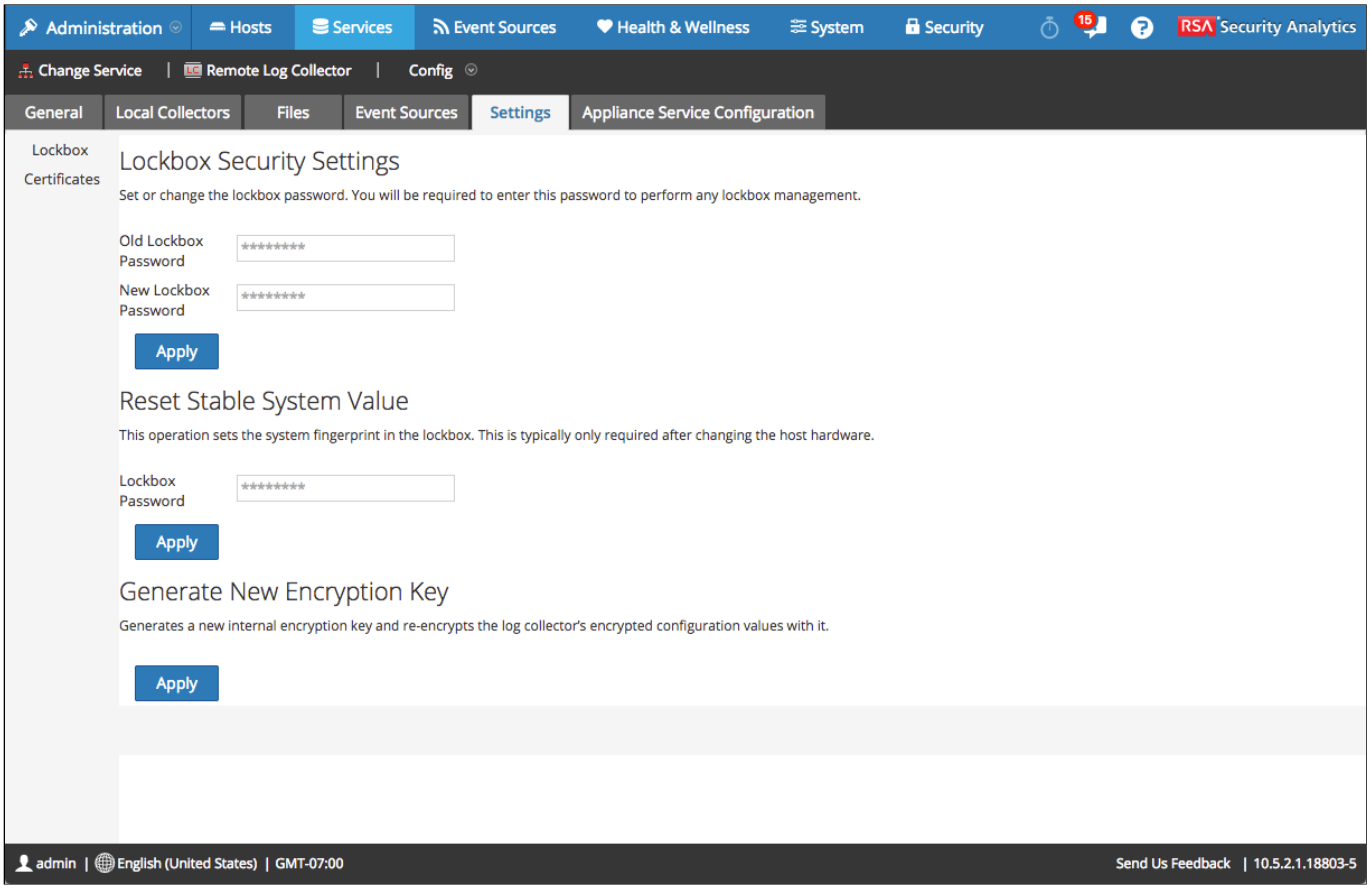
Each Log Collector has a built in password vault, or **lockbox**, which stores passwords which are used to connect to log event sources. The lockbox has two main components: a master **password**, and a **stable system value**.

The **stable system value** is a hardware and software fingerprint of the local Linux host. It is used to determine whether the **lockbox** has been moved to a new host in order to illicitly harvest its contents. Every time the Log Collector process starts, it checks to see whether the system fingerprint matches the **stable system value** of the **lockbox**. If not, the Log Collector will not be able to unlock lockbox, and it will not be able to collect logs from any event sources which have a saved password in the LC configuration.

The purpose of the **master password** is to allow the **lockbox** to be unlocked and recovered if the stable system value has changed. This typically happens during system updates, especially when there is a kernel update.

The first thing you should do when configuring a Log Collector for the first time is set the lockbox password! The second thing you should do is **store this password somewhere safe**! You do not ever want to be in a situation where log collection is down because you performed a system update and you can't collect logs because the stable system value has changed, and you don't have the password to the lockbox.

These things are configured from **Administration->Services-><YourVLC>->Config->Settings**.

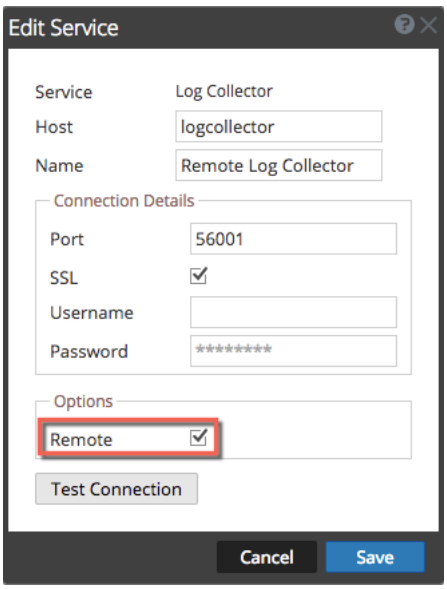


To set the initial lockbox password, enter the password in the **New Lockbox Password** field and click **Apply**. You don't need to enter anything in the **Old Lockbox Password** field.

If the stable system value has changed and the lockbox can't be unlocked (you will know this either from **/var/log/messages** or an alert in Health & Wellness), simply enter the lockbox password under **Reset Stable System Value**.

Telling SA that a LC is Remote

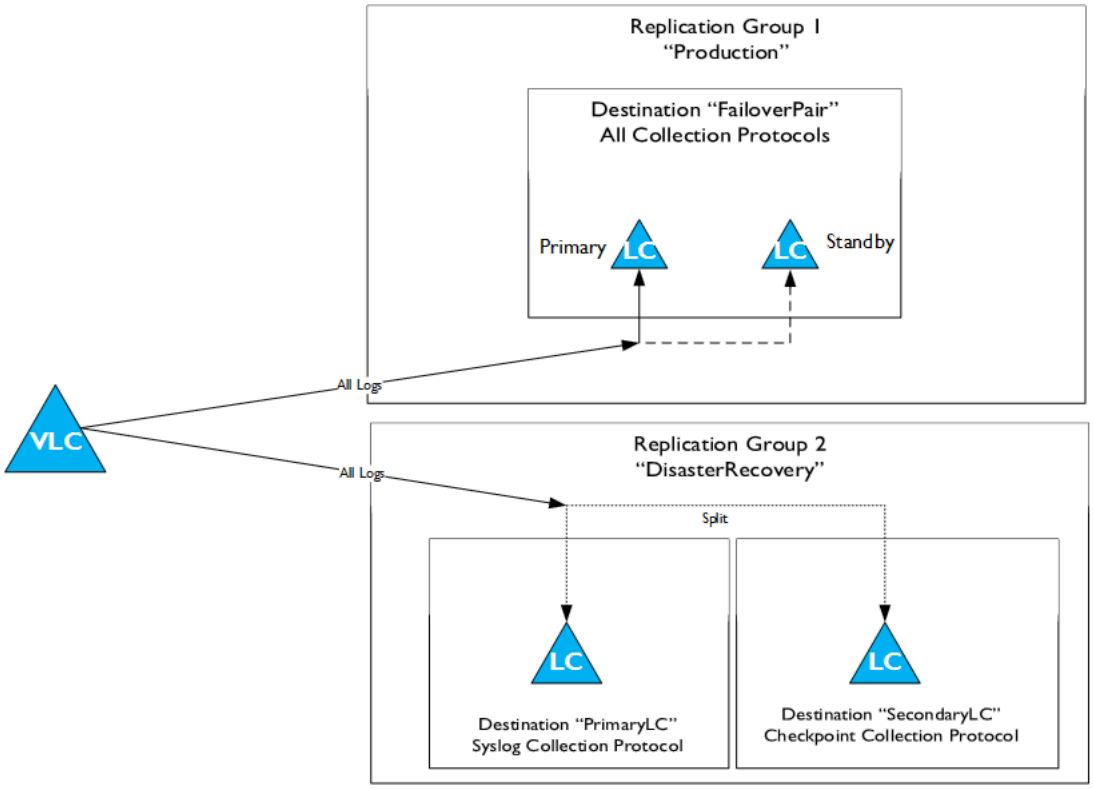
Though local and remote collectors run exactly the same Log Collector code, how they are used, the configuration they get, and the user experience differs between them. SA should detect whether a collector is local or remote, but sometimes it gets it wrong, or it may forget that you configured it to be a remote collector. Should this ever happen, a Log Collector is designated as being Remote from **Administration->Services.** Select your VLC from the list of services , and click the edit service button ( ). Tick the box where it says **Remote** and click **Save**.



Configure a VLC to Forward Logs to a Local Collector

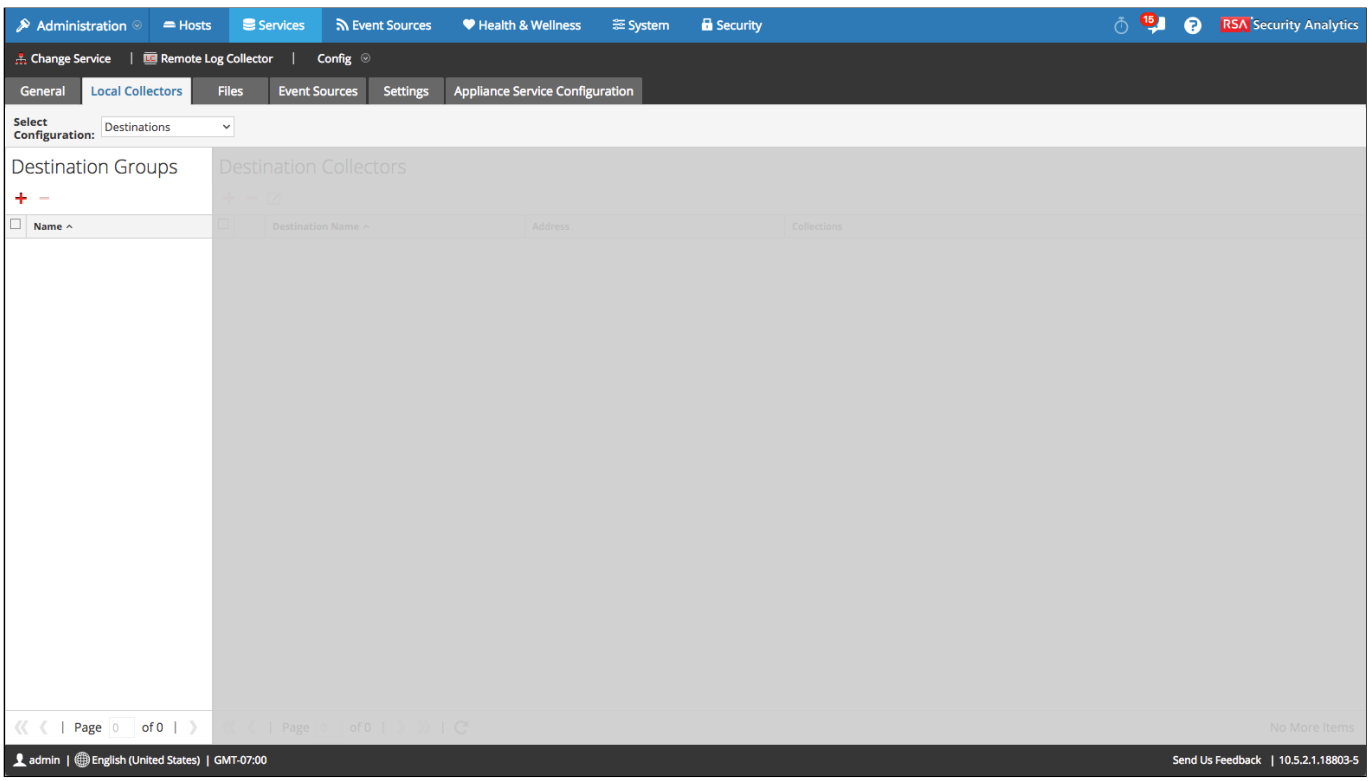
Log Collectors can be configured to push logs to a downstream LC, or a downstream LC can be configured to pull logs from an upstream VLC. For reasons we shan't get into here, we strongly recommend using the push model rather than the pull model. Log collection has some fairly robust features:

- Push logs downstream
- Pull logs from upstream
- Log replication between Destination Groups
- Failover between Local Collectors
- Balance logs between Local Collectors by collection protocol



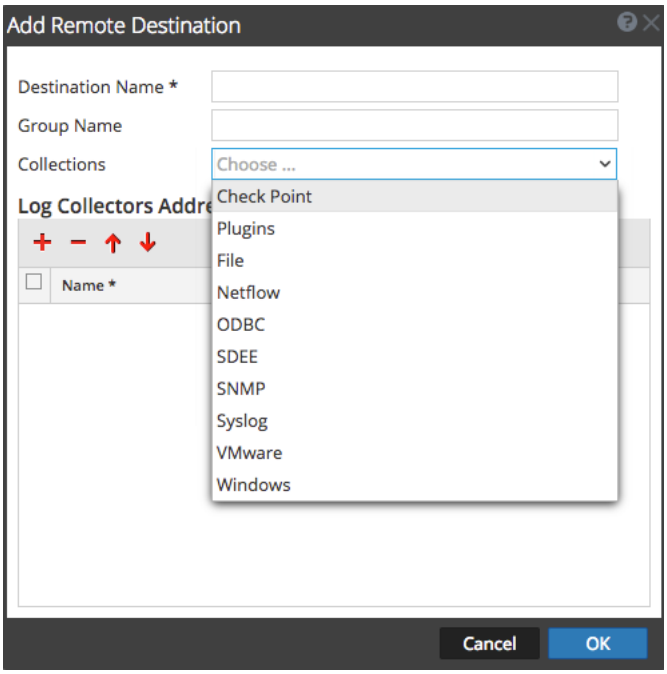
Shall we visit **Administration->Services->**<YourVLC>**->Config**? Now click the **Local Collectors** tab. Make sure you're looking at the VLC and not you Local Collector. If you don't see the **Local Collectors** tab, but instead see a **Remote Collectors** tab, you're either looking at the Local Collector, or your VLC hasn't been configured to be a Remote Collector under **Administration->Services**. It is here where we configure other collectors which we can either push logs to or pull logs from.

Full disclosure: the user experience here is particularly confusing and unintuitive. Please bear with me as I try to help you make sense of it.



Notice the dropdown called **Select Configuration**. From here, we can choose **Destinations** (the default), or **Sources**. **Destinations** means collectors that we will push logs to. **Sources** defines collectors that we will pull logs from. We shall concern ourselves only with **Destinations**.

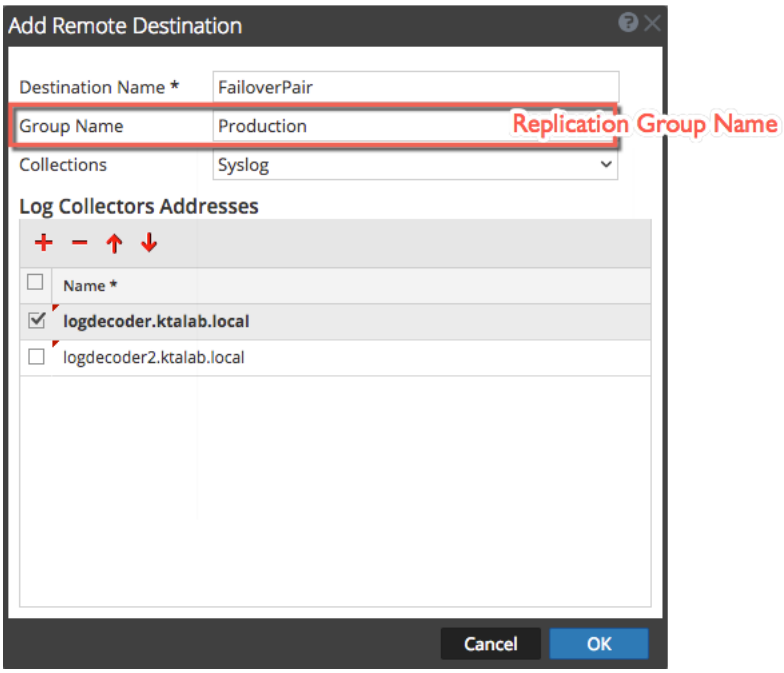
If we click the + button, we shall receive a dialog asking us to define a Group, Destination, Collections (collection protocols), and Log Decoders/Failover pairs.

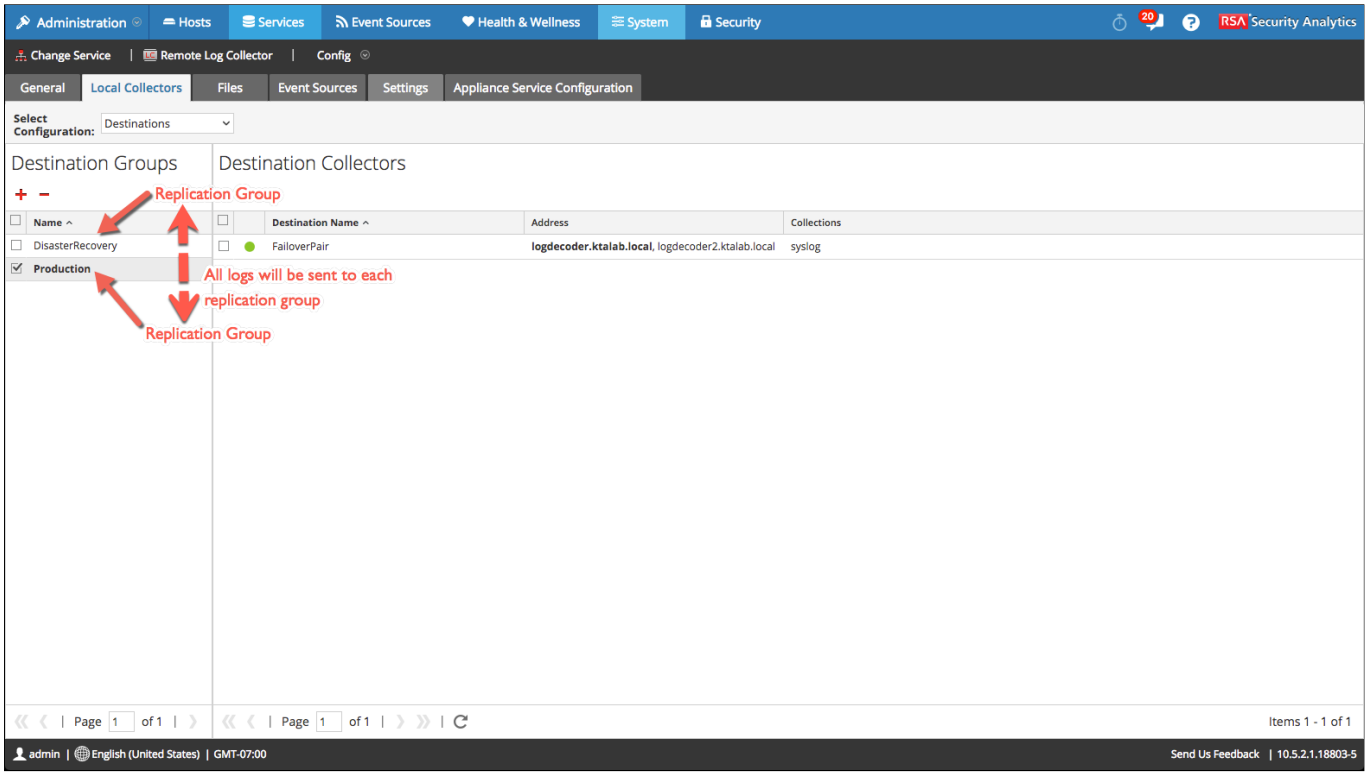


Replication With Destinations Groups

The '**Local Collectors**' page is split into two panes: **Destination Groups** and **Destination Collectors.** The purpose of **Destination Groups** is to replicate log traffic. Every log received by the collector will be pushed to **every** defined **Destination Group**. This is useful for disaster recovery scenarios.

We define a new group with the **Group Name** field in the **Add Remote Destination** dialog.

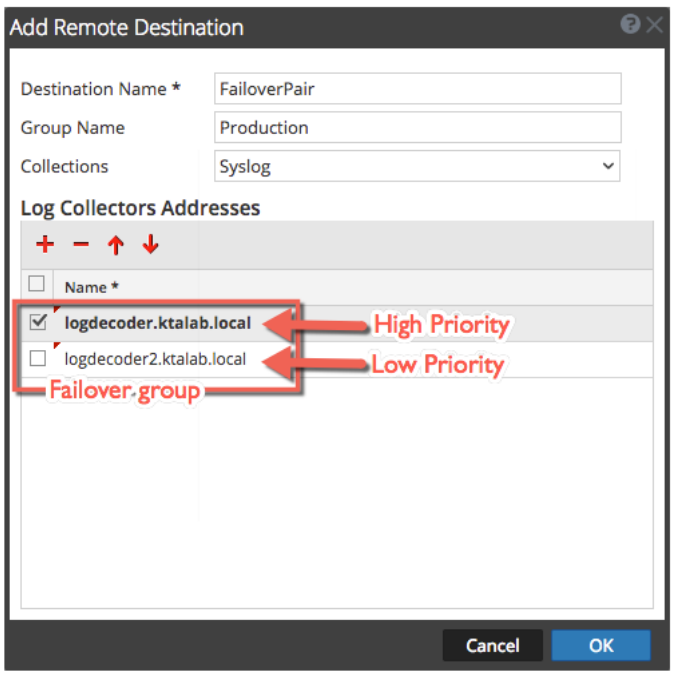


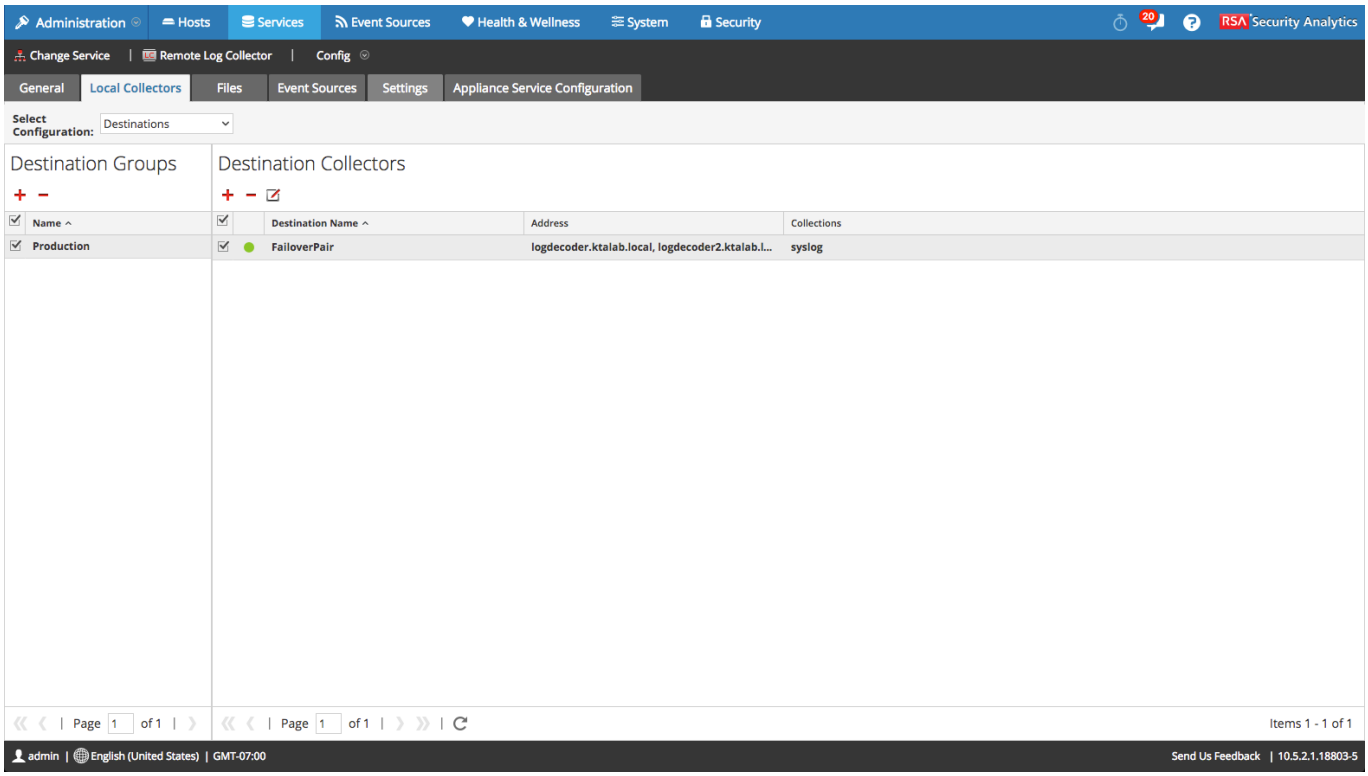


Two Replication Groups

Failover Between Local Collectors - 'Destinations'

Each replication group contains at least one '**destination**'. A **destination** is one or more Log Collectors to send log traffic to. If more than one Collector is defined within a single destination, this constitutes a failover group, prioritised highest to lowest. Logs will only be sent to a single collector in the failover group at a time, starting with the collector with the highest priority. If this Collector should go down, new logs will automatically swing over to the collector with the next highest priority.



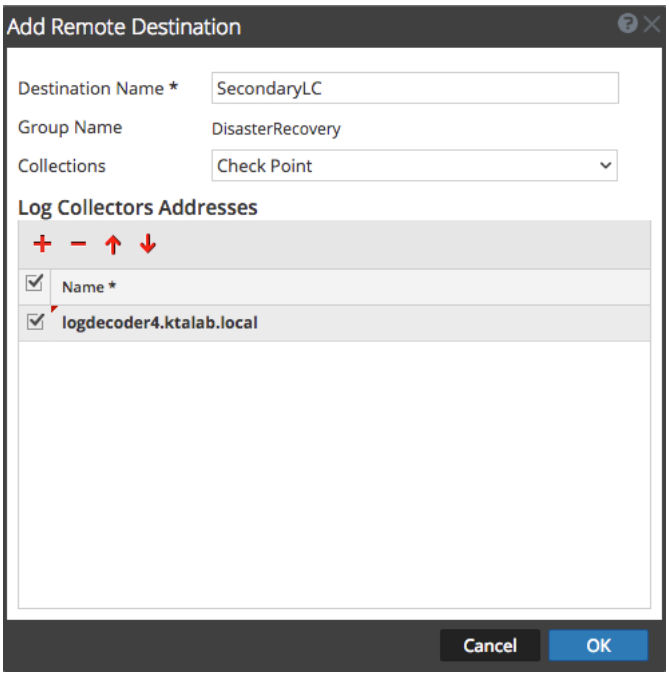


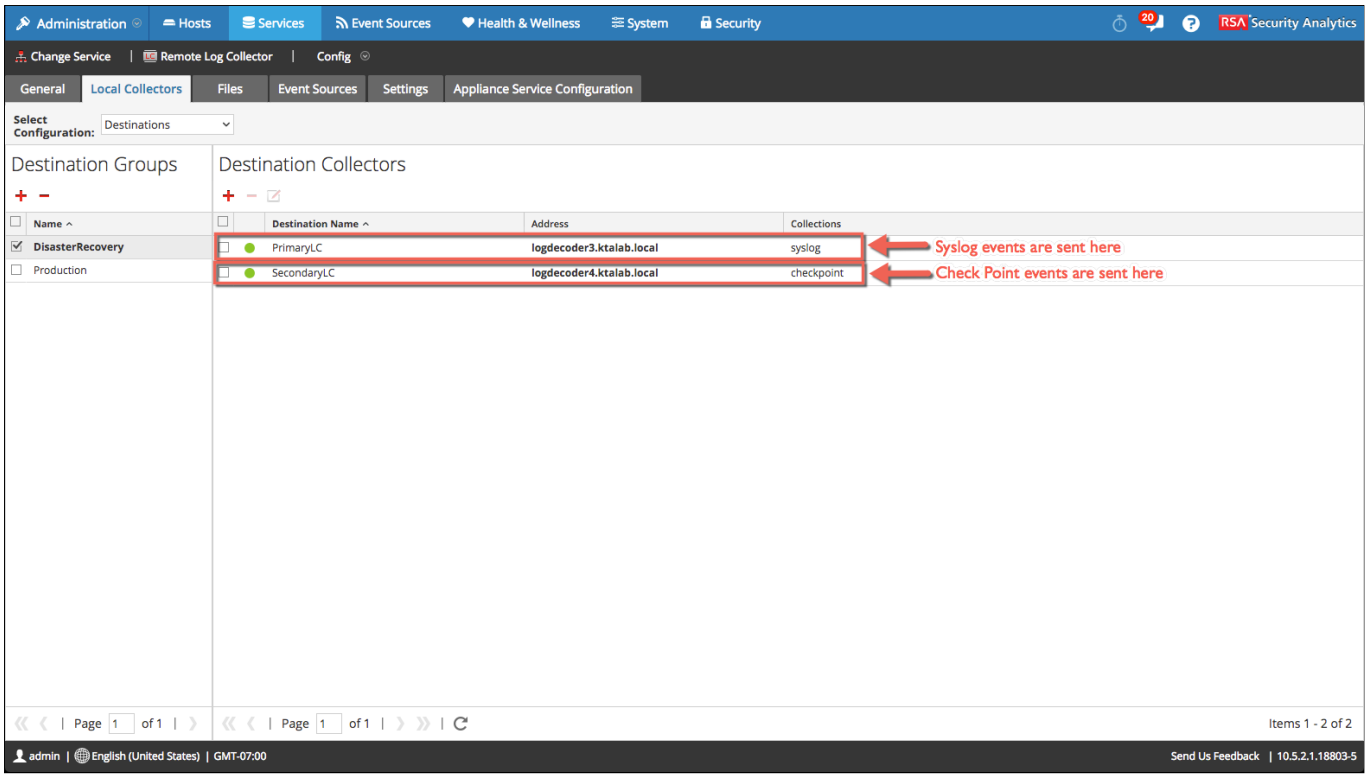
Single Replication Group (no replication) with Single Failover Pair

Load Balancing Logs Between Local Collectors by Collection Protocol

One **destination** is distinguished from another destination in the same replication group by its **collection protocols**. By creating more than one **destination** within a **replication group**, we can create a crude form of load balancing, split by **collection protocol**. For instance, we may have more than one Log Decoder in a data centre, and we wish split up the load by sending syslog logs to one LD, and Check Point logs to another.

To add a load balanced destination, click the + button under **Destination Collectors**.





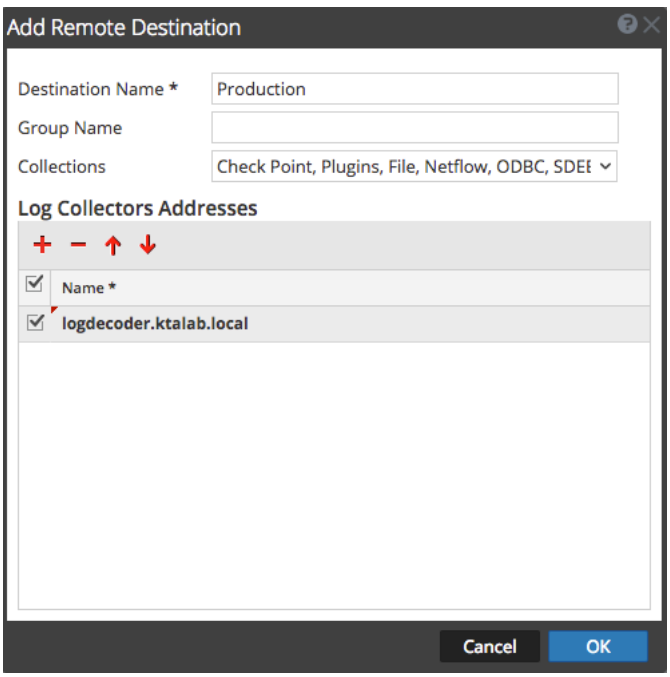
Two replication groups with DR site load balancing between two collectors, split by protocol

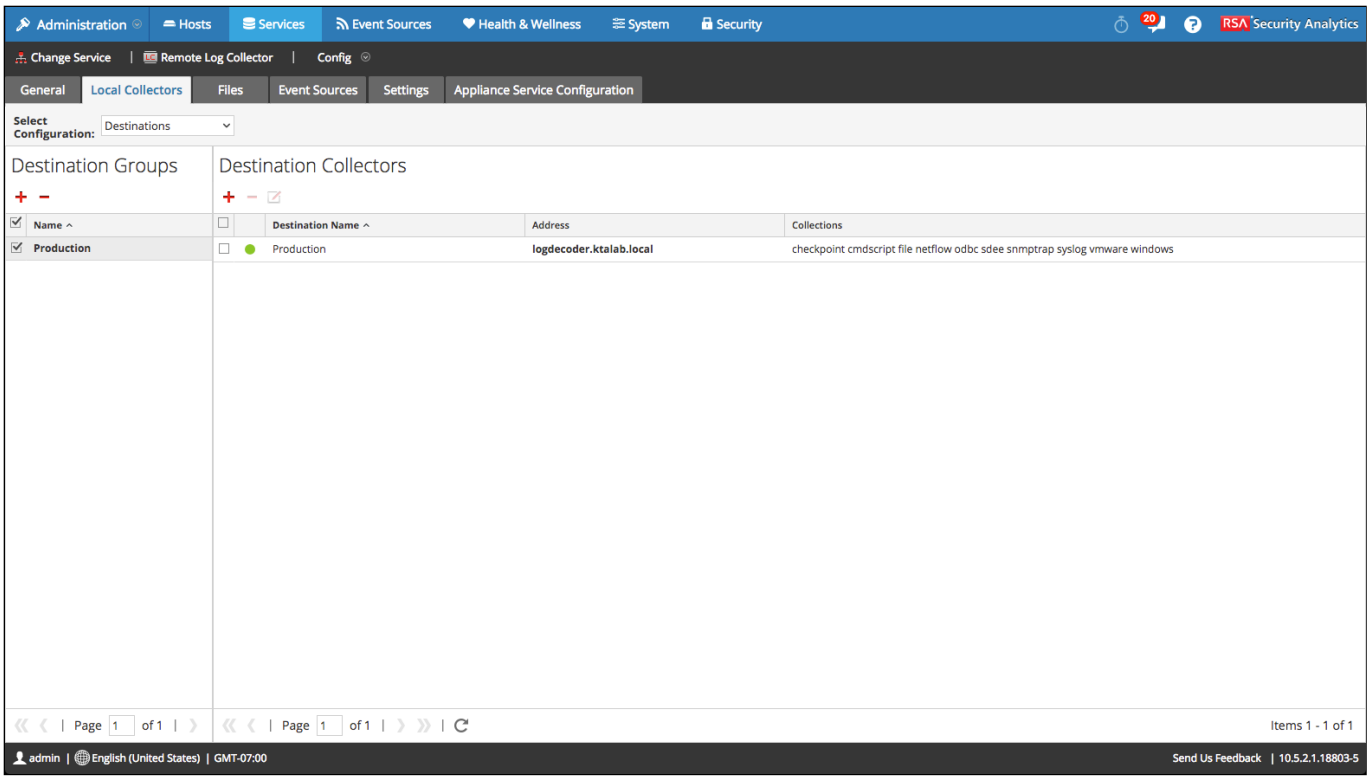
Real World Scenario

What I've just described regarding failover, load balancing, and replication, are not used terribly often in the real world. So let's build the typical and simple real-world scenario, consisting of one VLC to one Local Collector, with all collection protocols.

Now define a destination group. Click the + button and complete this form:

- **Destination Name**: must be composed of alphanumeric characters, and may contain no spaces.
- **Group Name** - In this simple scenario, this can be left blank. The replication group name will take the name of the destination.
- **Collections** - Choose all collection protocols from the dropdown
- **Log Collectors Addresses** - The local collector to send logs to. Should be the local collector on your Log Decoder



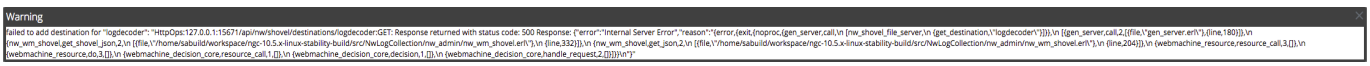


What Does Hitting OK Actually Do?

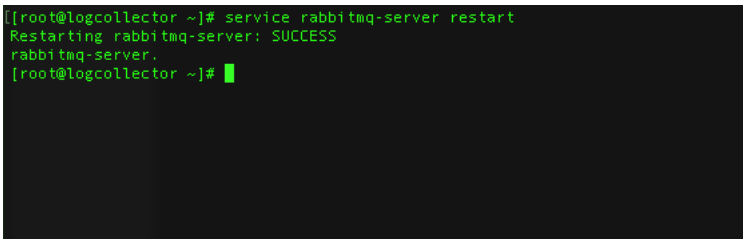
This step in fact triggers a reconfiguration of RabbitMQ on the VLC and the Local Collector it connects to, to enable RabbitMQ **Federation**, through which logs will be pushed.

What if it Goes Wrong?

This can occasionally go wrong, as evidenced by a pop-up error such as the below:



In my experience, this is almost always fixed by running this on the VLC via ssh: **# service rabbitmq-server restart** . Once this is finished, try again.

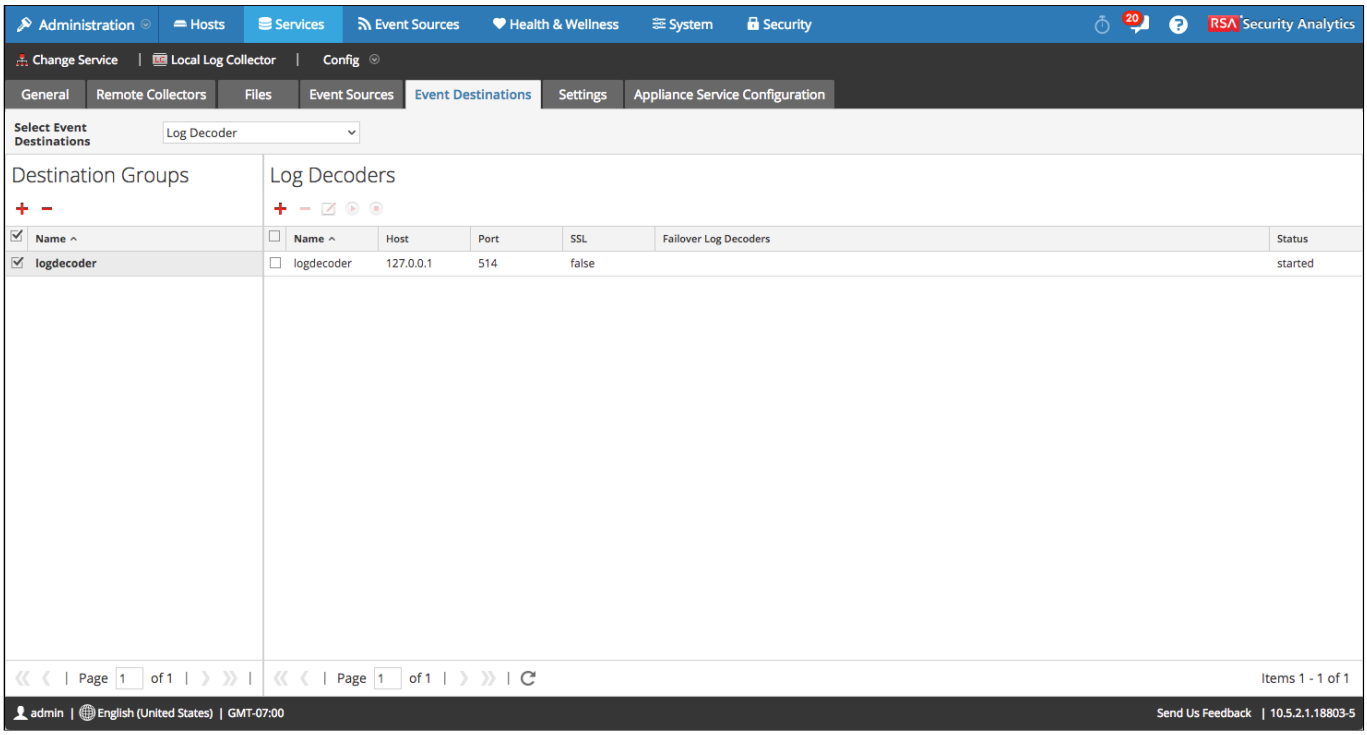


Event Destinations

Event Destinations are a concept particular to **Local Collectors**. An event Destination is one of two things:

- A Log Decoder
- An Identity Feed

Configuring an **Event Destination** instructs a Local Collector to transmit log events to said destination through the use of a sub-process called an **Event Processor**. It usually isn't necessary to configure an **Event Destination**, as the Local Collector by default adds the Log Decoder which is present on the local host. You can get here from **Administration->Services-><YourLocalCollector>->Config->Event Destinations**.

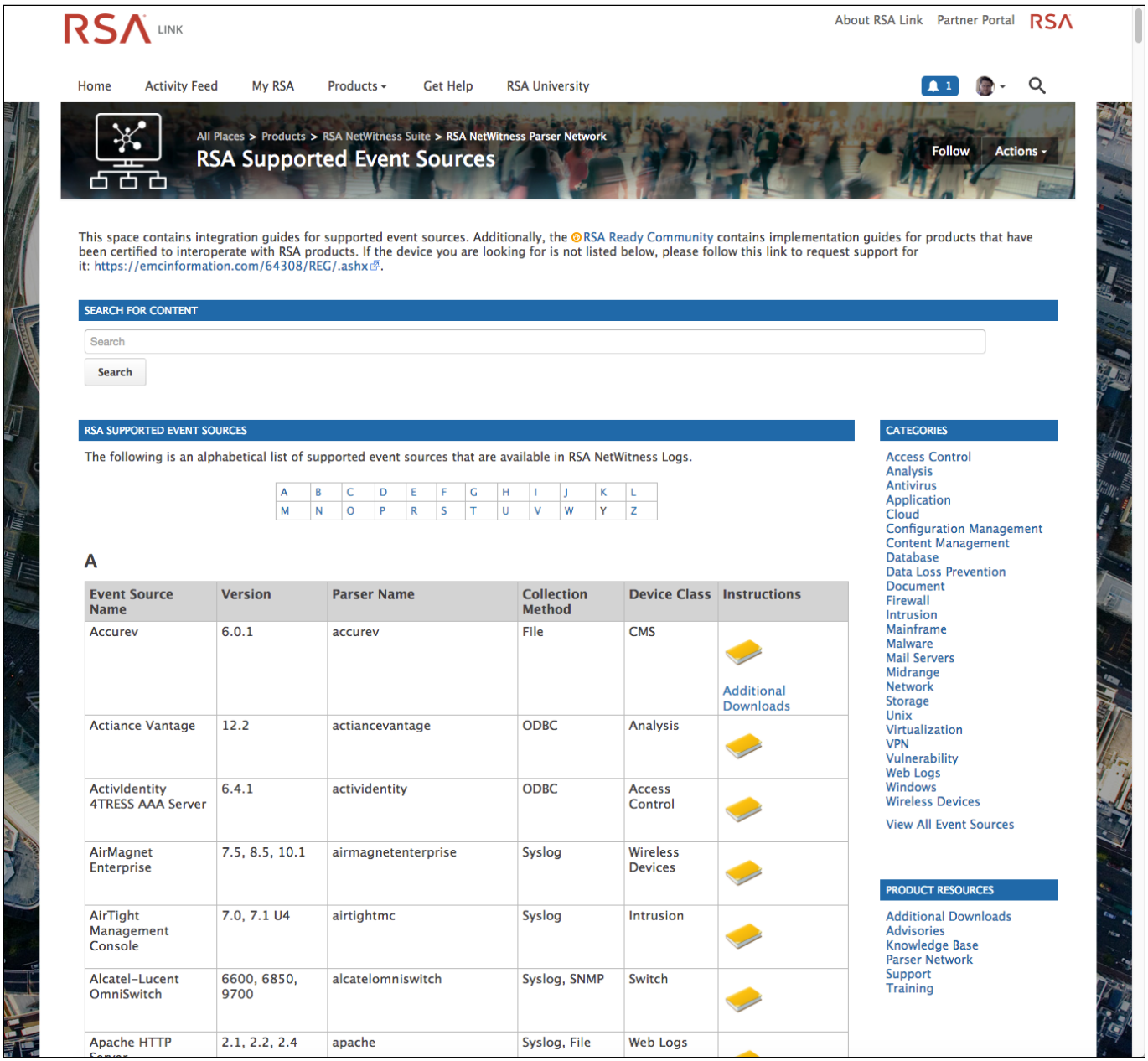


Default Event Destination

Identity Feeds will be discussed later on in the lesson.

Identify Needed Collection Configuration

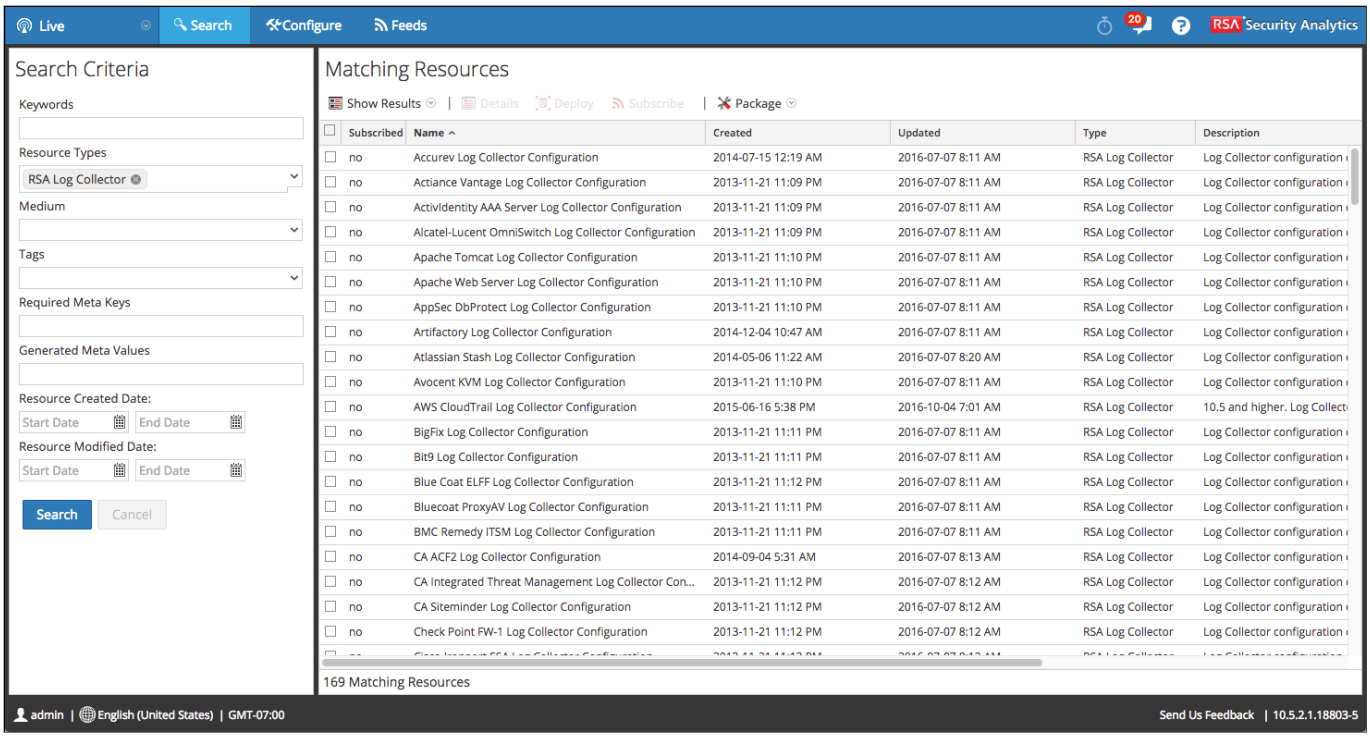
Before even beginning to configure collection for a particular event source type, you need to know that SA even supports parsing its logs properly. While RSA has published scores of log parsers, some of them might be outdated or only for particular versions of a product. For this, stop by the RSA Community site at https://community.rsa.com/community/products/netwitness/parser-network/event-sources. It lists all supported event sources, and it contains guides for the configuration of each type of event source. Be forewarned: **some of the guides are a bit rubbish**. You will just have to work through the ambiguity.



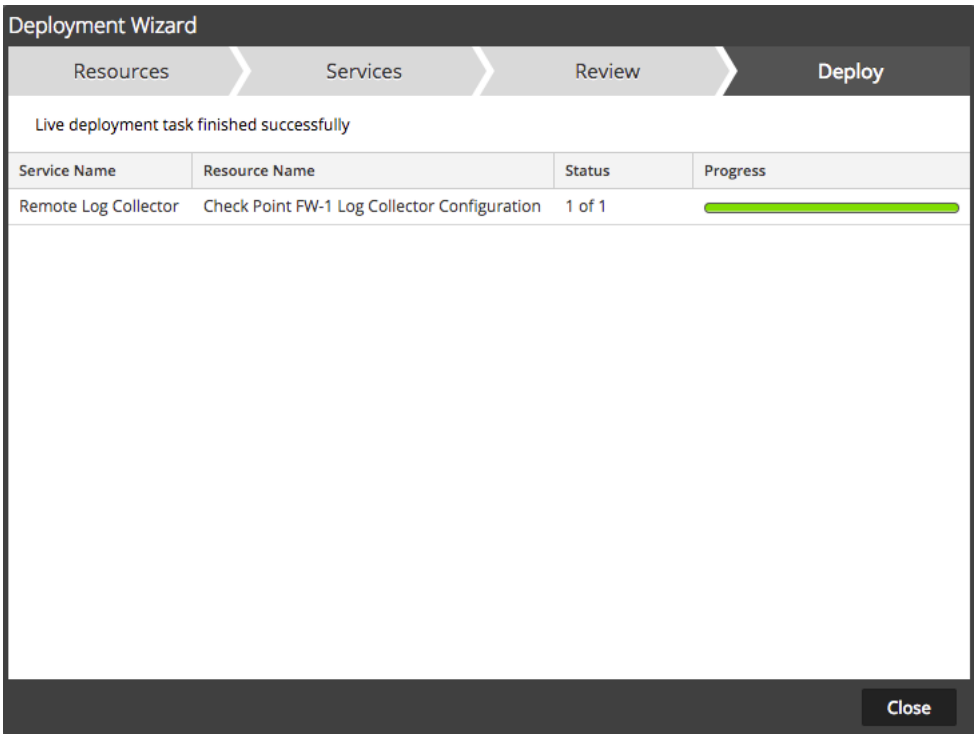
Identify and Deploy Live Content

When collecting logs from non-syslog-like event sources, it's important to remember that the Log Collector needs to know how to transform non-syslog traffic into syslog-style traffic that a Log Decoder can parse. For instance, SNMP is a protocol rather than a log format, so for any supported SNMP event source types, the Log Collector needs to know how to to translate this into a real log. When pulling from ODBC sources, the collector is pulling database entries, and needs to know how to map those raw data values into a log message.

Enter Log Collector **Live** Content. Have a look at **Live->Search.** Select the '**RSA Log Collector'** resource type and click **Search**.



Any time you are to onboard a new non-syslog device type, ensure you've deployed any Live content for it to your Log Collectors. It need only be applied to the devices which perform the collection, so likely only VLC's.



Configuring a Collection Protocol

Configuring collection protocols is one of the broadest topics in all of SA. It would be impractical to cover all scenarios and event source types. As such, I shall endeavour to constrain our lesson to a couple of simple-ish use-cases. In nearly all circumstances, log collection will require configuration in both the Log Collector and also in your own environment: servers, routers, switches, firewalls, workstations, etc.

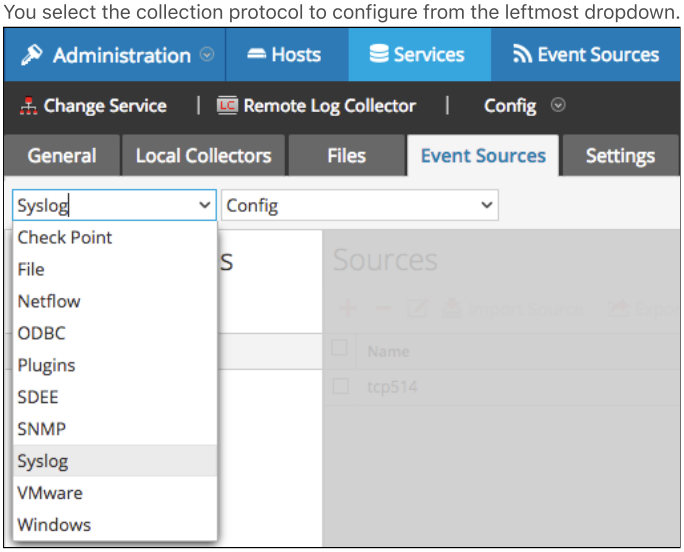
The high-level procedure is:

1. Consult the SA documentation to confirm your event source type and version is supported. If so:
2. Install any relevant Log Collector content from RSA Live.
3. Configure your event sources to either send logs to your VLC, or permit your VLC to pull logs from them.
4. Configure the **Config->Event Sources** tab of your VLC to collect data from your event source using the appropriate collection protocol.
5. Ensure your collection protocol is selected on your **Config->Local Collectors** tab, so that the log events get forwarded to your Local Collectors.
6. Start the collection service for your collection protocol in the System tab of your VLC.

Exercise - Configuring Syslog Collection

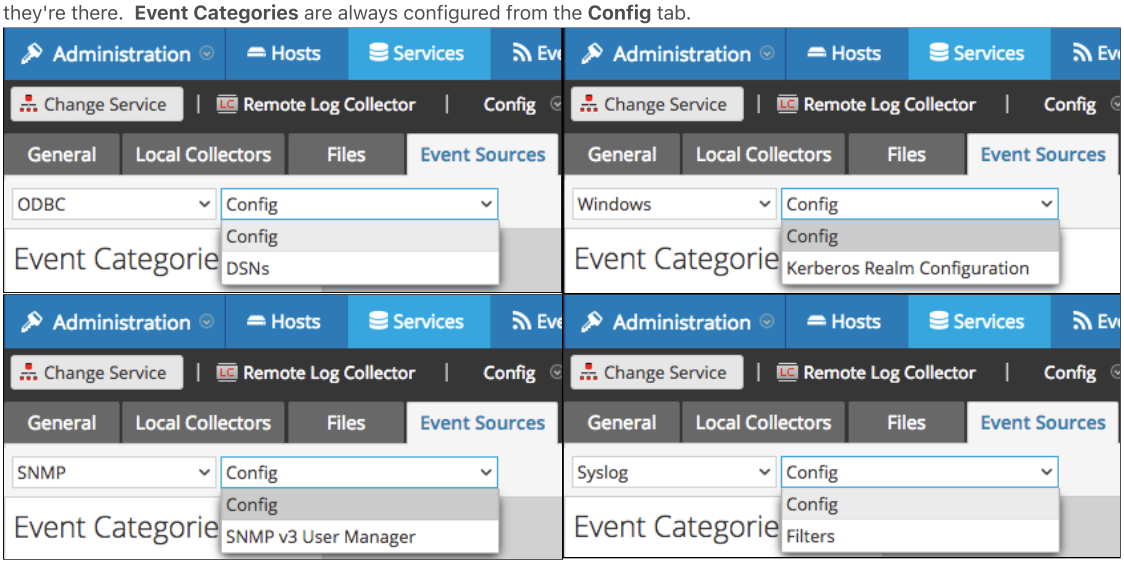
Now let's configure syslog collection. In our lab, there is syslog event data being sent to your VLC at regular intervals. We just need to collect it. This is a promiscuous-type collection protocol, so it needs only rudimentary configuration on the VLC. Syslog events do not require any collector content from RSA Live, though every event source type requires a Log Parser compatible it.

1. Navigate to **Administration->Services-><YourVLC>->Config.**
2. You select the collection protocol to configure from the leftmost dropdown. Choose **Syslog**.

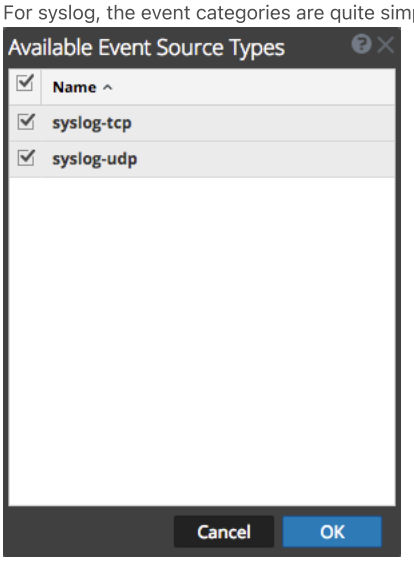


3. **Event Categories** is a slightly nebulous concept, whose purpose is dependant on which collection protocol you're configuring. For instance, for ODBC collection, it essentially represents application types (bit9, epolicy, mcafeedlp, etc).

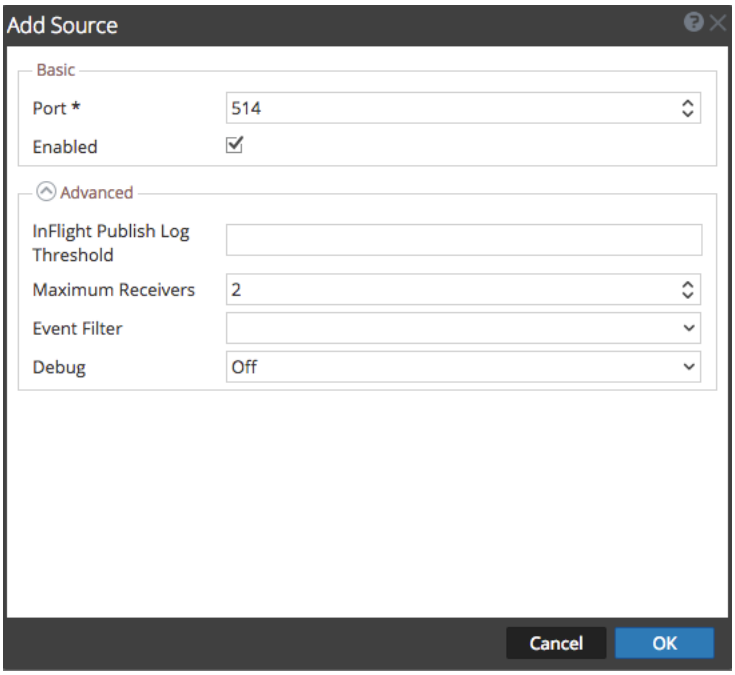
   From the righthand dropdown, there may be other protocol-specific tabs, like Kerberos Realms for Windows, DSN's for ODBC, SNMPv3 User Manager for SNMP collection, and Filters for Syslog. We shan't concern ourselves with them for our exercise, but you should at least know that they're there. **Event Categories** are always configured from the **Config** tab.



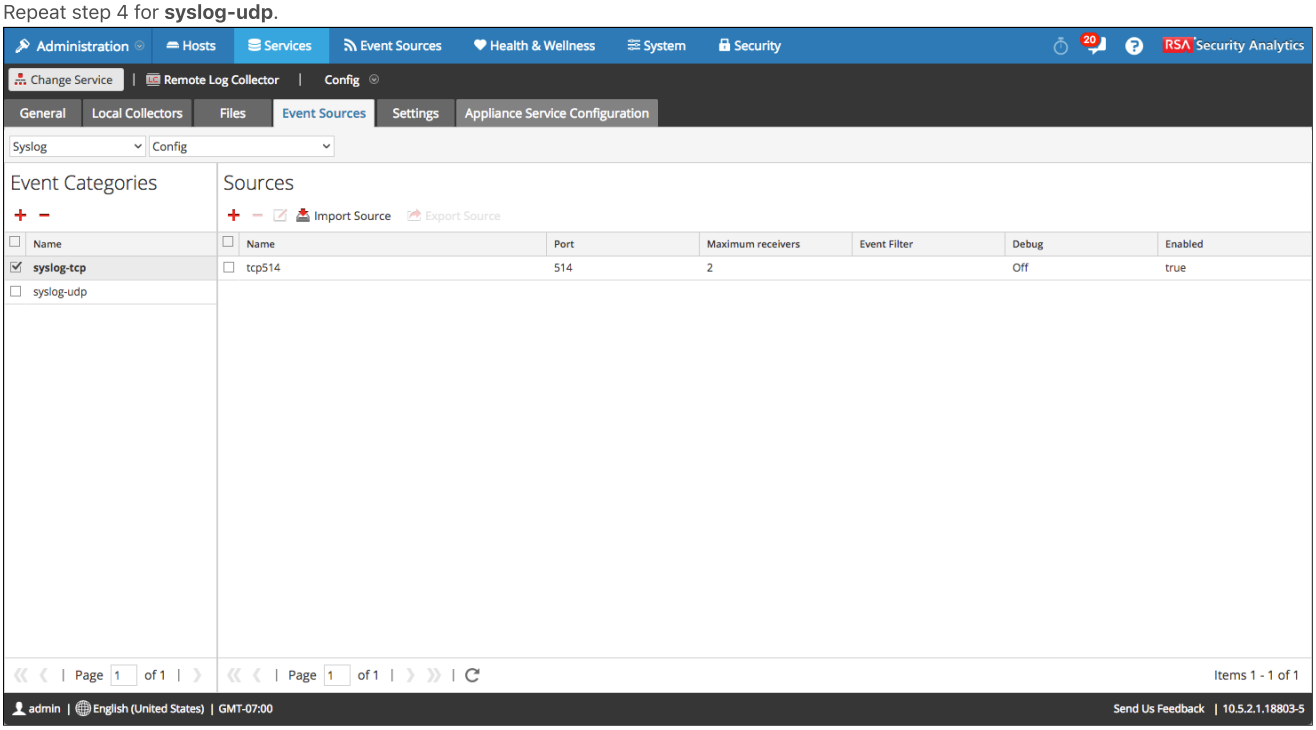
   For syslog, the event categories are quite simple: **syslog-tcp** and **syslog-udp**. Click the + button, and add both types. Click **OK**.



4. You'll now see **syslog-tcp** and **syslog-udp** listed under **Event Categories**. Now all we need to do is configure these two categories to listen on specific ports. Click on one of them, and notice the **Sources** tab light up. Now click the + button under **Sources**. Enter the default syslog port number of **514.** Leave the advanced settings at their defaults. Click **OK**.



5. Repeat step 4 for **syslog-udp**.



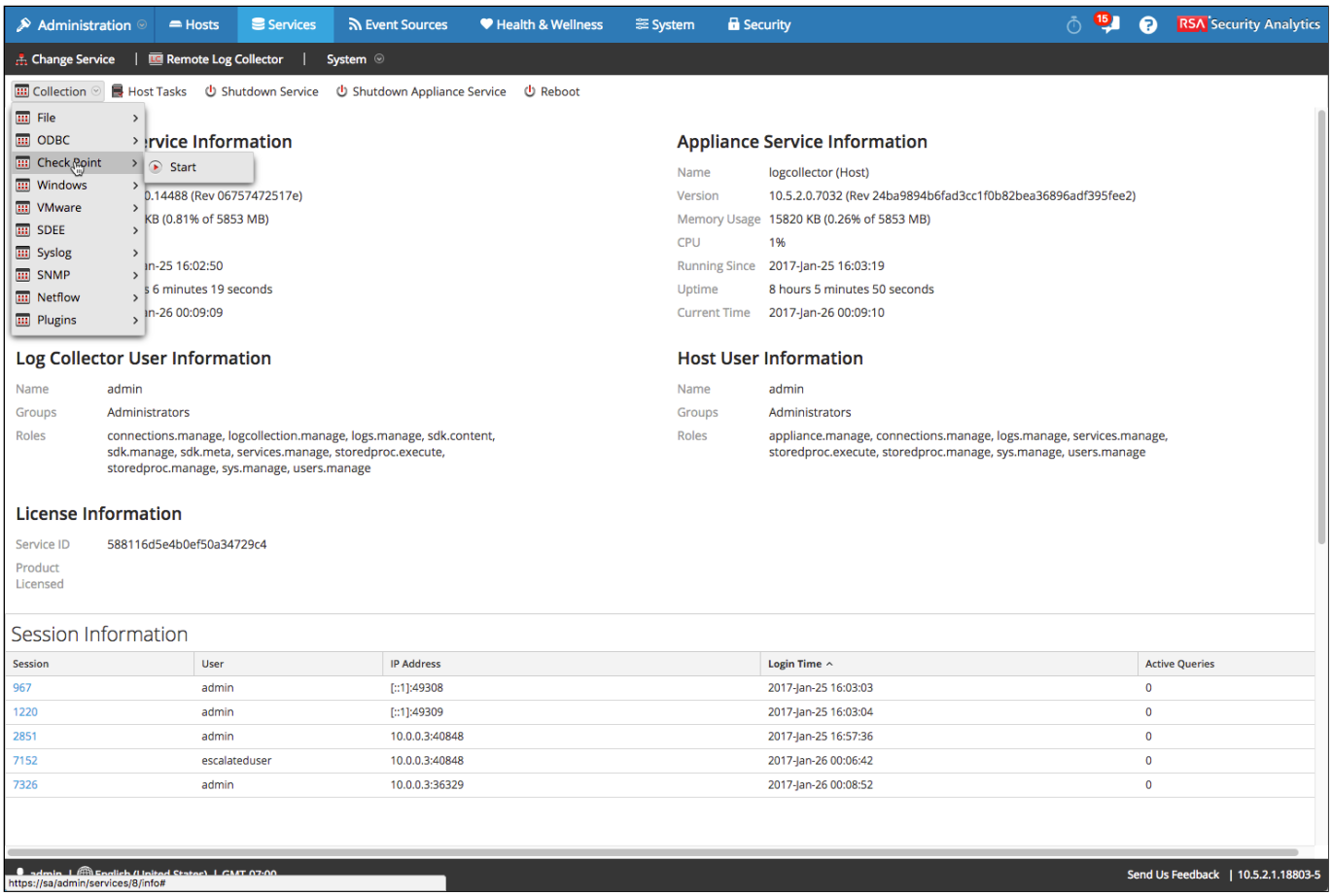
6. Congratulations, you've configured the syslog collection protocol! Follow the next section on starting a service to start the syslog Collection Service, and also configure it to autostart.

In your environment, you likely have more protocols to configure than syslog, but including them in our lab would be difficult at this juncture. We do aim to improve our offerings here in the months to come.

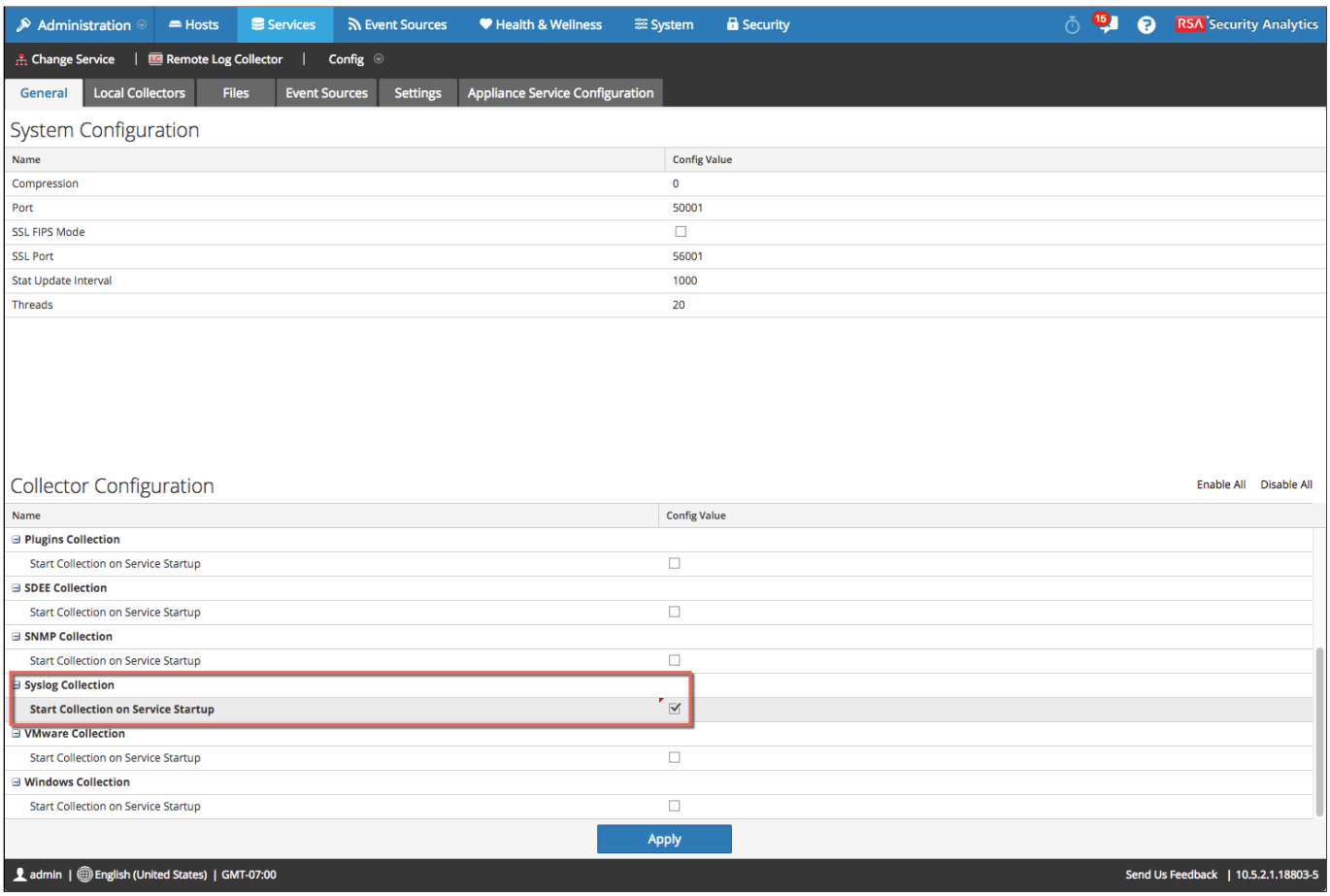
Top Tip: If you'd like to collect Windows event logs over WinRM, there can be significant administrative overhead to facilitate this in a Windows domain. An alternative to using WinRM is using a third-party product called **Snare**. by InterSect Alliance at https://www.intersectalliance.com/our-product/snare-agent. You install the Snare agent on your Windows hosts, and it will forward Windows event logs to your LC in syslog format. RSA does has a parser for Snare logs. There is cost associated with Snare, but it is considerably easier than configuring up WinRM. There is a caveat: if you have supported (read: parsed) third-party products which log to the Windows event log, you will not be able to use Snare for these, as RSA only publishes parsers for the native logging formats of applications. Snare is only supported for Windows logs and Linux logs. You can get around this limitation if you write a custom log parser, however.

Starting a Collection Service

Once you've configured your collection protocol, its collection service must then be started. You do this in **Administration->Services-><YourVLC>->System**, under the Collection dropdown. Collection services need only be started on VLC's, if you're only using VLC's for collection. There is no need to start them on Local Collectors.



Be sure to also configure the collection service to auto-start from **Administration->Services->**<YourVLC>**->Config** under **Collector Configuration**.



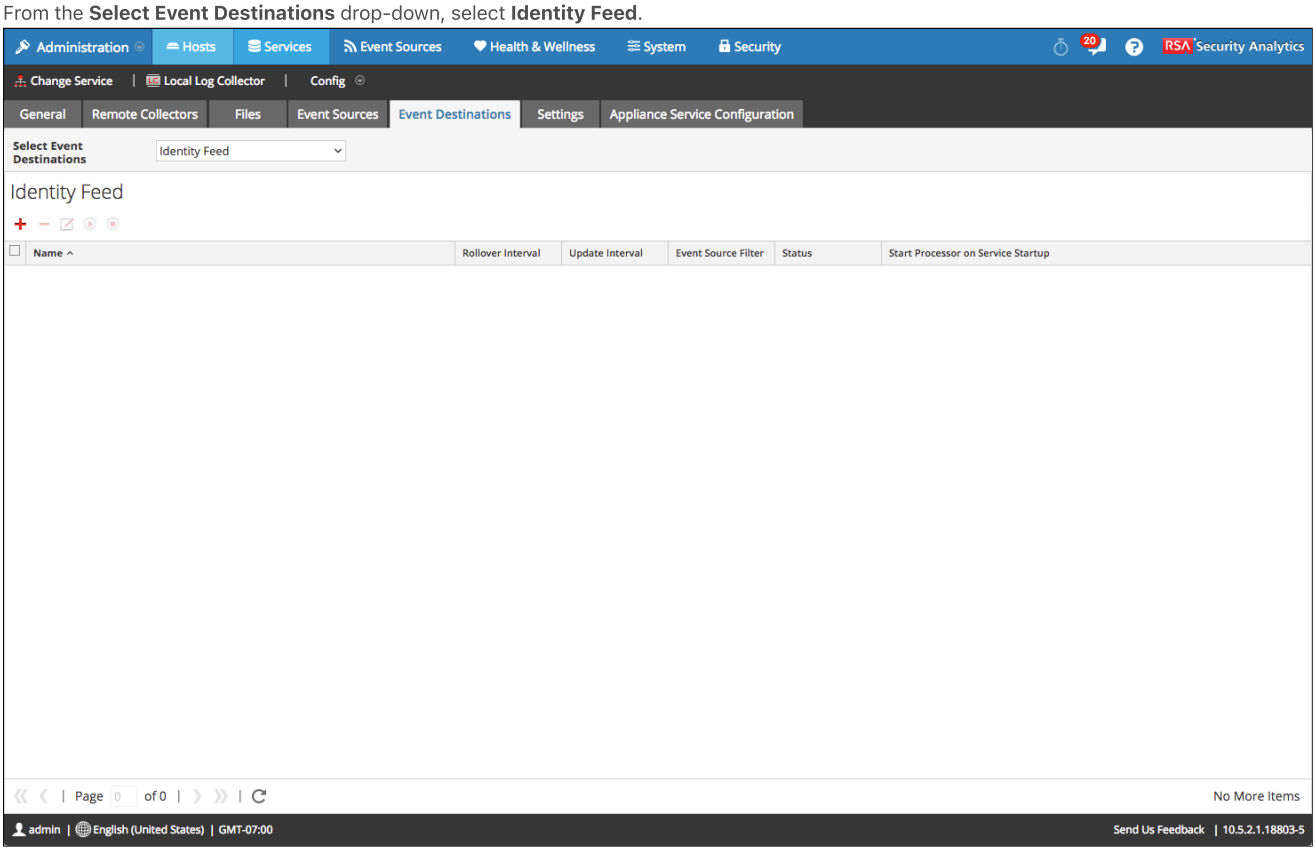
Identity Feeds

Identity feeds allow you to re-inject user and computer information from collected WinRM events back into Decoders. Setting up identity feeds is essentially a three-step process:

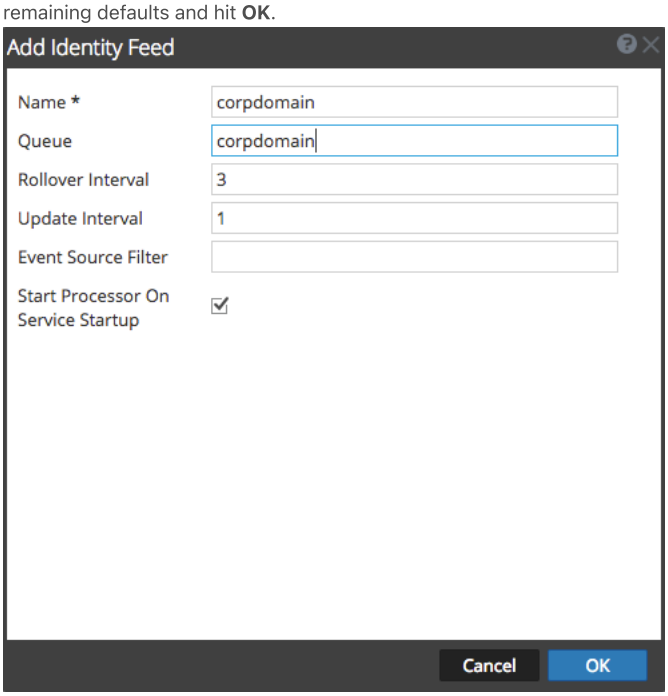
1. Configure Windows (WinRM) collection
2. Create an identity feed Event Processor in a Log Collector
3. Configure Live to fetch the identity feed and push it out to your Log Decoders.

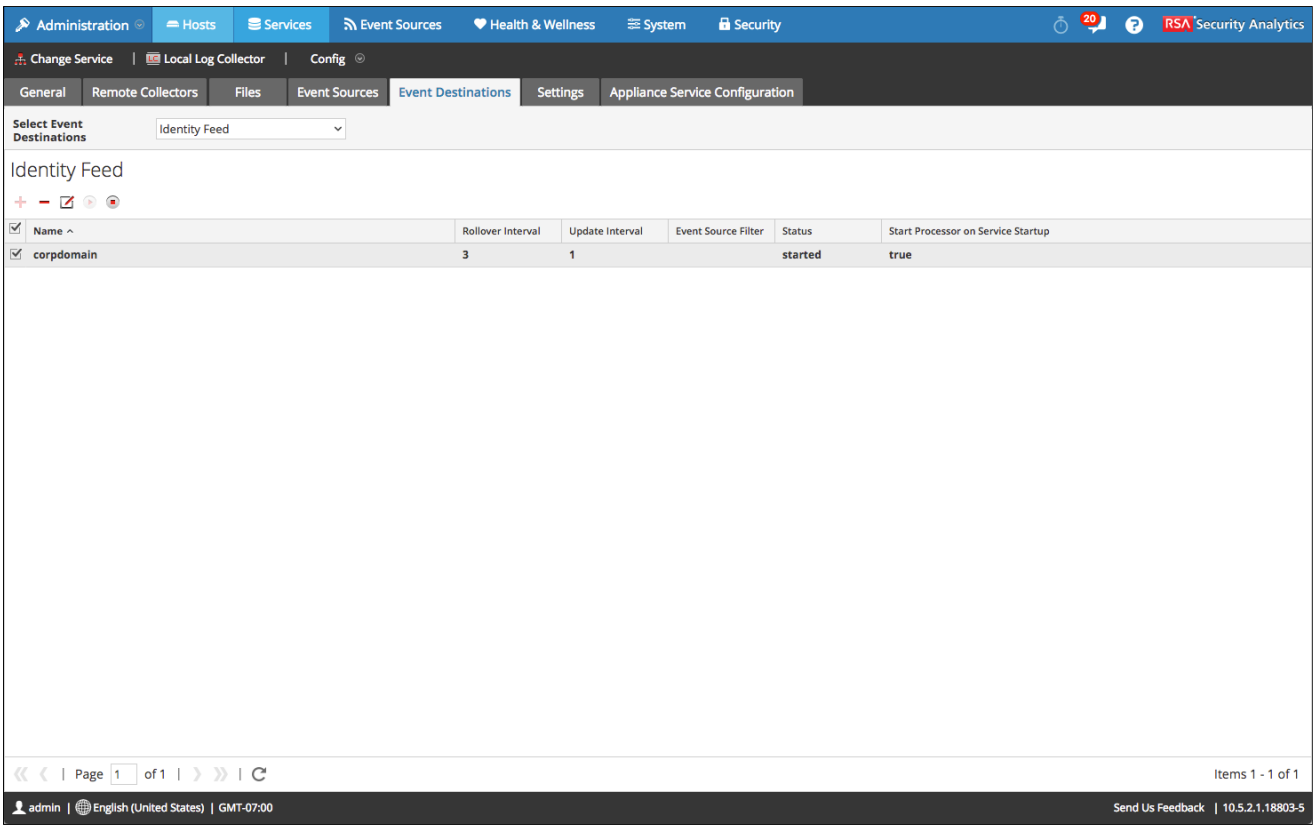
To configure the identity feed Event Processor on your Local Collector:

1. Navigate to **Administration->Services-><YourVLC>->Config->Event Destinations.**
2. From the **Select Event Destinations** drop-down, select **Identity Feed**.



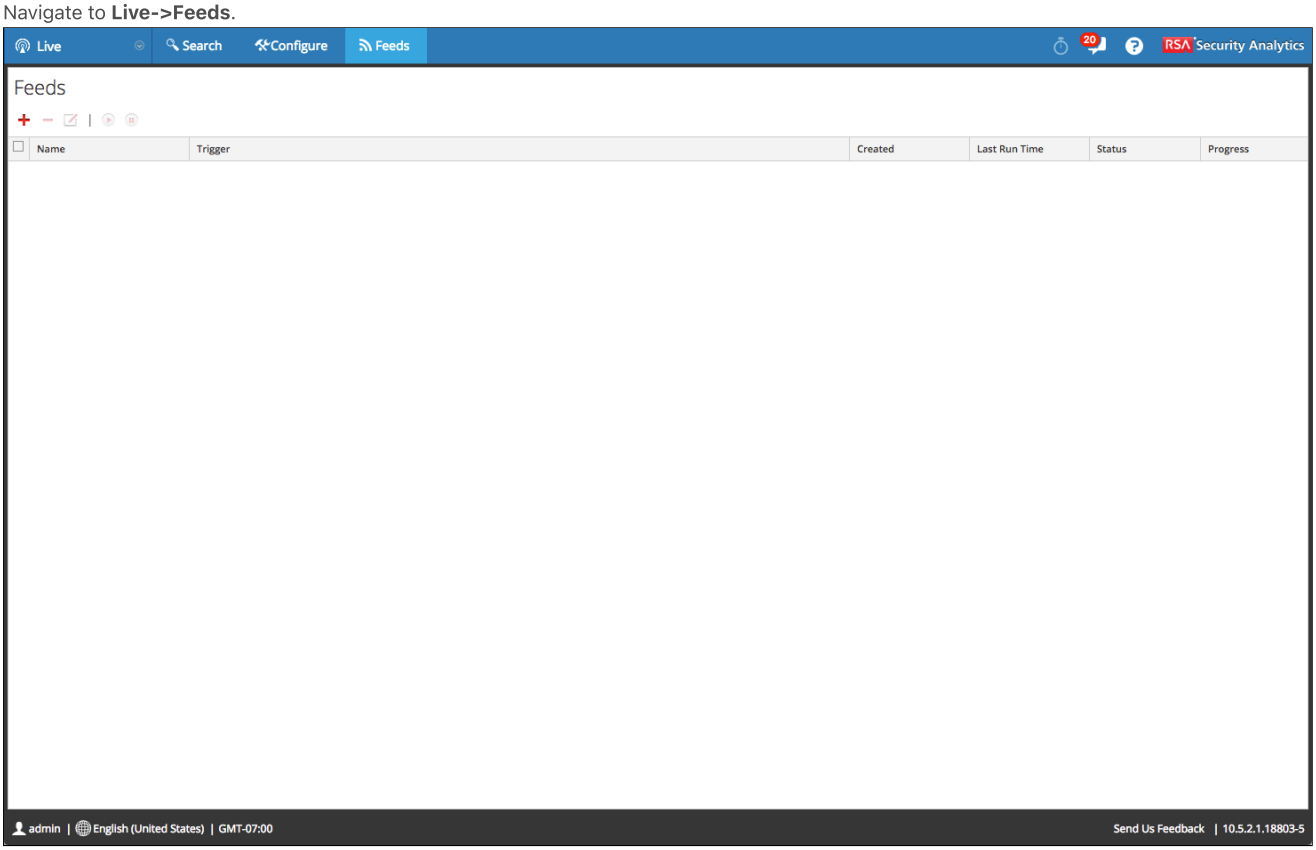
3. Click the + button.
4. In the Add Identity Feed dialog, enter a name for the feed (let's use '**corpdomain**'), and enter the same name in the queue field. Leave the remaining defaults and hit **OK**.





Pushing Your Identity Feed Out to Decoders

1. Navigate to **Live->Feeds**.



2. Click the ➕ button.
3. Under the **Feed Task Type**, select **Recurring.**
4. Enter a name for the feed, using only alphanumeric characters with no spaces.
5. Under URL, enter this URL (***Identity Event Processor Name*** is the name of the Identity Feed which we previously created in our Log Collector) : http://***<LogCollectorHost>***:50101/event-processors/***<Identity Event Processor Name>***?msg=getFile&force-content-type=application/octet-stream&expiry=600

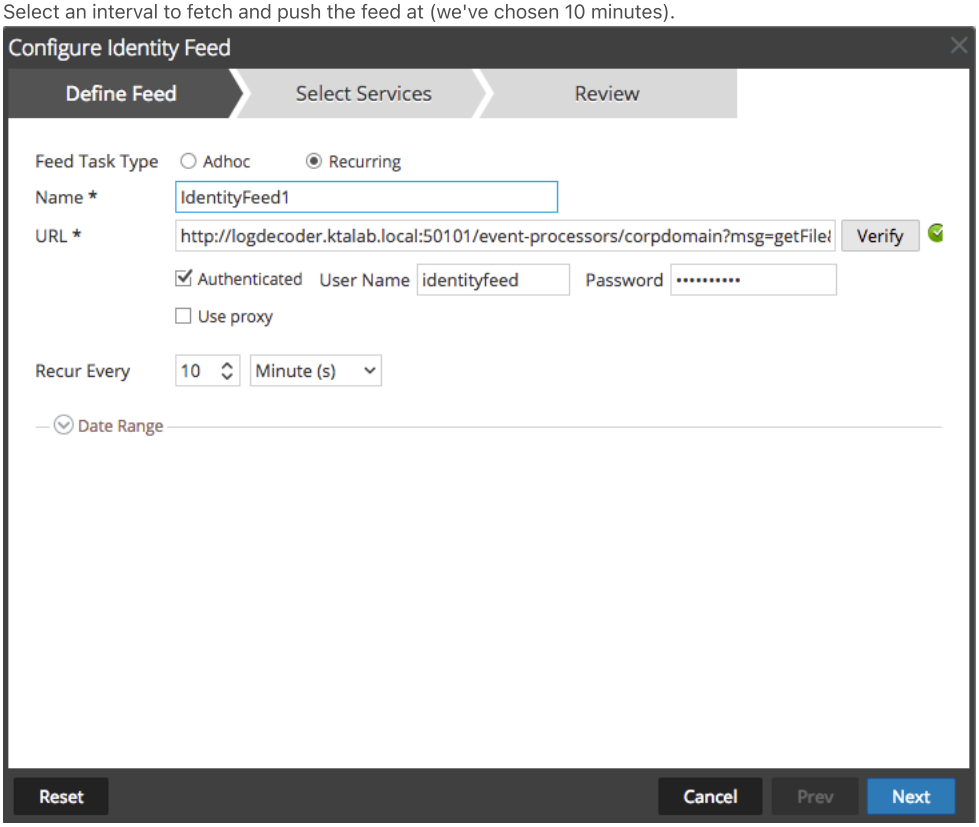
In our example, we'll use **http://*logdecoder.ktalab.local*:50101/event-processors/*corpdomain*?msg=getFile&force-content-type=application/octet-stream&expiry=600**

6. Tick the box for **Authenticated**, and enter a Log Collector-local user account to pull the identity feed with. You should not use an admin account, but rather create an account specifically for this purpose. In our example, we're using an account we've named '**identityfeed**'.
7. Click the Verify button to confirm it can fetch the feed file. If not, use the '**wget**' command from ssh to test:

```
[[root@sa ~]#
[[root@sa ~]# wget --user identityfeed --password=netwitness http://logdecoder.ktalab.local:50101/event- processors/corpdomain?msg=getFile&force- content-type=application/octet-stream&expiry=600'
--2017-01-27 19:29:21--  http://logdecoder.ktalab.local:50101/event-%20processors/corpdomain?msg=getFile&force-%20content-type=application/octet-stream&expiry=600
Resolving logdecoder.ktalab.local... 10.0.0.6
Connecting to logdecoder.ktalab.local|10.0.0.6|:50101... connected.
HTTP request sent, awaiting response... 401 Unauthorized
Connecting to logdecoder.ktalab.local|10.0.0.6|:50101... connected.
HTTP request sent, awaiting response... 401 Unauthorized
Authorization failed.
[[root@sa ~]#
[[root@sa ~]# wget --user identityfeed --password=netwitness 'http://logdecoder.ktalab.local:50101/event- processors/corpdomain?msg=getFile&force- content-type=application/octet-stream&expiry=600'
--2017-01-27 19:29:26--  http://logdecoder.ktalab.local:50101/event-%20processors/corpdomain?msg=getFile&force-%20content-type=application/octet-stream&expiry=600
Resolving logdecoder.ktalab.local... 10.0.0.6
Connecting to logdecoder.ktalab.local|10.0.0.6|:50101... connected.
HTTP request sent, awaiting response... 401 Unauthorized
Connecting to logdecoder.ktalab.local|10.0.0.6|:50101... connected.
HTTP request sent, awaiting response... 404 Not Found
2017-01-27 19:29:2 ERROR 404: Not Found.

[root@sa ~]#
```

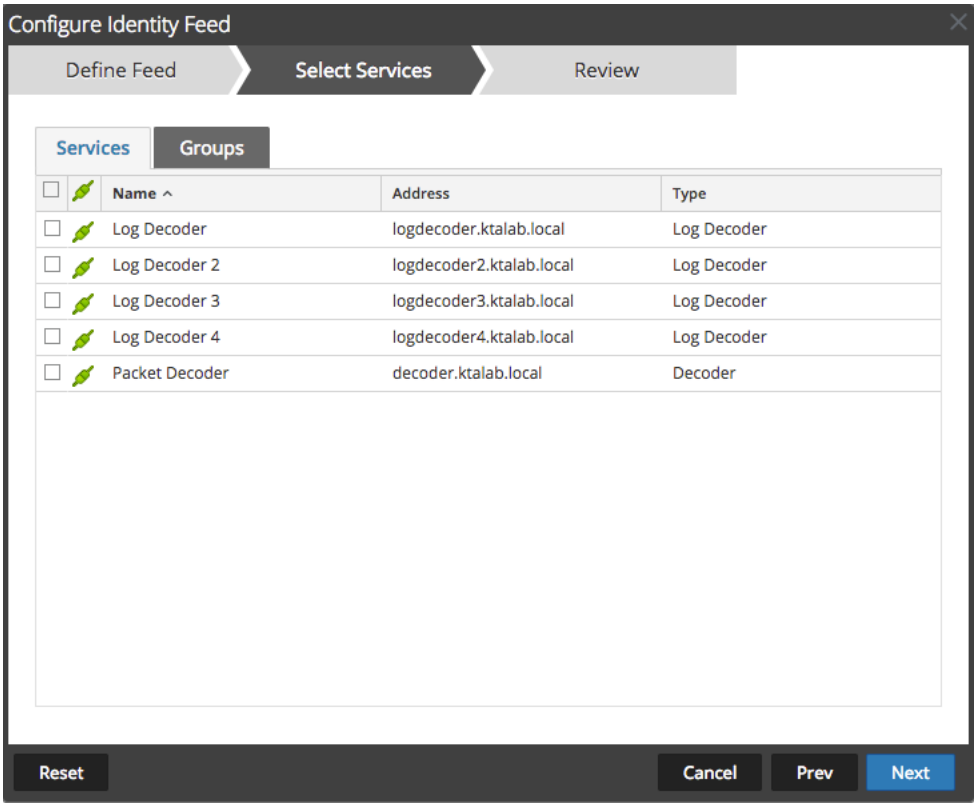
Wrong password and spaces in URL are causing Verify failure

8. Select an interval to fetch and push the feed at (we've chosen 10 minutes).

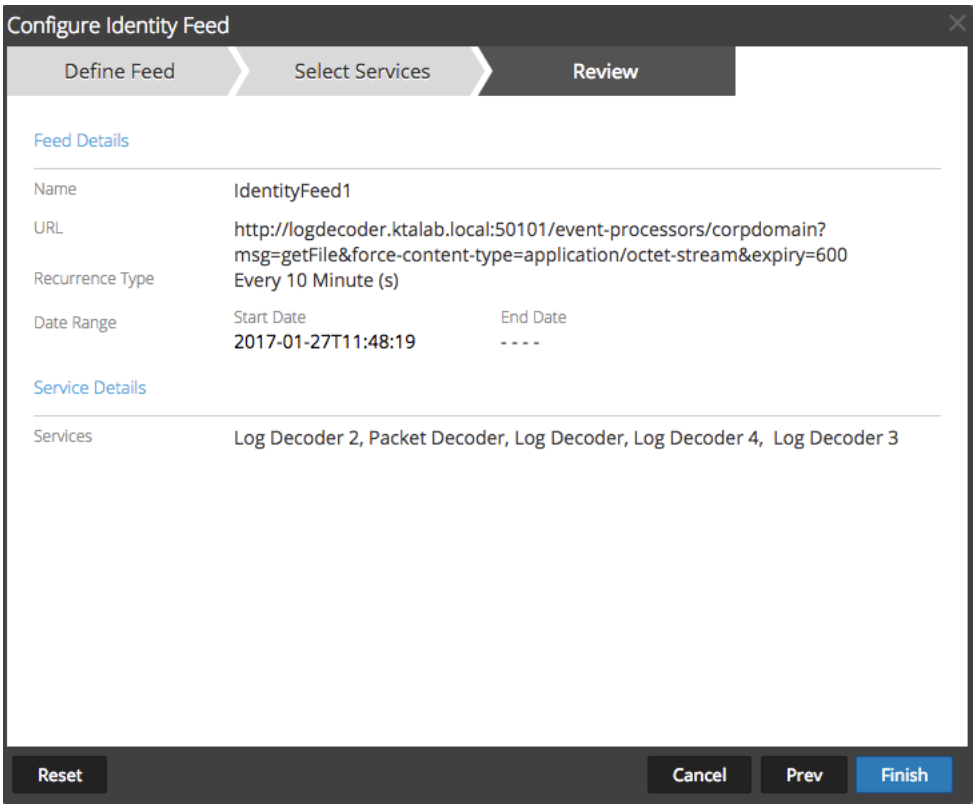


9. Click **Next.**

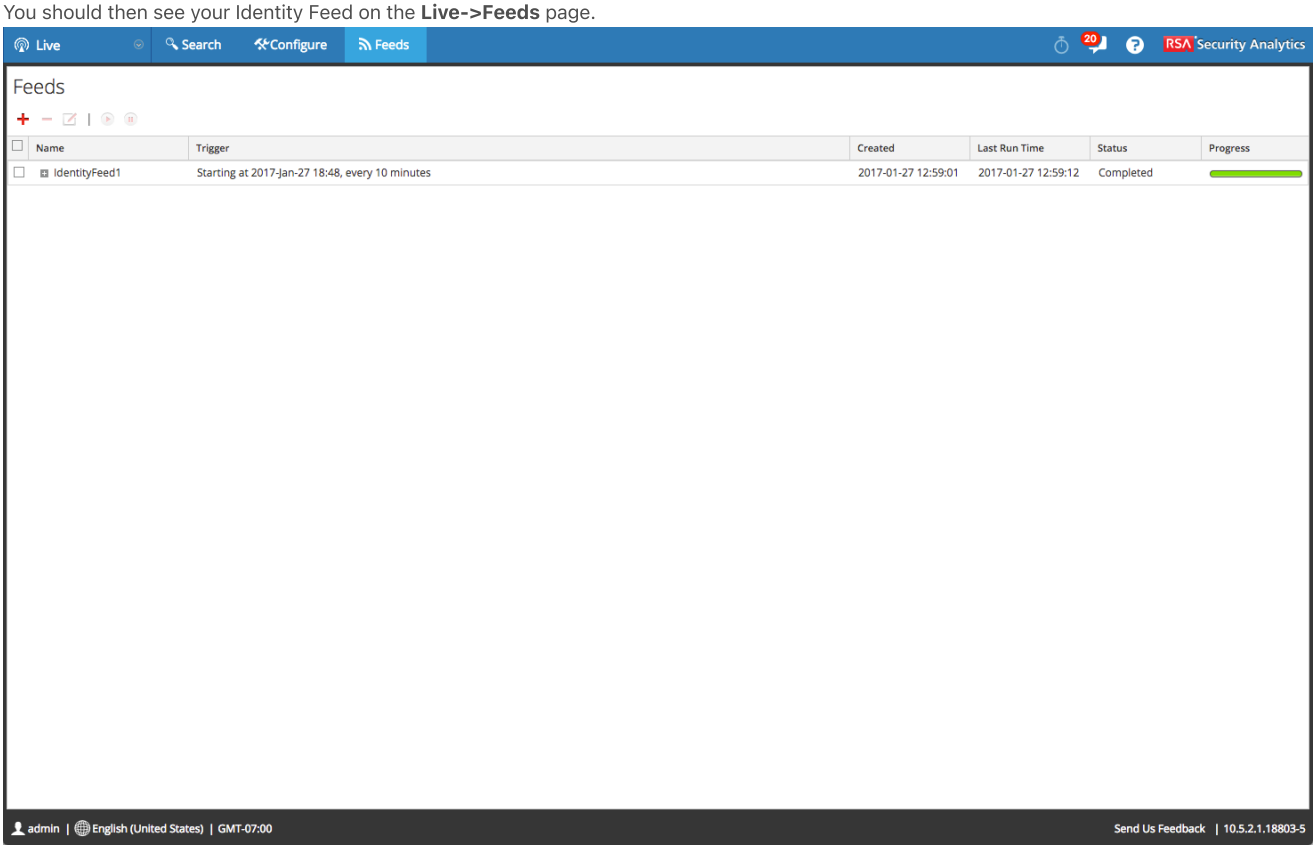
10. Choose the Decoders to push the feed to and click **Next**. All of them is usually a good choice.



11. Click **Finish**.



12. You should then see your Identity Feed on the **Live->Feeds** page.



Legacy Windows Collection

SA supports the collection of logs from legacy Windows sources through use of a Windows Log Collector. Legacy Windows is defined as Windows 2003 or older. I hope very much that you don't have any of these in your environment! The Windows Collector is a service which runs on a Windows server, which retrieves logs from these legacy sources and forwards them to a Local Collector. These are beyond the scope of this course, but I would be remiss not to mention them. For more information on installing and configuring Windows Log Collection, visit https://community.rsa.com/docs/DOC-63461.

Troubleshooting

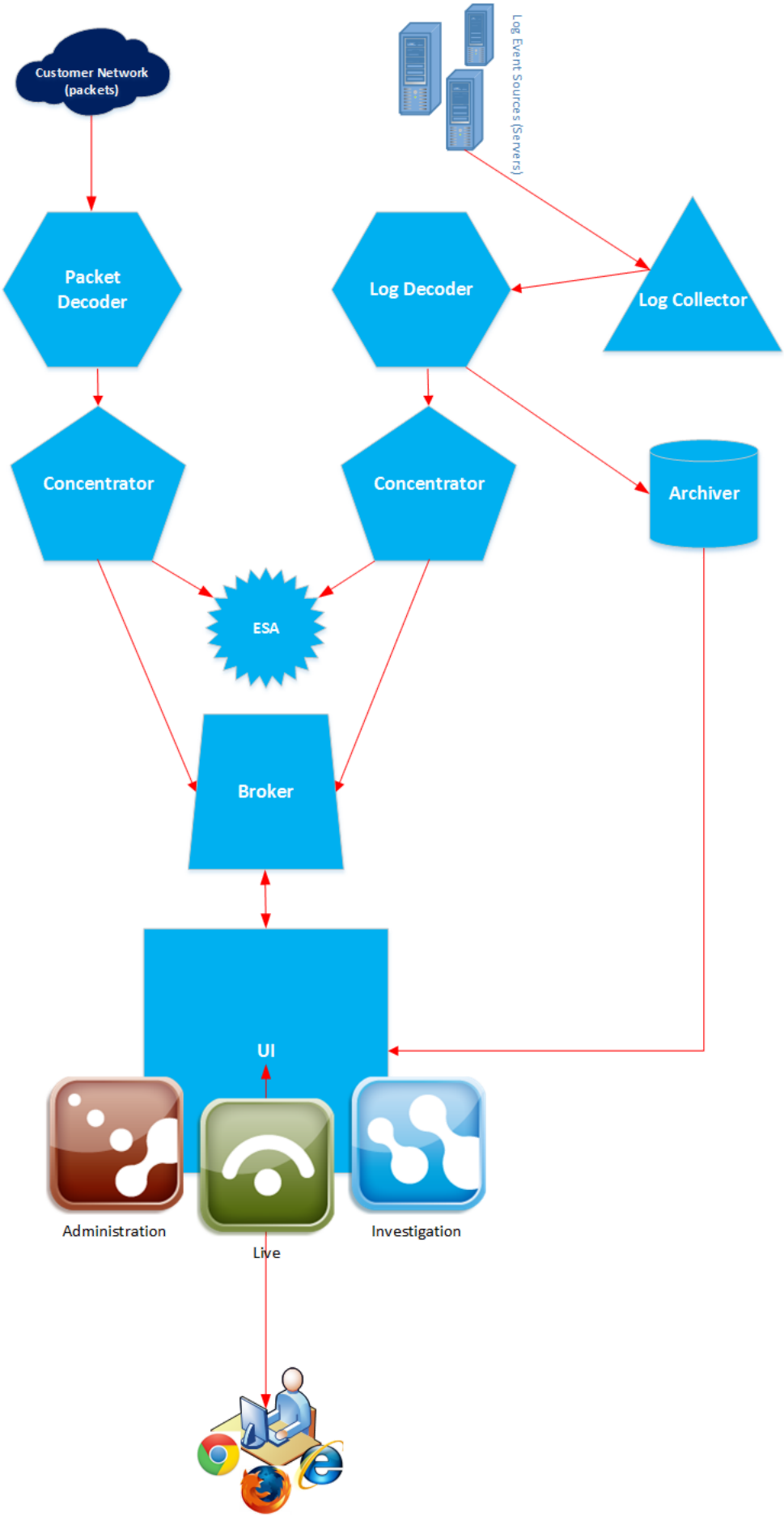
The best place to begin troubleshooting aLog Collector is the system log at **/var/log/messages**. Another good place is the RabbitMQ logfiles located in **/var/log/rabbitmq**. In most cases, if something goes wrong with a LC, messages in the logfile will shed more light on the problem. If you're not familiar with the tail, grep, and less commands, please get acquainted with them post haste!

17. Concentrator

Whilst the Decoder is the great author (or factory) of the SA suite, the Concentrator is its great librarian. The job of the Concentrator is to stream, or **aggregate**, meta from its upstream Decoder or Decoders, store this meta in its meta DB, and then index the meta so that it is easily query-able (searchable) and drill-able.

Rather than capturing data, the Concentrator aggregates data.

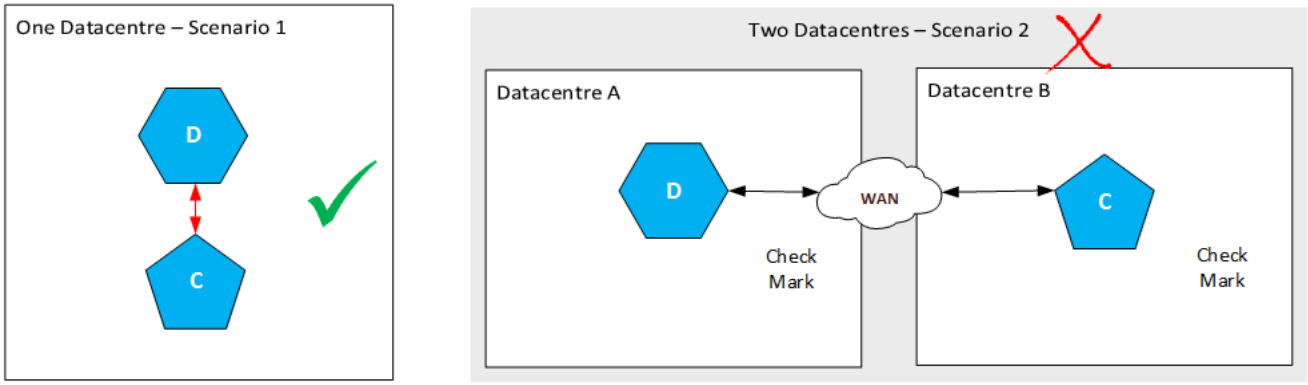
```
The Concentrator listens on port TCP 50005 / 56005 (SSL).
```



SA Stack

Placement

A Concentrator should never be placed in a different physical location than the Decoder(s) from which it aggregates. It should always be on the same physical LAN, and never aggregate over a WAN link.



One Good and One Bad Deployment Scenario

The Concentrator Index

Whereas the Decoder typically has no index to speak of, the Concentrator stores the physical indexes which are at the heart of SA's meta-based approach to security. The Decoder must only capture and parse data, the Concentrator must keep much of this data in memory. Next to ESA, the Concentrator is the SA suite's greatest memory hog, as it must store large chunks of its index in memory in order to keep it searchable.

Index Configuration File

The index configuration files are index-concentrator.xml and **index-concentrator-custom.xml** under **/etc/netwitness/ng**.

To Solid-State or Not to Solid-State?

The Concentrator generally also requires the fastest disks of any appliance in the SA suite. The Decoder's I/O is mostly sequential, reading and writing data in a nice serial order. The Concentrator's raison d'être is its index, and the nature therein is random I/O, as Analysts drill through and query meta unpredictably. For this reason, the Concentrator index really should reside on solid state drives (SSD's), which are spindle-less drives which use chips to store data rather than metal platters.

The Series 4 Exception

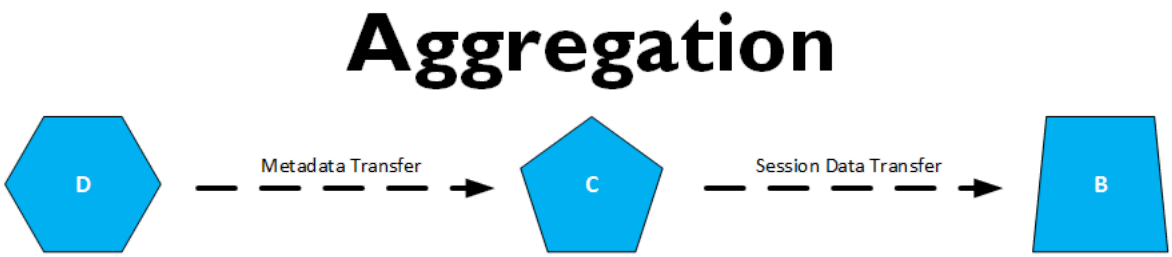
The exception to this is the Series 4 Hybrid, which most unfortunately did not possess any SSD's in its disk configuration, and consequentially was subject to dismal query performance. Most gratifyingly, RSA corrected this wee problem with its Series 5 Hybrid appliances.

The "Log Concentrator"

You may sometimes hear people using the term "Log Concentrator". This is a semantic and/or sales term used to indicate a Concentrator which is paired with a Log Decoder. There is no separate Log Concentrator service, or a technical difference between a Log or Packet Concentrator.

Aggregation

Aggregation is the process of streaming session data and meta data from an upstream SA Core service. Concentrators aggregate meta and session data from Packet Decoders and Log Decoders, and store them in their own local NWDB databases.



Configuring Aggregation

Before a Concentrator will aggregate meta and session data, the upstream service or services which it will aggregate from must first be configured by an Administrator. This can be done under **Administration -> Services -> <Concentrator> -> Config.**

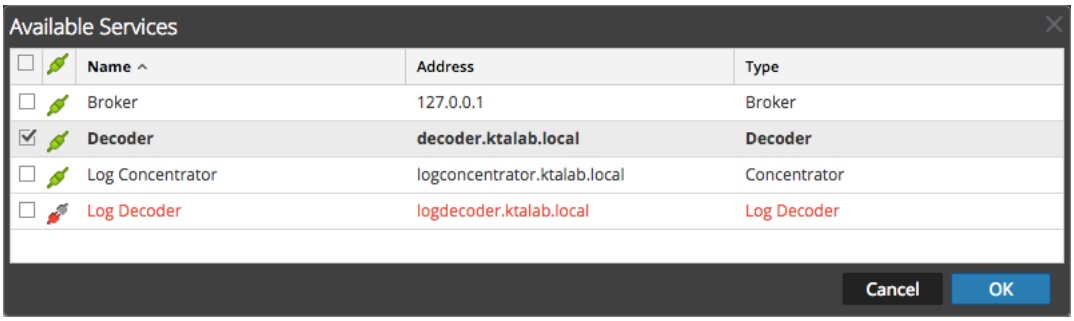
Note: Though it is possible for a Concentrator to aggregate from more than one upstream SA Core service, it is generally recommended to not do this except for low-bandwidth, low capture rate Decoders. This isn't to make more money, but rather is for performance reasons.

To add a an aggregation service:

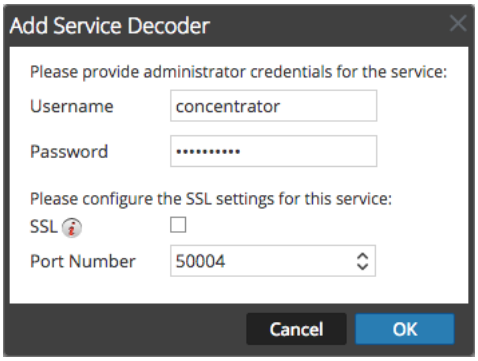
1. In the UI, navigate to **Administration -> Services -> <Concentrator> -> Config**

2. Under **Aggregate Services**, click the + button

3. Tick the box next to the upstream services to aggregate from and click **OK**.

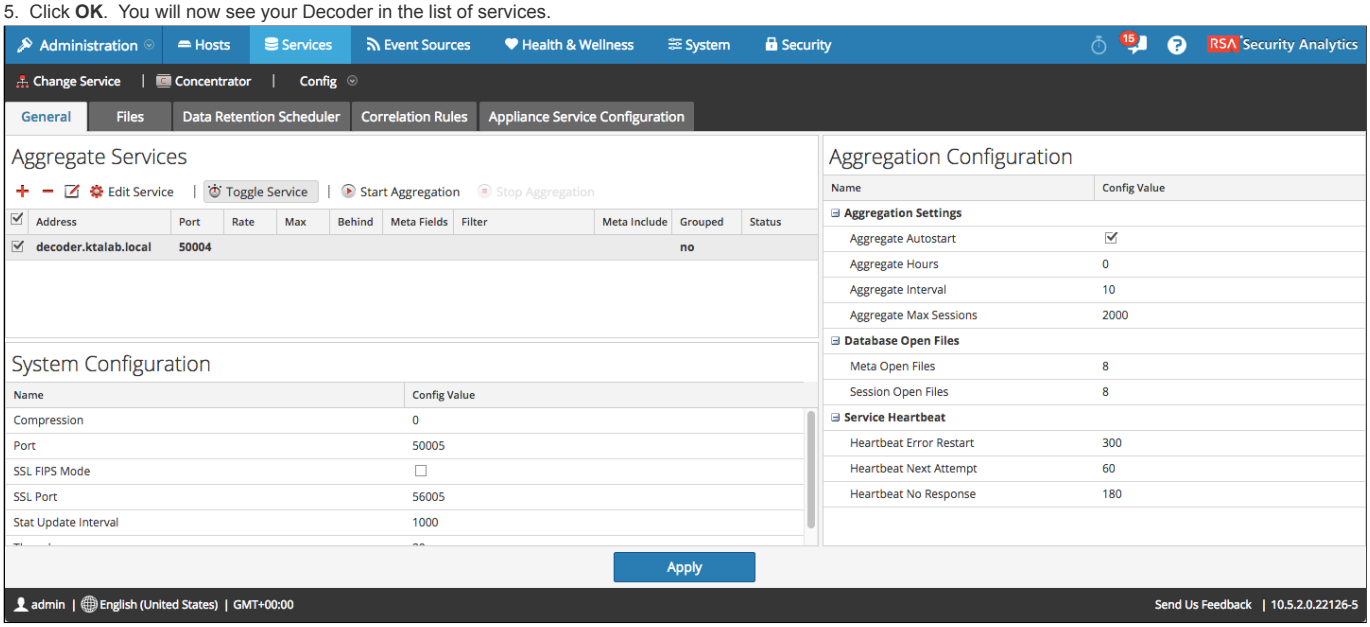


4. You will be asked for credentials that the Concentrator will use to connect to the upstream Decoder service or services. Remember to use an aggregation account, as opposed to the '**admin**' account.



Note: While technically the '**admin**' account can be used, and often is used erroneously in the field, one should use a dedicated aggregation account instead, with more limited permissions. This is for both security and auditing purposes. This must be defined locally on the Decoder service. I usually create an account called '**concentrator**' on each Decoder with the '**Aggregation**' role. Security will be covered later in the course.

5. Click **OK**. You will now see your Decoder in the list of services.



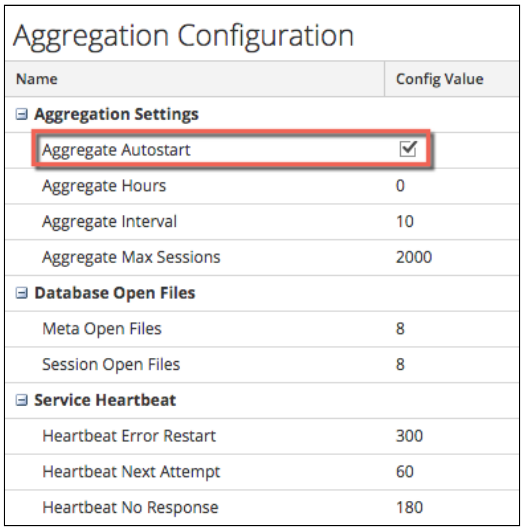
6. Click the Apply button.

Starting / Stopping Aggregation

From **Administration -> Services -> <Concentrator> -> Config,** click the Start Aggregation button. You should see the page start populating statistics from the upstream Decoder, and the status of the Decoder should change to **Consuming**.

Aggregate Autostart

Just like other Core services, a Concentrator doesn't by default automatically aggregate data from upstream services. **Aggregate Autostart** must be enabled by an Administrator. This can be done in the UI under **Administration -> Services -> <Concentrator> -> Config -> Aggregation Configuration.**



Toggling Aggregation for a Specific Service

For whatever reason, you may sometime wish to stop aggregation from a certain Decoder, but not all Decoders. This can done with the Toggle Service button on the **Administration -> Services -> <Concentrator> -> Config** page of the UI. You should see its status change in the **Status** column.

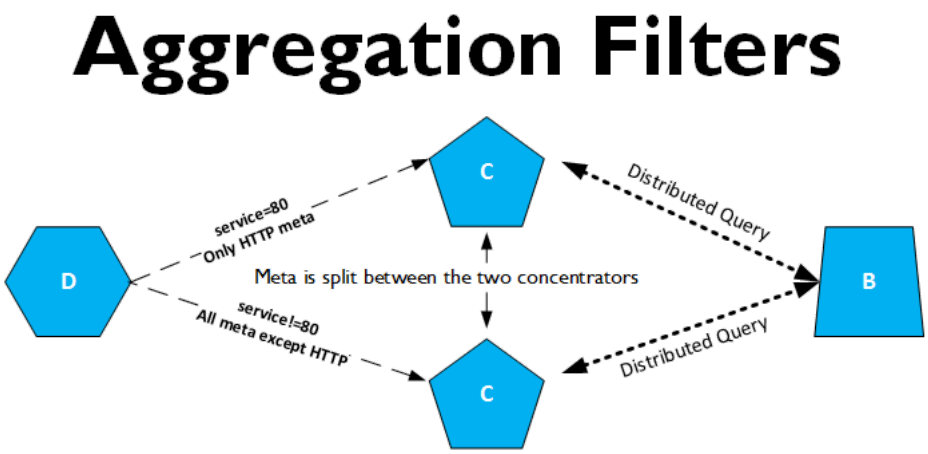
A Few Key Statistics

There are a few key statistics which you may want to keep an eye on when dealing with aggregation. They can be found at **Administration -> Services -> <Concentrator> -> Config -> Aggregation Configuration** in the UI.

- **Sessions Behind**: Unlike a Decoder, a Concentrator will never drop meta or sessions. The Concentrator tries to aggregate sessions and meta from a Decoder almost as soon as it is generated, but sometimes a Concentrator can lag behind its Decoders. A value of zero means the Concentrator is running in sync with its upstream Decoder. You generally want this number to be at or near zero. When starting aggregation for the first time, or after a prolonged aggregation outage, this number may be quite high, but you should see it decrease.
- **Meta Rate**: This value indicates the rate at which discrete meta values which are being aggregated. This is a measure of how much meta the upstream Decoder is generating.

Aggregation Filters

Rather than the default mode of aggregating all captured meta from a Decoder, it is possible to aggregate only certain meta keys (whitelist), or filter out certain meta keys (blacklist). There are cases where the meta rate of a given Decoder is so high that if a single Concentrator were to hold all of its meta, the query response times would be too slow. In this case, you can actually have two separate Concentrators aggregating from the same Decoder. One Concentrator may aggregate all meta except HTTP (service!=80), and the other Concentrator will only aggregate meta for HTTP sessions, with a filter of service=80.

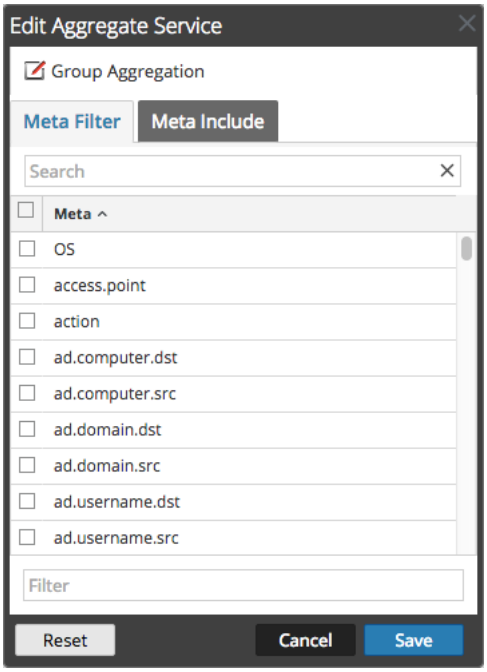


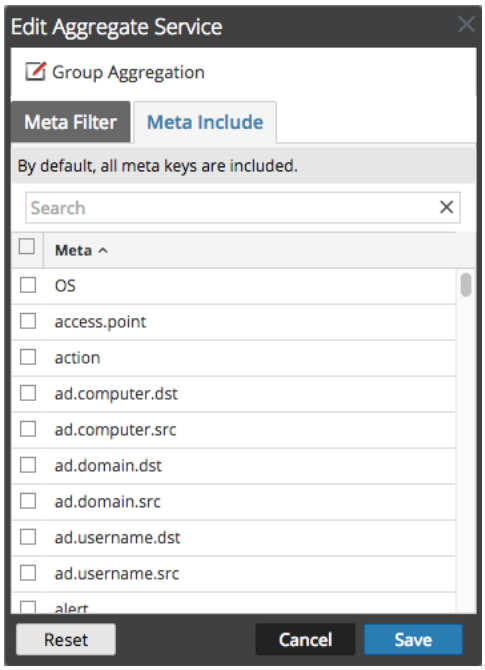
This can be configured by:

1. Select the tickbox next to the upstream Decoder that you wish to apply a filter to on the **Aggregate Services** portion of the **Administration -> Services -> <Concentrator> -> Config** screen.

2. Click the button.

3. Choose either the Meta Filter (blacklist) or Meta Include (whitelist) tab.



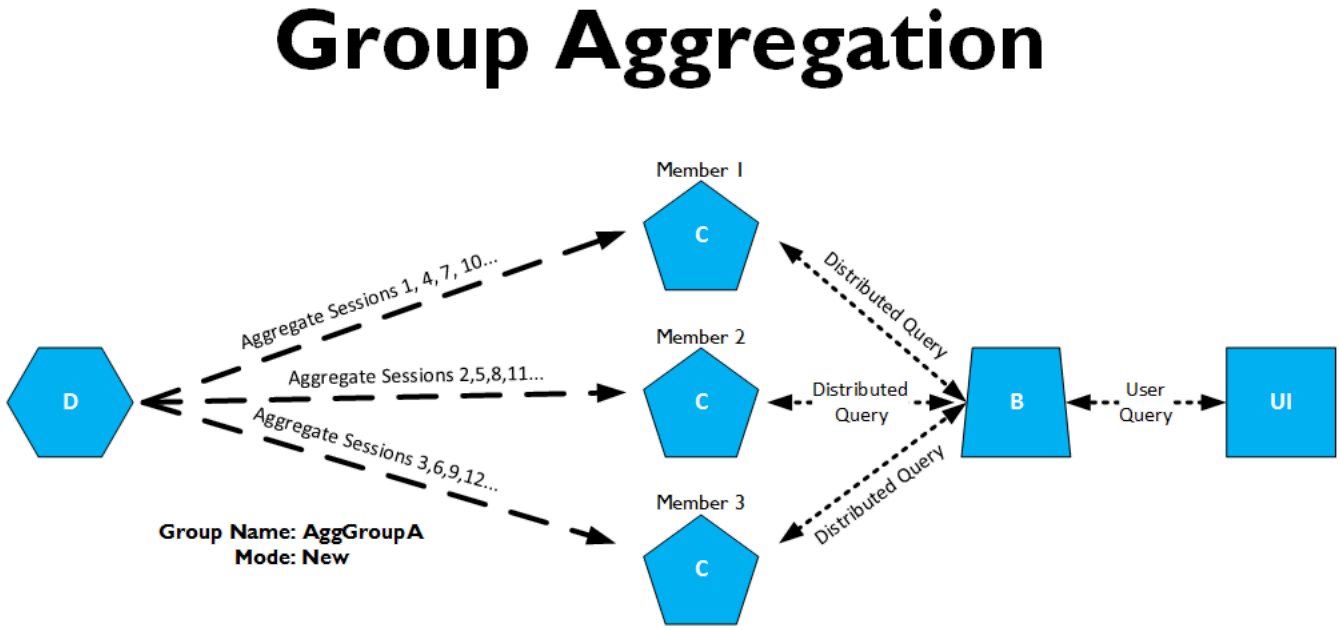


4. Depending on the tab you've chosen, choose either the meta keys you want to filter out (blacklist) or include (whitelist).

5. Click **Save**.

Group Aggregation

Group Aggregation or gang aggregation allows more than one Concentrator to aggregate from a single Decoder, and split the sessions evenly between them. Brokers can then aggregate from all Concentrator members of the gang.



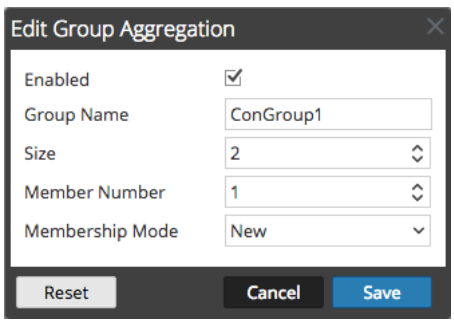
To create a Concentrator gang:

1. Select the tickbox next to the upstream Decoder that you wish to apply a filter to on the **Aggregate Services** portion of the **Administration -> Services -> <Concentrator> -> Config** screen.

2. Click the button.

3. Click the Group Aggregation button.

4. Here's where you define the Aggregation Group. You'll need to give it the following information:



- **Enabled** - tick this box
- **Group** Name - Pick a name for your aggregation gang. All members of the gang must also be configured with this name.
- **Size** - The number of nodes that the aggregation gang is comprised of. All members of the gang must be configured with the same value

- **Member Number** - This is the numeric gang node ID of the current Concentrator. This is unique to this Concentrator. Each member of the gang must have a unique, and sequential, node ID.
- **Membership Mode** - This can be either **New** or **Replace**. **Replace** (the default) means that this gang is brand new (all members are aggregating for the first time), and the node will aggregate sessions back to the beginning of time. New means that this node is being added to an existing gang, and will only start aggregating from the latest available session ID.

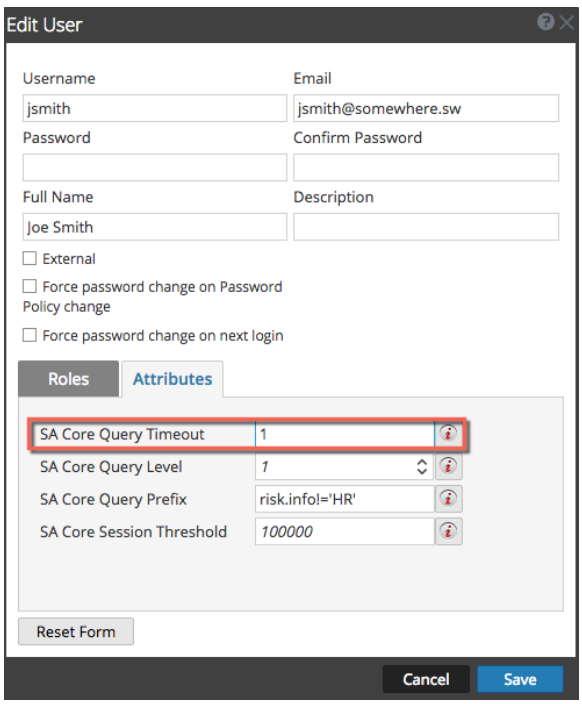
Also, you will need to ensure that the config node **/concentrator/config/aggregate.sessions.max** (in Explore view) matches between all members of the gang. This configures how many sessions each gang member shall aggregate at a time. For instance, if this value is 100 in a two-member gang, then node 1 will aggregate sessions 1-99, and node 2 will aggregate sessions 100-199.

Query Timeouts

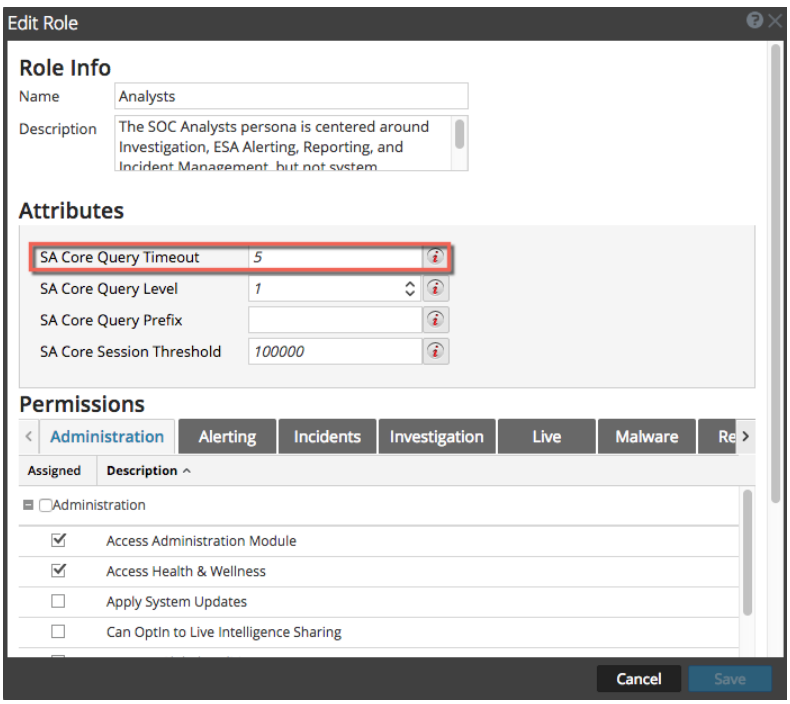
Large, poorly-formed queries over long timeframes can greatly hamper the performance of SA, so SA provides some safeguards to prevent these from killing a deployment. There are three settings which control how long queries are permitted to run. All query timeout settings are in minutes. One is configured in the UI in **Administration->Security->Users.** Another is in **Administration->Security->Roles,** and the last is configured via the legacy SA security model via local user accounts in SA Core services in **Administration->Services-> <*Service*> ->Security->Users**.

The hierarchy of these settings is thus:

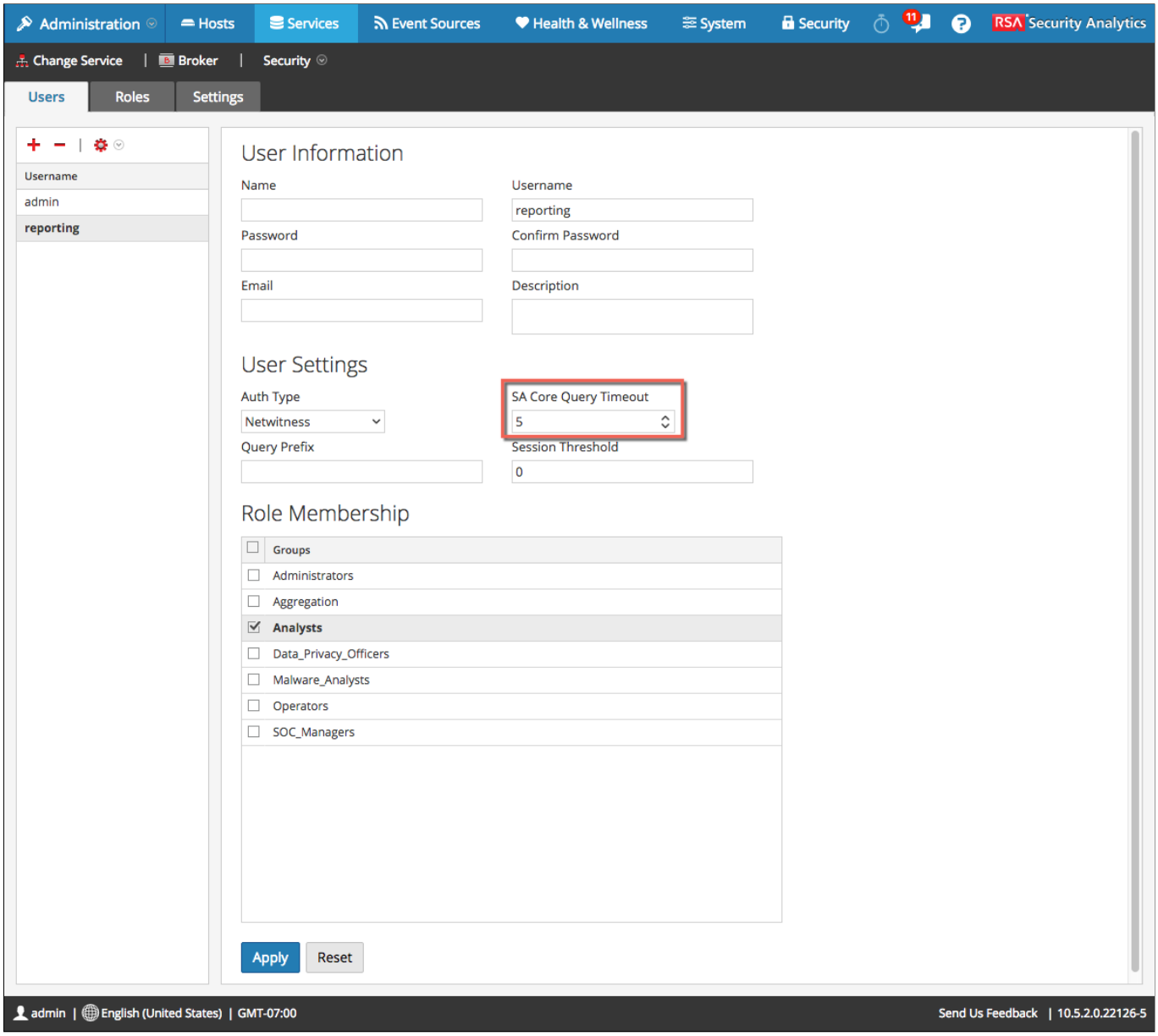
- Setting a query timeout for an SA system role applies to all members of that role.
- If a user is a belongs to multiple system roles, then the most restrictive value (lowest numerically) applies.
- A query timeout set for an individual system user will override query timeout settings for system roles.
- Query settings for local user accounts in SA Core services apply to legacy security model users, such as the Reporting Engine, the thick Investigator client, and queries via the REST API.



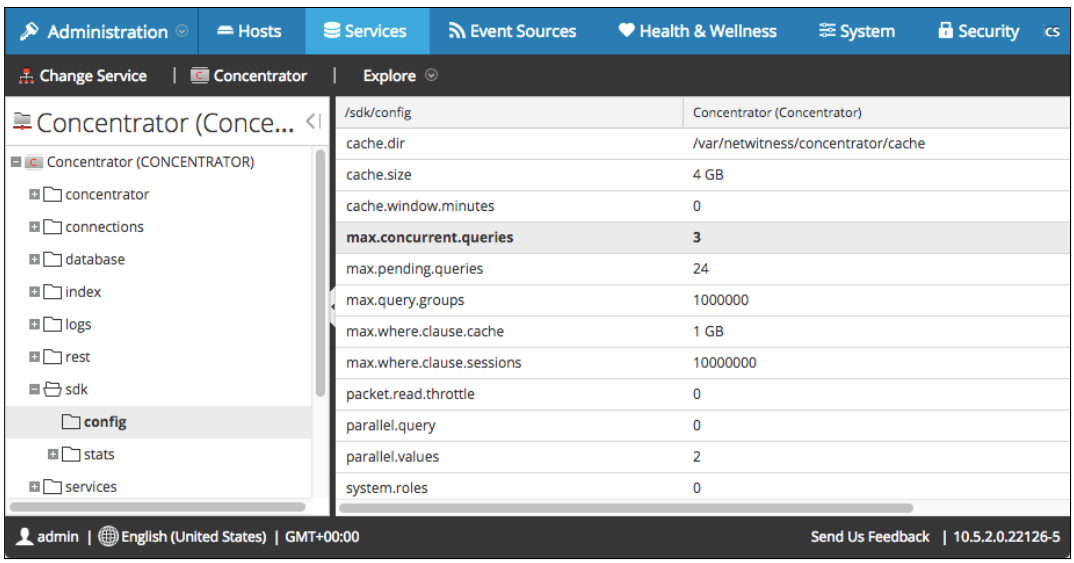
System User Query Timeout



System Role Query Timeout



SA Core Local User Query Timeout



max.concurrent.queries

One very important setting to understand is the explorer node **/sdk/config/max.concurrent.queries**. You will see this setting on most SA Core services, but it is most important on the Concentrator. **Max.concurrent.queries** controls how many queries are permitted to run simultaneously on the service's index. The default value is 3. This value can massively impact the performance of your Concentrator, so getting it right is very important. The

things to keep in mind are:

- the higher the value, more queries can run simultaneously.
- this means that with a higher value, more demand is potentially placed upon Concentrator memory and CPU resources, depending on how many queries typically run
- too low of a value means that it could be a while before your query is serviced, as new queries are serviced in the order in which they were submitted. If the queue isn't large enough, a few big jobs (reports or expensive queries) can make it seem like nothing is happening while using Investigation, as it spins whilst waiting for its queries to be serviced.
- too low of a value can also mean that a Concentrator's resources is under-utilised.

Getting **max.concurrent.queries** right means finding your Concentrator's performance "sweet spot", which can only be determined by watching your SA environment over time. and adjusting the value carefully.

Aggregate.hours

Most Concentrator-specific configuration nodes in the Concentrator API tree can be safely left alone, but one which occasionally may need to be modified is **/concentrator/config/aggregate.hours**. This setting only applies when aggregating from an upstream Decoder for the first time. A value of zero means aggregate all data from the Decoder, but a non-zero value tells the Concentrator the number of hours of meta to pull from the Decoder. This is used when there is so much data held on a Decoder that catching up would take far too long, so this gets the Concentrator back online more quickly.

You can find this in the explore view of the Concentrator, or in **Administration -> Services -> <Concentrator> -> Config** in the UI**.**

Aggregation Never Catches Up

In very rare circumstances, a Concentrator will not catch up with its Decoder, and its sessions behind never decreases or even increases. This generally happens when the host clocks are out of sync between the Concentrator and the Decoder. This really shouldn't these days as NTP configurations are now pushed out from the SA UI to all connected hosts, thus all SA hosts in a stack should synchronise to a common time source.

It could also happen if the meta rate is too high for the Concentrator to catch up, but I've seen this only one time in my career working with SA and NetWitness.

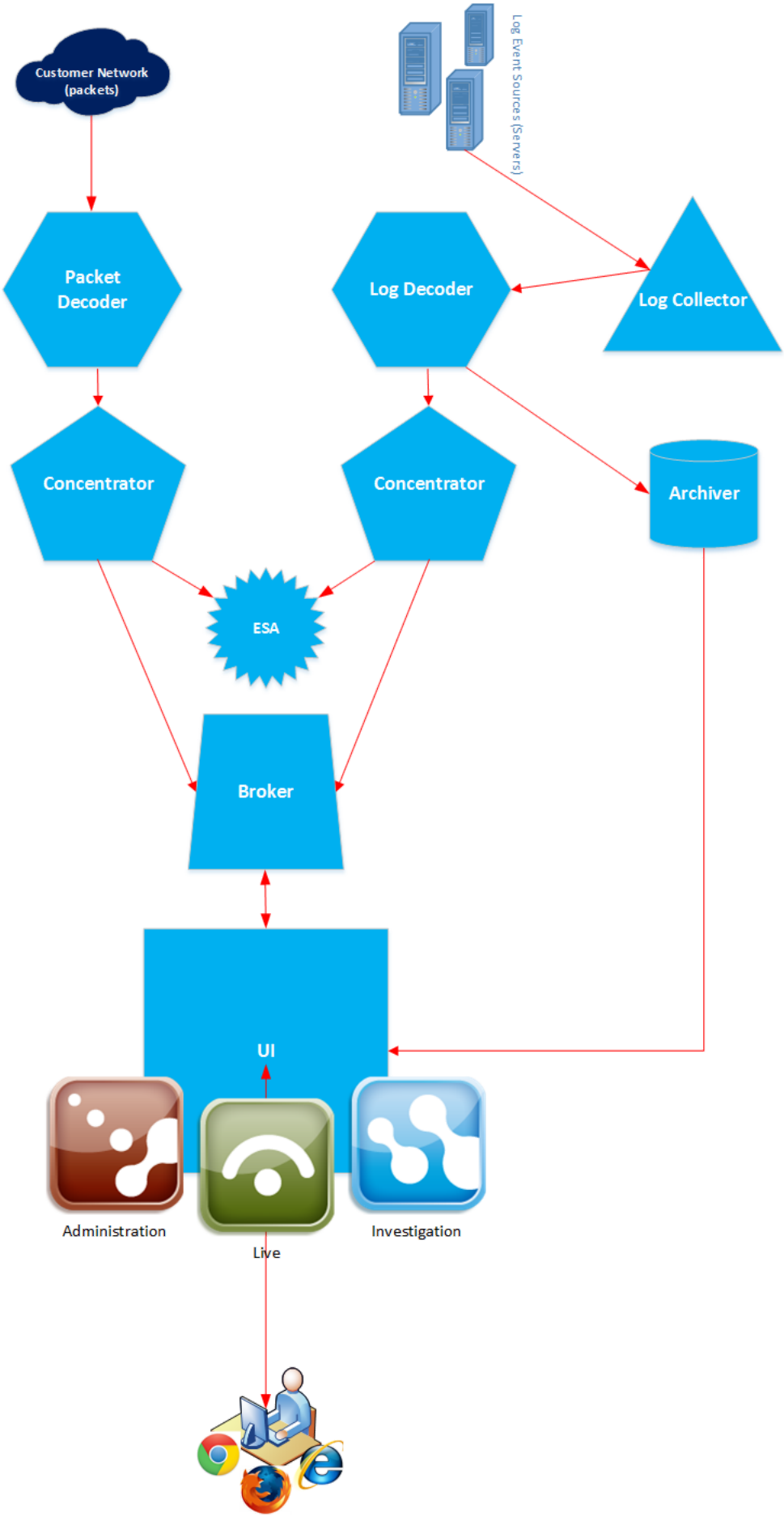
18. Broker

The SA Broker is the simplest service of the SA Core Stack. It is essentially a proxy which redirects queries issued to it to upstream services, which can be Concentrators and/or other Brokers. It allows an Investigator to focus an Investigation or Report on more than one SA Core service simultaneously.

```
The Broker listens on port TCP 50003 / 56003 (SSL).
```

From a reporting and investigation standpoint, interacting with a Broker is an identical experience to working with a Concentrator.

By employing more than one Broker, organisations can organise their SA deployment hierarchically and regionally. E.g. a multi-national company has 2 Decoders and 2 Concentrators in each of their respective New York, London, Delhi, and Sydney offices. By also placing a single Broker at each of these sites, one can focus an investigation on only the London site, or only the Delhi site. To take it a step farther, by placing another Broker in the mix which aggregates from the four regional Brokers, an investigation can then focus on the entire global SA deployment.



SA Stack

Hardware

Being very lightweight, Brokers can easily be run from nearly any type of hardware, provided enough RAM is present. They are ideal for running on VM's. You will find a Broker service is packaged with the SA UI server, which leads to a oft-used misnomer where the UI is called "the Broker". Please don't do this.

Placement

A Broker does not need to be co-located. It can be located nearly anywhere in the network, and has no issue with connecting to its upstream Concentrators via WAN link.

Broker Configuration

Despite dissimilar purposes, the Configuration of Brokers is very similar to that of Concentrators - nigh on identical, in fact.

Broker Aggregation

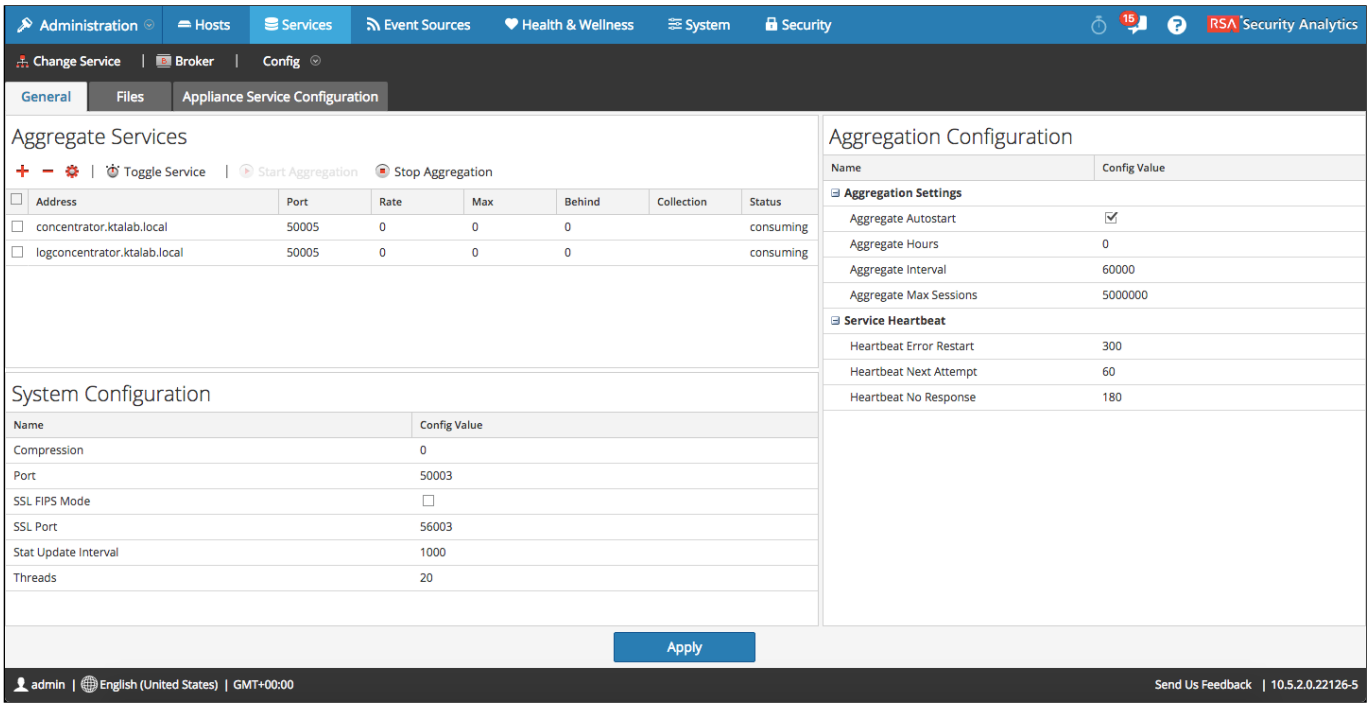
Brokers do not aggregate meta in the way that Concentrators do. The only things which a Broker consumes from its downstream services are Session ID's. A broker keeps track of which session ID's are available on all configured downstream services. Session ID's stored on each SA Core service is non-unique, meaning that a given session ID can exist on multiple services, all referring to different captured sessions. By keeping track of these session ID's, it is able to map a session ID from itself to a given remote session ID.

Example: A Broker is configured to aggregate from Concentrator A and Concentrator B. A user conducting an investigation against the Broker finds an interesting session with a session ID of 777, which maps to session ID 368 on Concentrator A. He then finds another interesting session on the Broker with a session ID of 778, but this maps to session 409 on Concentrator B. It is the Broker's job to keep track of this mapping so that session ID's are always consistent when conducting investigations and querying data.

<insert diagram here>

Configuring Broker Aggregation

The process of configuration of aggregation on a Broker is identical to that of a Concentrator, except that it's done through the **Administration -> Service -> *<Broker>* -> Config** page of the UI, and instead of aggregating from Decoders, one will aggregate from Concentrators and possibly other Brokers. This includes **aggregate.autostart** and the starting and stopping of aggregation.



The main difference you'll find between Concentrator and Broker aggregation is that Broker aggregation take a matter of seconds to aggregate from its upstream services. This is because Brokers don't consume metadata during aggregation. The Broker only consumes ranges of session ID's, so it knows what session ID's are present on each connected upstream service. That's it.

Aggregation Statistics

Because Brokers are only aggregating session ranges from its upstream sources, its session and meta rates will appear very low in both the Aggregate Services view and in Health & Wellness. This is norml.

Broker Databases, or Lack Thereof

Because a Broker doesn't consume anything but session ID's, it has no conventional NWDB databases, and requires minimal disk space. It does not have a conventional index, either. A quick look at **/var/netwitness/broker/index** will show some files, but these are quite small and not at all the same as the index files you would see on a Concentrator. They end with .nwmapdb, and store the session ID ranges which have been consumed by the Broker for each upstream Core service.

```
[[root@sa ~]# cd /var/netwitness/broker/index/
[[root@sa index]#
[[root@sa index]# ls -l
 total 296
 -rw-------. 1 root root  2776 Nov  4 17:19 concentrator.ktalab.local.50005-000000001.nwmapdb
 -rw-------. 1 root root   112 Nov  4 17:19 concentrator.ktalab.local.50005-000000002.nwmapdb
 -rw-------. 1 root root   112 Nov  4 17:19 concentrator.ktalab.local.50005-000000003.nwmapdb
 -rw-------. 1 root root 31216 Nov  4 17:19 concentrator.ktalab.local.50005-000000004.nwmapdb
 -rw-------. 1 root root 34384 Nov  4 17:19 concentrator.ktalab.local.50005-000000005.nwmapdb
 -rw-------. 1 root root 16456 Nov  4 17:19 concentrator.ktalab.local.50005-000000006.nwmapdb
 -rw-------. 1 root root 26392 Nov  4 17:19 concentrator.ktalab.local.50005-000000007.nwmapdb
 -rw-------. 1 root root 13576 Nov  4 17:19 concentrator.ktalab.local.50005-000000008.nwmapdb
 -rw-------. 1 root root 19192 Nov  4 17:19 concentrator.ktalab.local.50005-000000009.nwmapdb
 -rw-------. 1 root root 15592 Nov  4 17:19 concentrator.ktalab.local.50005-000000010.nwmapdb
 -rw-------. 1 root root 33232 Nov  4 17:19 concentrator.ktalab.local.50005-000000011.nwmapdb
 -rw-------. 1 root root  9904 Nov  4 19:41 concentrator.ktalab.local.50005-000000012.nwmapdb
 -rw-------. 1 root root    18 Nov  4 17:19 concentrator.ktalab.local.50005db.lock
 -rw-------. 1 root root    16 Nov  4 19:41 concentrator.ktalab.local.50005.nwvalid
 -rw-------. 1 root root   688 Nov  4 17:16 logconcentrator.ktalab.local.50005-000000001.nwmapdb
 -rw-------. 1 root root   112 Nov  4 17:16 logconcentrator.ktalab.local.50005-000000002.nwmapdb
 -rw-------. 1 root root  5080 Nov  4 17:16 logconcentrator.ktalab.local.50005-000000003.nwmapdb
 -rw-------. 1 root root  2992 Nov  4 17:16 logconcentrator.ktalab.local.50005-000000004.nwmapdb
 -rw-------. 1 root root  1912 Nov  4 19:29 logconcentrator.ktalab.local.50005-000000005.nwmapdb
 -rw-------. 1 root root    18 Nov  4 17:16 logconcentrator.ktalab.local.50005db.lock
 -rw-------. 1 root root    16 Nov  4 19:41 logconcentrator.ktalab.local.50005.nwvalid
 -rw-------. 1 root root    41 Nov  4 19:41 master.nwindex
 [root@sa index]#
```

A Broker's Index Directory

```
[[root@concentrator ~]# cd /var/netwitness/concentrator/index/
[[root@concentrator index]# ls
 managed-values-0  managed-values-10  managed-values-3  managed-values-5  managed-values-7  managed-values-9
 managed-values-1  managed-values-2   managed-values-4  managed-values-6  managed-values-8
[[root@concentrator index]#
[[root@concentrator index]# ls managed-values-0/
 access.point.nwindex     device.type.nwindex      ip.dstport.nwindex          risk.info.nwindex
 action.nwindex           did.nwindex              ip.proto.nwindex            risk.suspicious.nwindex
 ad.computer.dst.nwindex  directory.nwindex        ip.src.nwindex              risk.warning.nwindex
 ad.computer.src.nwindex  domain.dst.nwindex       ipv6.dst.nwindex            search.text.nwindex
 ad.domain.dst.nwindex    domain.src.nwindex       ipv6.proto.nwindex          server.nwindex
 ad.domain.src.nwindex    ec.activity.nwindex      ipv6.src.nwindex            service.nwindex
 ad.username.dst.nwindex  ec.outcome.nwindex       language.nwindex            sinterface.nwindex
 ad.username.src.nwindex  ec.subject.nwindex       lc.cid.nwindex              site.cat.nwindex
 alert.id.nwindex         ec.theme.nwindex         link.nwindex                sourcefile.nwindex
 alert.nwindex            email.dst.nwindex        logon.type.nwindex          sql.nwindex
 alias.host.nwindex       email.nwindex            managed-values-0.manifest   ssl.ca.nwindex
 alias.ip.nwindex         email.src.nwindex        match.nwindex               ssl.subject.nwindex
 alias.ipv6.nwindex       error.nwindex            medium.nwindex              subject.nwindex
 alias.mac.nwindex        eth.dst.nwindex          msg.id.nwindex              summary.db
 attachment.nwindex       eth.src.nwindex          msg.nwindex                 tcp.dstport.nwindex
 browser.nwindex          eth.type.nwindex         obj.name.nwindex            tcp.srcport.nwindex
 buddy.nwindex            event.cat.name.nwindex   obj.type.nwindex            threat.category.nwindex
 category.nwindex         event.class.nwindex      org.dst.nwindex             threat.desc.nwindex
 cid.nwindex              event.desc.nwindex       org.nwindex                 threat.source.nwindex
 city.dst.nwindex         event.source.nwindex     org.src.nwindex             time.nwindex
 city.src.nwindex         event.type.nwindex       orig_ip.nwindex             tld.nwindex
 client.nwindex           extension.nwindex        OS.nwindex                  udp.dstport.nwindex
 content.nwindex          feed.category.nwindex    paddr.nwindex               udp.srcport.nwindex
 country.dst.nwindex      feed.desc.nwindex        page.db                     user.dst.nwindex
 country.src.nwindex      feed.name.nwindex        parse.error.nwindex         username.nwindex
 crypto.nwindex           filename.nwindex         password.nwindex            user.src.nwindex
 database.nwindex         filetype.nwindex         policy.name.nwindex         version.nwindex
 device.class.nwindex     filter.nwindex           process.nwindex             virusname.nwindex
 device.group.nwindex     found.nwindex            query.nwindex               vis.level.nwindex
 device.host.nwindex      fullname.nwindex         reference.id.nwindex        wlan.channel.nwindex
 device.ip.nwindex        group.nwindex            referer.nwindex             wlan.ssid.nwindex
 device.ipv6.nwindex      ip.addr.nwindex          result.code.nwindex
 device.name.nwindex      ip.dst.nwindex           rid.nwindex
```

A Concentrator's Index Directory

The Broker Index

Though Brokers don't have a full index, they still have an index configuration files, just like a Concentrator. A Broker's index configuration files should match the index config of all of its downstream services. If a downstream key is missing from the Broker's index config, the data for that key will not be query-able through that Broker.

Index Configuration File

The index configuration files are [index-broker.xml]() and **index-broker-custom.xml** under **/etc/netwitness/ng**.

Top Tip: When configuring custom meta keys on a Broker, a mismatch in key format type (i.e. **Text**, **IPv4**, etc) and level (i.e. **IndexKeys** vs **IndexValues**) between said Broker and its upstream services will cause a query against that key to return no results. See the lesson on the Index.

19. RSA Live

Sometimes it's easier to have someone else write security content for you, and in this RSA is happy to oblige. **RSA Live** (formerly referred to as NetWitness Live), is a content distribution system service for Security Analytics. From here, one can download parsers, feeds, threat intelligence, reports, and rules, which can greatly enhance the analytical capabilities of your organisation.

Live is a subscription-based service which requires an account provided by either RSA or one of its channel partners.

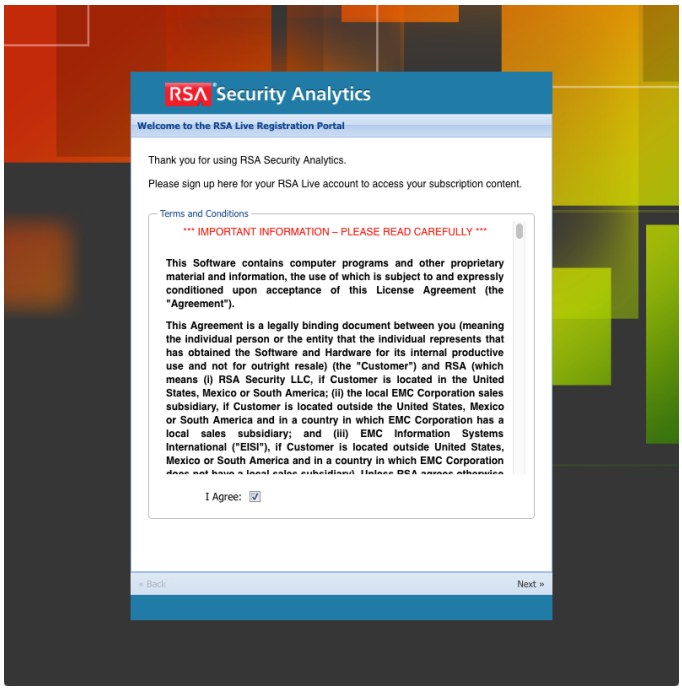
This lesson isn't so much about what specific content is available in Live, but it is more about the live workflow.

Registering for a Live Account

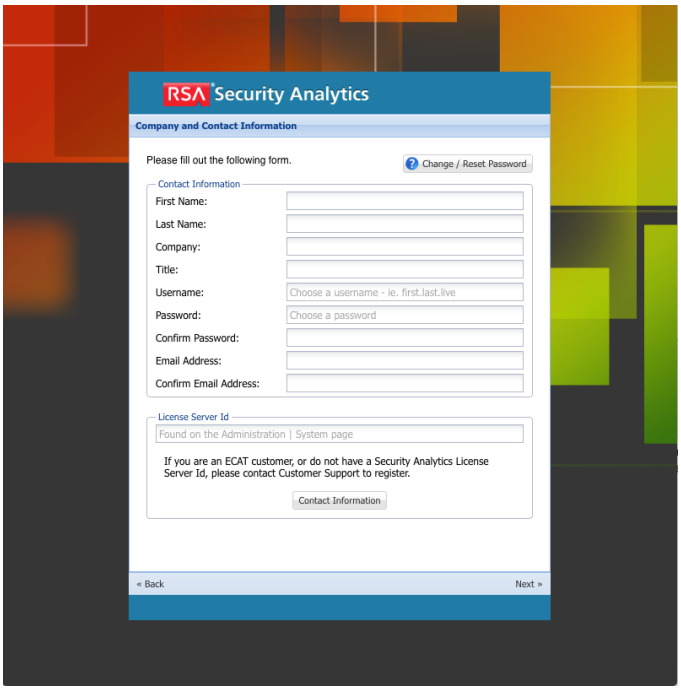
To register for a Live account, fill out the form at https://cms.netwitness.com/registration. I've had bad luck using Chrome with this site, but Firefox should work well.

You will need to have installed a full SA license to pass the second registration page, and you will also need to enter your SA License Server ID from **Administration->System->Info** in the UI. You will not be able to complete the registration if you have just installed SA for the first time and only have the demo license installed. Please see the lesson on Licensing in order to install an SA license.

Top Tip: Every Live ID is tied to a specific SA license server ID. If you're re-using an SA head system and you don't know the previous Live credentials, you will not be able to complete the Live registration. While RSA Support may be able to help with this, the easiest workaround is to change the license server ID. Should you ever need help in doing this, we should be glad to help you.



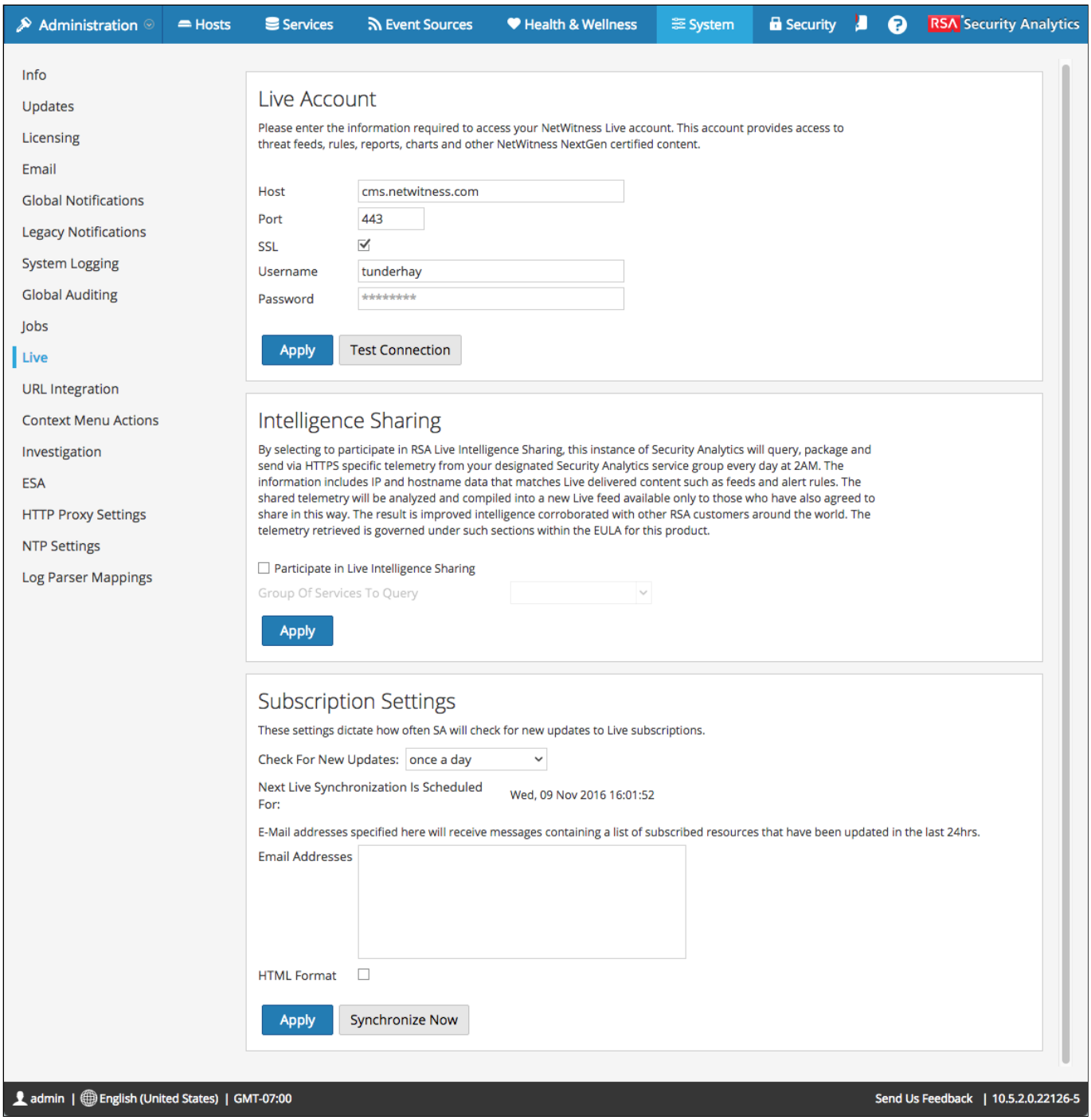
Live Terms & Conditions



Live Registration Form

Configuring Live

Before Live can be used, it must first be configured. To do this, navigate to **Administration->System->Live** in the UI.



Here you must enter your credentials to access Live content. The host is always **cms.netwitness.com**.

Live Intelligence Sharing

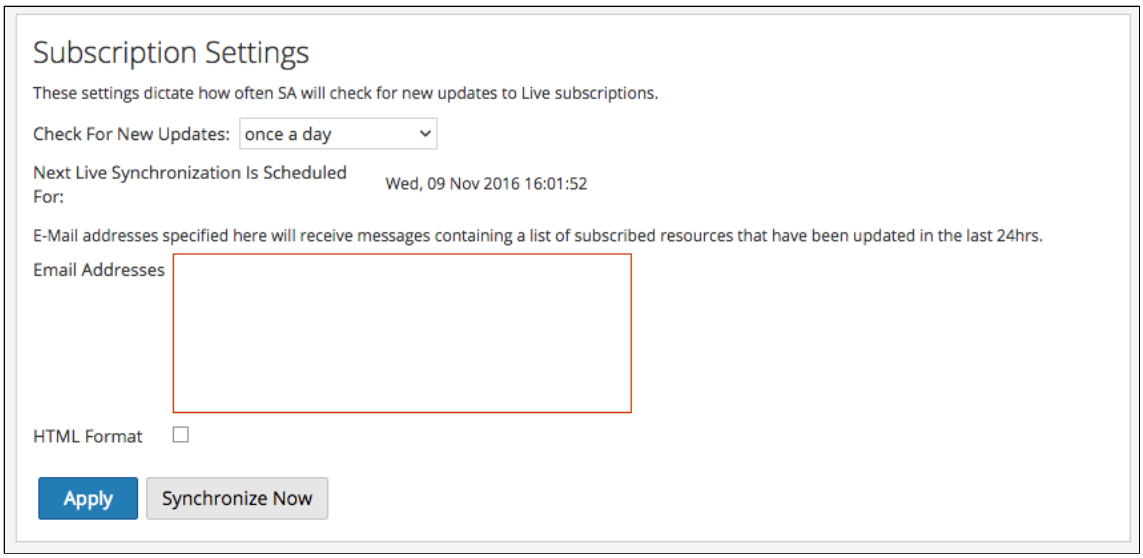
As you can see from the screenshot, Live has the capability of periodically sending IP and hostname telemetry which matches content delivered by Live back to RSA. This is then used to improve RSA's live content, which is then distributed in the form of a Live feed. This feed is only available to customers who've agreed to participate in the program. You can see the EULA (End User License Agreement) for SA for specifics on the telemetry.

Note that this is **turned on** by default, and if you do not want to participate you must manually switch it off.

If you do use Intelligence Sharing, in the dropdown, you must designate a Service Group which will be queried by SA for the telemetry data.

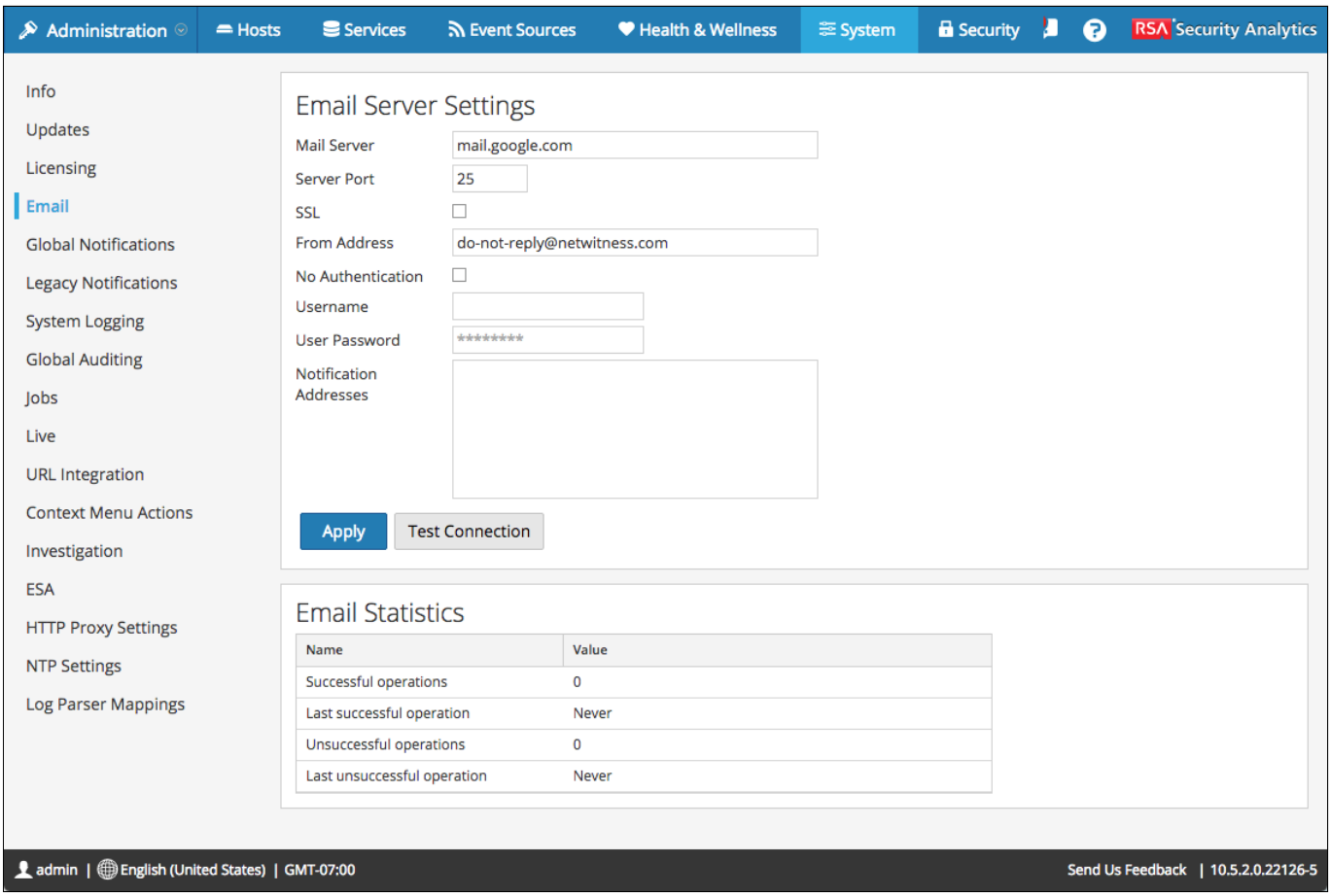
Subscription Settings

Here you can specify the interval at which Live will check for new content updates for subscribed content. If the subscribed content has been added to a Live Deployment group, then any newly-updated content will be pushed out automatically to the Deployment Group. Take notice that **this overwrites existing Live content**.



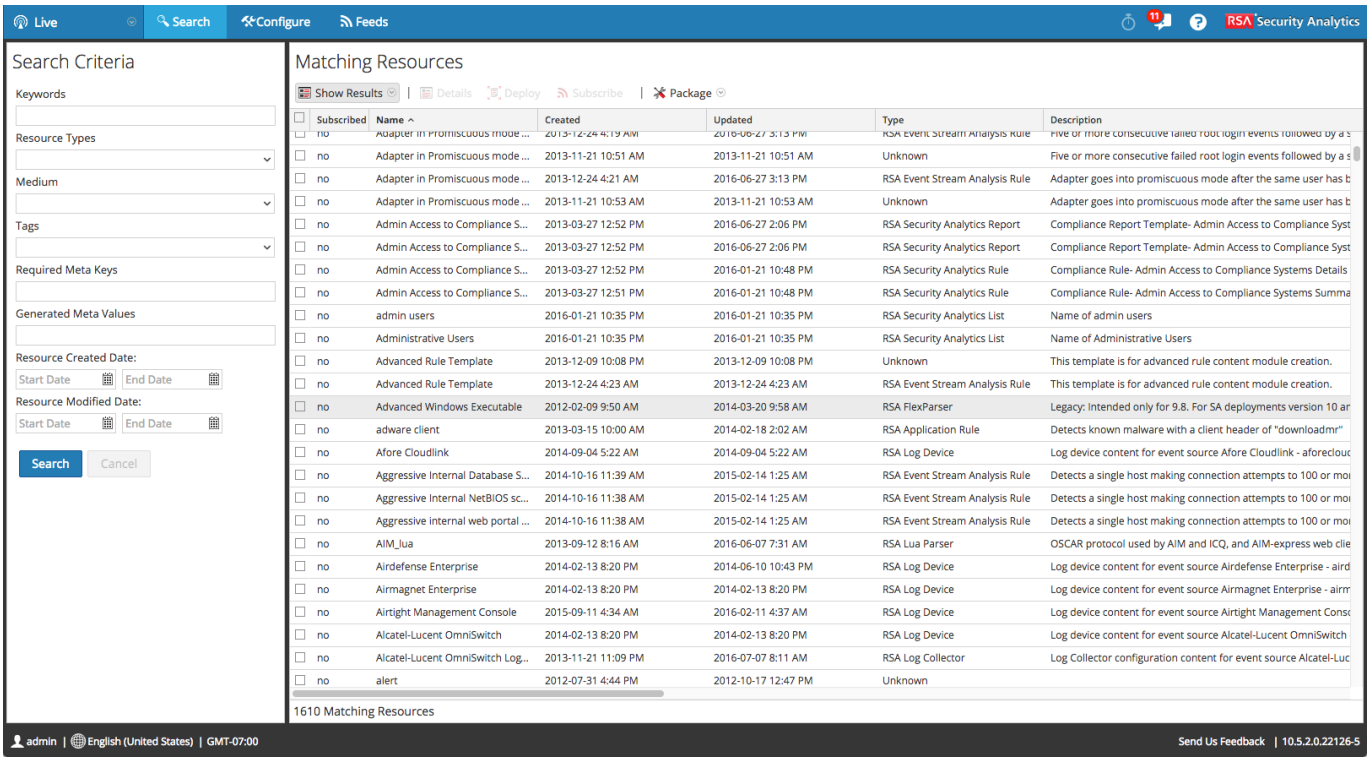
If you add email addresses to the Email Addresses list, they will receive an email from Live every time Live content is updated. In order to use this, you must configure your mail server settings in **Administration->System->Email** within the UI.

Top Tip: One should never modify Live content directly after it has been deployed to an SA service. Instead, make a duplicate of the Live content and modify the duplicate. Be sure to disable the original Live content if you do this, or you could end up with duplicate values in your meta.



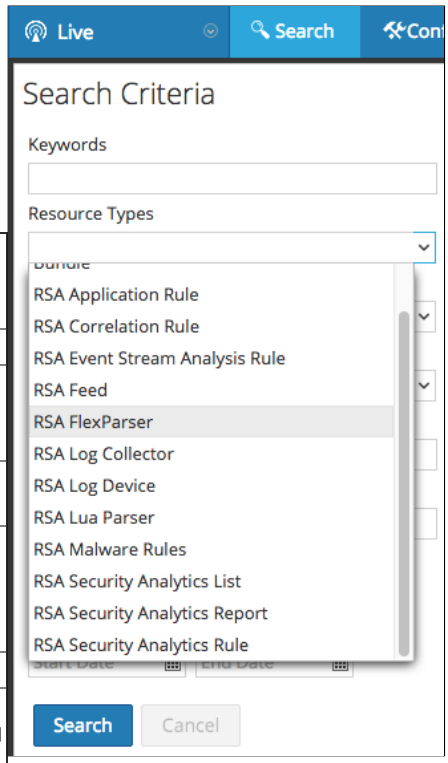
Accessing Live

To access Live within SA, navigate to **Live->Search** from the Start Menu.



Live Content Types

At the time of writing, these are the following content types available from Live. You can filter on these from the "Resource Types" dropdown of **Live->Search**.



Advanced Analytics (Warehouse)	These are Reporting Engine rules for the RSA Warehouse, which is a Hadoop-based product.
Bundle	Not used, as of the time of writing.
Application Rule	Application rules which can be used in Decoders and/or Log Decoders. When deploying, pay close attention to the content name. Content for Log Decoders starts with 'NWFL_'. You don't want to deploy Log content on a Packet Decoder.
Correlation Rule	These are simple correlation rules which can run on both Decoders or Concentrators.
Event Stream Analysis Rule	These are Esper (EPL) rules for ESA. Pay close attention to each description when deploying, as you may need to make changes to some rules after deploying it. ESA rules from Live always deploy in trial mode.
Feed	These are Decoder feeds which can be used in either Packet or Log Decoders.
Flex Parser	These are legacy packet parsers based on the XML-based FLEX language. These were mainly used in NetWitness 9.8. Some still may be useful, but most have been superseded by LUA-based equivalents.
Log Device	These are log parsers for Log Decoders
Lua Parser	These are LUA language parsers for Packet Decoders.
Malware Rules	YARA rules for Malware analytics
Security Analytics List	Reporting Engine lists which can be used in RE rules
Report	RE reports
Rule	RE rules for NWDB (SA Core) data sources

Why not take a few minutes now to browse through some content that's available on Live?

Getting Information on Live Content

It's possible to get more information on specific Live content by double-clicking on it from **Live->Search**. This will give you information on dependencies, other content which might conflict, which meta keys the content will populate, and in some cases it will also give the meta values which the content generates.

From the Info view, you can also download an individual .zip file of the content by using the **Download** button.

Top Tip: Properly installing Live Content is often not as simple as merely deploying it. It is important to pay attention to conflicts, and also to add meta keys which the content may need to your index configuration files. SA does not presently have a mechanism for dealing with these things automatically.

Live | Search | Configure | Feeds | 11 | RSA Security Analytics

Download | Subscribe | Deploy | Service Locator

windows executable

type	RSA Lua Parser
created	2013-10-18 7:53 AM
updated	2015-12-15 12:53 PM
description	Identifies windows executables, and analyzes them for anomolies and other suspicious characteristics. There may be non-nefarious reasons for many of these alerts. Alerts indicate that the executable in question may warrant further scrutiny. Alerts should not be interpreted as a definitive determination that an executable is malicious. DEPENDENCIES Parsers * FeedParser Feeds * alertids_info * alertids_suspicious * alertids_warning CONFLICTS Parsers * advanced_windows_executable * CMS_windows_executable * encoded_file_fingerprinting KEYS * alert.id - mapped to risk meta * filetype - 'windows executable', 'x86 pe', 'windows dll', 'x64pe' RISK VALUES info

admin | English (United States) | GMT-07:00 Send Us Feedback | 10.5.2.0.22126-5

Live | Search | Configure | Feeds | 11 | RSA Security Analytics

Download | Subscribe | Deploy | Service Locator

RISK VALUES

info
* exe dos loader missing
* exe abnormal e_cblp
* exe abnormal e_cp
* exe abnormal e_crlc
* exe abnormal e_cparhdr
* exe abnormal e_minalloc
* exe abnormal e_maxalloc
* exe abnormal e_ss
* exe abnormal e_sp
* exe abnormal e_csum
* exe abnormal e_ip
* exe abnormal e_cs
* exe abnormal e_ovno
* exe abnormal e_res1_1
* exe abnormal e_res1_2
* exe abnormal e_res1_3
* exe abnormal e_res1_4
* exe abnormal e_oemid
* exe abnormal e_oeminfo
* exe abnormal e_res2_1
* exe abnormal e_res2_2
* exe abnormal e_res2_3
* exe abnormal e_res2_4
* exe one dos header anomaly
* exe two dos header anomalies
* exe two sections
* exe timestamp zero
* exe abnormal number of symbols

suspicious
* exe abnormal loader flags
* exe dos loader missing
* exe pe offset extremely high
* exe three dos header anomalies
* exe four dos header anomalies
* exe timestamp close to zero
* exe timestamp before 1999

admin | English (United States) | GMT-07:00 Send Us Feedback | 10.5.2.0.22126-5

Installing Content From Live

In a certain sense, the object of Live is to install its content in your SA installation. To do this, there are two options:

1. Deploy the content manually. You accomplish this directly from the **Live->Search** page.

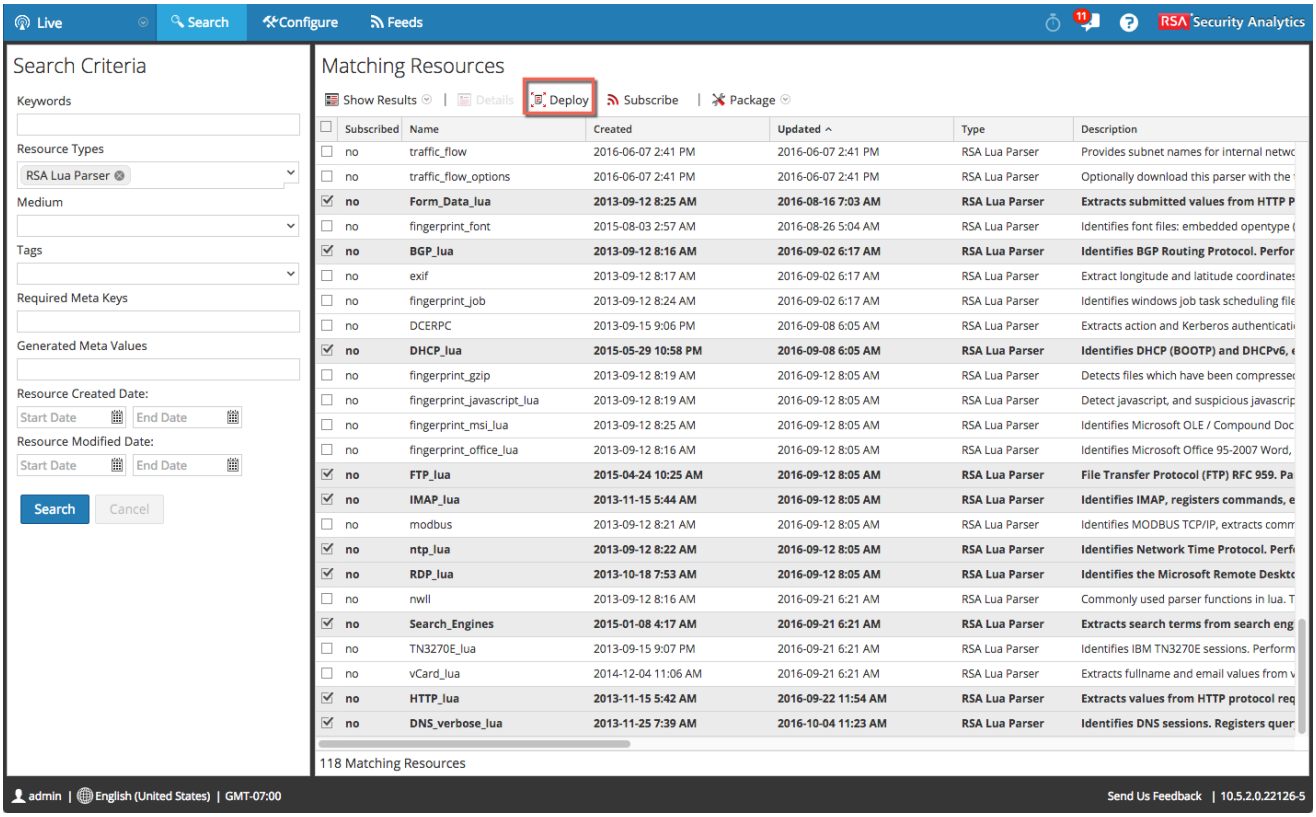
2. Use a Live **Deployment Group**. Deployment Groups are an extension of the Service Groups concept which allow you to group SA services together. Service Groups work on a subscription model, whereby Live automatically pushes subscribed content out to all members of the Deployment Group.

Deploying Content Manually

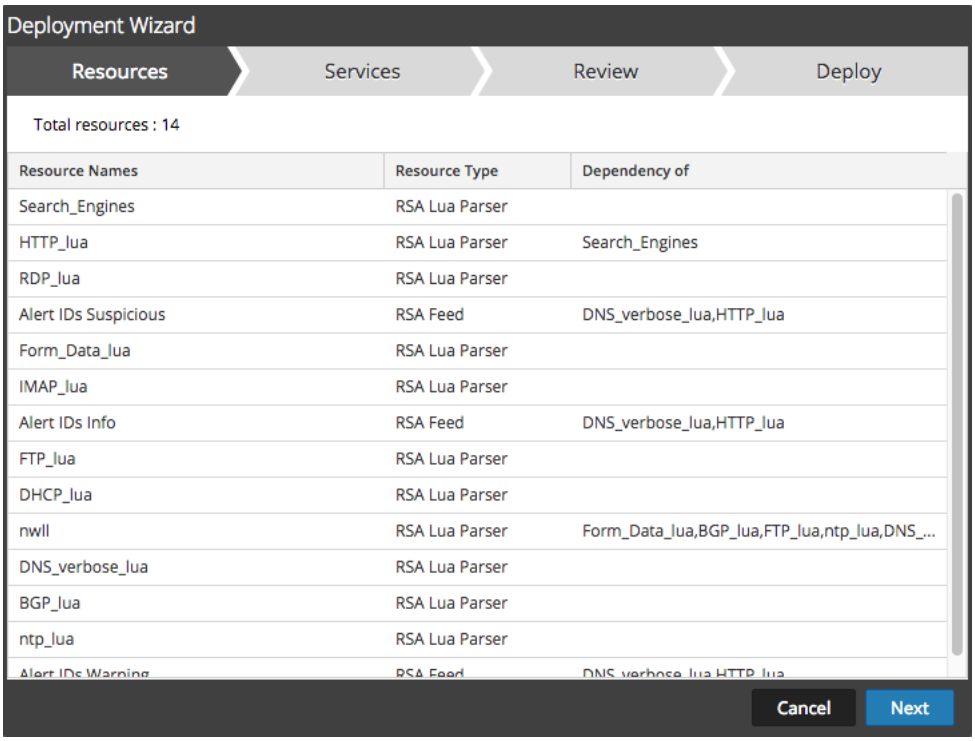
Deploying Live content manually is very straightforward, and is the most common method of initially deploying Live content. To do this:

1. From **Live->Search**, select the content types you want to deploy from the Resource Types dropdown and click **Search**. You can select more than one type in this dropdown. This should narrow down the list quite a bit.

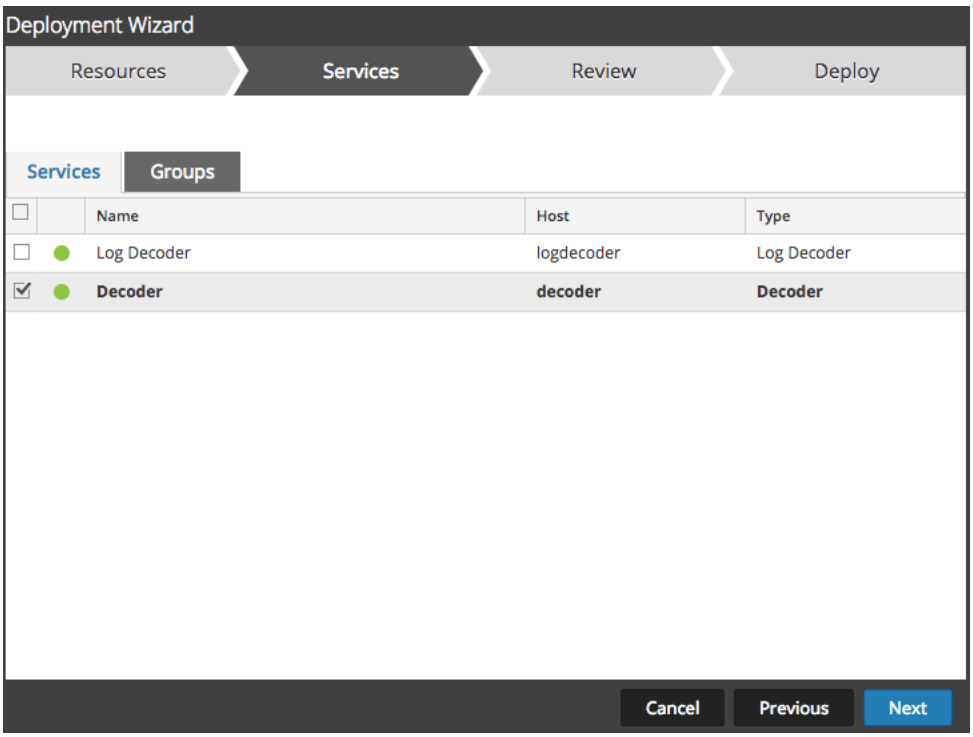
2. From the search results, use the tick boxes to select the content which you want to deploy. Click the **Deploy** button.



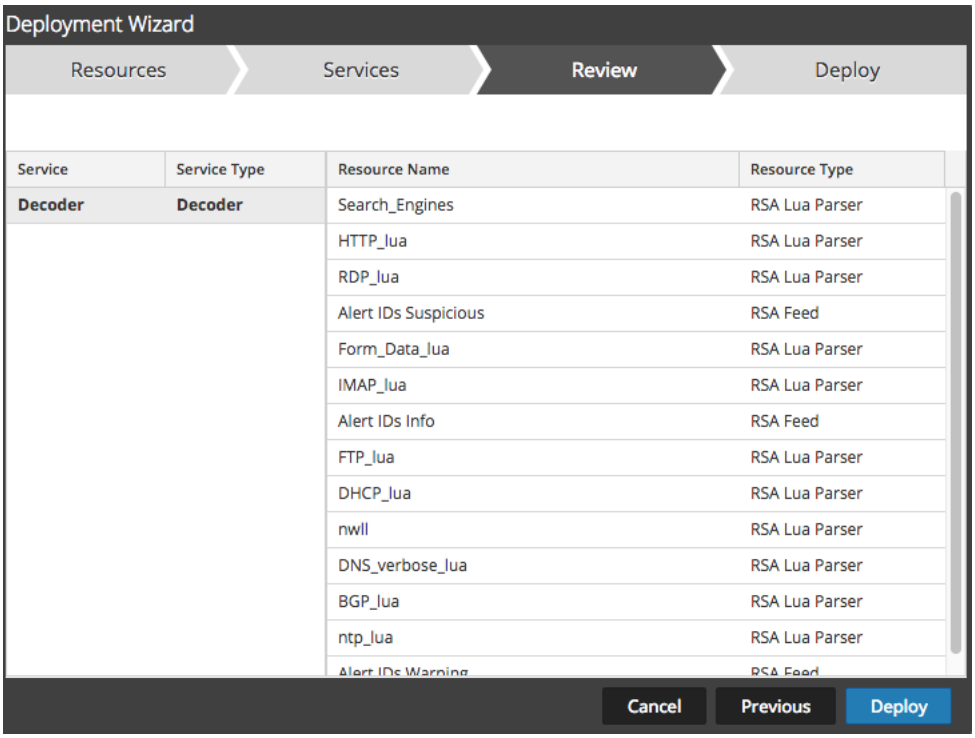
3. The **Deployment Wizard** now appears, displaying the selected content in the **Resources** tab. Click Next.



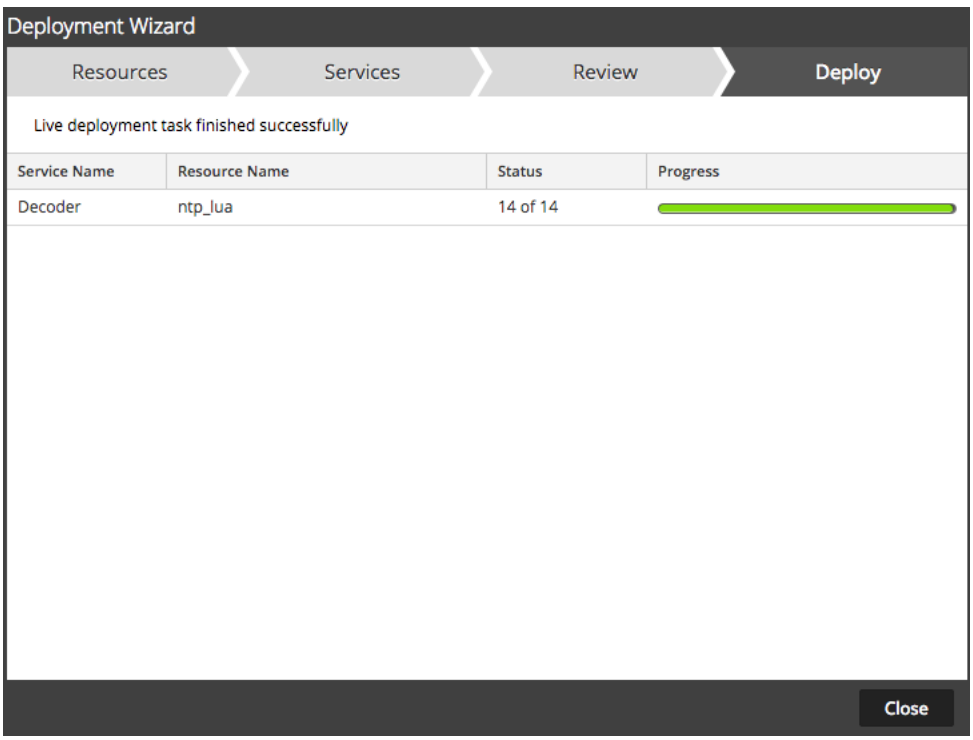
4. Select the SA Core services you wish to deploy the content to from the **Services** tab. Click Next.



5. Review your services and selected resources from the **Review** tab, and click **Deploy**.



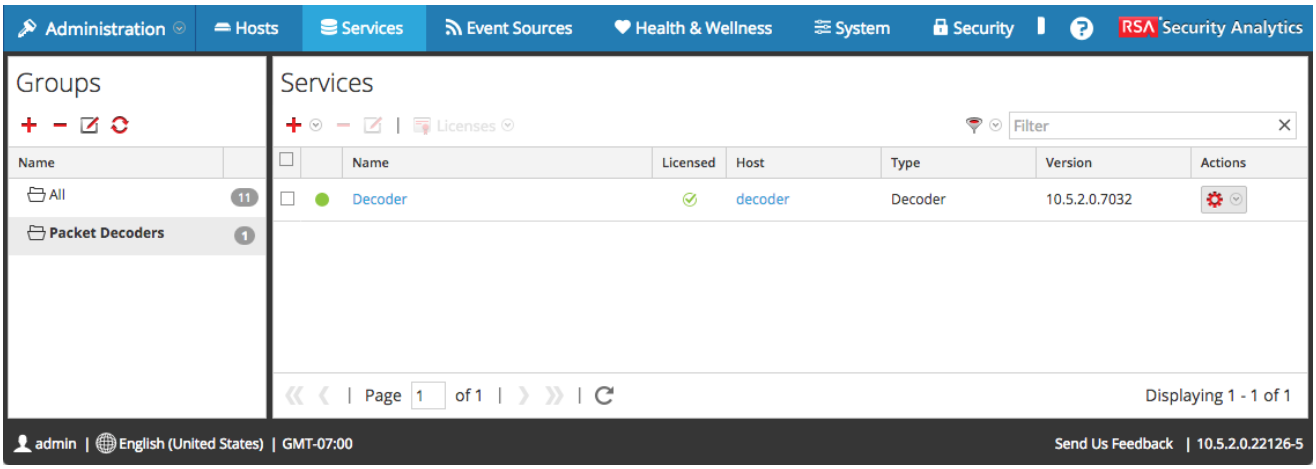
6. From the Deploy tab, you will see a progress bar indicating the status of the content deployment.



Deploying Content With a Deployment Group

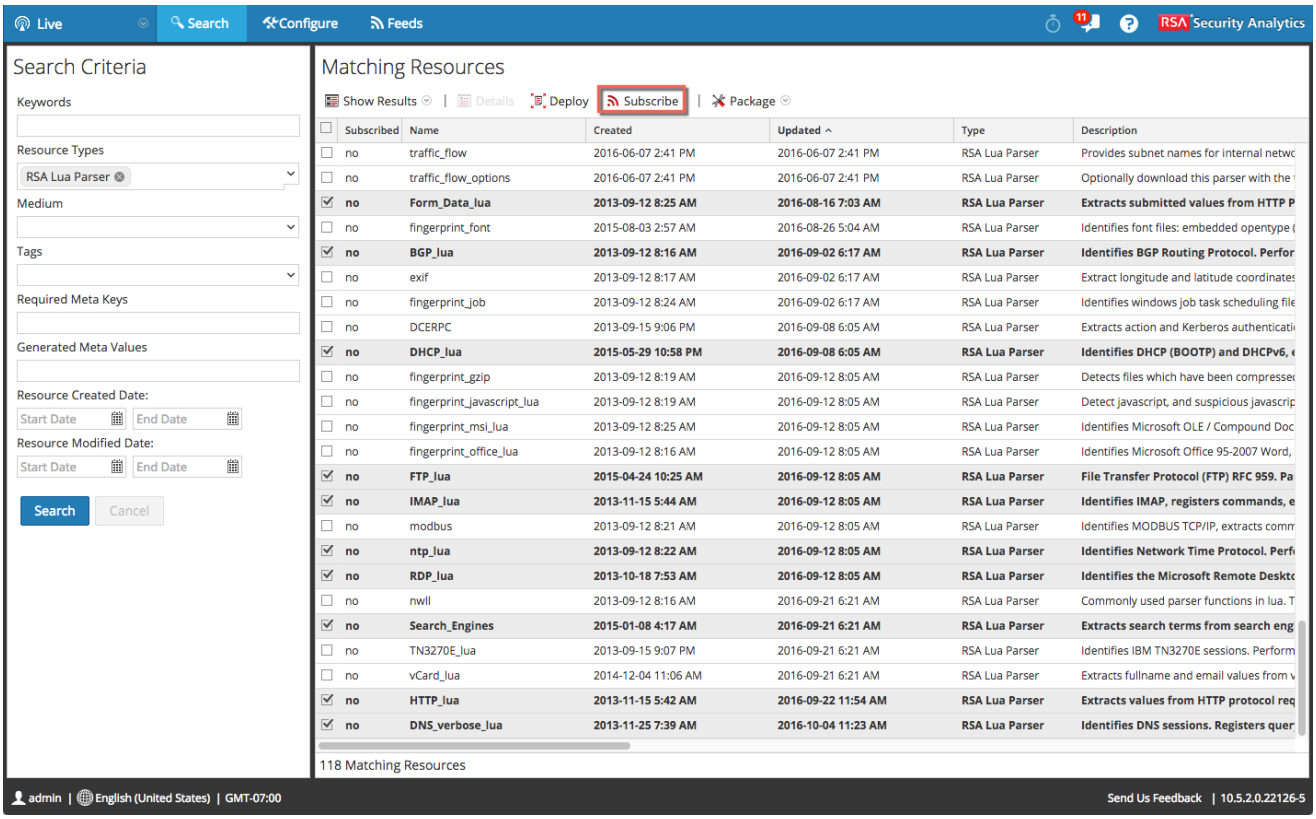
Using Deployment Groups is a slightly more lengthy procedure, but is powerful because it keeps your deployed content up-to-date when RSA publishes an update to subscribed Live content.

1. In **Administration->Services**, create a Service Group or groups containing the services for which you want to keep Live content in sync. See the lesson on UI Administration for instructions on how to do this. For instance, create a Service Group called "Packet Decoders", and drag all of your Packet Decoder services into it.

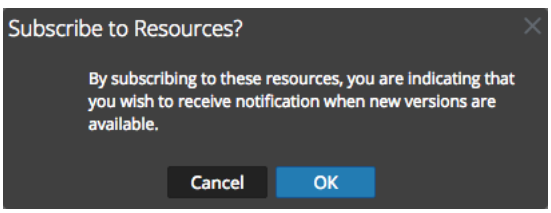


2. From **Live->Search**, select the content types you want to automatically keep in sync from the Resource Types dropdown and click **Search**. You can select more than one type in this dropdown. This should narrow down the list quite a bit.

3. From the search results, use the tick boxes to select the content which you want to add to your Deployment Group. A good time to do this is actually immediately after manually deploying the content. Click **Subscribe**.



4. Click **OK** on the acceptance box that pops up and wait for the "**Subscribing selected resources**" text which hovers over the Search screen to disappear.



5. Click the Configure tab, or go to **Live->Configure**. You will land on the **Subscriptions** tab. From here, you will see which Live content you've subscribed to. Notice that much of the content has an expander, which if you click on it, it will show content dependencies which automatically were selected when you subscribed to the content in the previous steps. You'll see that many of the LUA parsers have a dependency of 'nwll', which is a library of sorts which contains commonly-used functions which are used by other LUA parsers.

   Top Tip: The mere act of subscribing to content does not necessarily mean that the content has been deployed yet or will be deployed after it's been updated by RSA in Live.

Live | Search | Configure | Feeds | RSA Security Analytics

Subscriptions | Deployments

Details

Name	Type	Version	Updated	Description
Form_Data_lua	Lua Parser	0.6	2016-08-16 7:03 AM	Extracts submitted values from HTTP POST actions. S...
BGP_lua	Lua Parser	0.3	2016-09-02 6:17 AM	Identifies BGP Routing Protocol. Performs identificati...
DHCP_lua	Lua Parser	0.2	2016-09-08 6:05 AM	Identifies DHCP (BOOTP) and DHCPv6, extracts hosts ...
FTP_lua	Lua Parser	0.5	2016-09-12 8:05 AM	File Transfer Protocol (FTP) RFC 959. Parses only FTP c...
IMAP_lua	Lua Parser	0.3	2016-09-12 8:05 AM	Identifies IMAP, registers commands, errors, usernam...
ntp_lua	Lua Parser	0.3	2016-09-12 8:05 AM	Identifies Network Time Protocol. Performs identificat...
RDP_lua	Lua Parser	0.3	2016-09-12 8:05 AM	Identifies the Microsoft Remote Desktop Protocol. Per...
Search_Engines	Lua Parser	0.2	2016-09-21 6:21 AM	Extracts search terms from search engine queries. Se...
HTTP_lua	Lua Parser	0.6	2016-09-22 11:54 AM	Extracts values from HTTP protocol request and resp...
nwll	Lua Parser	0.16	2016-09-21 6:21 AM	Commonly used parser functions in lua. This file itself...
HTTP_lua	Lua Parser	0.6	2016-09-22 11:54 AM	Extracts values from HTTP protocol request and resp...
Alert IDs Info	Decoder Feed	0.17	2016-08-09 1:32 PM	AlertID to name mappings for informational alerts
Alert IDs Suspicious	Decoder Feed	0.21	2016-08-09 1:32 PM	Name to AlertIDs mappings for suspicious alerts
Alert IDs Warning	Decoder Feed	0.19	2016-09-01 12:56 PM	Name to AlertID mappings for warning alerts
nwll	Lua Parser	0.16	2016-09-21 6:21 AM	Commonly used parser functions in lua. This file itself...
DNS_verbose_lua	Lua Parser	0.4	2016-10-04 11:23 AM	Identifies DNS sessions. Registers query and respons...
Alert IDs Info	Decoder Feed	0.17	2016-08-09 1:32 PM	AlertID to name mappings for informational alerts
Alert IDs Suspicious	Decoder Feed	0.21	2016-08-09 1:32 PM	Name to AlertIDs mappings for suspicious alerts
Alert IDs Warning	Decoder Feed	0.19	2016-09-01 12:56 PM	Name to AlertID mappings for warning alerts
nwll	Lua Parser	0.16	2016-09-21 6:21 AM	Commonly used parser functions in lua. This file itself...

admin | English (United States) | GMT-07:00 Send Us Feedback | 10.5.2.0.22126-5

6. Click the **Deployments** tab. The Deployments tab displays SA's Service Groups, which are Deployment Groups as well. Click on the Packet Decoders Group in the Groups pane. You'll see the + button illuminate in the Subscriptions pane. Now click on the + button.

Live | Search | Configure | Feeds | RSA Security Analytics

Subscriptions | Deployments

Groups

Name	Address	Type
Packet Decoders		

Subscriptions

Name	Created	Updated	Type	Description

admin | English (United States) | GMT-07:00 Send Us Feedback | 10.5.2.0.22126-5

Add Subscription

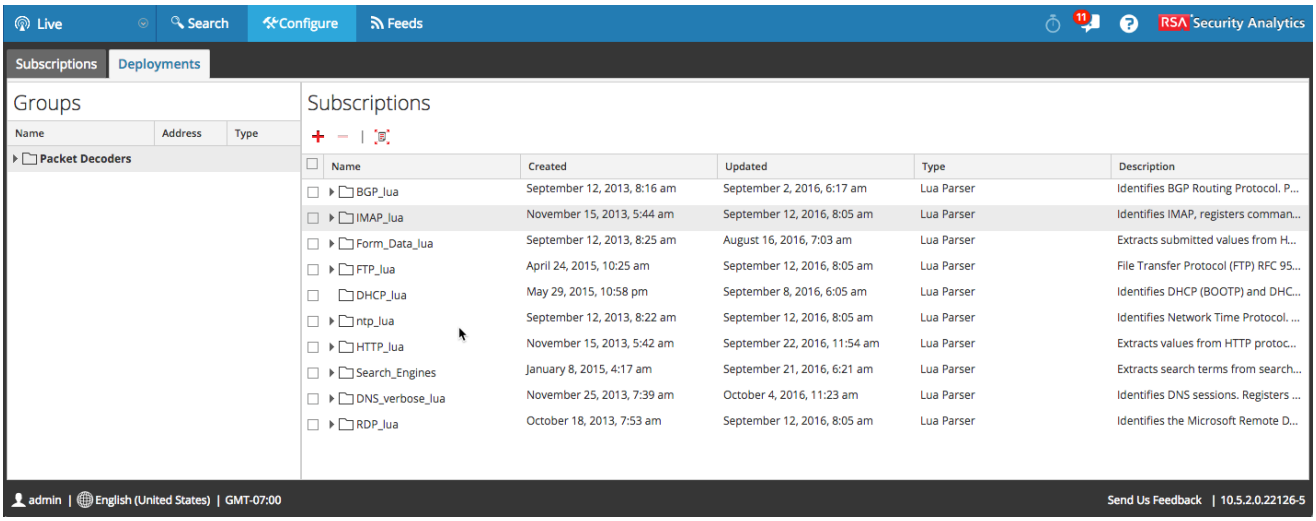
Name	Type
Form_Data_lua	Lua Parser
BGP_lua	Lua Parser
DHCP_lua	Lua Parser
FTP_lua	Lua Parser
IMAP_lua	Lua Parser
ntp_lua	Lua Parser
RDP_lua	Lua Parser
Search_Engines	Lua Parser
HTTP_lua	Lua Parser
DNS_verbose_lua	Lua Parser

Close | Save

7. Now the **Add Subscription** dialog opens. This dialog can be deceptively tricky to use, as it will contain all subscribed content which does not already belong to the selected Deployment Group, and if there's a lot of it, it can be difficult to pick out the content you want to add to the group. It is by default much too small, though it can be expanded by dragging its corners. If there are multiple content types, it often works best to select the Type column header to sort by content type, to make it easier to locate the content you want to add to the group.

   Select the content you want to add to the Deployment Group, then click **Save**.

8. You should now see all content which you've selected in the Subscriptions pane. To deploy all content in the Deployment Group, click the **Sychronise** button, which looks like .



9. All content in the Deployment Group will now, in the background, be pushed out to all applicable services in the Service Group.

   If a Service Group contains more than type of Device, say Log Decoders and Packet Decoders, Live is smart enough to push only LUA parser content to the Packet Decoder group members and Log Device content to the Log Decoder group members. However, in the case of Application Rules, Live cannot distinguish between packets-only and logs-only app rules, and will push all app rules to both Log Decoders and Packet Decoders.

   Top Tip: The UI logfile is a good place to find informational and error messages concerning the downloading and deploying Live content. This can be found at
   /var/lib/netwitness/uax/logs/sa.log.

```
2016-11-09 00:24:53,156 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: alertids_suspicious.zip UUID: 7587fdd4-d1f6-4f37-a91a-63625125a561 to service: Decoder
2016-11-09 00:24:53,262 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: alertids_info.zip UUID: ac241a8e-c98e-406d-b5de-a3965cf79973 to service: Decoder
2016-11-09 00:24:53,324 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: dhcp.zip UUID: b647e1af-6d75-4fca-b283-fc757c3ab011 to service: Decoder
2016-11-09 00:24:53,415 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: nwll.zip UUID: 859d943a-4d02-496c-8ca4-5868441384c5 to service: Decoder
2016-11-09 00:24:53,482 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: alertids_warning.zip UUID: 67e52493-77cb-44f8-ab9b-1a6085db3f96 to service: Decoder
2016-11-09 00:24:53,546 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: search_engines.zip UUID: fe3091e3-1715-40ee-b7a0-203ef776bd13 to service: Decoder
2016-11-09 00:24:53,615 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: http.zip UUID: 282dd1ff-89cd-4376-9152-a13a54e8c852 to service: Decoder
2016-11-09 00:24:53,670 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: rdp.zip UUID: bae409e2-68e1-4bf4-b2c5-faa59d1a7f66 to service: Decoder
2016-11-09 00:24:53,738 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: formdata.zip UUID: 209312e4-771b-432c-9514-7b8e866b9925 to service: Decoder
2016-11-09 00:24:53,838 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: imap.zip UUID: 1fbf9c85-1430-4684-bcfa-4b73d822ec52 to service: Decoder
2016-11-09 00:24:53,902 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: ftp.zip UUID: 4aac5926-3615-475a-aa4f-ebd5c5dc7c0c to service: Decoder
2016-11-09 00:24:53,963 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: dns_verbose.zip UUID: 3fb51ab6-9645-444a-b2a7-65058a260a3f to service: Decoder
2016-11-09 00:24:54,059 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: bgp.zip UUID: 63d7da97-7134-4c70-b966-5c51eb4462e4 to service: Decoder
2016-11-09 00:24:54,169 [Live Task Deployment 331400217] INFO  com.rsa.smc.sa.live.job.task.DeploymentTask - Successfully deployed
live resource file: ntp.zip UUID: 62d3b06b-5280-455c-8418-15443c08c0f5 to service: Decoder
2016-11-09 01:11:45,407 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription nwll is
already deployed for group
2016-11-09 01:11:45,408 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription nwll is
already deployed for group
2016-11-09 01:11:45,408 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription nwll is
already deployed for group
2016-11-09 01:11:45,409 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription nwll is
already deployed for group
2016-11-09 01:11:45,409 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription nwll is
already deployed for group
2016-11-09 01:11:45,409 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription nwll is
already deployed for group
2016-11-09 01:11:45,409 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription HTTP_lua is
already deployed for group
2016-11-09 01:11:45,409 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription Alert IDs
Warning is already deployed for group
2016-11-09 01:11:45,410 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription Alert IDs
Suspicious is already deployed for group
2016-11-09 01:11:45,410 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription nwll is
already deployed for group
2016-11-09 01:11:45,410 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription Alert IDs
Info is already deployed for group
2016-11-09 01:11:45,410 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription Alert IDs
Warning is already deployed for group
2016-11-09 01:11:45,410 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription Alert IDs
Suspicious is already deployed for group
2016-11-09 01:11:45,410 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription nwll is
already deployed for group
2016-11-09 01:11:45,410 [qtp684874119-5338] INFO  com.rsa.smc.sa.live.web.controller.LiveManageController - Subscription Alert IDs
Info is already deployed for group
```

Using Live in an Air Gap Environment

It is still possible to use Live if your SA server does not have access to the Internet. This requires a second instance of the UI running which does have Internet access. You can do this by creating an SA UI server instance in a virtual machine, running an SA trial license perpetually.

By creating **Content Packages** with the secondary SA server that has Internet access, you can then take the Content Package file, which contains all the needed Live content, back to the airgapped UI server, from where you can deploy it to your SA services.

Top Tip: Content Packages do not work with the Subscription / Content Group model. It is purely manual deployment.

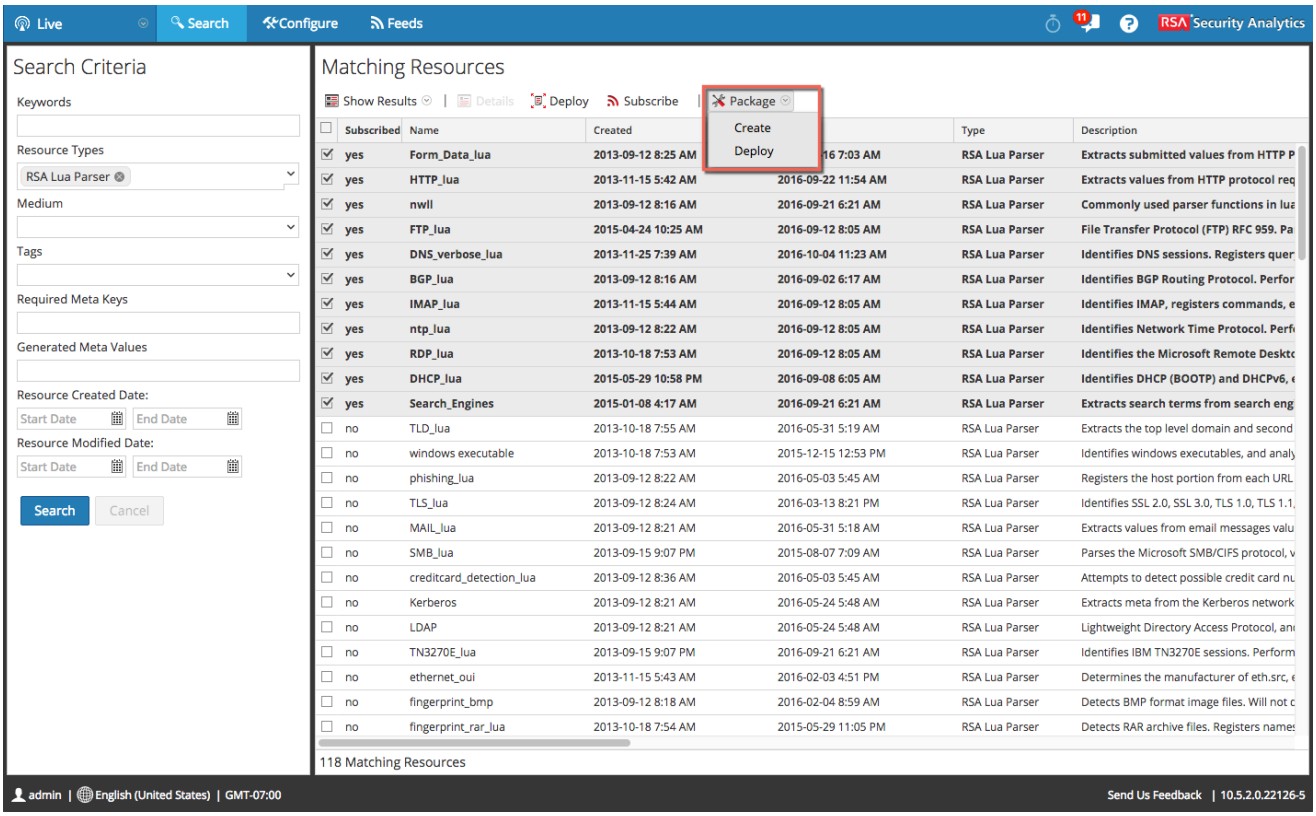
To Create a Content Package

1. From Internet-connected SA UI instance, navigate to **Live->Search**, then select the content types you want to deploy from the Resource Types dropdown and then click **Search**. You can select more than one type in this dropdown. This should narrow down the list quite a bit.

   Top Tip: Though you can't Shift-Click or Control-Click to quickly multi-select here, after ticking one box you can multi-select by using Shift-Arrow,

which will select each subsequent row from the Matching Resources pane.

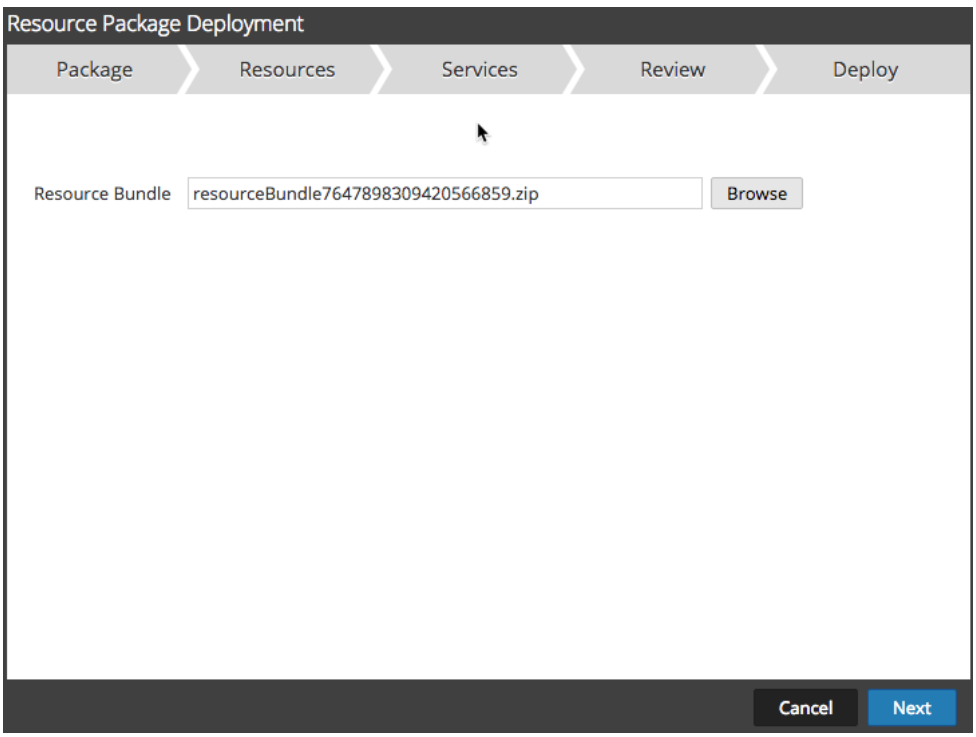
2. Click the **Package** menu, then **Create.** A file named **resourceBundle<*somelongnumber*>.zip** (e.g. resourceBundle7647898309420566859.zip) will then be created by Live and automatically downloaded by your web browser.



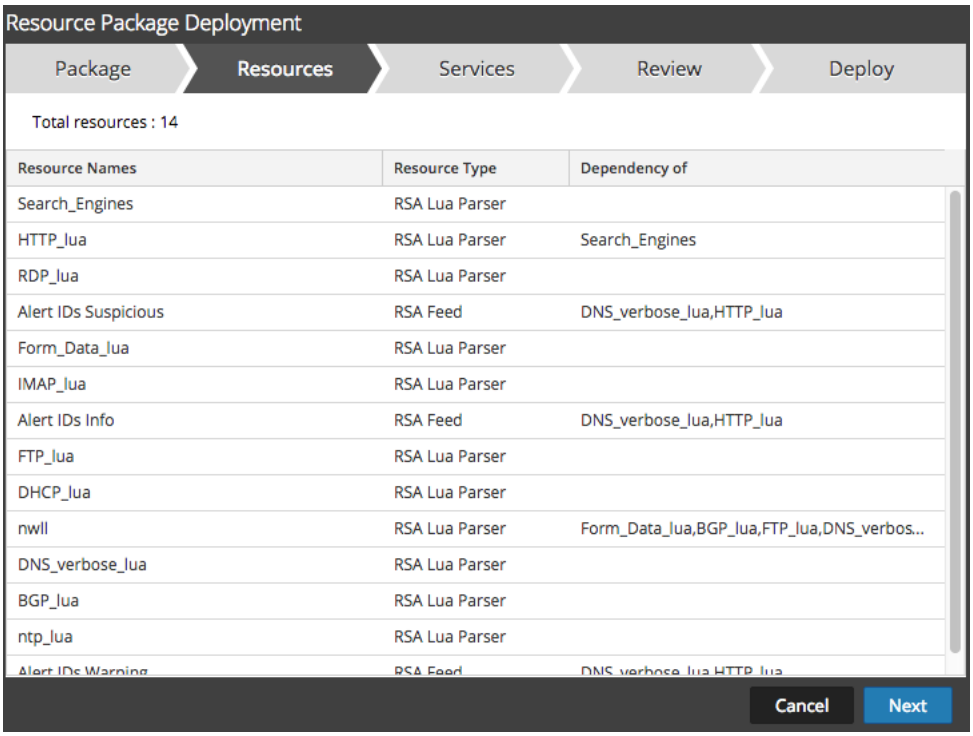
3. Using your favourite method of moving the file into your airgapped environment (USB stick, floppy disk, Morse Code, etc), copy the Live package file to the workstation you use to control your airgapped SA server.

4. From your airgapped SA server, Navigate to **Live->Search**, then click on the **Package** menu and select **Deploy**. The **Resource Package Deployment** wizard then appears.

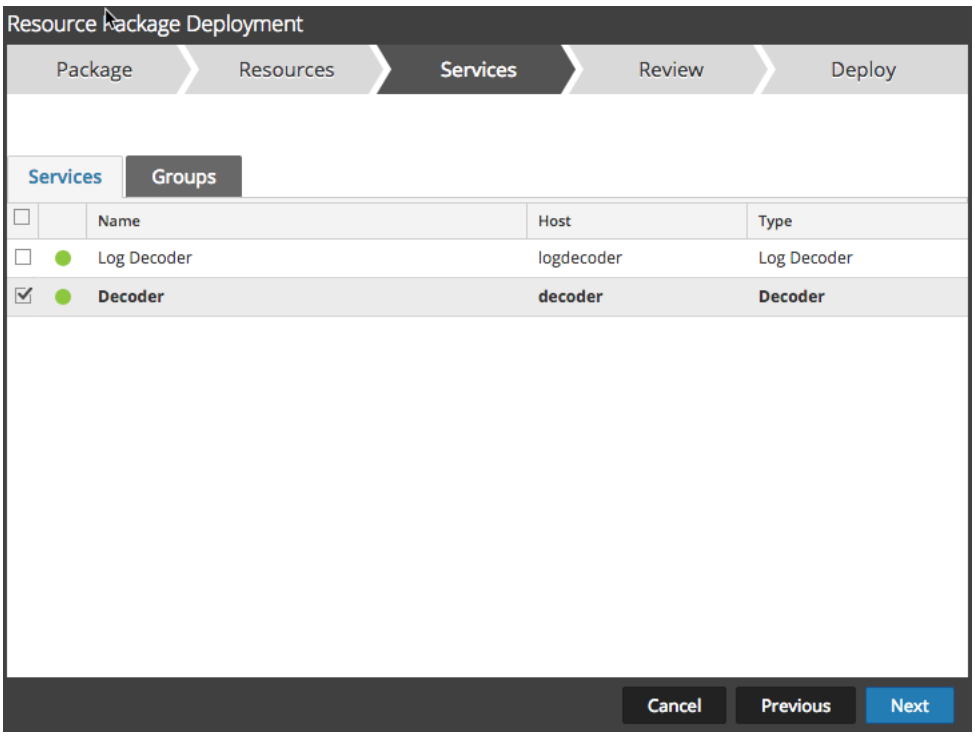
5. Click the Browse button and select your Content Package file. Click **Next**.



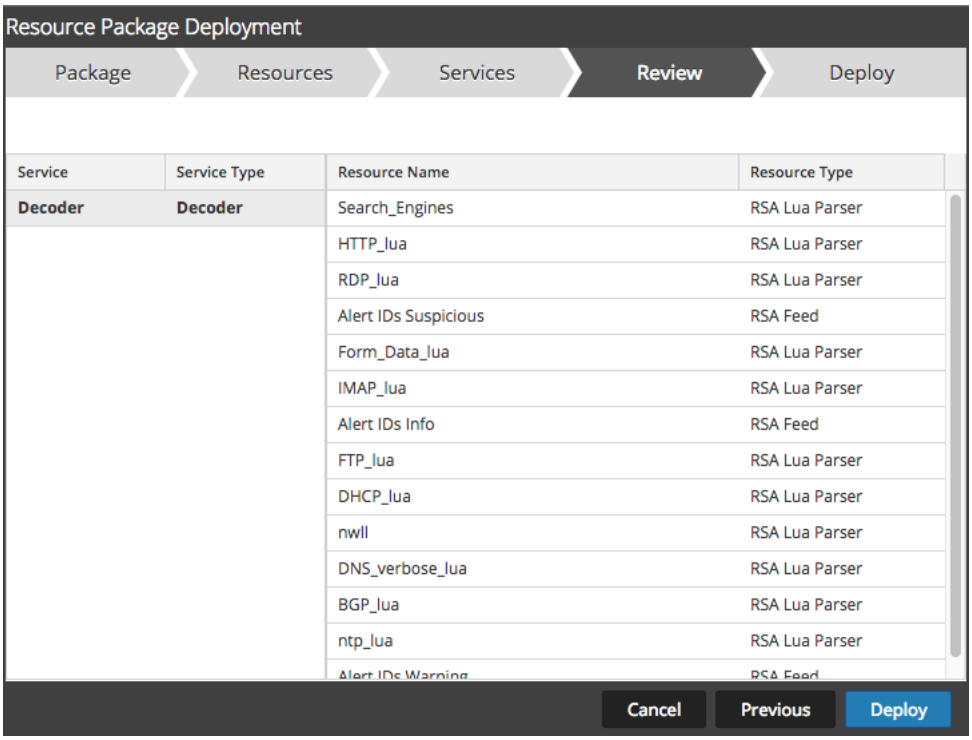
6. The Live resources contained in the package file will be shown in the **Resources** tab. Click **Next**. You will not be given a choice of which resources to deploy. All content from the package will be deployed.



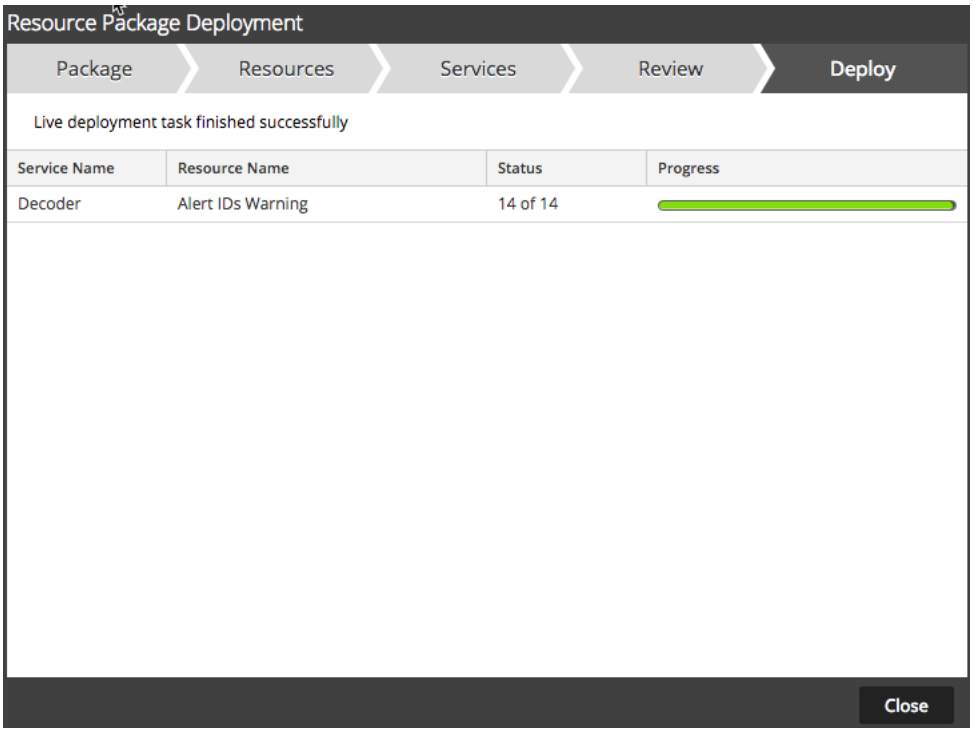
7. Select the SA services to which you would like to deploy the Live content from the **Services** tab.



8. Review your selections from the **Review** tab, and click **Deploy**.



9. From the **Deploy** tab, you will see a progress bar indicating the status of the content deployment. Click **Close**. Your Live content is now deployed in your airgapped environment.



20. Thick Investigator Client

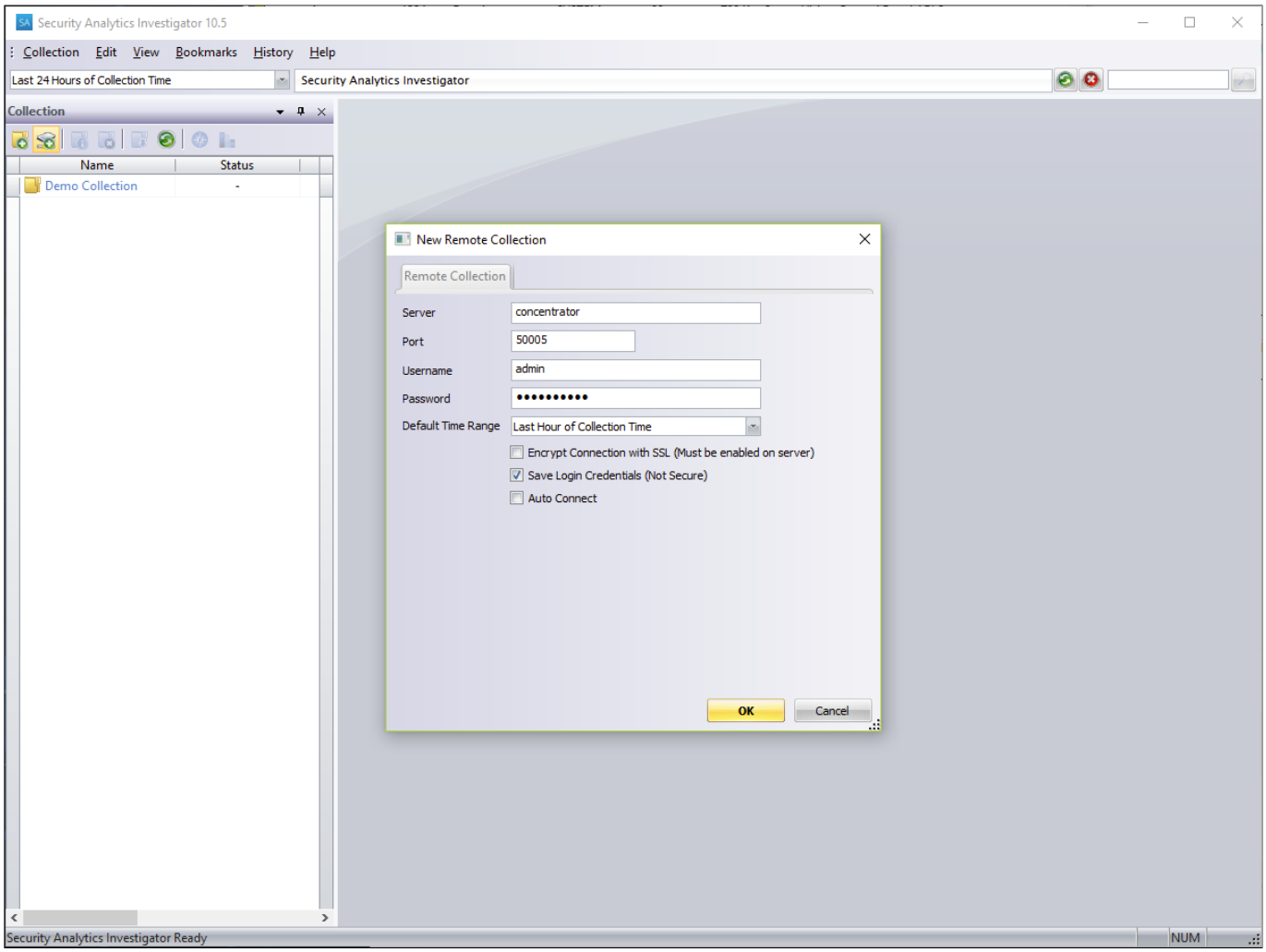
Using Legacy Thick Investigator to Inspect PCAP's and Test Content

The legacy Investigator thick client can still be an extremely useful tool for inspecting PCAP files and also testing custom content, like feeds, app rules, and parsers.

The Investigator client can be downloaded from RSA Link at https://community.rsa.com. It runs only on Windows, and as of the time of writing, the latest version available is 10.5.1.0.

Unlike the full SA suite, the Investigator client has what are called "Local Collections". Rather than connecting directly to a Concentrator to view captured data, a pcap can be imported into the thick client and its contents browsed through in almost exactly the same way as navigating a Concentrator's meta and reconstructing sessions. It can also capture data off the wire from the workstation it's running on, just like a Decoder. The thick Investigator client essentially has an indexing Decoder built into it, but instead of running NwDecoder, it runs an executable called NwInvestigator.exe. The data is stored in a data file which internally uses the same NWDB database files as SA.

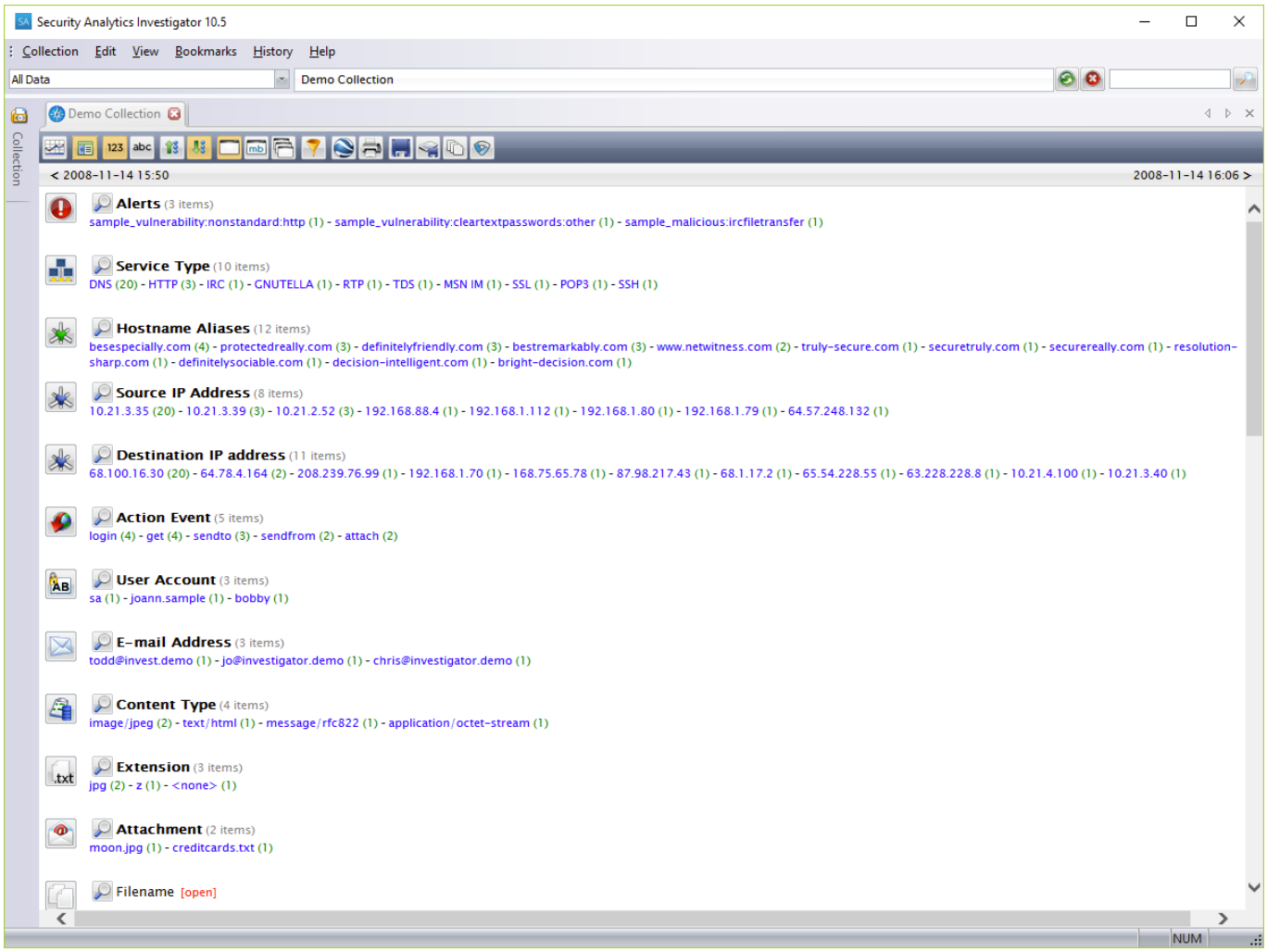
The thick client can also connect to existing SA infrastructure through the use of "Remote Collections".



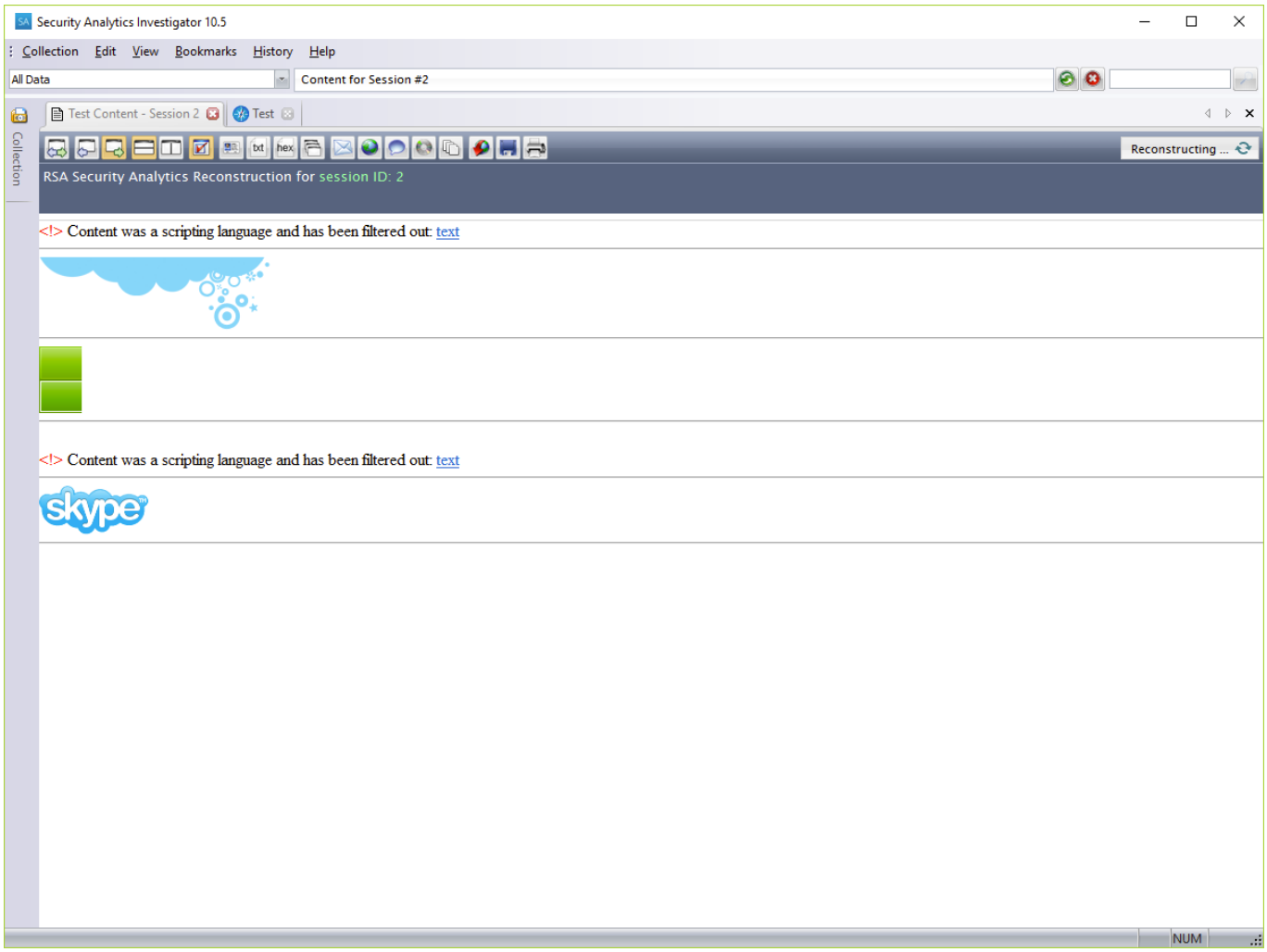
Note: Once you've installed the 10.5.1 thick client, in order to activate it, you will need to connect to a remote collection (i.e. a Decoder, Concentrator, or Broker). This ensures that people downloading it have SA. After you've done this, you will be able to create local collections.

Browsing the Demo Collection

Let's have a look at Investigator's built-in demo collection. Double-click on it to connect to it, then double-click again to browse the collection. You can drill through and query the data in the same way as the SA thin client. You can also reconstruct sessions in a similar way, by clicking on the number to the right of a meta value.



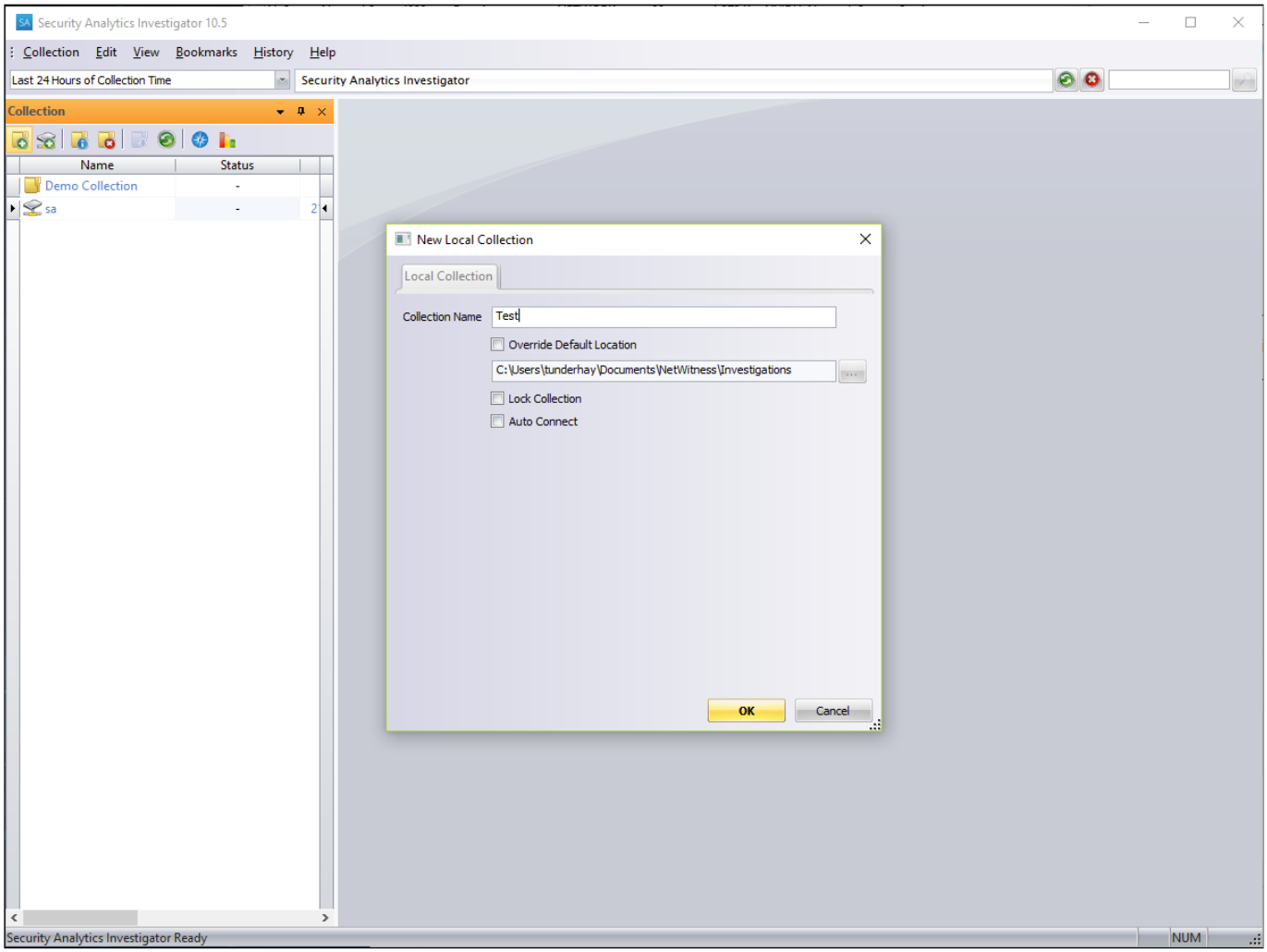
Demo Collection Meta



A Reconstructed Session

Creating a Local Collection and Importing a PCAP

1. Click the New Local Collection Button, give it a name, and click 'OK'.

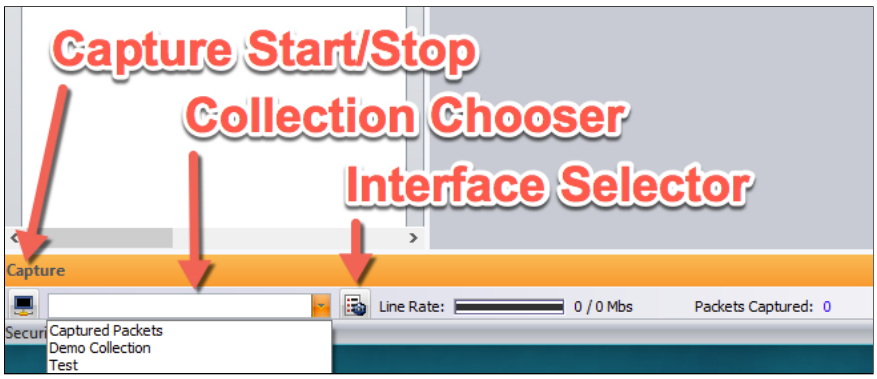


2. Double-click on the newly created collection to connect to it. Then right-click on it and click "Import Packets". Now choose a pcap file and click 'OK'. You will see a progress bar next to the collection name.

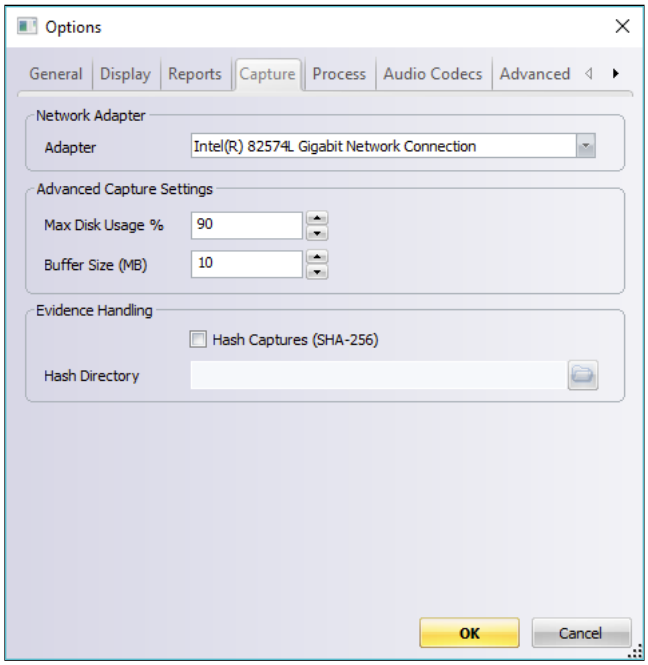
3. Once this has finished, double-click the collection again to browse it.

Capturing Off the Wire

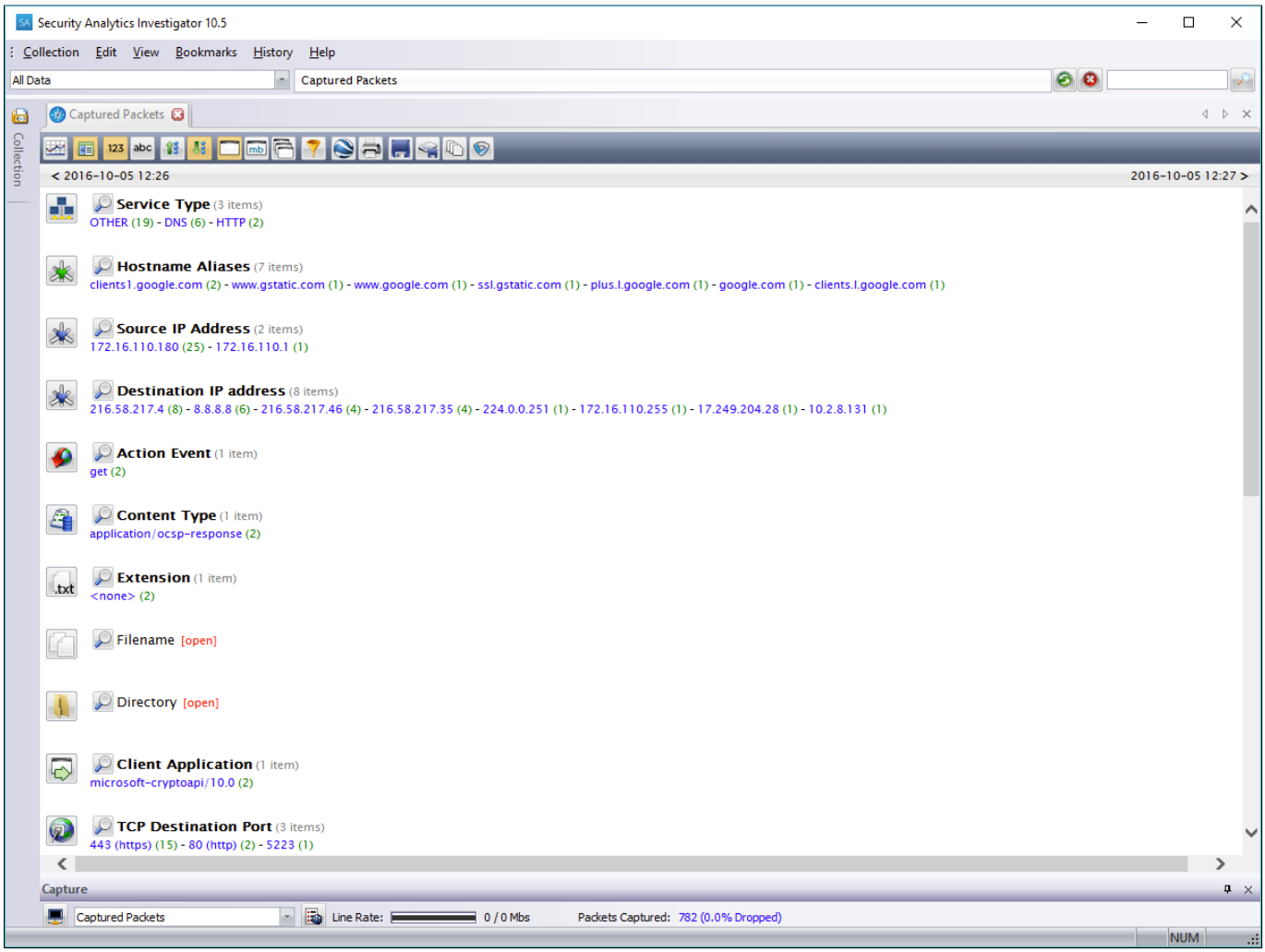
If you optionally installed WinPCAP during the Investigator install process, you may also capture packets off the wire of your Windows workstation.



1. Click the View menu, then Capture Bar. You will see the capture bar appear at the bottom of your Investigator window.
2. Choose a collection to save the captured packets to from the drop-down
3. Click the Interface Selector to choose the appropriate NIC to capture from



4. Click the Start/Stop Capture button
5. Do some activity on your PC, such as browse a web page
6. Click the Start/Stop Capture button to end capture
7. Double-click the collection name to browse the meta and packets of what you just captured



Installing Live Content

Thick Investigator can use app rules, feeds, and parsers from NetWitness Live in the same fashion as SA.

1. You first open Investigator preferences in **Edit -> Options**

2. Click the Advanced tab

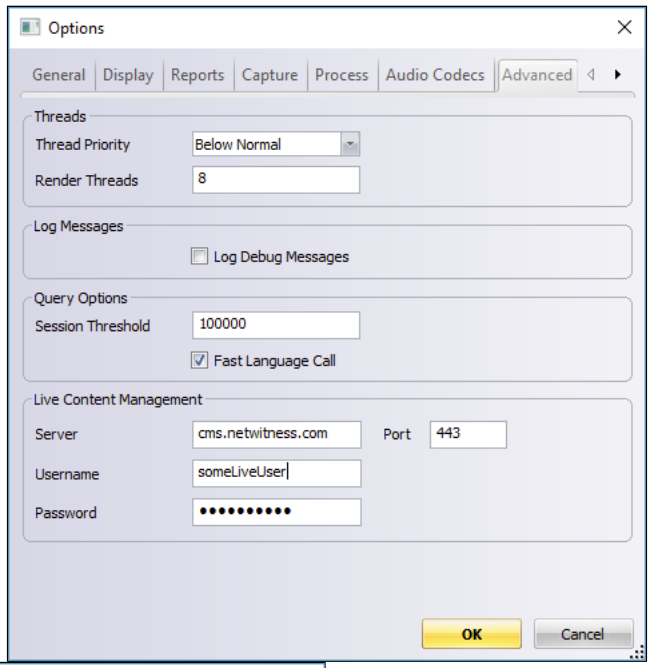
3. Enter your Live Credentials under Live Content Management, with the server name as **cms.netwitness.com** and the port as **443**. Click 'OK'.

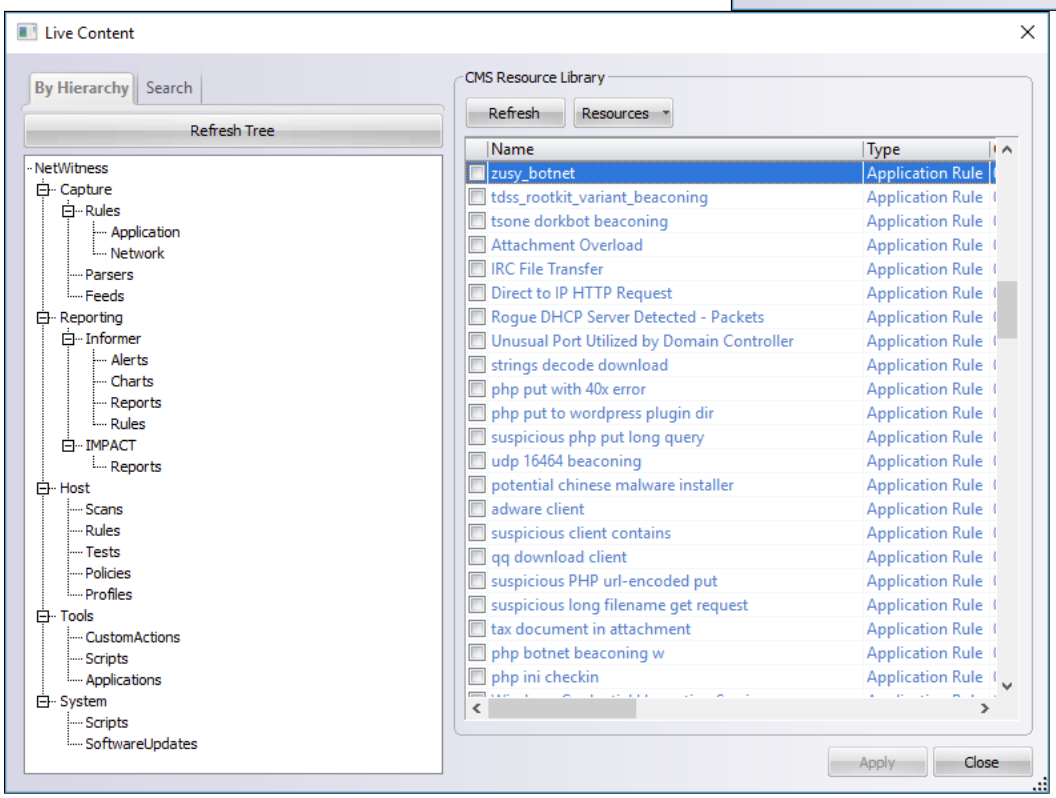
4. Click **View -> Live Content -> Resources**

5. Navigate through the tree, and choose the content you wish to deploy by clicking the tickboxes. Not all content will be applicable to Investigator (Reporting, etc).

Technical Note: The parsers you see in the Live resources window will unfortunately only display legacy FLEX parsers, but not LUA parsers. It *should* be possible to install LUA parsers manually in the parsers directory.

6. Click **Apply** to download and install the content into Investigator. The next time you process a local collection, either by importing a PCAP file or reprocessing a collection, this content will be used during the parsing.





Reprocessing a Collection

If you've changed the content configuration of Investigator, either through a new or updated feed, parser, app rule, or index configuration, you may want to update an existing Local Collection's meta using this new content. You can't do this on a Decoder, but you can with Investigator.

1. Connect to a Local Collection by double-clicking on it.

2. Right-click on the collection and click Reprocess Collection. You will see a progress bar indicating the re-parsing of the collection.

3. Click Yes at the warning.

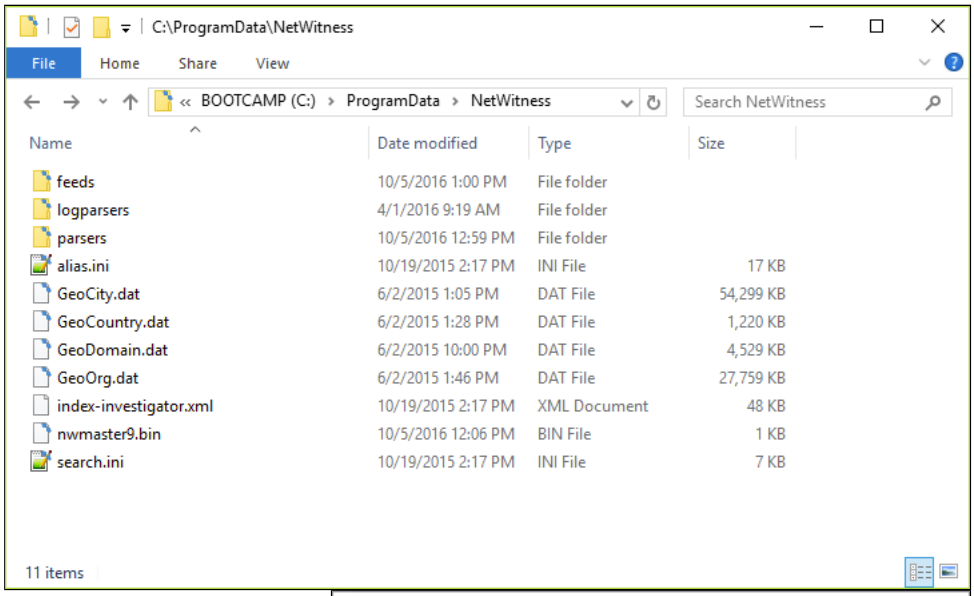
4. Choose a new name for the reprocessed Local Collection and click OK.

5. You will see a progress bar next to the new (reprocessed) Local Collection indicating how far along the parsing is. When this is finished, you can double-click on the collection to browse its meta.

Customising Investigator's Engine Configuration

Just like a Decoder, the thick Investigator client stores its feeds, parsers, and index configuration in one directory. This directory is C:\ProgramData\NetWitness. Here you see the same things as you would see on a Decoder, with the exception of the engine's main configuration XML file (the equivalent of NwDecoder.cfg), which is stored in the Windows registry.

The files in this directory have **no bearing on Remote Collections**. They only apply when parsing new packet data in Local Collections, or when reprocessing a Local Collection.



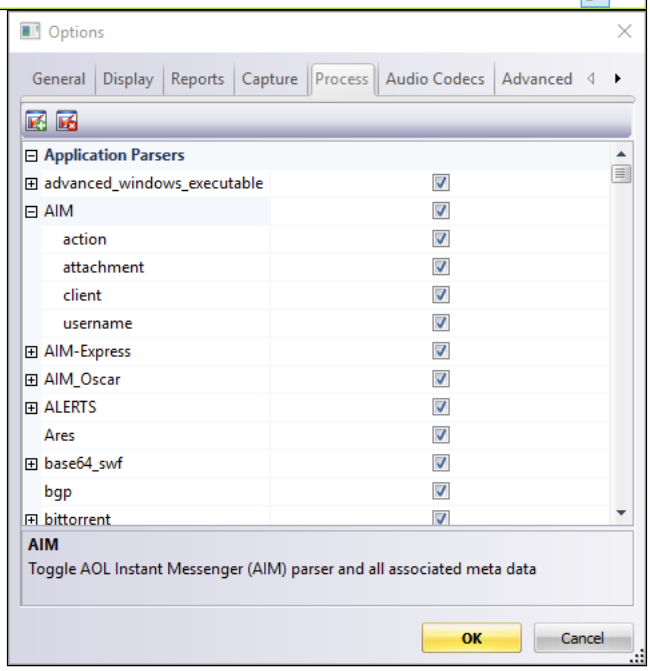
Choosing Parsers

Like a full Decoder, you can choose which of the installed feeds and parsers you want to enable.

1. Click **Edit -> Options** (warning: if you have a lot of content installed, this could take a while to come up - be patient)

2. Click the **Process** tab

3. Just like in the thin client, the parser names are the left-hand side and whether they are enabled is on the right. Click the + sign to expand a parser to see which meta keys that it will generate meta under. Furthermore, you can disable individual meta keys under which a parser generates meta, if you only need specific meta types from a given parser.



Customising the Investigator Index Configuration

This is the same as for any other SA Core service, only the index configuration file is **C:\ProgramData\NetWitness\index-investigator.xml.** Use your favourite **UNIX-linefeed compatible** text editor to modify it.

C:\ProgramData\NetWitness\index-investigator.xml - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

index-investigator.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<language level="IndexNone" defaultAction="Auto">
    <!--
    Index Language Definition File 10.3 (10-4-2013)

    NOTE: This file should not be updated by customers.  It will no longer be backed
    up and restored during updates.  Instead, all customer specific changes should be
    made to index-concentrator-custom.xml.  This file will not be touched during the upgrade
    process and any keys in that file will override this file.

    <language> root node which specifies any defaults found in key nodes
        <key> child node of language which can have the following attributes:
            name = Name of the key (max is 16 chars)
            description = Description of the key (max is 256 bytes, strings encoded as UTF-8)
            format = Variant format of the value which can be:
                Int8, UInt8, Int16, UInt16, Int32, UInt32, Int64, UInt64, UInt128
                Float32, Float64, TimeT, Binary, Text, IPv4, IPv6, MAC
            valueMax = maximum number of unique values allowed per index slice (typically 1 to 4 slices are created per day)
                       zero means unbounded
            defaultAction = Default investigation action for each report: Open, Close, Auto, Hidden

            <value> child node of key which specifies overrides at the value level:
                name = Specific value for the override
                action = keep|filter

            aliases
                alias - list of aliases for a value to human readable name
    -->

    <!-- time needs to always be indexed at value level -->
    <key description="Time" format="TimeT" level="IndexValues" name="time" valueMax="0" />

    <!-- concentrator values used during aggregation -->
    <key description="Remote Session ID" format="UInt64" level="IndexKeys" name="rid" />
    <key description="Concentrator Source" format="Text" level="IndexValues" name="cid" valueMax="256" />
    <key description="Decoder Source" format="Text" level="IndexValues" name="did" valueMax="256" />

    <!-- Risk assessment keys -->
    <key description="Alerts" format="Text" level="IndexValues" name="alert" valueMax="100000" defaultAction="Open"/>
    <key description="Risk: Informational" format="Text" level="IndexValues" name="risk.info" valueMax="250000" defaultAction="Open"/>
    <key description="Risk: Suspicious" format="Text" level="IndexValues" name="risk.suspicious" valueMax="250000" defaultAction="Open"/>
    <key description="Risk: Warning" format="Text" level="IndexValues" name="risk.warning" valueMax="250000" defaultAction="Open"/>

    <!-- Feed information definitions -->
    <key description="Feed Name" format="Text" level="IndexKeys" name="feed.name"/>
    <key description="Feed Category" format="Text" level="IndexKeys" name="feed.category"/>
    <key description="Feed Description" format="Text" level="IndexKeys" name="feed.desc"/>
    <key description="Threat Source" format="Text" level="IndexKeys" name="threat.source"/>
    <key description="Threat Category" format="Text" level="IndexKeys" name="threat.category"/>
    <key description="Threat Description" format="Text" level="IndexKeys" name="threat.desc"/>

    <!-- Standard keys -->
    <key description="Service Type" format="UInt32" level="IndexValues" name="service" valueMax="75" defaultAction="Open">
        <aliases>
            <alias format="$alias" value="0">OTHER</alias>
            <alias format="$alias" value="20">FTPD</alias>
            <alias format="$alias" value="21">FTP</alias>
            <alias format="$alias" value="22">SSH</alias>
            <alias format="$alias" value="23">TELNET</alias>
```

eXtensible Markup Language file | length : 48204 lines : 750 | Ln : 1 Col : 1 Sel : 0 | 0 | UNIX | UTF-8 | INS

21. Notifications

Security Analytics has robust, if somewhat complex, notification mechanisms. A notification mechanism is some kind of message sent from the SA system to people who need to know something, such as when an important security alert has fired, a health alarm, or when an event source has stopped sending logs.

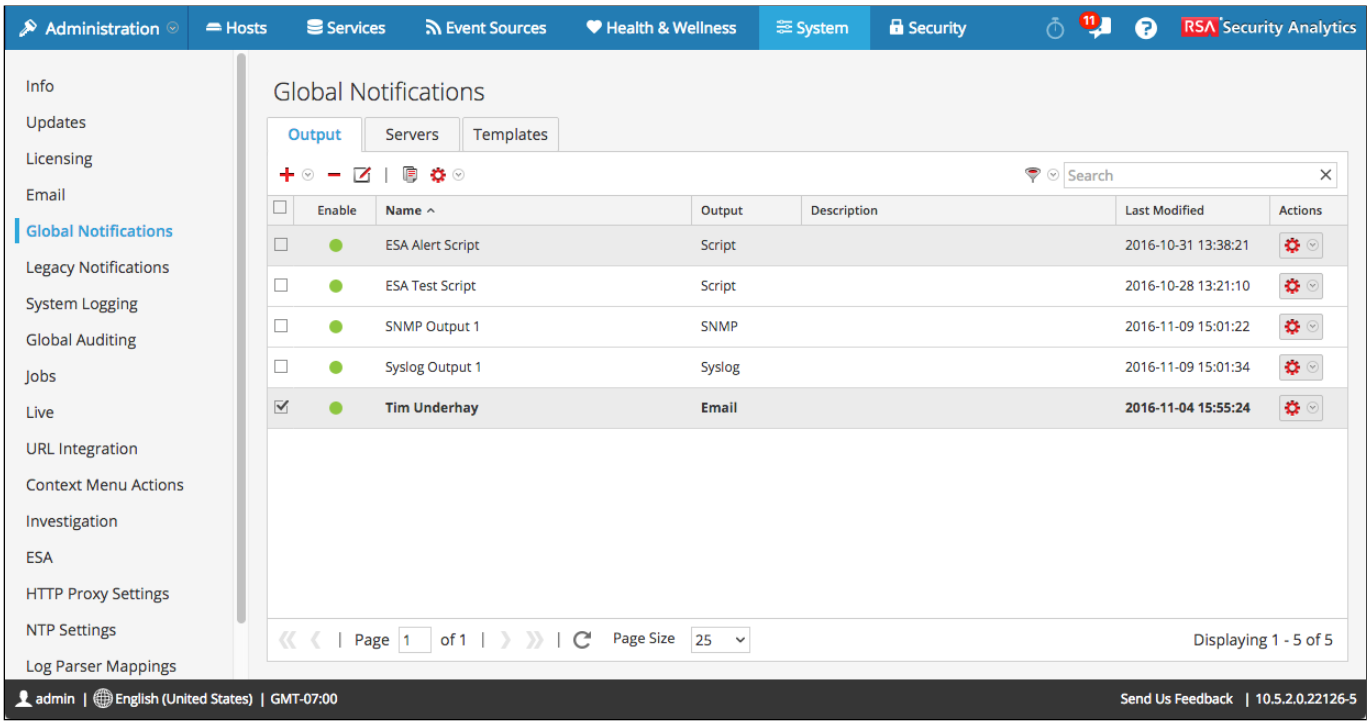
There are two notification paradigms:

- Global Notifications
- Legacy Notifications

SA is inconsistent with its user experience around notifications, though this appears to be improving with every release. Some parts of the product use Global Notifications, and other parts use disparate Legacy Notifications.

Global Notifications

Global Notifications is the newer and primary method in which SA sends notifications. It is configurable in a centralised manner. Its centralised settings are configured from **Administration->System->Global Notifications**.



Types of Global Notifications

It supports these notification types:

- **Email**
- **SNMP Trap**
- **Syslog Message**
- **A Custom Script** - This is only supported by ESA

Where Are Global Notifications Used in SA?

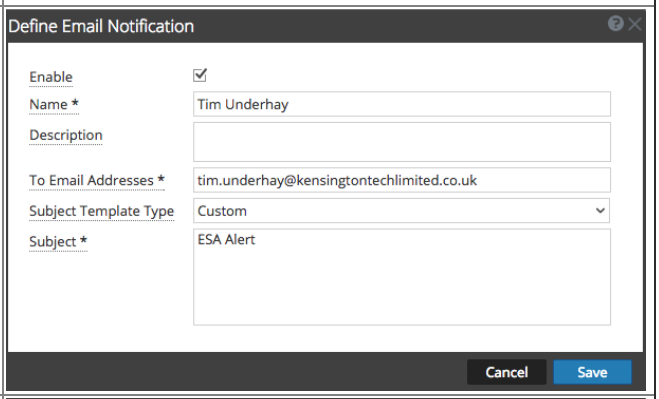
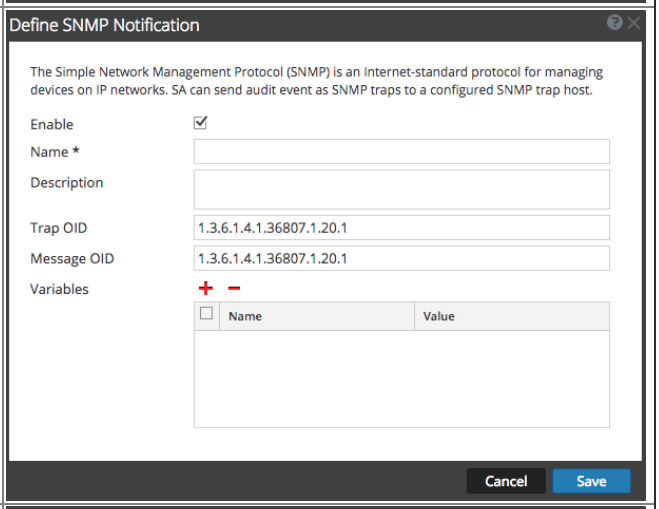
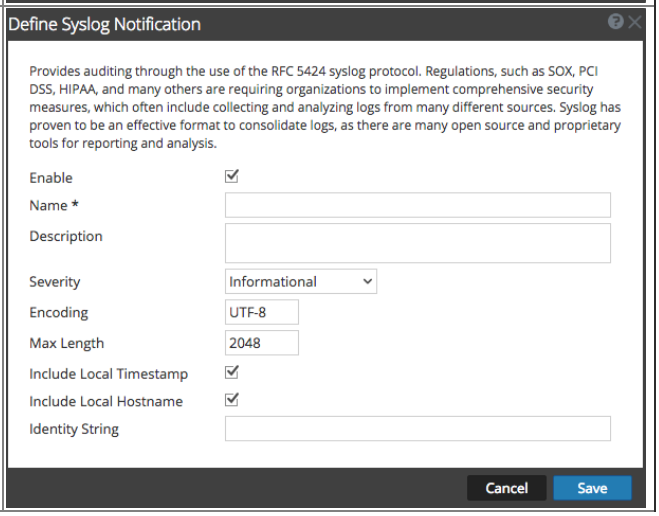
Global Notifications are used by the following parts of the UI:

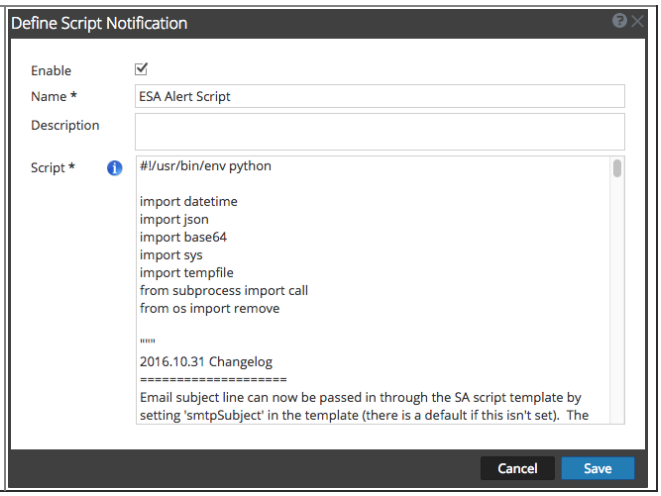
- Event Source Monitoring
- Health & Wellness
- Alerts (Event Stream Analysis)
- Incident Management
- Global Audit Logging (supports only syslog)

Global Notification Concepts

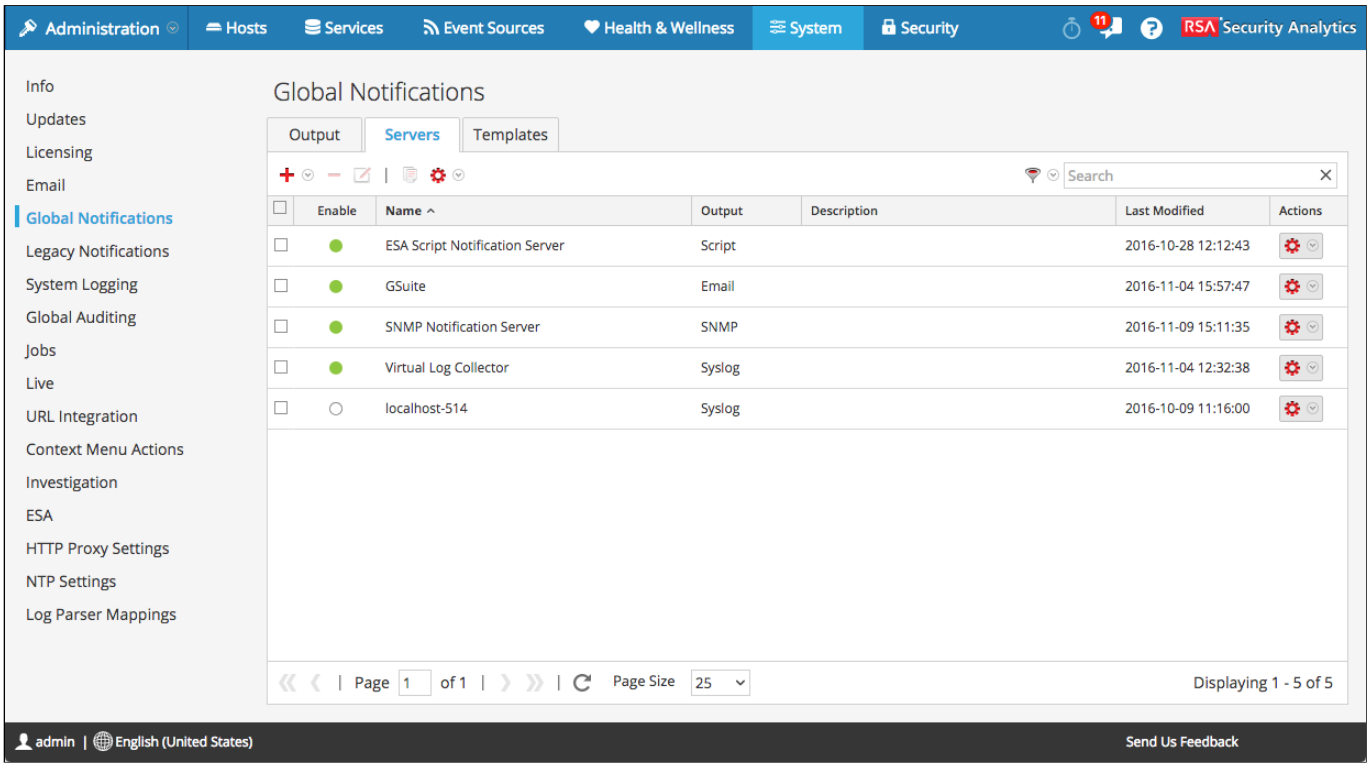
Global Notifications are oriented around three concepts, namely outputs, servers, and templates.

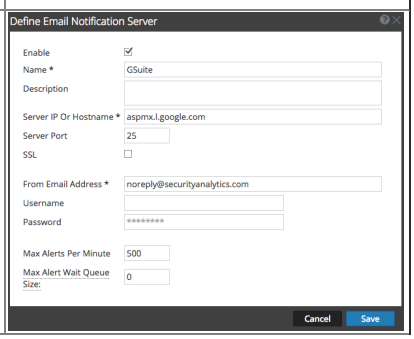
- **Output** - A slightly nebulous concept. Outputs define parameters specific to each notification type.

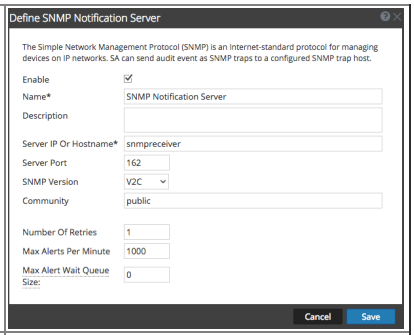
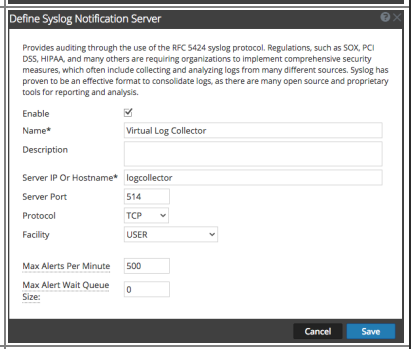
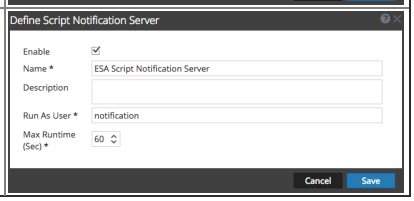
Sub-Type	Description	Image
Email	This is each recipient's email address and an email subject line.	
SNMP	The SNMP trap and message OID, and any custom variables.	
Syslog	Parameters such as severity level, encoding, message length, hostname and timestamp.	

Script	The script body, including shebang (#!) line.	

- **Servers** - Servers are fairly self-explanatory. They define servers that Global Notifications will send notifications to.



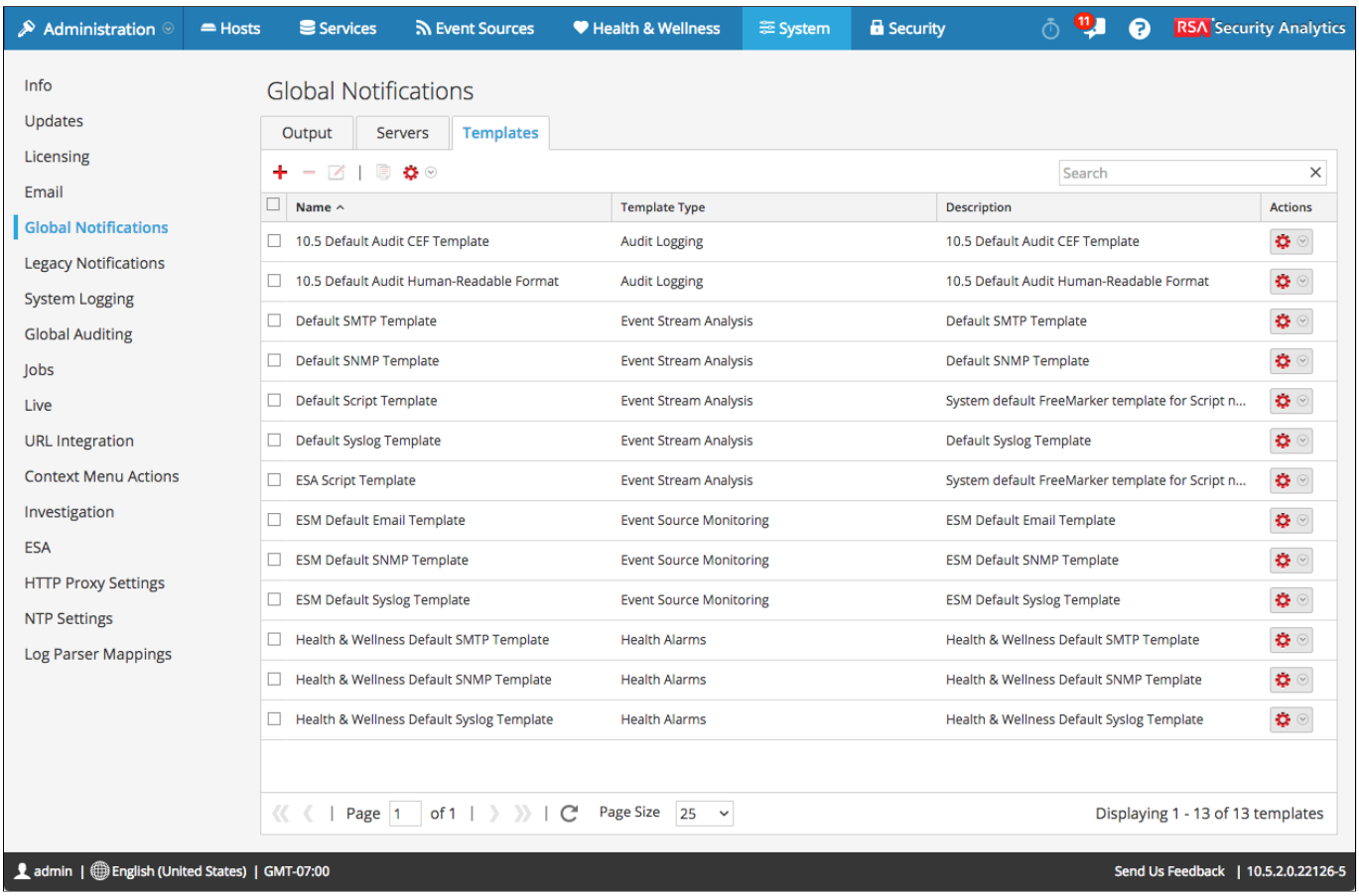
Sub-Type	Description	Image
Email	Defines an SMTP server and its authentication properties	

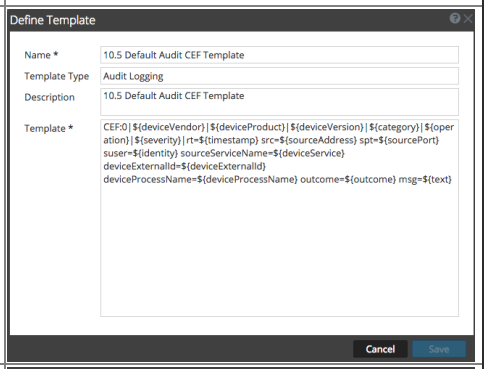
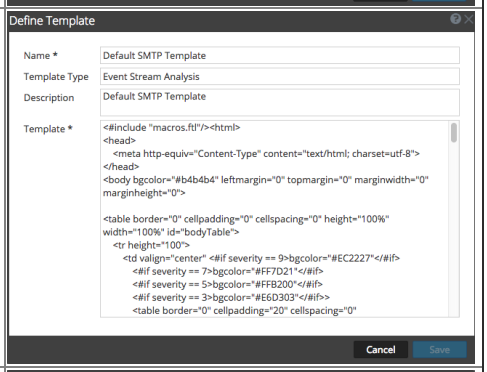
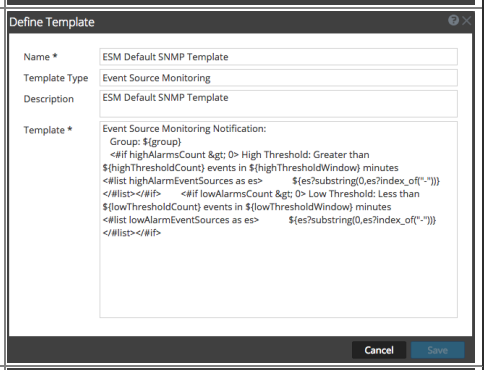
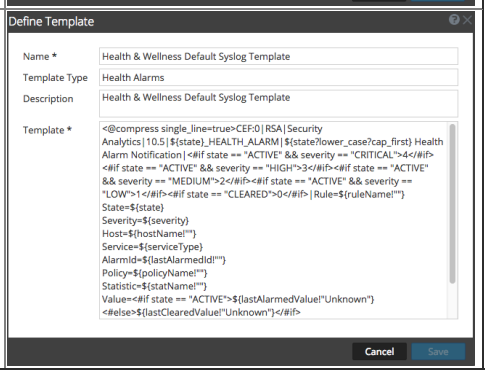
SNMP	Defines an SNMP trap receiver server and its settings.	
Syslog	Defines a syslog server, its protocol, and logging facility. This can also be an SA log device; either a Log Decoder or Log Collector.	
Script	As is really no server associated with scripts as it's something that gets run rather than sent, this only defines the user the script will run under, and how long it will be permitted to run for until it's killed.	

- **Templates** - Templates define the body of what is sent in a notification. They can contain variables which are specific to each service which sends the notification. Variables are defined using **${*varName*}** syntax. There are predefined templates for Audit Logging, Event Stream Analysis, Event Source Monitoring, and Health Alarms. Every template specifies a type which binds it to one of these uses.

  Different templates can be defined for different output types. For instance, there are three different templates for Health & Wellness: one for Email, one for SNMP, and one for Syslog

  You cannot modify out-of-the-box templates. You must create a duplicate and modifying the copy.



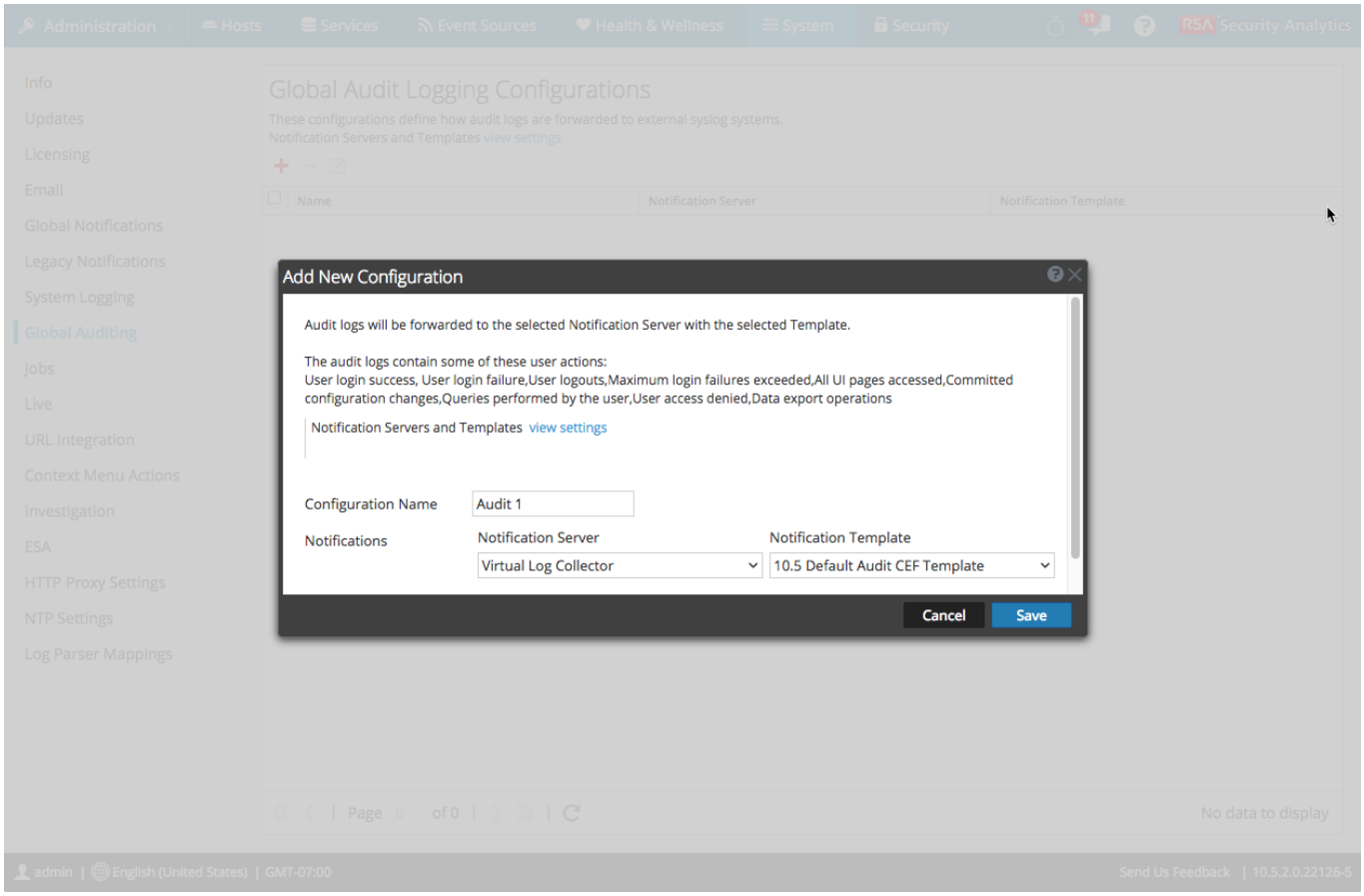
Template Type	Description	Image
Audit Logging	Template used for audit logging, which only supports syslog notifications. There are two out-of-the-box kinds: **A CEF template** - The format specified by this template can be parsed by SA's CEF parser, but its format is difficult for a human to read. **A human-readable format** - If you are sending audit logs to a syslog server or other application for which humans will be reading raw audit logs, use this.	
Event Stream Analysis	Includes templates for email, snmp, syslog, and script. It is the only SA component that supports script notifications.	
Event Source Monitoring	Templates for Event Source Monitoring. Supports email, snmp, and syslog notifications.	
Health Alarms	Templates for Health & Wellness policy alarms. Supports email, snmp, and syslog notifications.	

How Do I Use Notifications?

Notifications themselves are configured from within each applicable UI component. The workflow for configuring a notification is roughly the same within each component:

1. Click the ➕ Button and select the notification type (email, syslog, snmp, or script).
2. From the **Notification** column, choose an Output which you defined in Global Notifications.
3. From the **Notification Server** column, you guessed it, choose a Global Notification Server.
4. From the **Template** column, choose a Template you defined for this component in Global Notifications.
5. Click **Apply** or **Save.**

Here are a few screenshots of notifications within their respective UI components.



Audit Log Notifications

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Alarms | Monitoring | Policies | System Stats Browser | Event Source Monitoring | Settings

Policies

- Archiver (1)
- Broker (1)
- Concentrator (1)
- Decoder (1)
- Event Stream Analysis (1)
- Host (1)
  - *SA Host Monitoring Policy*
- IPDB Extractor (1)
- Incident Management (1)
- Log Collector (1)
- Log Decoder (1)
- Malware Analysis (1)
- Reporting Engine (1)
- Warehouse Connector (1)
- Workbench (1)

Rules

Define the conditions under which you want to trigger an alarm for the Security Analytics health problems (definition includes severity, statistic the alarm applies to, threshold, and threshold at which the alarm clears). After you define the alarm rule, enable or disable the alarm.

Enable	Name	Severity	Category	Statistic	Threshold
	Critical Usage o...	Critical	FileSystem	Mounted Filesystem Disk Usage Per...	Alarm > 75% for 0 MINUTES
	Filesystem is Full	Critical	FileSystem	Mounted Filesystem Disk Usage Per...	Alarm = 100% for 0 MINUTES
	High Filesystem...	High	FileSystem	Mounted Filesystem Disk Usage Per...	Alarm > 95% for 0 MINUTES
	High System Sw...	High	SystemInfo	Swap Utilization	Alarm > 50% for 5 MINUTES
	High Usage on ...	High	FileSystem	Mounted Filesystem Disk Usage Per...	Alarm > 60% for 0 MINUTES
	Host Unreacha...	Critical	ProcessInfo	Appliance Down	Alarm = True for 0 MINUTES
☑	**Power Supply ...**	**High**	**SystemInfo**	**Power Supply Status**	**Alarm !Regex Match ^ok$\|^ns$|^Presence detected$...**
	RAID Logical Dri...	High	DiskRaid	Logical Drive State	Alarm Regex Match .*Degraded.* for 0 MINUTES
	RAID Logical Dri...	Critical	DiskRaid	Logical Drive State	Alarm Regex Match .*Offline.*|.*Failed.*|.*Unknown.* f...
	RAID Logical Dri...	Medium	DiskRaid	Logical Drive State	Alarm Regex Match .*Rebuild.* for 0 MINUTES
	RAID Physical D...	Critical	DiskRaid	Physical Drive State	Alarm !Regex Match .*Online.*|.*Hotspare.*|.*Rebuild....
	RAID Physical D...	High	DiskRaid	Physical Drive Predictive Failure Cou...	Alarm > 1 for 0 MINUTES
	RAID Physical D...	Low	DiskRaid	Physical Drive State	Alarm Regex Match .*Rebuild.* for 0 MINUTES
	RAID Physical D...	High	DiskRaid	Physical Drive State	Alarm Contains Unconfigured(good) for 0 MINUTES
	SD Card Failure	Critical	SystemInfo	SD Card Status	Alarm != ok for 0 MINUTES

Policy Suppression

Define the time periods to suppress policies.

Time Zone: UTC (GMT+00:00)

Days	Time Range
Policy suppression is not allowed	

Notification

Notify responsible parties when the alarm triggers. Choose each notification type and destination here.

Notification Settings

Output	Recipient	Notification Server	Template
EMAIL	**Tim Underhay**	**GSuite**	**Health & Wellness Default SMTP Template**

admin | English (United States) | GMT-07:00 — Send Us Feedback | 10.5.2.0.22126-5

Health & Wellness Notifications

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Manage | Monitoring Policies

Groups

Order	Group Name
1	**All Unix Event Source(s)**
2	All Windows Event Source(s)
3	Critical Windows Event Source(s)
4	PCI Event Source(s)
5	Quiet Event Source(s)

Monitoring Policy for **All Unix Event Source(s)**

Save

☑ Enable — Last Modified **2016-10-09 11:10:51**

Thresholds

Define a low threshold or high threshold or both.

Low Threshold	High Threshold
< 0 events in 60 Minutes	> Number events in Time Minutes

Notifications

Notify responsible parties when the alarm triggers. Choose each notification type and destination here.

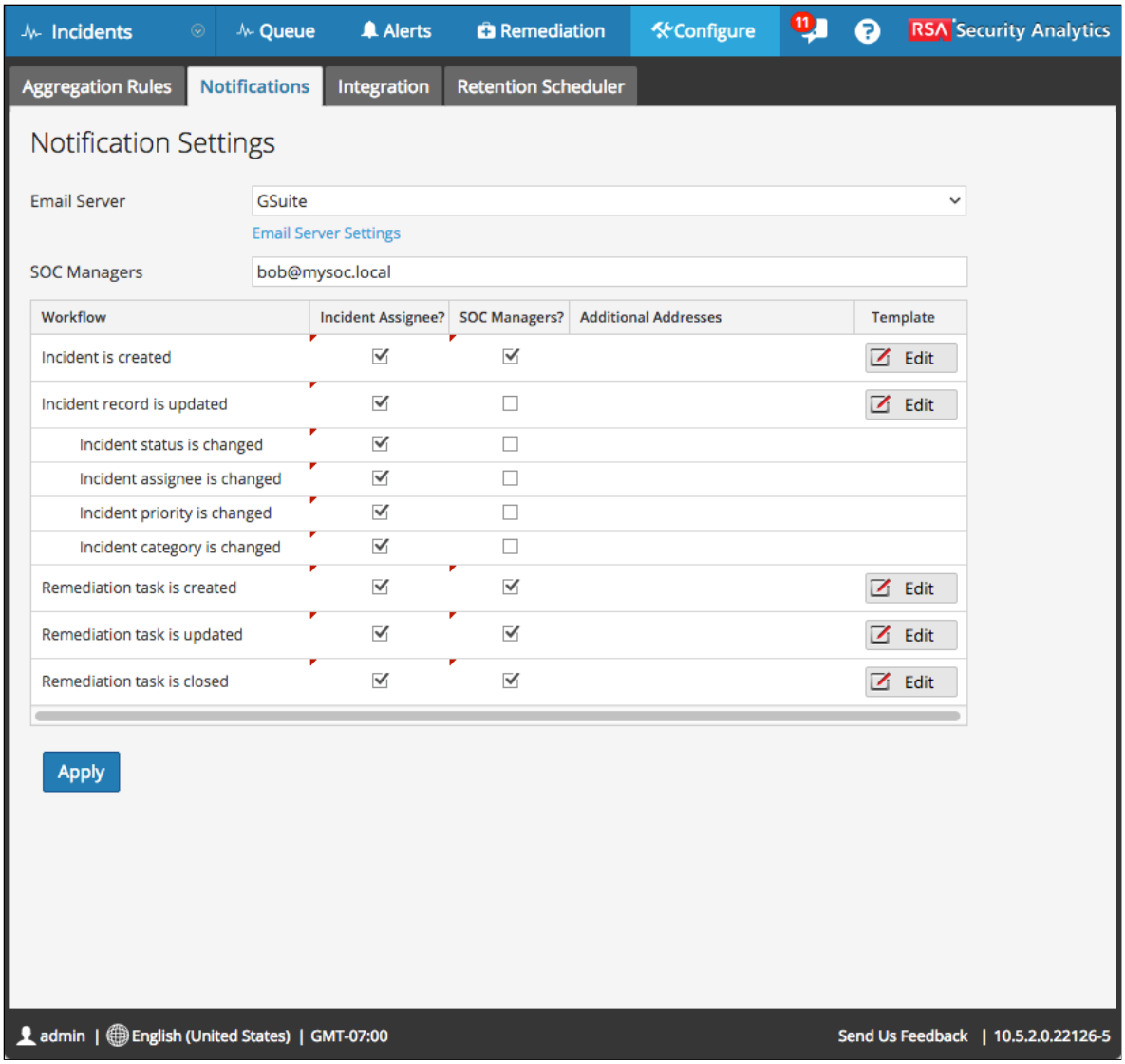
Notification Settings

Output	Recipient	Notification Server	Template
EMAIL	**Tim Underhay**	**GSuite**	**ESM Default Email Template**

☑ Output Suppression of every 60 minutes

admin | English (United States) | GMT-07:00 — Send Us Feedback | 10.5.2.0.22126-5

Event Source Monitoring Notifications



Incident Management Notifications

The IM Exception

As you can see from the screenshot above, Incident Management notifications use a different workflow than the other components, but uses a Global Notifications Email Server. This workflow be covered in the lesson on Incident Management.

Legacy Notifications

"Hodge-podge" is a good term to describe legacy notifications. It's really a collection of disparate user experiences for different areas of the UI. RSA is trying to eliminate these incongruous experiences, but haven't managed to do so yet.

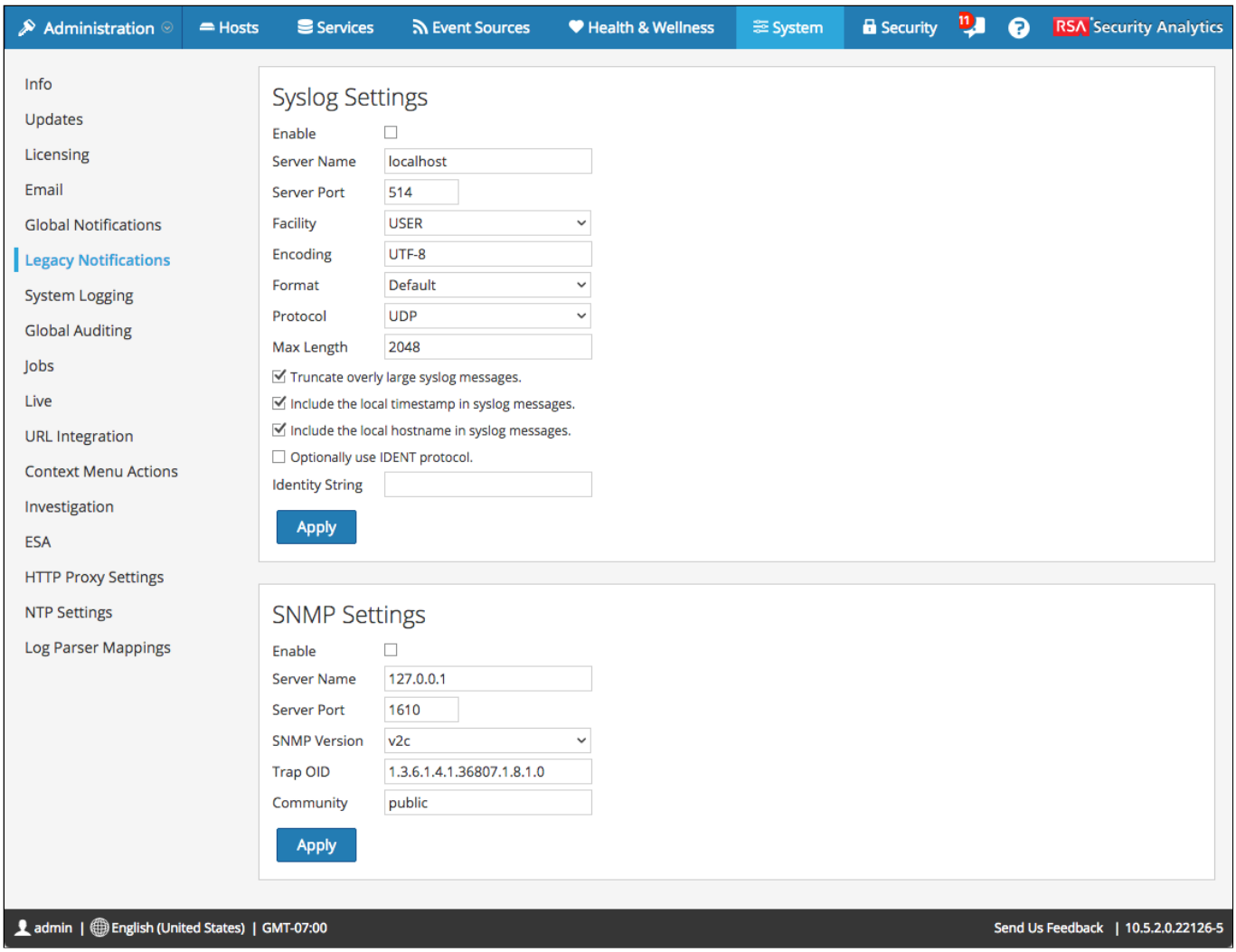
These are the different UI areas which still make use of legacy notifications:

- Licensing
- Legacy Event Source Monitoring
- Legacy Health & Wellness
- Live
- Password Expiration
- Reporting Engine (it has its very own experience)

Configuring Legacy Notifications

Legacy syslog and SNMP server and output settings are configured in **Administration->System->Legacy Notifications**, while legacy email server settings are configured under **Administration->System->Email**.

Legacy notifications do not utilise the concept of templates, but rather the body of output is hard-coded into every application that sends through these mechanisms. They also do not generally have the ability to define recipients for each UI area but rather will be blasted out to every recipient defined in **Administration->System**. This excludes password expiry emails, which are sent to the email address defined for the user whose password is about to expire. Each area of SA which uses Legacy Notifications will only give you the ability to toggle each notification type on or off, but nothing more (excepting the Reporting Engine).



Legacy Notification Syslog and SNMP Trap Settings

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

- Info
- Updates
- Licensing
- Email
- Global Notifications
- Legacy Notifications
- System Logging
- Global Auditing
- Jobs
- Live
- URL Integration
- Context Menu Actions
- Investigation
- ESA
- HTTP Proxy Settings
- NTP Settings
- Log Parser Mappings

Email Server Settings

Mail Server	mail.google.com
Server Port	25
SSL	☐
From Address	do-not-reply@netwitness.com
No Authentication	☐
Username	
User Password	********
Notification Addresses	tim.underhay@kensingtontechlimited.co.uk

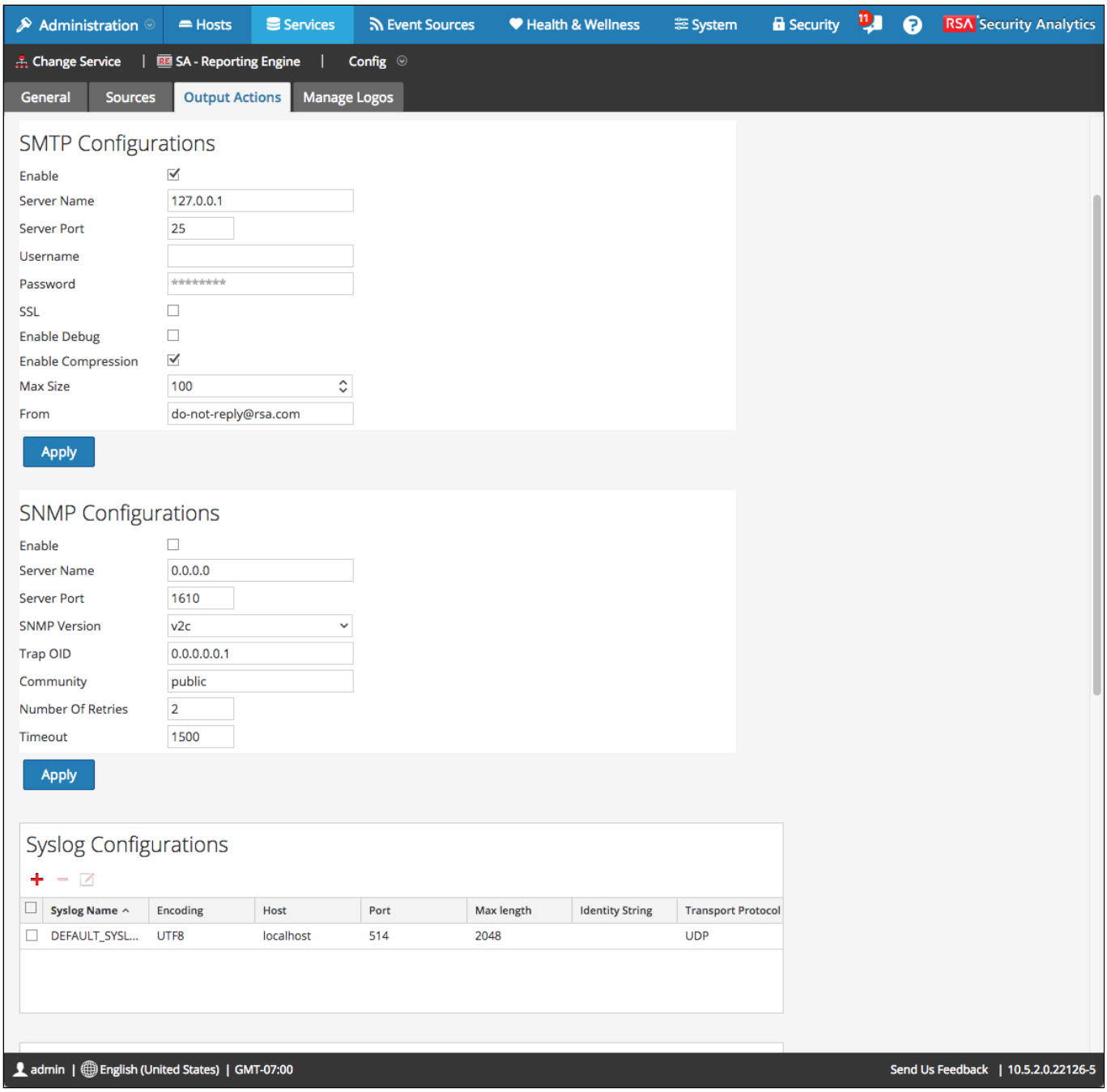
Apply | Test Connection

Email Statistics

Name	Value
Successful operations	0
Last successful operation	Never
Unsuccessful operations	0
Last unsuccessful operation	Never

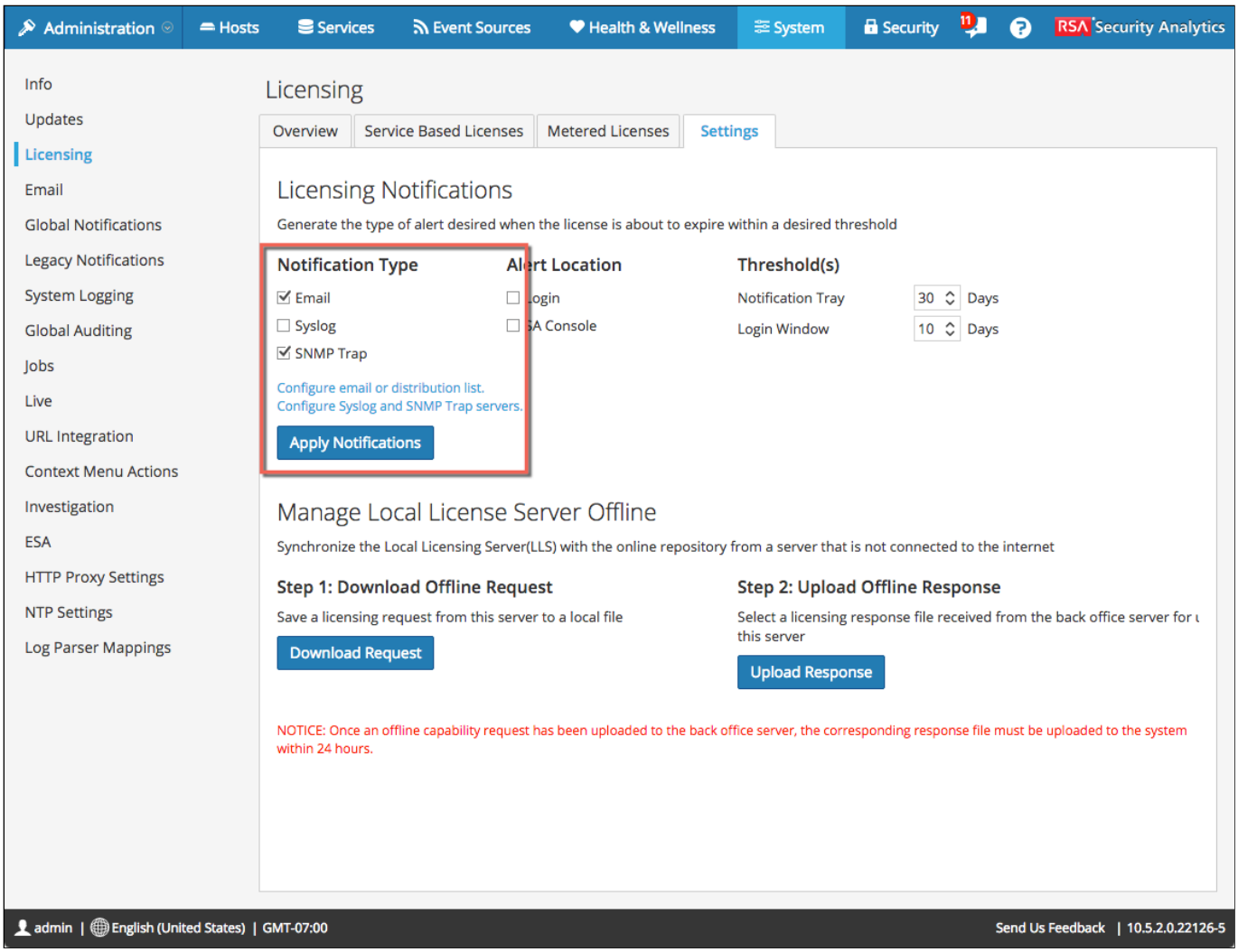
admin | English (United States) | GMT-07:00 Send Us Feedback | 10.5.2.0.22126-5

Legacy Email Notifications

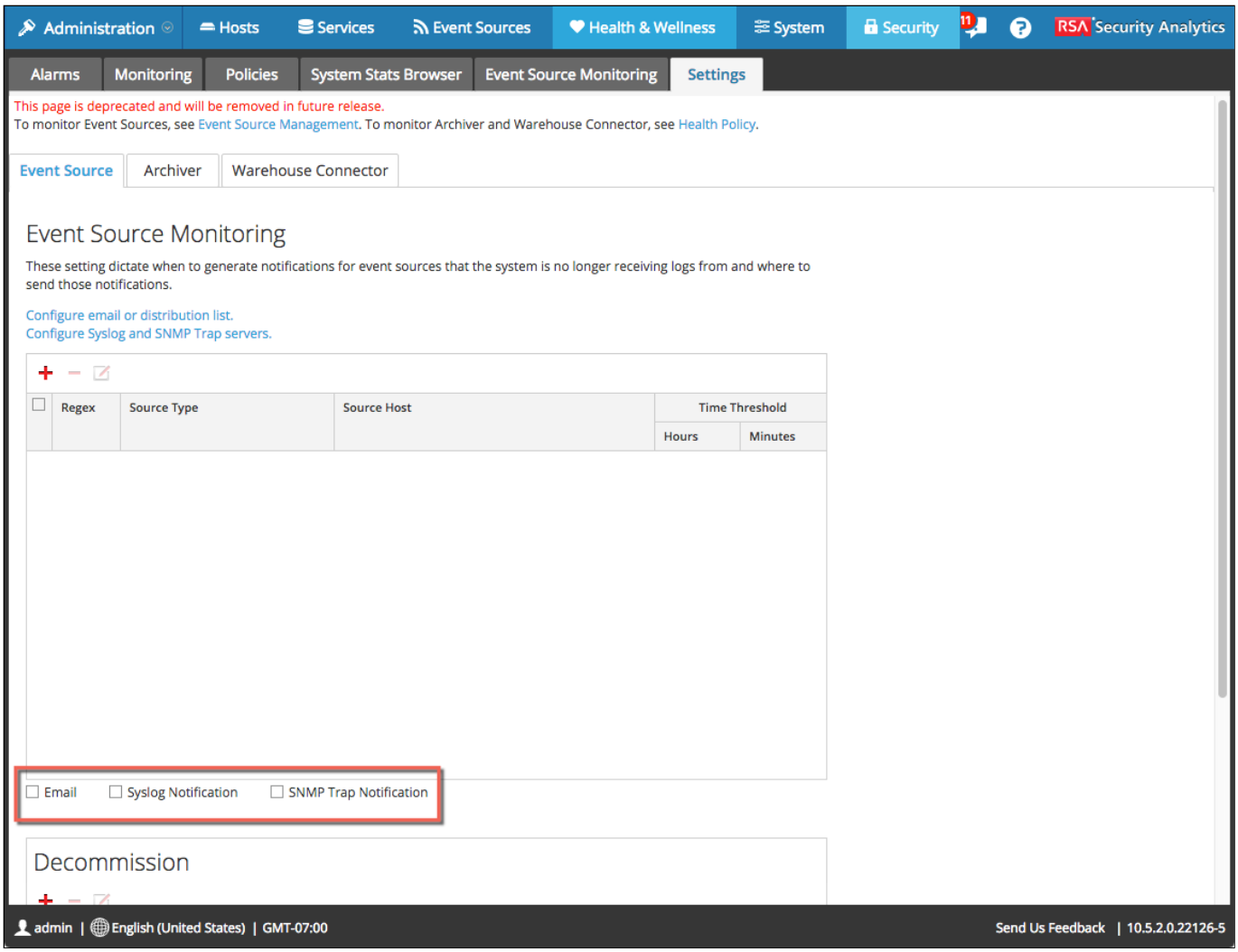


Reporting Engine Notifications

Licensing Notifications



Legacy Event Source Monitoring Notifications



Legacy Health & Wellness Notifications

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Alarms | Monitoring | Policies | System Stats Browser | Event Source Monitoring | Settings

This page is deprecated and will be removed in future release.
To monitor Event Sources, see Event Source Management. To monitor Archiver and Warehouse Connector, see Health Policy.

Event Source | Archiver | Warehouse Connector

Archiver Monitoring

These settings allow for notifications to be generated when critical thresholds concerning Archiver aggregation and storage have been met.

Aggregation Status

Notify After

For ☐ Failed ☐ Offline

Aggregation Connection

Notify After Failing For Minutes

Storage Connection

Notify After Failing For Minutes

Storage Capacity

Storage Threshold By ◉ Space ○ Time

When Storage Size Is % Full When Warm Storage Size Is % Full

Notification Type

Configure email or distribution list.
Configure Syslog and SNMP Trap servers.

☐ SA Console ☐ Email ☐ Syslog Notification ☐ SNMP Trap Notification

Apply Cancel

admin | English (United States) | GMT-07:00 Send Us Feedback | 10.5.2.0.22126-5

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Alarms | Monitoring | Policies | System Stats Browser | Event Source Monitoring | Settings

This page is deprecated and will be removed in future release.
To monitor Event Sources, see Event Source Management. To monitor Archiver and Warehouse Connector, see Health Policy.

Event Source | Archiver | Warehouse Connector

Warehouse Connector Monitoring

These settings allow for notifications to be generated when critical thresholds concerning Warehouse Connector and storage have been met.

Source or Destination Status

Notify Offline For [ ] Minutes

Stream Status

Notify Stopped For [ ] Minutes
Disk Is [ ] % Full
Source Is Behind [ ] Sessions
Rejected Folder Size Is [ ] % Full
Number Of Files In Permanent Failure Folder [ ]

Notification Type

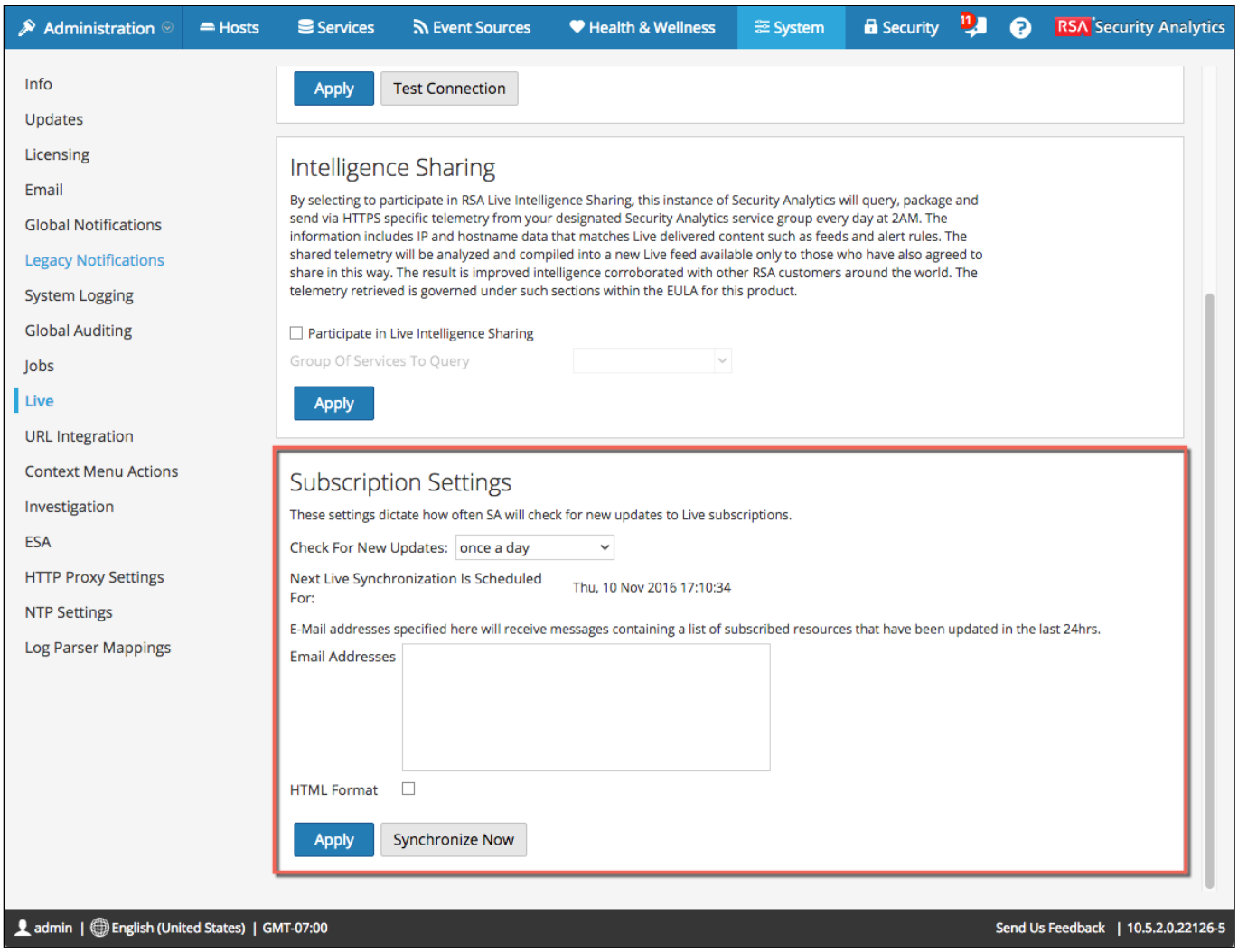
Configure email or distribution list.
Configure Syslog and SNMP Trap servers.

☐ SA Console ☐ Email ☐ Syslog Notification ☐ SNMP Trap Notification

Apply | Cancel

admin | English (United States) | GMT-07:00 Send Us Feedback | 10.5.2.0.22126-5

Live


Administration
Hosts
Services
Event Sources
Health & Wellness
System
Security
RSA Security Analytics
Info
Updates
Licensing
Email
Global Notifications
Legacy Notifications
System Logging
Global Auditing
Jobs
Live
URL Integration
Context Menu Actions
Investigation
ESA
HTTP Proxy Settings
NTP Settings
Log Parser Mappings
Apply
Test Connection
Intelligence Sharing
By selecting to participate in RSA Live Intelligence Sharing, this instance of Security Analytics will query, package and send via HTTPS specific telemetry from your designated Security Analytics service group every day at 2AM. The information includes IP and hostname data that matches Live delivered content such as feeds and alert rules. The shared telemetry will be analyzed and compiled into a new Live feed available only to those who have also agreed to share in this way. The result is improved intelligence corroborated with other RSA customers around the world. The telemetry retrieved is governed under such sections within the EULA for this product.
Participate in Live Intelligence Sharing
Group Of Services To Query
Apply
Subscription Settings
These settings dictate how often SA will check for new updates to Live subscriptions.
Check For New Updates:
once a day
Next Live Synchronization Is Scheduled For:
Thu, 10 Nov 2016 17:10:34
E-Mail addresses specified here will receive messages containing a list of subscribed resources that have been updated in the last 24hrs.
Email Addresses
HTML Format
Apply
Synchronize Now
admin | English (United States) | GMT-07:00
Send Us Feedback | 10.5.2.0.22126-5

22. SA Security

The SA security model can be a bit tricky, but we'll aim to help you understand it here. It's important for administrators to understand the security model of SA, as it can greatly impact the functionality of SA. As the data which SA contains is highly sensitive, it can also affect your organisation's compliance with laws and regulations.

Terminology

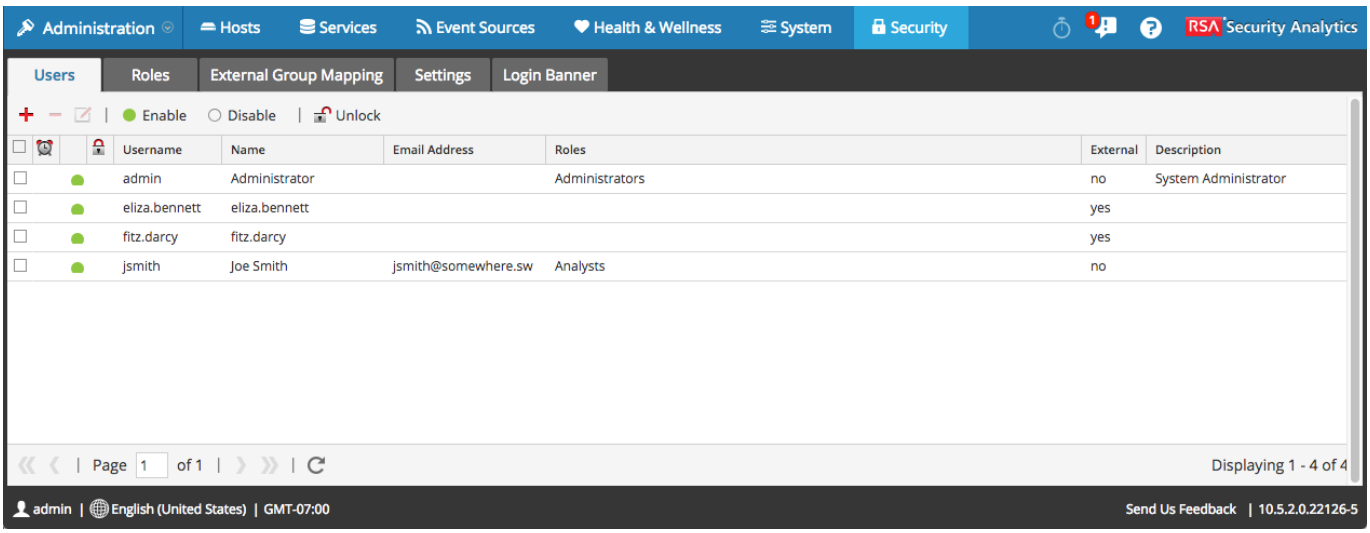
Firstly, let's define some common terminology that will enable us to discuss SA security.

Connections

A connection is the transport mechanism whereby one program connects to another program. SA is oriented around connections, whether it be from a user's web browser to the UI service, or whether it's from one system service to another system service, such as in aggregation.

Users

Users are those who initiate connections. These can be either humans, or they themselves can be programs (system services).



Authentication (Auth-N)

Authentication, or Auth-N for short, the first part of the connection process which answers the question: "is this user who he says he is?". In theory, a user can prove this based on one or many factors, like a password, a security token, or a certificate. If the answer to this question is no, then the connection process will not proceed further.

SA is capable of authenticating users against these back-ends:

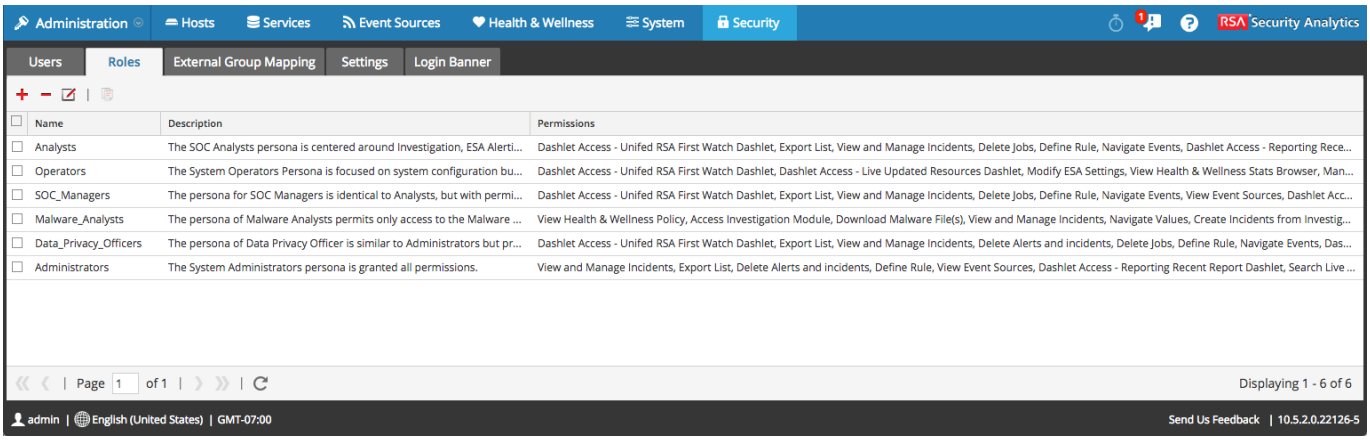
- **Local** - User is defined within SA. The default.
- **Active Directory** - User is defined within a Windows Active Directory Domain. The UI comes with native support for this, which uses LDAP under-the-bonnet. Configuration of this will be discussed in a separate lesson.
- **PKI** - Mainly used by military and the defense industries, PKI authentication is used for smartcard authentication. Configuration of PKI auth is not within the scope of this course.
- **PAM** - Pluggable Authentication Module. This is an OS-provided subsystem for using third-party modules for authentication. The most common uses are for RSA SecurID and for an OpenLDAP backend. This is usually the most difficult to configure, and should be avoided if one of the above options is available to you. This also makes use of the NSS system library. Configuration of this is not within the scope of this course.

Authorization (Auth-Z)

Authorization, or Auth-Z, is the second part of the connection process which answers the question: "Now that user X has proven who he is, what should he be able to do with the system?". Auth-Z is a process distinct from authentication.

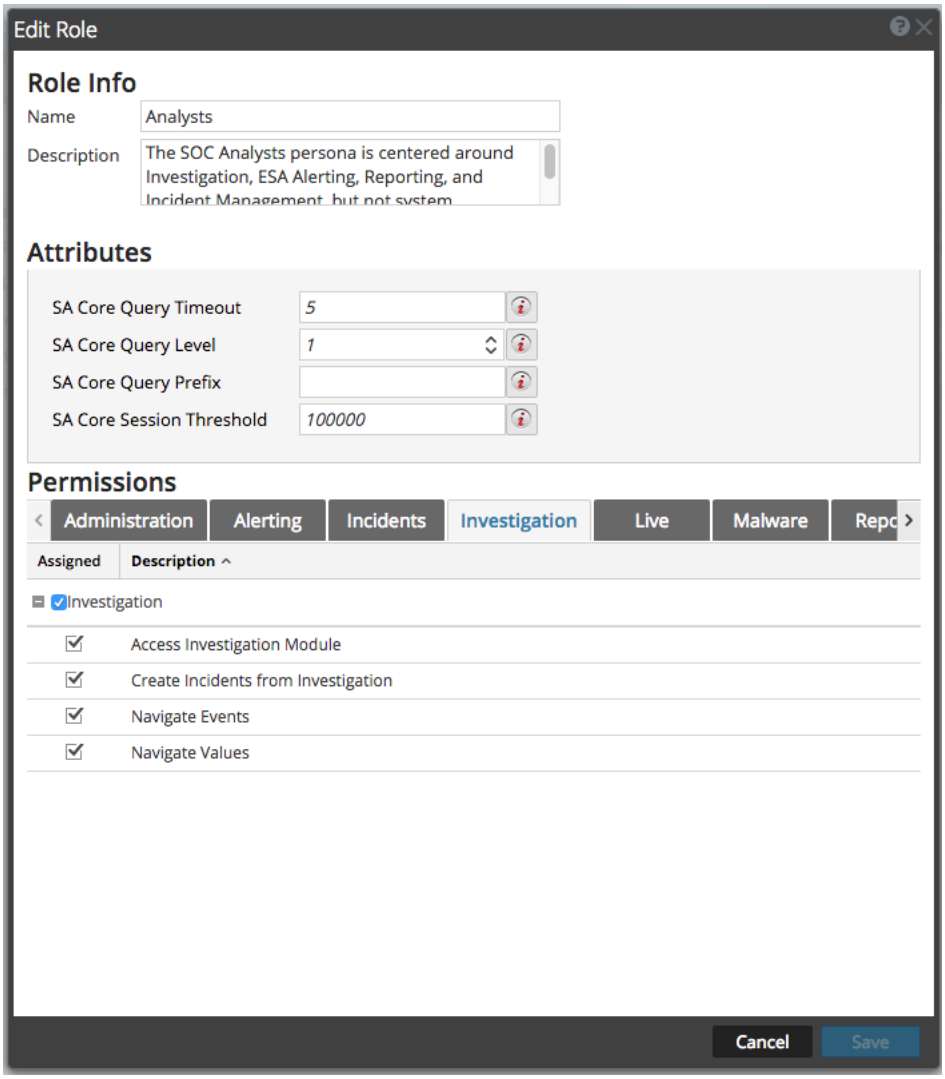
Roles

In SA, a Role is a logical grouping of permissions which help form the basis of authorization. For instance, from a high level, we can say that user X is assigned to role Analysts, therefore he will be allowed to access the Investigations features.



Permissions

Permissions are low-level objects which are the specific things which a role's members may be allowed to do. There are different permissions for the UI defined in global Roles. These are broken up into tabs, with each tab corresponding to a different section of the UI. There are also permissions for Core services, defined in each Core service.

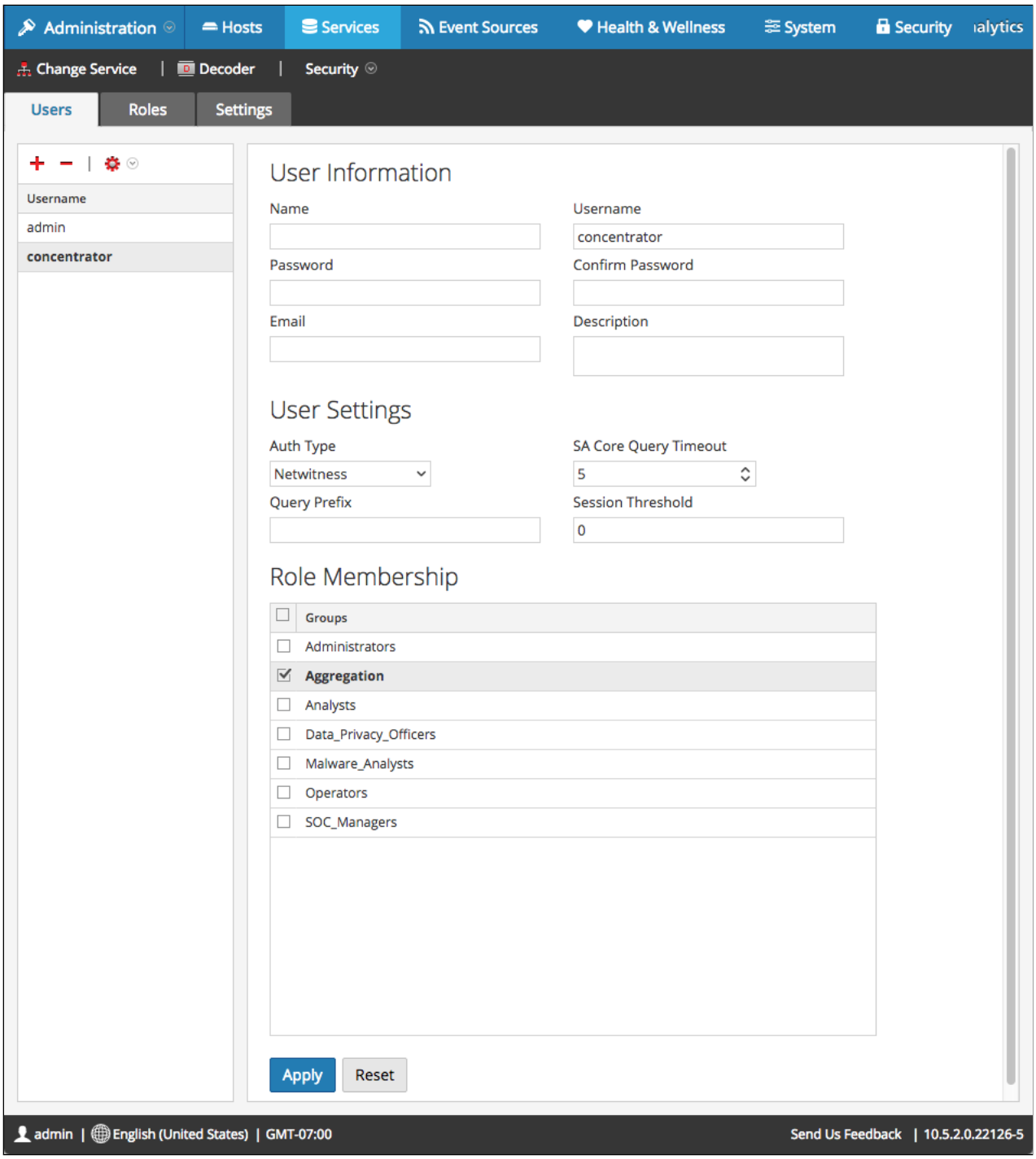


Where Are Users and Roles Defined in SA?

In SA, users and roles can be defined in one of two places: either centrally (on the UI server itself), or locally on each individual SA Core service, which possesses its own security database. Which gets used and when is described below in discussions of SA's two different security models, which most confusingly, operate at the same time.

Users and roles which are defined globally in SA can be found in **Administration->Security** in the UI (pictured above).

Users defined locally for each Core service are found in **Administration->Services->*<SomeCoreService>*->Security**.

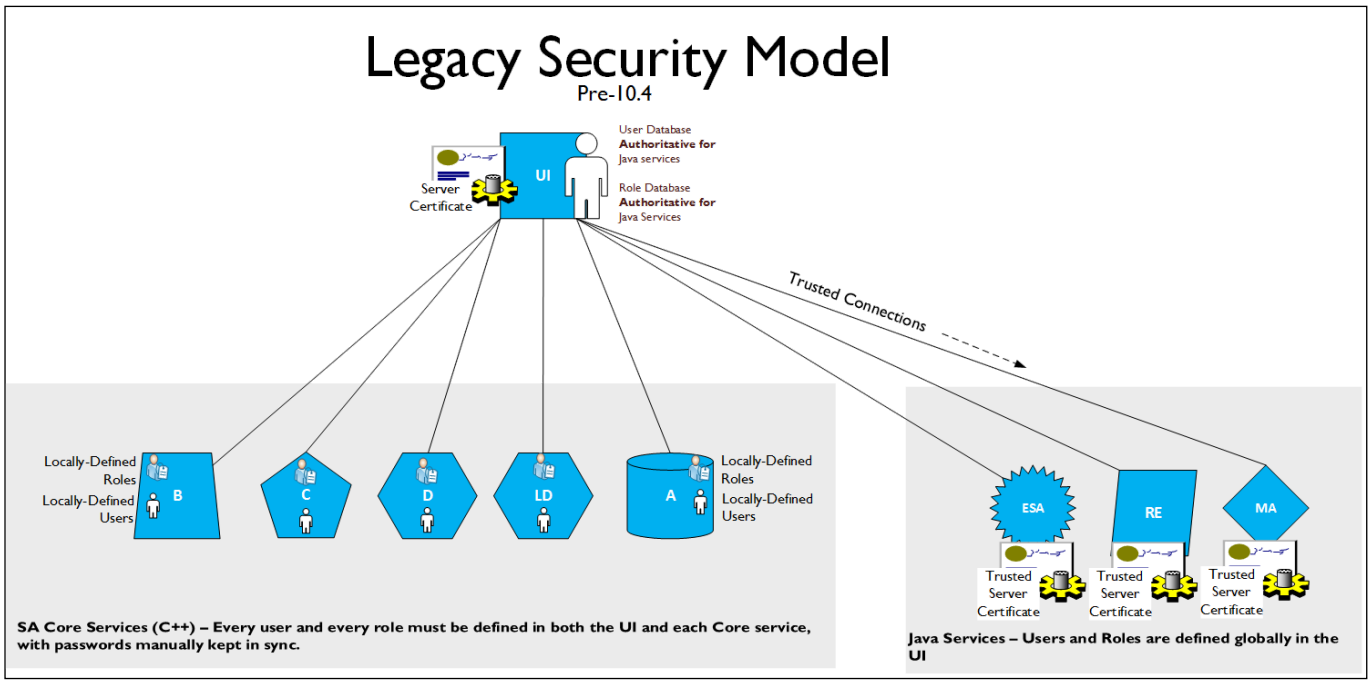


Legacy Distributed Security Model

The legacy security model is what was used in SA prior to version 10.4. In this model, every user must be defined locally in both the UI and also on every SA Core service. Every Core service was its own castle, essentially. It was a management nightmare, as user passwords often had to be kept in sync manually between Core services, and they could easily get out of sync if you weren't careful. And if using external authentication, such as with Active Directory, PAM had to be configured on every single Core service. Not fun at all.

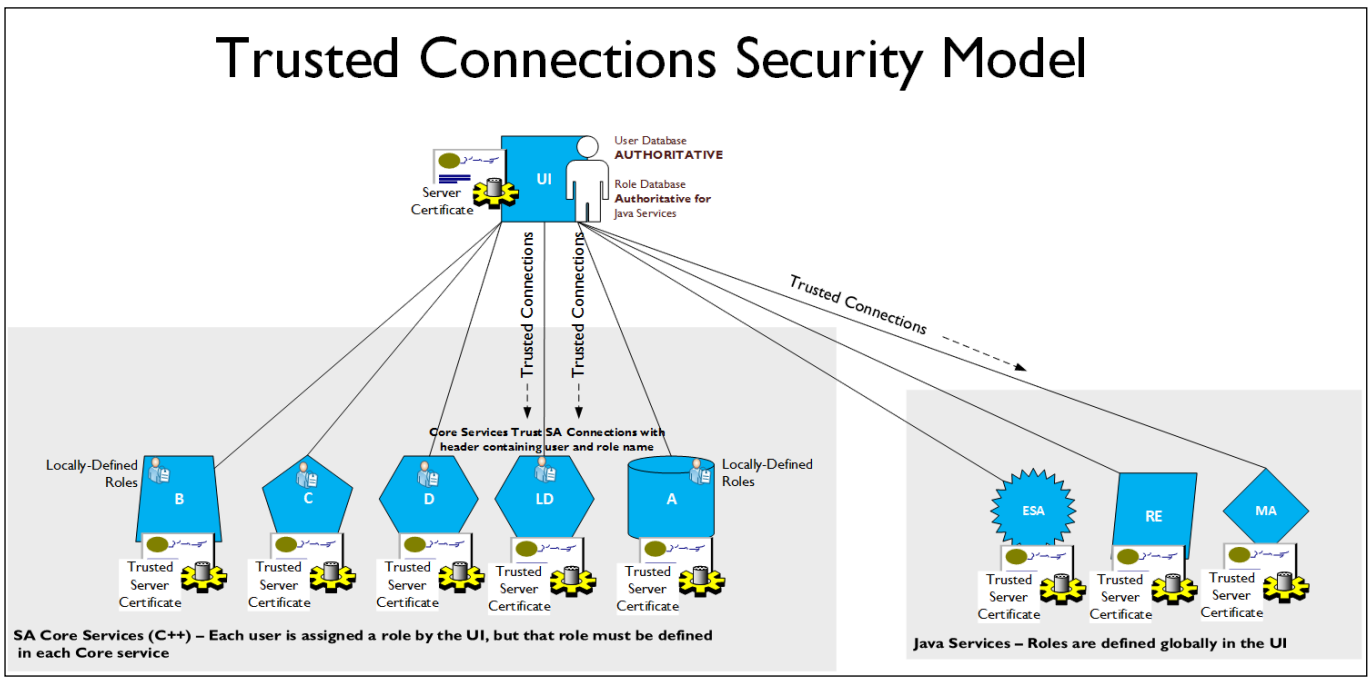
In the legacy security model, both authentication of the end user and authorization occur in both the UI and on every individual SA Core service.

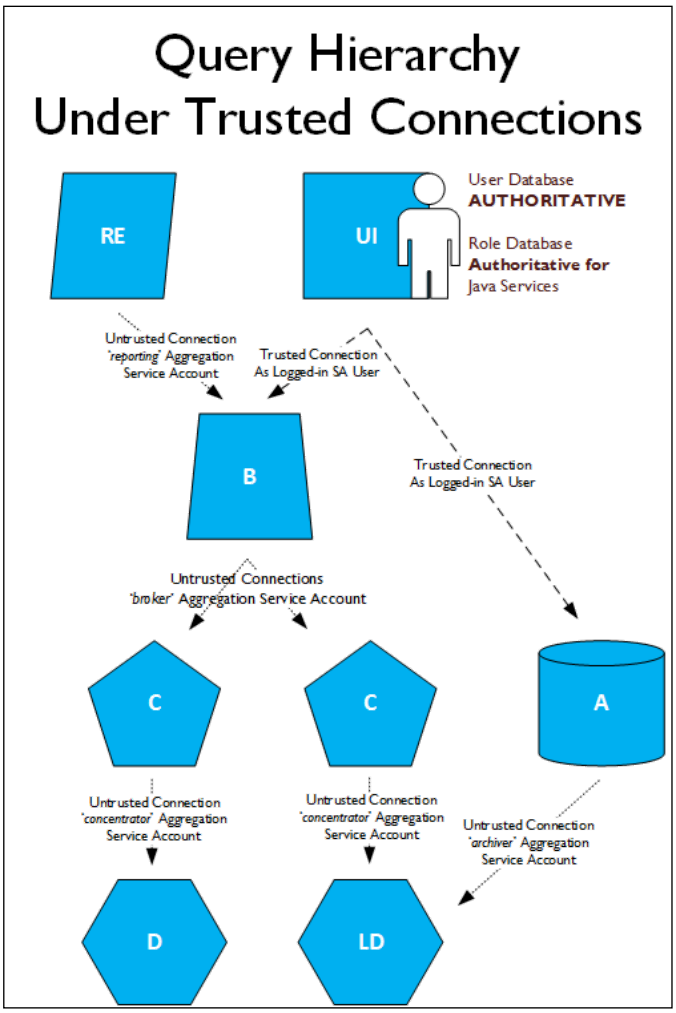
In the legacy model, only Java-based services like the Reporting Engine and Malware Analysis utilised the users and roles defined in globally the UE.



Trusted Connections Security Model

The Trusted Connection security model was introduced in Security Analytics 10.4, and it greatly simplified the management problems of the legacy security model. In the trusted connections model, users are defined globally in the UI, and each SA Core service is configured to trust connections which originate from the UI service. The process of authenticating the end user happens in the UI itself, so end users no longer need be defined locally in every Core service. However, the roles that the user belongs to must still be defined locally in each Core service. The reason for this is because when a user connects to a Core service from the UI, the process of authorization still occurs in the Core service.





Anatomy of a Trusted Connection

A trusted connection is simply an SA connection with a security header attached to it. This security header contains the following fields, which the Core service must honour, if the client certificate is trusted by the server:

Server Certificate	The public SSL certificate from the UI's java keystore for trusted connections. This is used to validate whether the Core service should trust the client. If it does not, the connection reverts to a legacy connection
Username	The username of the user connecting. This need not be defined locally in the Core service.
User Roles	A list of roles which the user belongs to. These must be defined within the Core service. If none of the roles are defined in the service, the connection attempt is defined.
Connection Attributes	These are things like query timeout, query level, query prefix, and session threshold.

A Mix: The Role of the Legacy Security Model Today

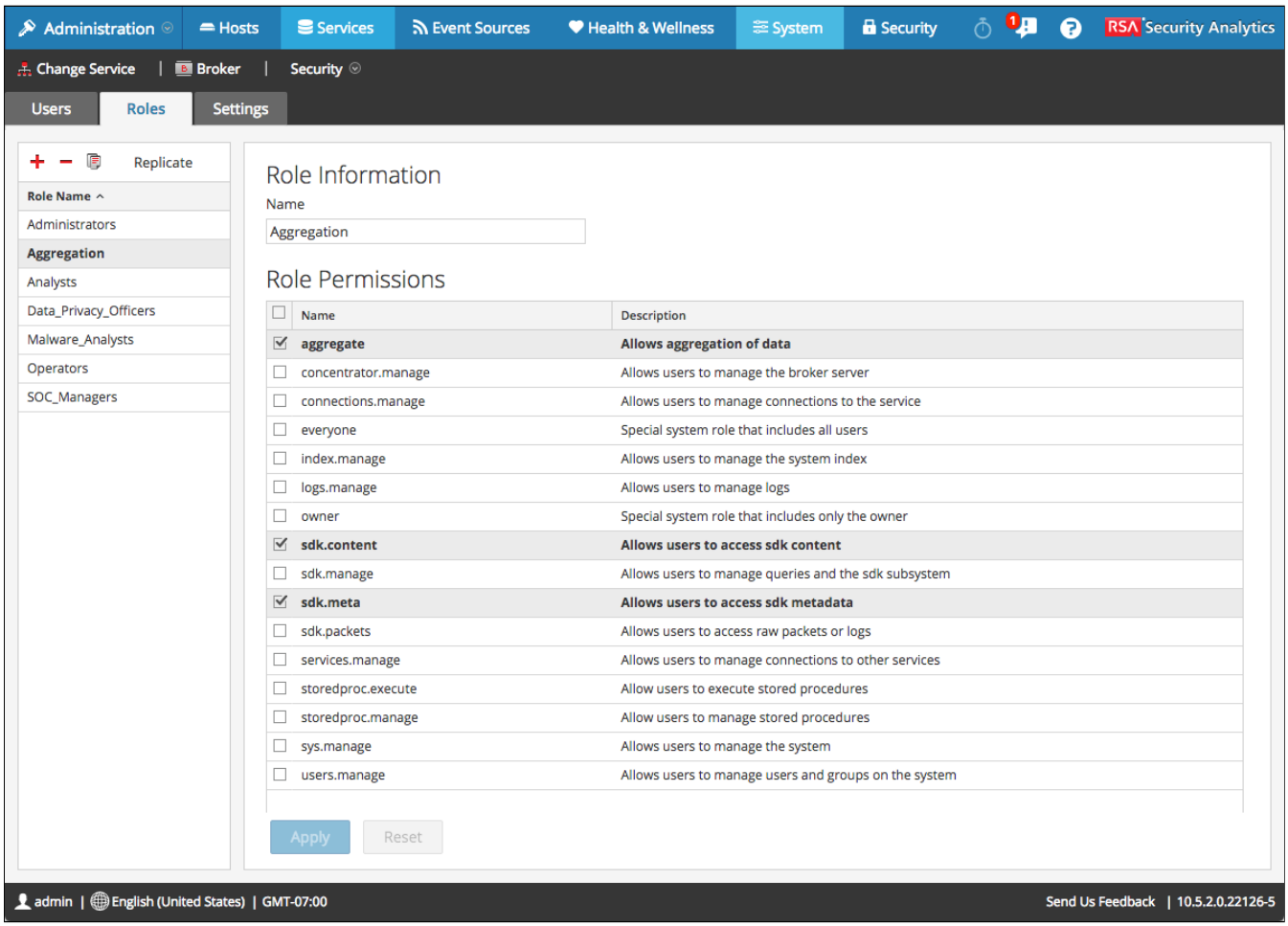
As noted before, the legacy security model still has its place in SA today, when working with these scenarios.

- **Aggregation Service Accounts** - When you have one Core service connecting to another Core service, this uses the Legacy Security Model. E.g., when a Broker aggregates sessions from Concentrators or other Brokers, it must connect using a locally-defined account on the upstream Core service. Or when a Concentrator aggregates session and meta data from upstream Decoders and Log Decoders, a local account on the Decoder must be used. For an aggregation connection like this, these are called **aggregation accounts,** or **service accounts**.

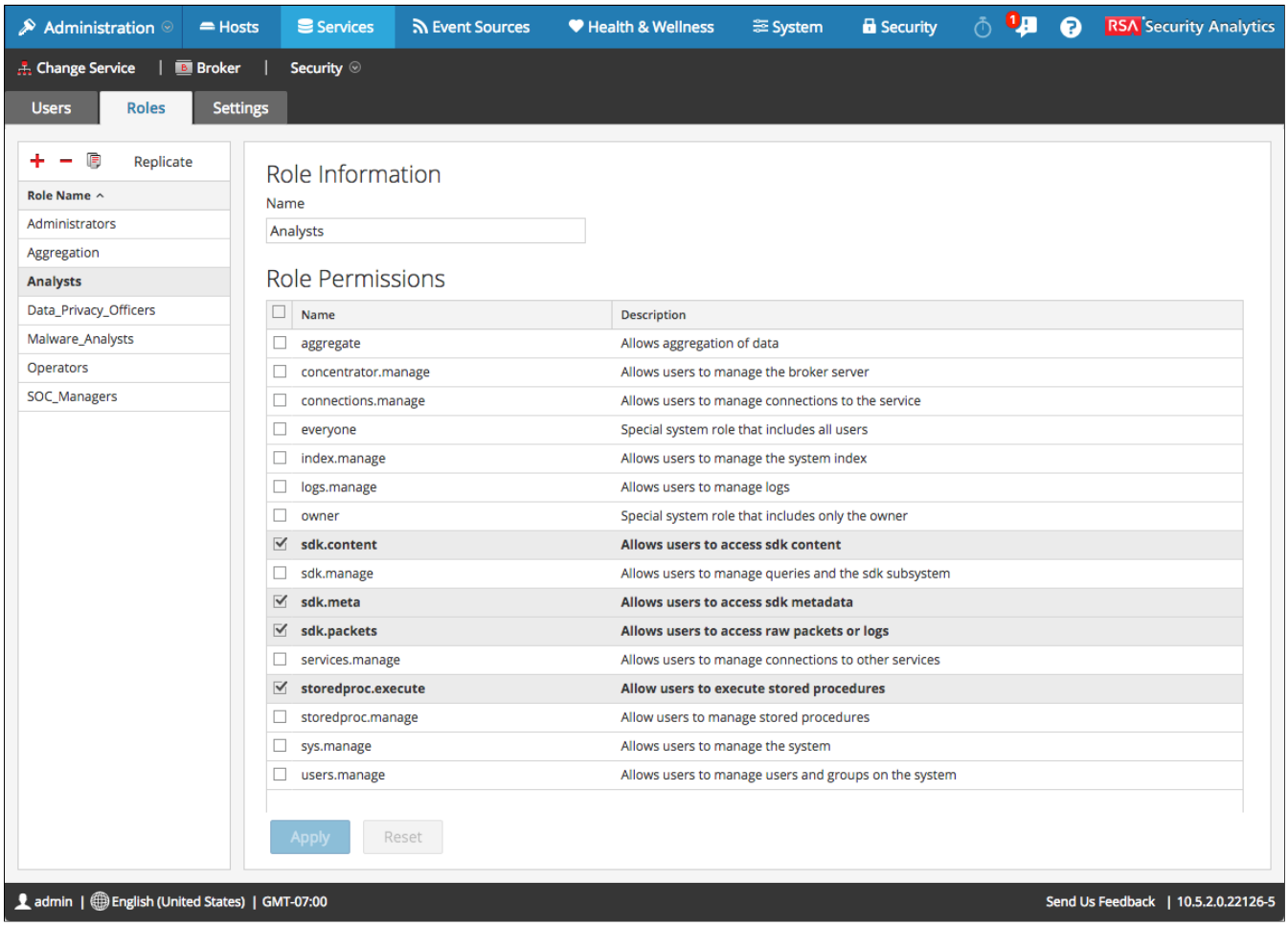
  Aggregation accounts are also used for performing queries through a downstream Core service. I.e. if a user fires off a query in Investigation to a Broker, it uses a trusted connection to connect to the Broker. The Broker, in turn, will distribute this query to all of its connected upstream services. When it proxies the query like this, the Broker will use its aggregation account, which is defined on each of the upstream services.

  ESA also uses a service account to aggregate its event data from Core services.

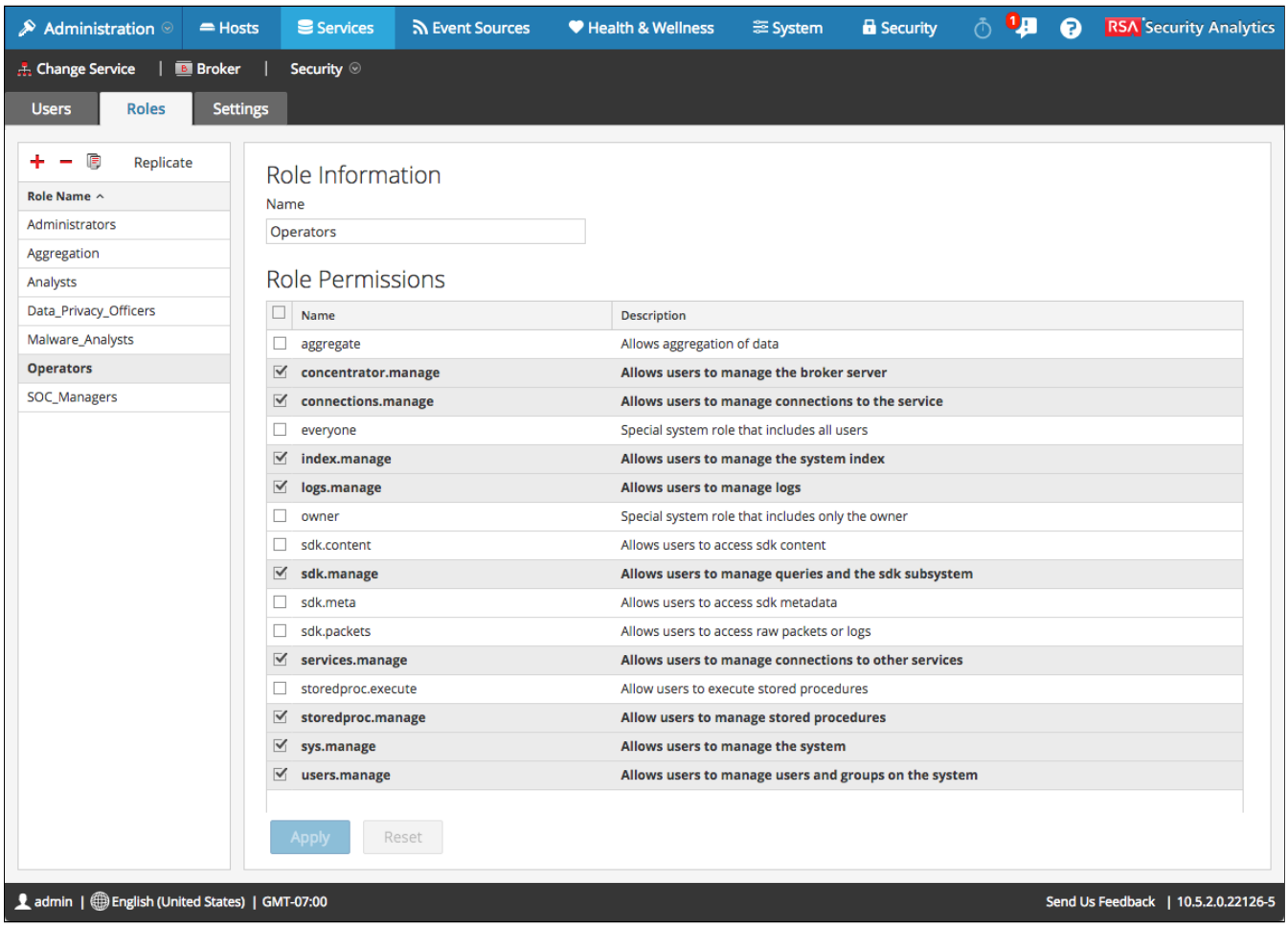
  Out-of-the-box, there is a role defined in most Core services called '**Aggregation**'. This has the minimum permissions necessary to perform aggregation.



- **Analyst Service Accounts** - The Reporting Engine uses an Analyst service account to query SA Core services. Connections from the legacy thick Investigator client also must use a service account. These accounts typically have the '**Analysts**' system role.



- **API Service Accounts** - These are used for making SDK calls from either NwConsole or the REST interface. The SDK is an extremely powerful way of working with SA from the command line or from a script. When issuing an API call, the connection must first be authenticated with a local service account. The role needed depends on the type of operation that's being performed. Most operations on the /sdk API node will require at least Analysts role permissions, whereas reading or modifying configuration settings will require Operators role permissions.



Common Service Accounts

These are the service accounts which I commonly create during a typical SA deployment:

Top Tip: Please, please, please - do NOT use the **admin** account for your service accounts!!!

Name	Defined On:	Purpose
concentrator	Decoders, Log Decoders	Aggregation from Concentrators
broker	Concentrators	Aggregation from Brokers
reporting	Concentrators, Brokers	NWDB queries from the Reporting Engine
esa	Concentrators	Aggregation account for ESA
archiver	Log Decoders	Aggregation account for Archiver

Out-of-the-Box Roles

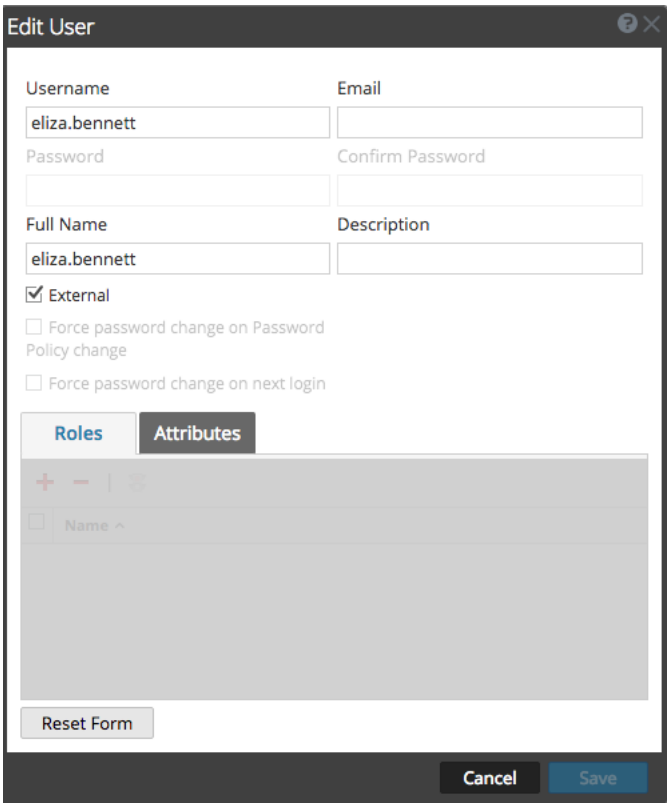
SA comes with some predefined roles which map to common user personas for SA. We'll borrow the description of each directly from the UI.

- **Administrators** - The System Administrators persona is granted all permissions.
- **Analysts** - The SOC Analysts persona is centered around Investigation, ESA Alerting, Reporting, and Incident Management, but not system configuration.
- **Operators** - The System Operators Persona is focused on system configuration but not Investigation, ESA Alerting, Reporting, and Incident Management.
- **SOC_Managers** - The persona for SOC Managers is identical to Analysts, but with permissions necessary to configure Incident Management.
- **Malware_Analysts** - The persona of Malware Analysts permits only access to the Malware Analysis module.
- **Data_Privacy_Officers** - The persona of Data Privacy Officer is similar to Administrators but primarily focused on configuring and viewing sensitive data.

External Users

Every AD or external user who logs into SA will automatically have a user account created for them under **Security->Users.** Their **external** flag will be set to **yes,** and the roles tab of their user profile will be greyed out, as the roles are mapped automatically for external users.

Top Tip: When using external authentication sources, case sensitivity in the username can matter. For instance, if you login as jsmith when the username is jSmith in Active Directory, the login will fail. If this is a problem, you can tick the box for '**Case Insensitive User Name**' in **Administration->Security->Settings**.



system.roles

System Roles, or system.roles, are a way of providing finer-grained control over access to individual meta keys on each Core service. With this, additional permissions appear in each Core security role, one for each meta key, under **sdk.meta.*X***, where X is the meta key name.

If the setting is configured to whitelist, in order to access specific meta keys, a user's role must contain the permission for the given key. If the setting is configured to blacklist, if a user's role has a a given **sdk.meta.*X*** permission selected, then the user will not be able to access that meta key. Administrative overhead is lower with blacklist, as it only requires subtracting a permission rather than having to add all desired permissions.

Activating system.roles

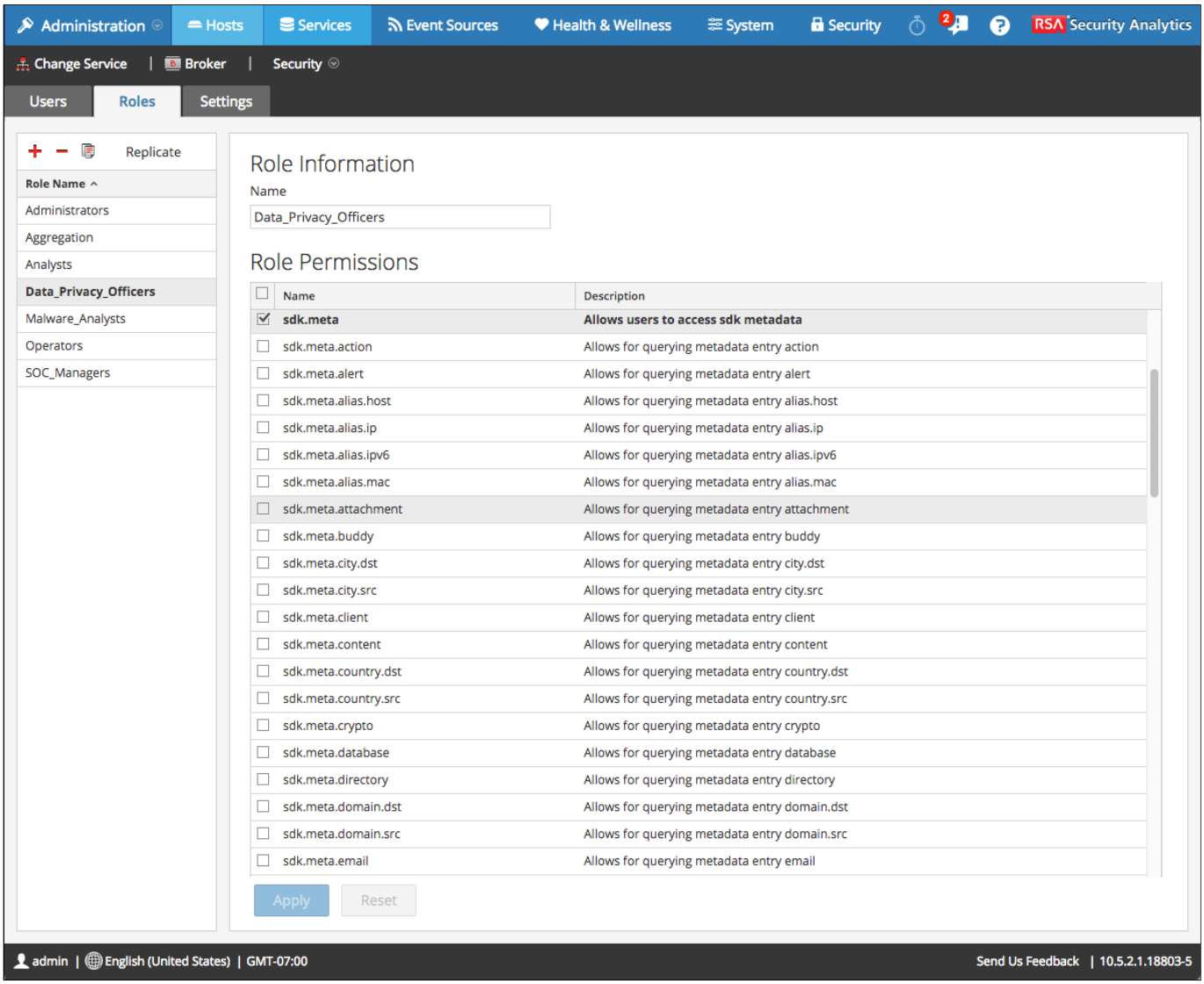
System roles are activated in the Explorer view of Core services.

1. In **Administration->Services**, select a service and go to **View->Explore**.
2. Navigate to the **/sdk/config** folder
3. Change the value of **system.roles** from 0 to one of these values:

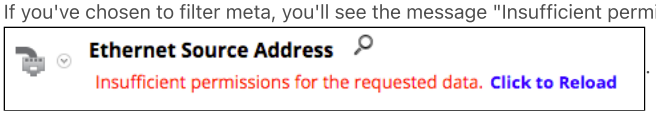
Value	Explanation
0	Disabled
1	Whitelist, with both meta and content filtered. If the sdk.meta.X permission for the given key is selected for a user's role, the user will be able to access both the meta of this key, and the raw content of sessions with this key (raw packets or logs). When viewing meta in Event Reconconstruction, only meta keys which have been whitelisted will be displayed.
2	Whitelist meta filtered. If the sdk.meta.X permission for the given key is selected for a user's role, the user will be able to access the meta of this key, but they will not be able to access the raw session content. When viewing meta in Event Reconstruction, only meta keys which have been whitelisted will be displayed.
3	Whitelist Content filtered. If the sdk.meta.X permission for the given key is selected for a user's role, the user will be able to access the raw content (either packets or logs) of sessions where all existing keys of the session have been whitelisted for the given role, and the user will be able to access all meta for the session.

4	Blacklist both meta and content. If the sdk.meta.X permission for the given key is selected for a user's role, the user will neither be able to access the session meta for this key, nor the raw content of sessions with this key (raw packets or logs). When viewing meta in Event Reconstruction, meta keys which have been blacklisted will not be displayed.
5	Blacklist meta filtered. If the sdk.meta.X permission for the given key is selected for a user's role, the user will not be able to access the meta of this key, but they will be able to access the raw session content. When viewing meta in Event Reconstruction, meta keys which have been blacklisted will not be displayed.
6	Blacklist Content filtered. If the sdk.meta.X permission for the given key is selected for a user's role, then the user will not be able to access the raw content (either packets or logs) of sessions where this key exists, but they will be able to access all meta for the session.

4. Under **Administration->Services->*<ChooseAService>*->Config->Security->Roles,** choose a role and then select the **sdk.meta.*X*** keys you want to either whitelist or blacklist, depending on how you've configured **/sdk/config/system.roles**.

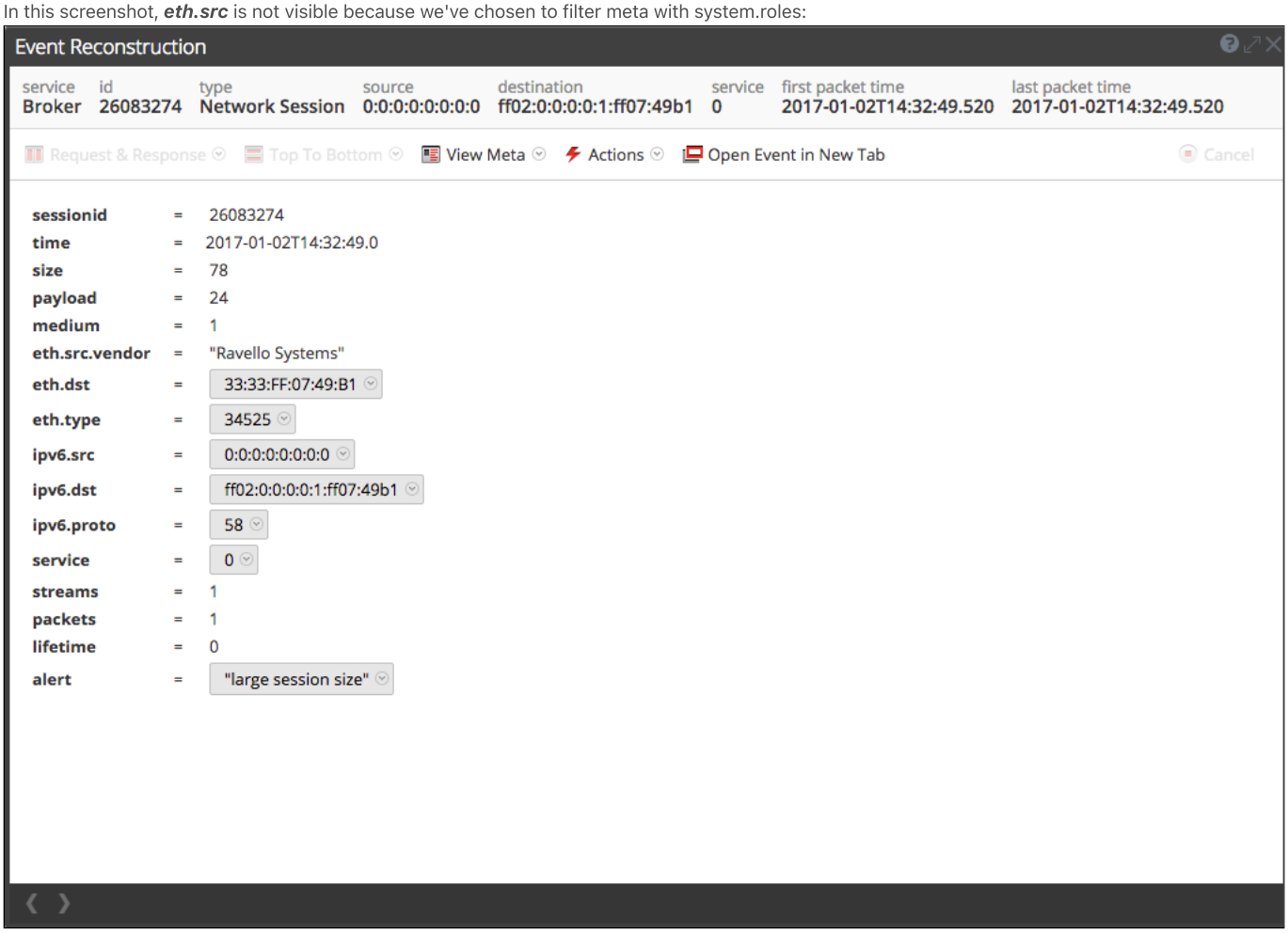


If you've chosen to filter meta, you'll see the message "Insufficient permissions for the requested data" in Investigation:

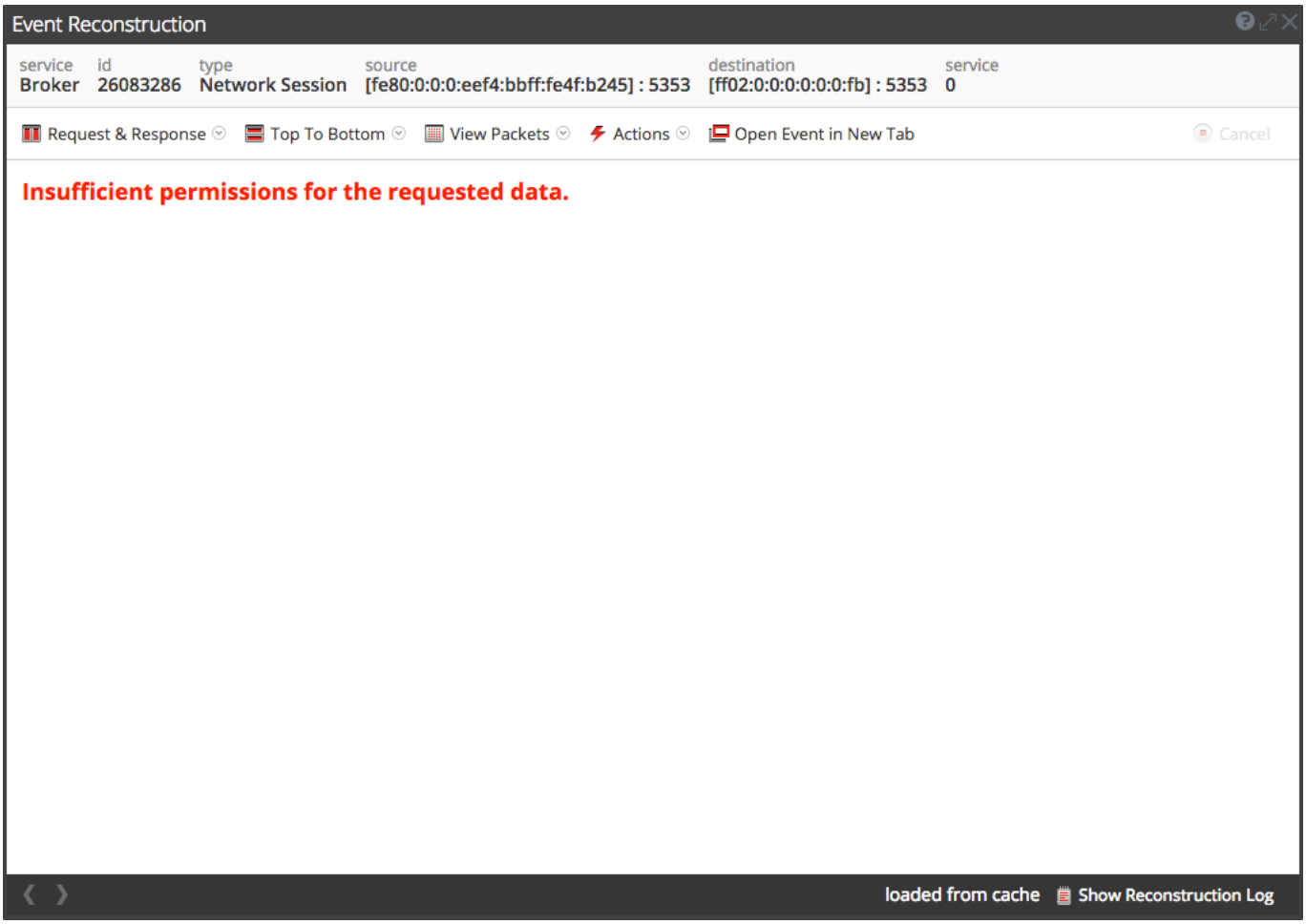


. The meta for the selected key will also not be visible in Session Reconstruction

In this screenshot, ***eth.src*** is not visible because we've chosen to filter meta with system.roles:

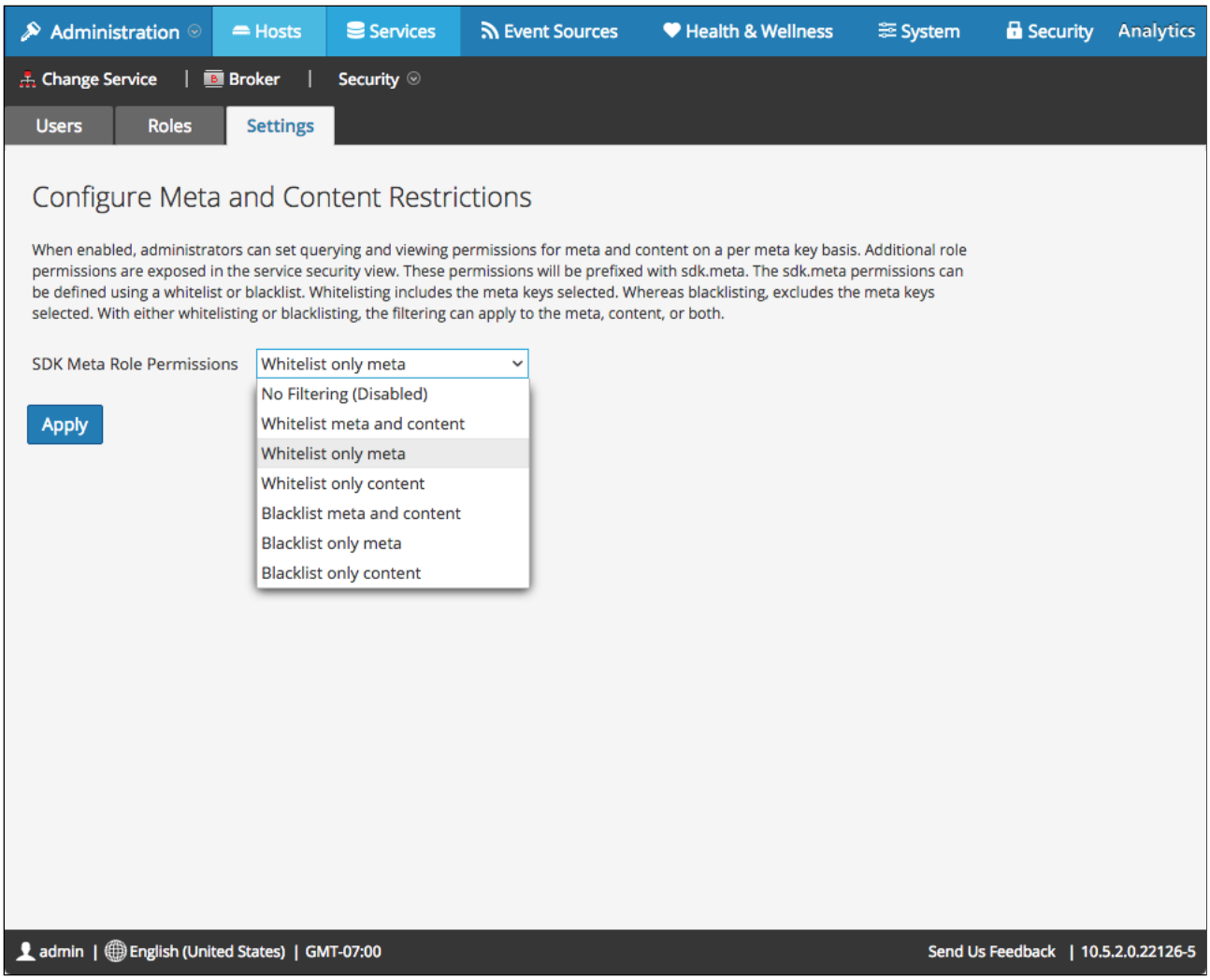


If we've chosen to filter content for a given meta key, when we try to access the raw content (packets, hex, text, web, mail, or files) for a session where this key exists, then we'll again get "Insufficient permissions for the requested data":



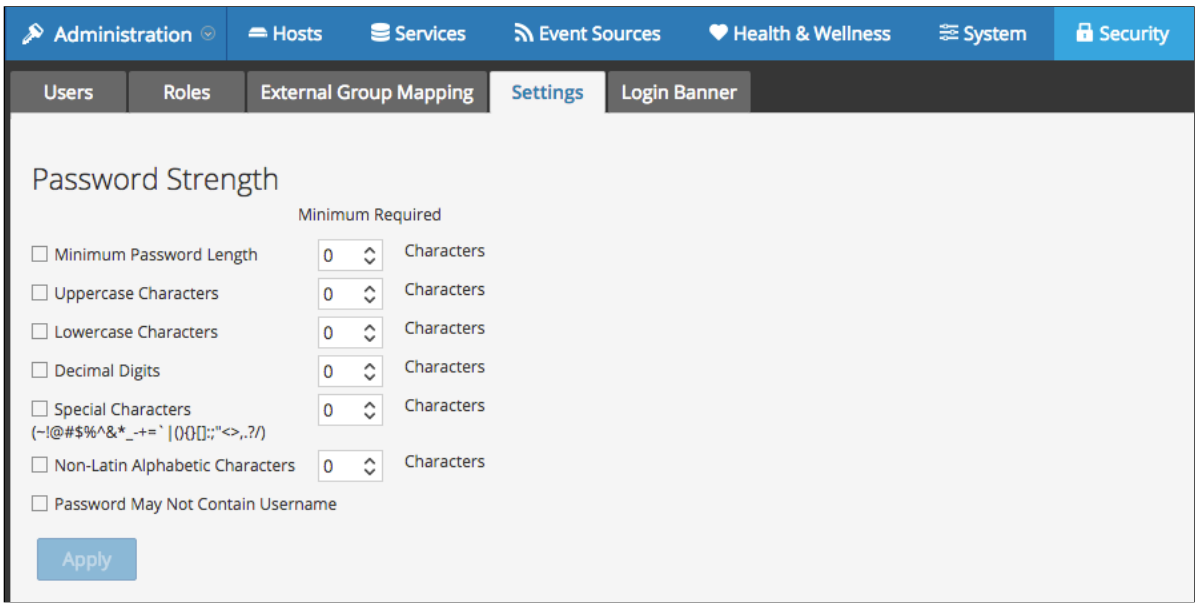
An Easier Way

This can also be configured in a more user-friendly way from **Administration->Services->*<ChooseAService>*->Config->Settings**.



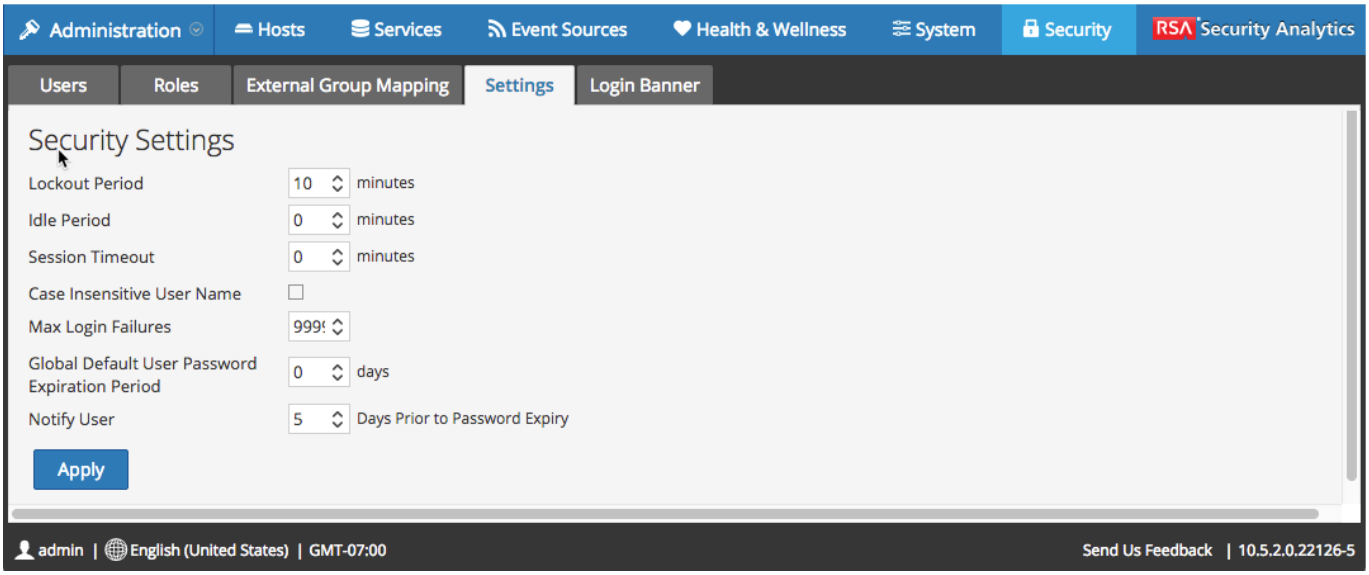
Password Strength Policy

Many enterprises require compliance with a predefined corporate password policy. SA provides the ability to enforce this on SA users when they change their passwords. Note that this functionality only applies to internal SA users, not external users, such as those on PAM or Active Directory. To access these settings, navigate to **Administration->Security->Settings**.



UI Security Settings

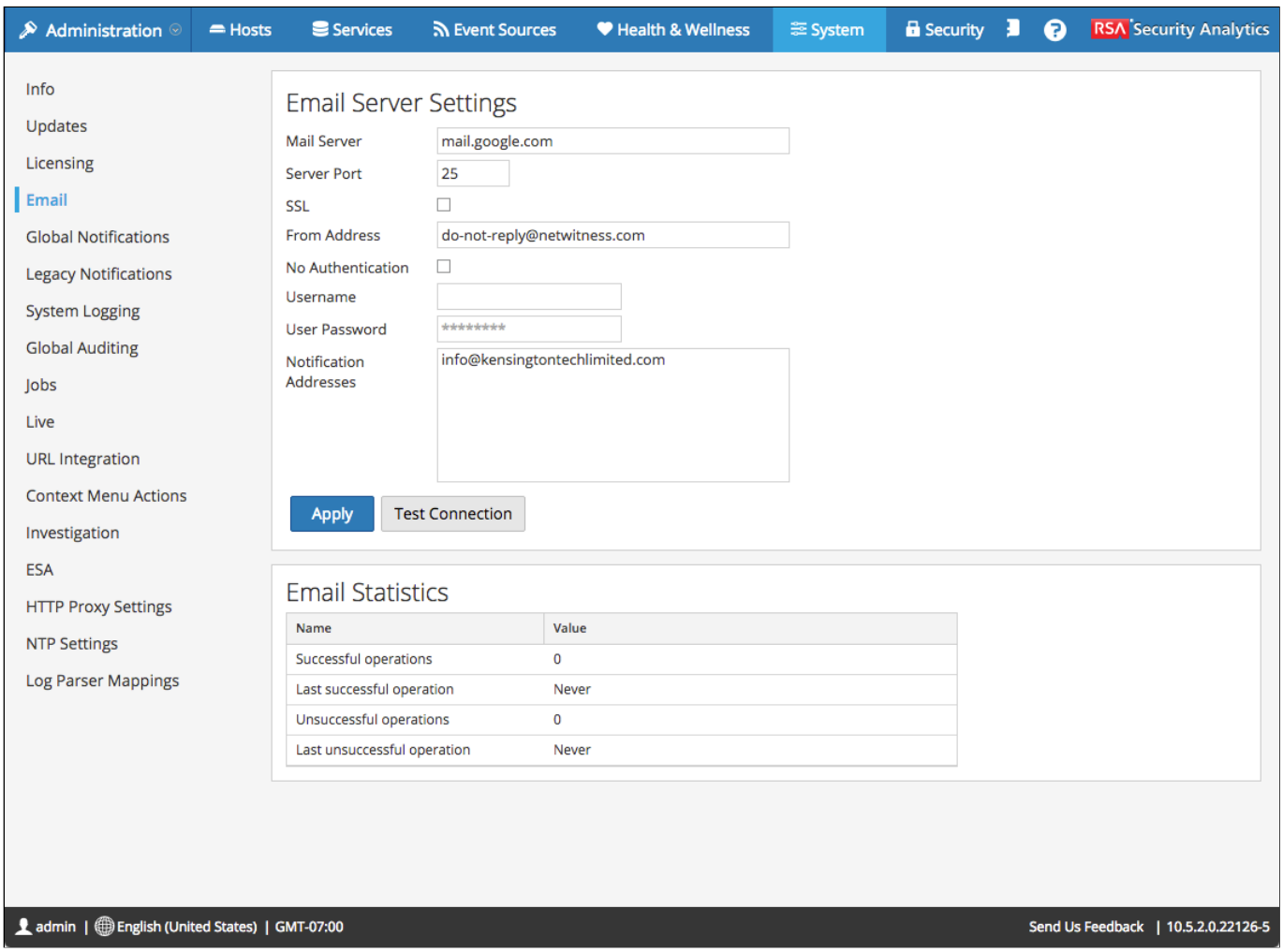
One may also configure parameters for security of the UI, such as lockout period, idle period, and session timeout. These can also be accessed from **Administration->Security->Settings**.



Password Expiry

SA can be configured to expire internal users' passwords. This can be configured in **Administration->Security->Settings,** under **Global Default User Password Expiration Period** (see above image). SA will send an email to the user's configured email address N days prior to expiry. This is configured just below the expiration period, under **Notify User.**

Top Tip: In order to use the email notification feature, an SMTP server must first be configured under **Administration->System->Email**.



23. Data Privacy

Due to stringent laws which do not allow for the the collection or attribution of personally identifying information, this is a topic most relevant to organisations in European Union countries, and as such, we shall only teach this for orgs that have expressed interest in it. Security Analytics allows a security org to obfuscate personally-identifiable information, so that security data can still be collected, but not attributed to a specific individual. We shan't discuss whether this actually improves a security posture (it doesn't), but should you have a requirement to use it, we shall teach you to configure it.

By now, you're all familiar with meta keys, index configuration, and granular system.roles permissions for SA Core. Data privacy utilises all of these concepts, and introduces one more: meta **value hashing**. First, let's discuss hashing at a high level, then get into how it applies to SA.

What Is This Hashing Business?

'Hashing' is a cryptographic operation which generates output of a string of letters and numbers, from input of either a string or binary value. The string produced by the hashing function is consistent, and will always produce the same output for the same input. Hashing is a one-way operation, meaning that a hashed value cannot be reversed back into its original value. This is unlike encryption, which is a two-way operation, meaning that something which is encrypted is capable of being decrypted. Hashes are used primarily for two purposes: obfuscating a value, and ensuring the integrity of a file. By running a hashing function on a file, you can ensure that it hasn't been modified, as modifying a file will completely alter its hashed value. A hashing function will always produce output of a same length, regardless of the length of the input. In theory, hashes should not be vulnerable to collisions. In hashing, a collision is where two different inputs produce identical outputs. In practice, some of the older hashing algorithms can in theory produce collisions.

Some common hashing algorithms are SHA1, SHA256, and MD5.

The Problem With Hashes

Though hashes are not mathematically reversible, they are still vulnerable to attack, because of something called **Rainbow Tables**. Otherwise known as a **Dictionary Attack**, a rainbow table is effectively a pre-calculated brute force attack, in table form. The table contains hash, value pairs. If you have a hash and want to know the original value of the hash, you simply look up the hash value in the table. Here's a basic rainbow table for the letters a through n, using SHA-1 hashing:

```
3f786850e387550fdab836ed7e6dc881de23001b,a
89e6c98d92887913cadf06b2adb97f26cde4849b,b
2b66fd261ee5c6cfc8de7fa466bab600bcfe4f69,c
e983f374794de9c64e3d1c1de1d490c0756eeeff,d
094e3afb2fe8dfe82f63731cdcd3b999f4856cff,e
a9fcd54b25e7e863d72cd47c08af46e61b74b561,f
a5938ace3f424be1a26904781cdb06d55b614e6b,g
244aa7266b3f5a08321b403b2c59baeba5539b19,h
397d543883c5cb5019a0ed08acba13fcb26261c2,i
3eedd8854d1e67a41f4a4aaf55ea0af427a65090,j
141b3ea3f03f5b011095f59a650c8883b9e56b44,k
cde25b5e10ad99822ac2c62b8e01b4d8af3e01d5,l
9a40b379906cd7befb8cfb7b084ac5852a1898c4,m
ab0680a89434e6411d9838b675ea87c2c11c379e,n
```

What's to be Done About Rainbow Tables?

Rainbow tables are clever, but they're fairly easily defeated through use of something called a **salt**. A salt is set of 'random' characters which get appended to the original value to be hashed, prior to the execution of the hash function. This has the effect of completely changing the output of the hash. Salts must be kept secret, but this is usually easily done. If an attacker does not know what salt value is being used by the hash function, he would need separate rainbow tables for every conceivable salt value under the sun. This isn't really feasible with current computing technology.

Here's the unsalted SHA-1 hash value for 'a':

```
3f786850e387550fdab836ed7e6dc881de23001b
```

Here is the SHA-1 hash value for 'a' with a salt of 'kgrf6db$' applied

```
0f91155179af47aee0316247f23b12cbdd18a202
```

Okay, So What Does All This Have to Do with SA?

Data Privacy in SA works essentially by hashing an original meta value, and then storing the hashed value in a separate meta key. Any meta key can be obfuscated in this manner: ip addresses, usernames, hostnames, etc. The original, unhashed value can still be stored in its original meta key, or it can also be configured to be discarded. If you choose to save the original meta values, you can use the system.roles to restrict access to the unhashed meta keys. Typically this is restricted to an SA user persona called the **Data Privacy Officer**, or **DPO**. You can also choose to configure a salt value which will be applied to all values prior to hashing.

Who is this Data Privacy Officer?

The Data Privacy Officer persona has two essential functions: to configure data privacy (the permissions to do this should be exclusive to the DPO role) , and when an original meta value is required as part of an investigation, he will be able to access this value.

What are the Drawbacks of Data Privacy?

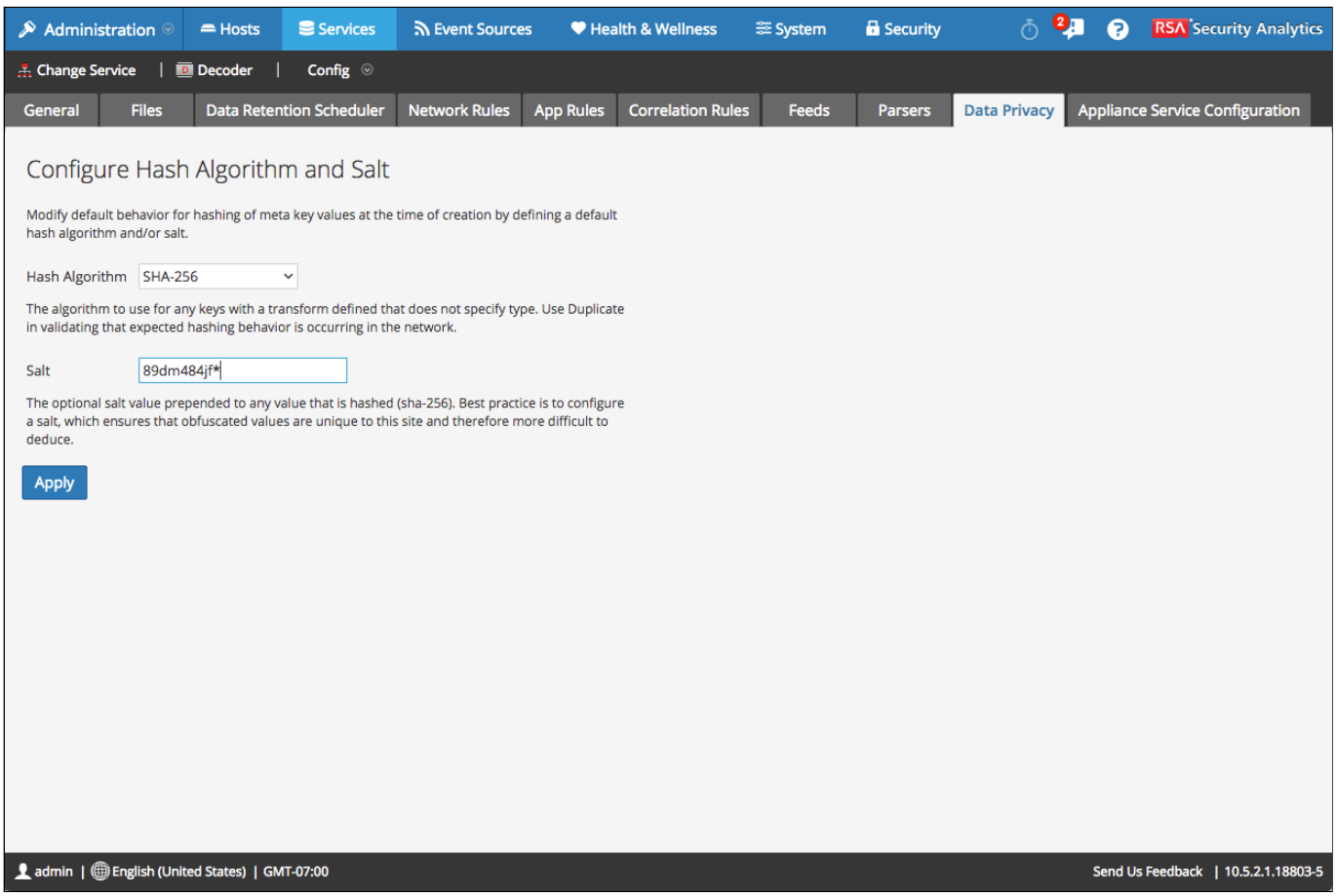
Enabling Data Privacy can significantly complicate investigations. No longer can a simple name or IP address be remembered by an Analyst. They must now keep track of near-meaningless hash values. It also limits what feeds can be applied to your data, and can hamper ESA correlation as well.

How do I Configure Data Privacy in SA?

It's multi step process:

1. Choose which meta keys you would like to obfuscate with hashed values. Here are some suggestions: **alias.host, ip.src, ip.dst, ipv6.src, ipv6.dst, username, ad.username.src, ad.username.dst, ad.computer.src, ad.computer.dst, user.src, user.dst**

2. Choose a hashing algorithm and configure a hash salt on each of your Decoders or Log Decoders from **Administration->Services->*<Service>*->Config->Data Privacy**. This is usually **SHA-256**.



Modify the index settings of your Decoder or Log Decoder to:

- Add meta keys to be used to store obfuscated (hashed) meta values. The usual practice is to add the suffix '**.hash**' to the end of the original meta key name. The **token="true"** attribute tells the UI that values of this key are aliases for original values, and they may not be very interesting to an Analyst. This may cause the UI to alter the way the key's values get displayed so that they're a bit easier on the eyes :

```
<key description="Hostname Aliases Hash" level="IndexValues" name="alias.host.hash" format="Text" valueMax="2500000" token="true" />
<key description="Source IP Address Hash" level="IndexValues" name="ip.src.hash" format="Text" valueMax="15000000" token="true" />
<key description="Destination IP address Hash" level="IndexValues" name="ip.dst.hash" format="Text" valueMax="15000000" token="true"
/>
<key description="Source IPv6 Address Hash" level="IndexValues" name="ipv6.src.hash" format="Text" valueMax="2500000" token="true"
/>
<key description="Destination IPv6 address Hash" level="IndexValues" name="ipv6.dst.hash" format="Text" valueMax="2500000"
token="true" />
<key description="User Account Hash" level="IndexValues" name="username.hash" format="Text" valueMax="2500000" token="true" />
<key description="E-mail Address Hash" level="IndexValues" name="email.hash" format="Text" valueMax="2500000" token="true" />
```

- Activate hashing on chosen original meta keys and store the hashed values to the new meta keys which you just created. This is done by adding a **<transform destination="myNewKey"/>** subkey to the original meta key. Also add the attribute **protected="true"** to the original meta keys on your Decoders, Log Decoders, and Concentrators. This tells the Core service to tightly control any release of values from the key.

```
<key description="Hostname Aliases" level="IndexValues" name="alias.host" format="Text" valueMax="2500000" protected="true">
<transform destination="alias.host.hash"/></key>
<key description="Source IP Address" level="IndexValues" name="ip.src" format="IPv4" valueMax="15000000" protected="true"><transform
destination="ip.src.hash"/></key>
<key description="Destination IP address" level="IndexValues" name="ip.dst" format="IPv4" valueMax="15000000" protected="true">
<transform destination="ip.dst.hash"/></key>
<key description="Source IPv6 Address" level="IndexValues" name="ipv6.src" format="IPv6" valueMax="2500000" protected="true">
<transform destination="ipv6.src.hash"/></key>
<key description="Destination IPv6 address" level="IndexValues" name="ipv6.dst" format="IPv6" valueMax="2500000" protected="true">
<transform destination="ipv6.dst.hash"/></key>
<key description="User Account" level="IndexValues" name="username" format="Text" valueMax="2500000" protected="true"><transform
destination="username.hash"/></key>
<key description="E-mail Address" level="IndexValues" name="email" format="Text" valueMax="2500000" protected="true"><transform
destination="email.hash"/></key>
```

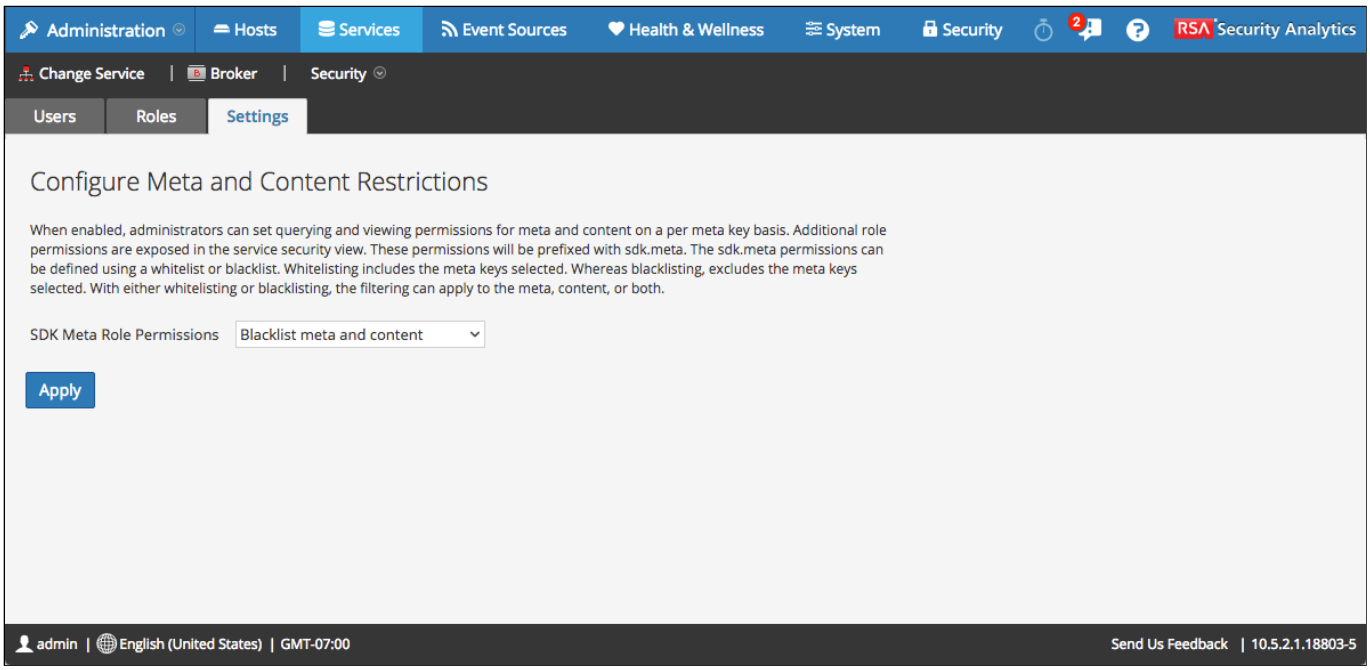
4. Restart your Decoders and Log Decoders

5, Add the new hash meta keys to your Concentrators' index configuration and restart the services.

6. Once the Concentrators have been restarted, restart your Brokers so that they will learn the new hash keys from your Concentrators.

7. Activate **system.roles** security on your Concentrators and Brokers with one of these settings. We recommend a value of 4, which blacklists both meta and content, as this is the most secure with the least administrative overhead. You may do so with the API node **/sdk/config/system.roles**, or it can be accomplished from **Administration->Services->*<yourService>*->Security->Settings**.

Value	Explanation
0	Disabled
1	Whitelist, with both meta and content filtered. If the sdk.meta.X permission for the given key is selected for a user's role, the user will be able to access both the meta of this key, and the raw content of sessions with this key (raw packets or logs). When viewing meta in Event Reconconstruction, only meta keys which have been whitelisted will be displayed.
2	Whitelist meta filtered. If the sdk.meta.X permission for the given key is selected for a user's role, the user will be able to access the meta of this key, but they will not be able to access the raw session content. When viewing meta in Event Reconstruction, only meta keys which have been whitelisted will be displayed.
3	Whitelist Content filtered. If the sdk.meta.X permission for the given key is selected for a user's role, the user will be able to access the raw content (either packets or logs) of sessions where all existing keys of the session have been whitelisted for the given role, and the user will be able to access all meta for the session.
4	Blacklist both meta and content. If the sdk.meta.X permission for the given key is selected for a user's role, the user will neither be able to access the session meta for this key, nor the raw content of sessions with this key (raw packets or logs). When viewing meta in Event Reconstruction, meta keys which have been blacklisted will not be displayed.
5	Blacklist meta filtered. If the sdk.meta.X permission for the given key is selected for a user's role, the user will not be able to access the meta of this key, but they will be able to access the raw session content. When viewing meta in Event Reconstruction, meta keys which have been blacklisted will not be displayed.
6	Blacklist Content filtered. If the sdk.meta.X permission for the given key is selected for a user's role, then the user will not be able to access the raw content (either packets or logs) of sessions where this key exists, but they will be able to access all meta for the session.



8. In **Administration->Security->Users**, add the **Data_Privacy_Officers** role to an existing user who you wish to be your DPO, or create a new user with this role.

Top Tip: Be sure to do this step before modifying your Administrator role permissions. You don't want to lock your whole organisation out of the ability to manage restricted settings!

9. Protect your data privacy configuration and enforce content restrictions by configuring the local security roles in your Concentrators and Brokers to:

- Permit the **Data_Privacy_Officers** role to access all meta and content for all keys (only necessary if you've whitelisted meta).
  Restrict non Data_Privacy_Officers roles from configuring SA Core meta key permissions by removing the permissions:
  - **index.manage** - this removes the ability of role members to modify the index configuration from the UI, so a non-DPO cannot disable meta obfuscation by editing the index configuration file. This does not prevent a user from doing so via the command line, however.
  - **sdk.manage -** this removes the ability of role members to configure the **/sdk** branch of the API tree, so a non-DPO cannot disable **system.roles** restrictions
  - **users.manage -** this removes the ability of role members to configure SA Core users and Groups, so a non-DPO cannot give themselves permission to obfuscated keys
- Restrict all roles except **Data_Privacy_Officers** so that they may only access obfuscated meta keys. This screenshot shows blacklisted keys and their obfuscated siblings.

Administration | Hosts | Services | Health & Wellness | Security | RSA Security Analytics

Change Service | Concentrator | Security

Users | Roles | Settings

Replicate

Role Name

- Administrators
- Aggregation
- Analysts
- Data_Privacy_Officers
- Malware_Analysts
- Operators
- SOC_Managers

Role Information

Name

Administrators

Role Permissions

	Name	Description
☐	sdk.meta.group	Allows for querying metadata entry group
☐	sdk.meta.ip.addr	Allows for querying metadata entry ip.addr
☑	**sdk.meta.ip.dst**	**Allows for querying metadata entry ip.dst**
☐	sdk.meta.ip.dst.hash	Allows for querying metadata entry ip.dst.hash
☐	sdk.meta.ip.dstport	Allows for querying metadata entry ip.dstport
☐	sdk.meta.ip.proto	Allows for querying metadata entry ip.proto
☑	**sdk.meta.ip.src**	**Allows for querying metadata entry ip.src**
☐	sdk.meta.ip.src.hash	Allows for querying metadata entry ip.src.hash
☑	**sdk.meta.ipv6.dst**	**Allows for querying metadata entry ipv6.dst**
☐	sdk.meta.ipv6.dst.hash	Allows for querying metadata entry ipv6.dst.hash
☐	sdk.meta.ipv6.proto	Allows for querying metadata entry ipv6.proto
☑	**sdk.meta.ipv6.src**	**Allows for querying metadata entry ipv6.src**
☐	sdk.meta.ipv6.src.hash	Allows for querying metadata entry ipv6.src.hash
☐	sdk.meta.language	Allows for querying metadata entry language
☐	sdk.meta.latdec.dst	Allows for querying metadata entry latdec.dst
☐	sdk.meta.latdec.src	Allows for querying metadata entry latdec.src
☐	sdk.meta.lc.cid	Allows for querying metadata entry lc.cid
☐	sdk.meta.lifetime	Allows for querying metadata entry lifetime
☐	sdk.meta.link	Allows for querying metadata entry link

Apply | Reset

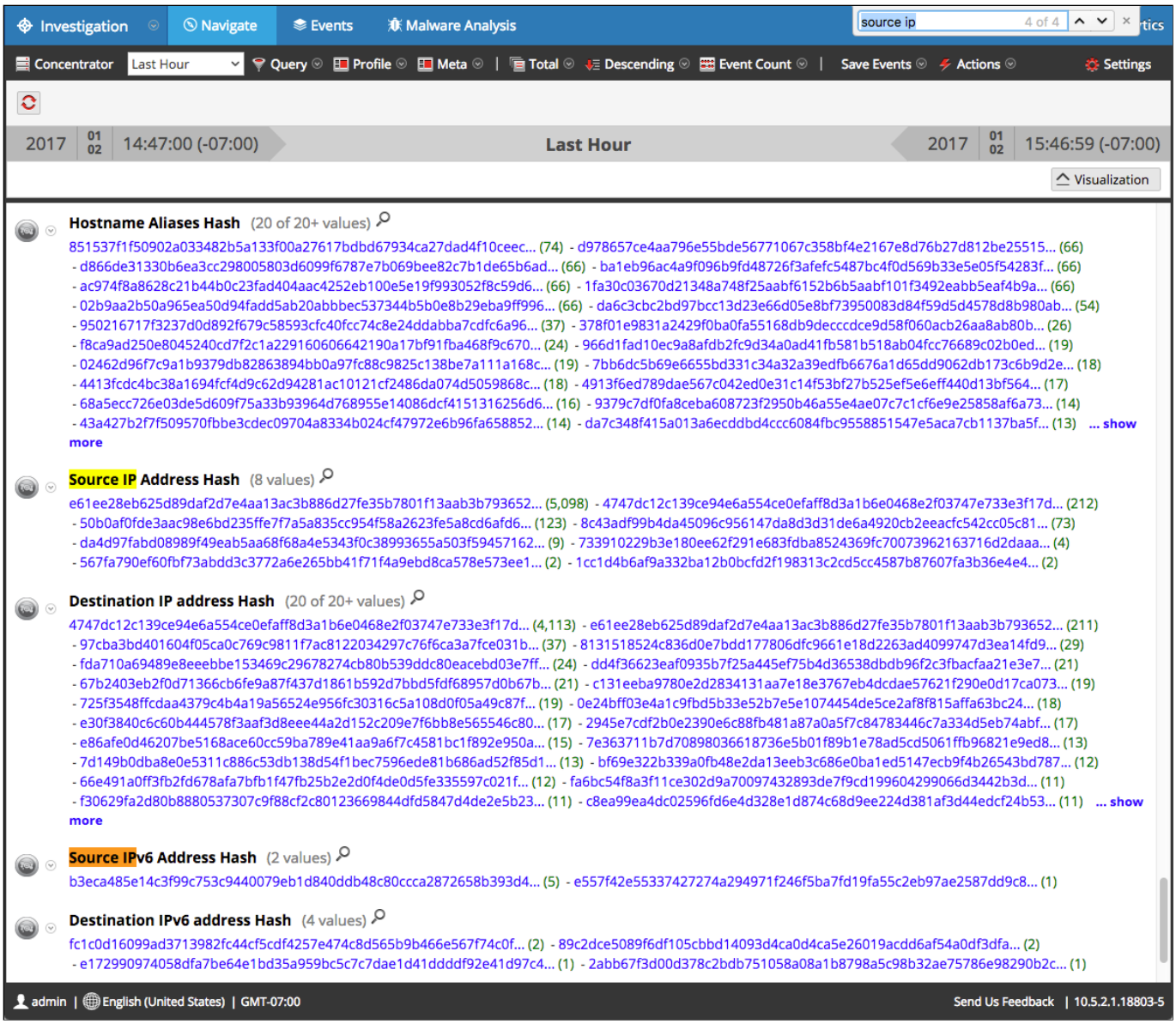
dpo | English (United States) | GMT+00:00 | Send Us Feedback | 10.5.2.1.18803-5

If you've done everything properly, when you access Investigation from a non-DPO account, you should see this message for original unobfuscated meta keys :

Source IP Address

Insufficient permissions for the requested data. **Click to Reload**

You should also see obfuscated meta keys with the suffix "Hash":



Now only users with the **Data_Privacy_Officers** role should be able to see the original unobfuscated meta keys.

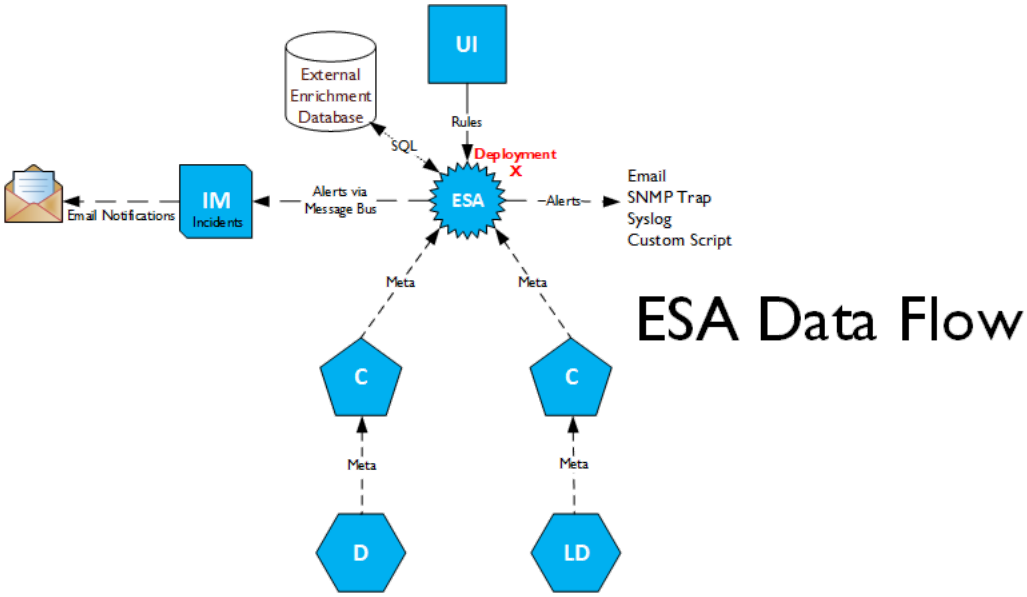
24. Event Stream Analysis

Intro

Event Stream Analysis, or **ESA**, is RSA's correlation engine for Security Analytics. Its job is to aggregate meta, and generate security alerts by running it as a data stream through rules. These rules can come from RSA Live, or they can be written from scratch. ESA is an implementation of the open source Esper stream processing engine, by EsperTech. This engine uses the Esper Processing Language, or **EPL**. EPL is a type of programming language that combines aspects of SQL and Java, to allow for the correlation of meta across multiple events, or in SA parlance, sessions. Unlike Application Rules, which are deployed at the Decoder level, ESA can correlate meta from multiple sessions in a single rule.

The goal of this lesson isn't to familiarise you with the EPL language itself, as that, on its own, could be the subject of an entire course. Rather, we shall delve into the administrative aspects of ESA, including the deployment of some Live ESA rules, and alerting.

ESA streams meta from Concentrators. It doesn't care whether the meta originates at a Log Decoder or a Packet Decoder.



ESA Use Cases

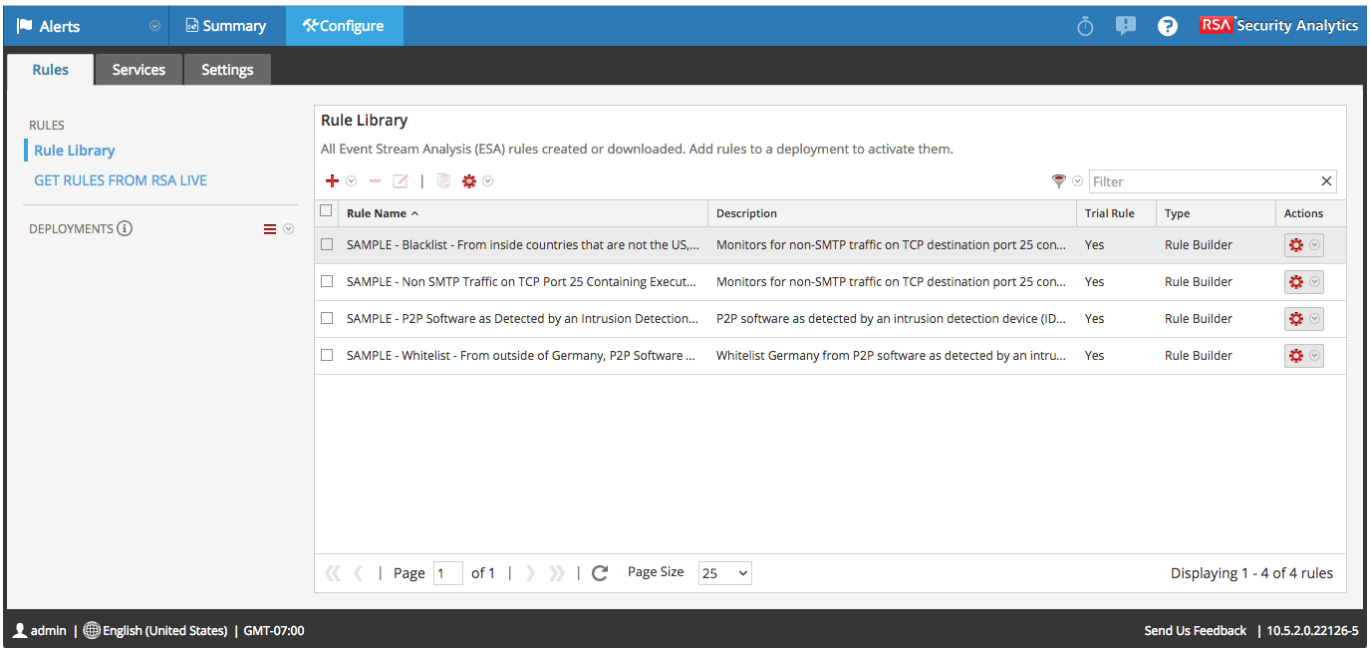
Examples of use cases where ESA might come in handy are:

- Multiple login failures followed by a successful login
- Download of PDF file followed by download of an executable file
- User account created, logged in, and deleted within an hour
- Clearing of an audit log
- Multiple firewall denies from source followed by permit
- Brute force login attempts

These are just a few use cases. ESA is capable of doing so much more.

Configuring ESA

ESA can be accessed primarily through the **Alerts** area of the UI. Let's navigate to ESA's **Configure** area.



Rule Library

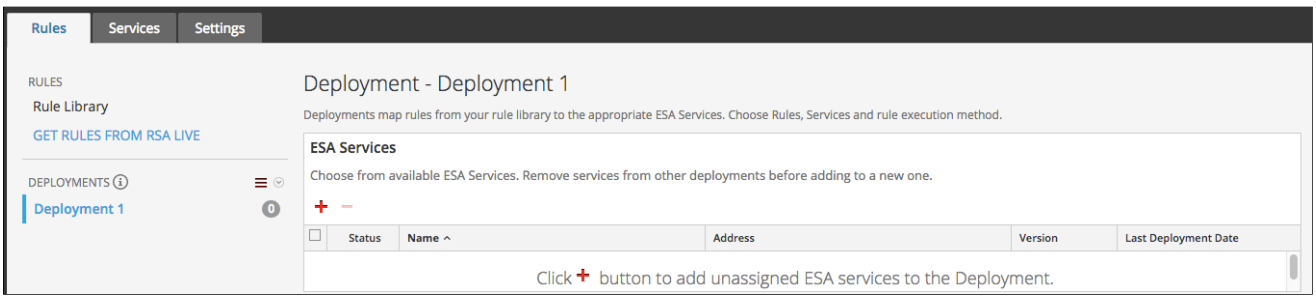
The first thing which you'll see is this **rule library**. This is the area from where ESA **Deployments** will draw their rules. It's also the area where one can modify ESA rules. Notice that out-of-the-box, there are four sample rules. We'll come into the individual rules a bit later.

Deployments

Every individual ESA service must belong to a **Deployment**. This is a logical grouping of a subset of library rules. Every service which belongs to a Deployment will get all receive the same rules. Essentially, a Deployment is a way of keeping rules synchronised between ESA services without having to deploy all rules to all services.

1. To create a Deployment, click the ≡ dropdown, and then **Add**.

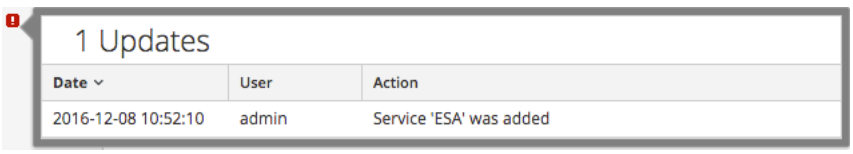
2. Give your deployment a name and hit Enter.



3. Under **ESA Service**, click the + sign.

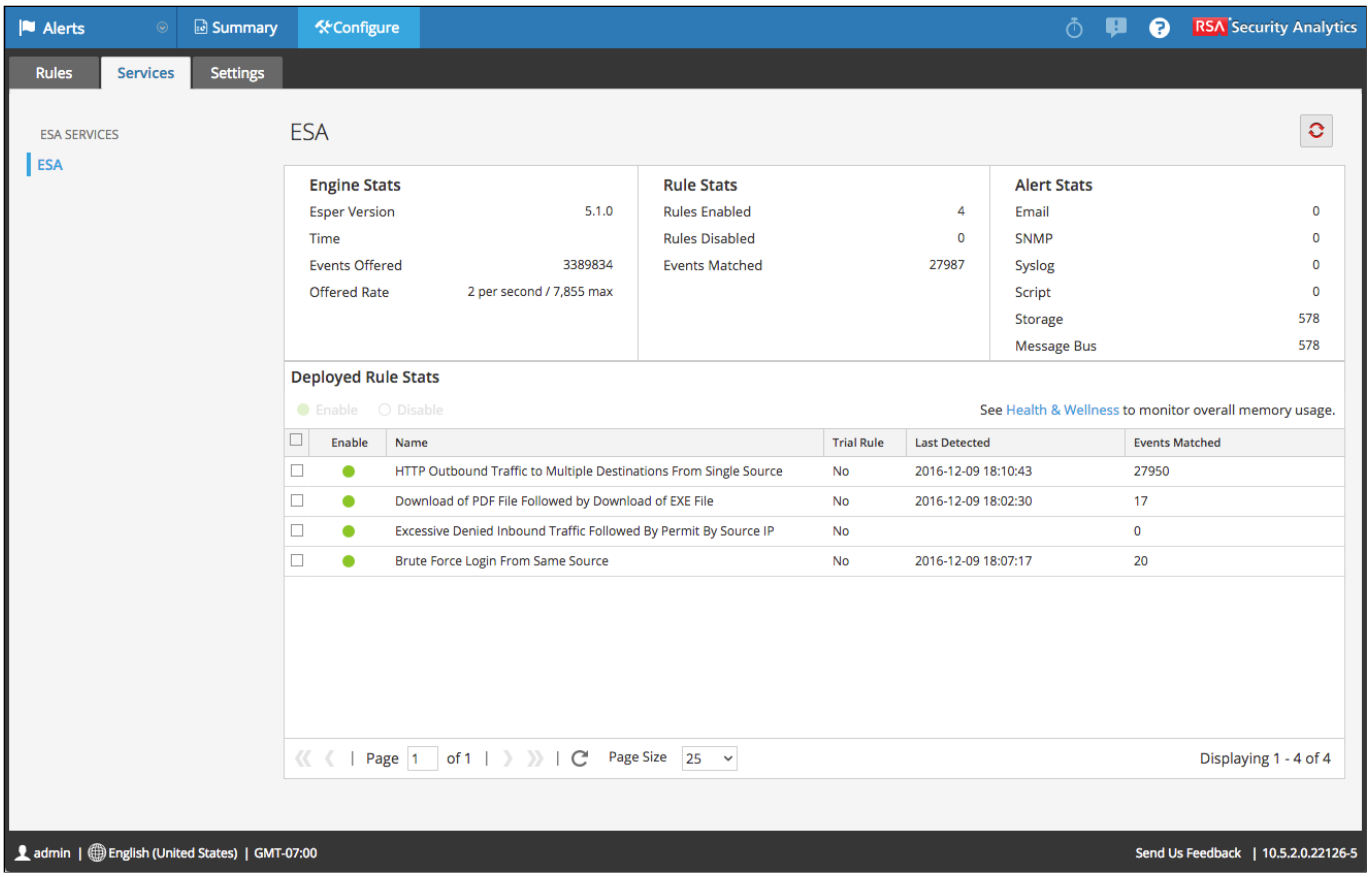
4. Select an ESA service to add to the Deployment and hit **Save**.

After this, you'll see a ! sign, which signifies that a Deployment has been updated, and requires being pushed out to the ESA services. If you click the symbol, you'll be shown what updates have been made to the Deployment.



Services Tab

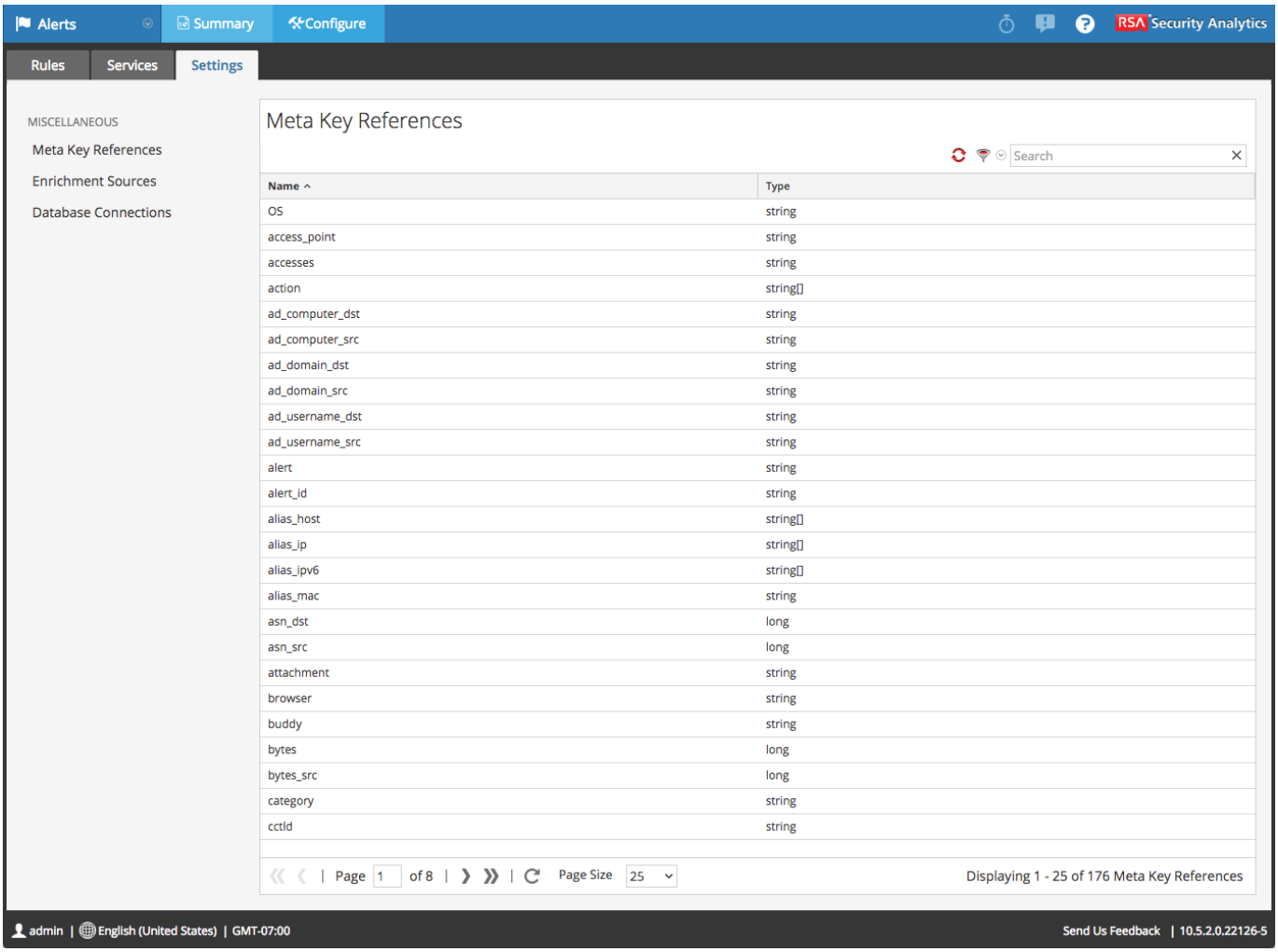
Clicking on the Services tab of **Alerts->Configure**, we'll find some information about every individual ESA service. You can select individual services from the left-hand pane. It will show the Esper version, engine stats, rule stats, and alert stats for the selected services. It will also show every rule deployed to the service, whether it's enabled, in trial mode, when it last detected something, and how many events matched the rule.



Settings Tab

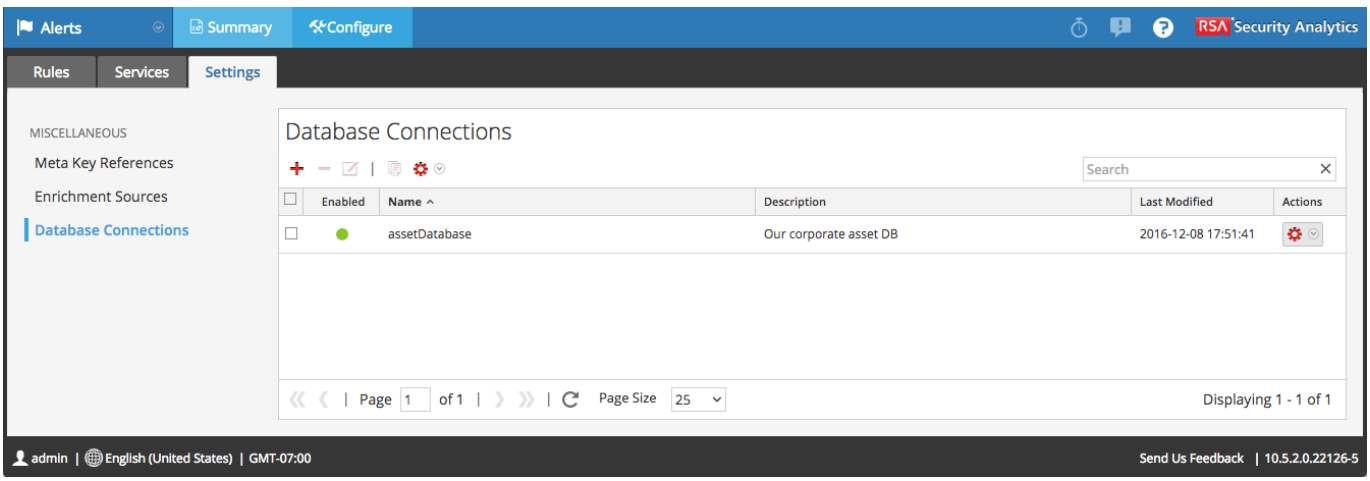
Clicking over to the Settings tab, we first see the **Meta Key References** tab. This is essentially all of the meta keys which ESA has detected from its upstream Concentrators. If you make an index configuration change to them, hit the refresh button here so that ESA can make use of any new keys.

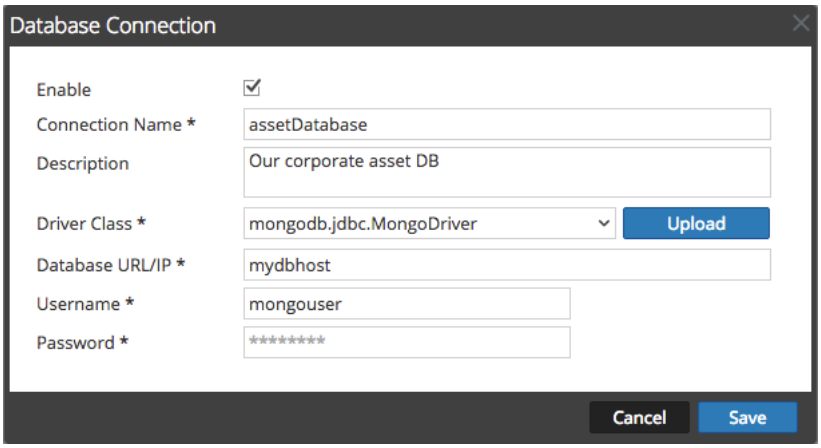
Top Tip: Notice that meta key names in ESA have had there periods replaced by underscores. I.e. **tcp.dstport** becomes **tcp_dstport**. This is because in EPL (which wasn't actually written by RSA), the period is an operator character, which has the function of accessing child objects. This is fairly standard in many programming languages.



Database Connections

Click on **Database Connections** inside **Alerts-Configure->Settings**. Here is where one may define connections to external databases. These can be used in ESA rules to lookup against external tables. It is capable of using MongoDB, PostgreSQL, and JDBC. You can also upload your own driver, in jar class form.

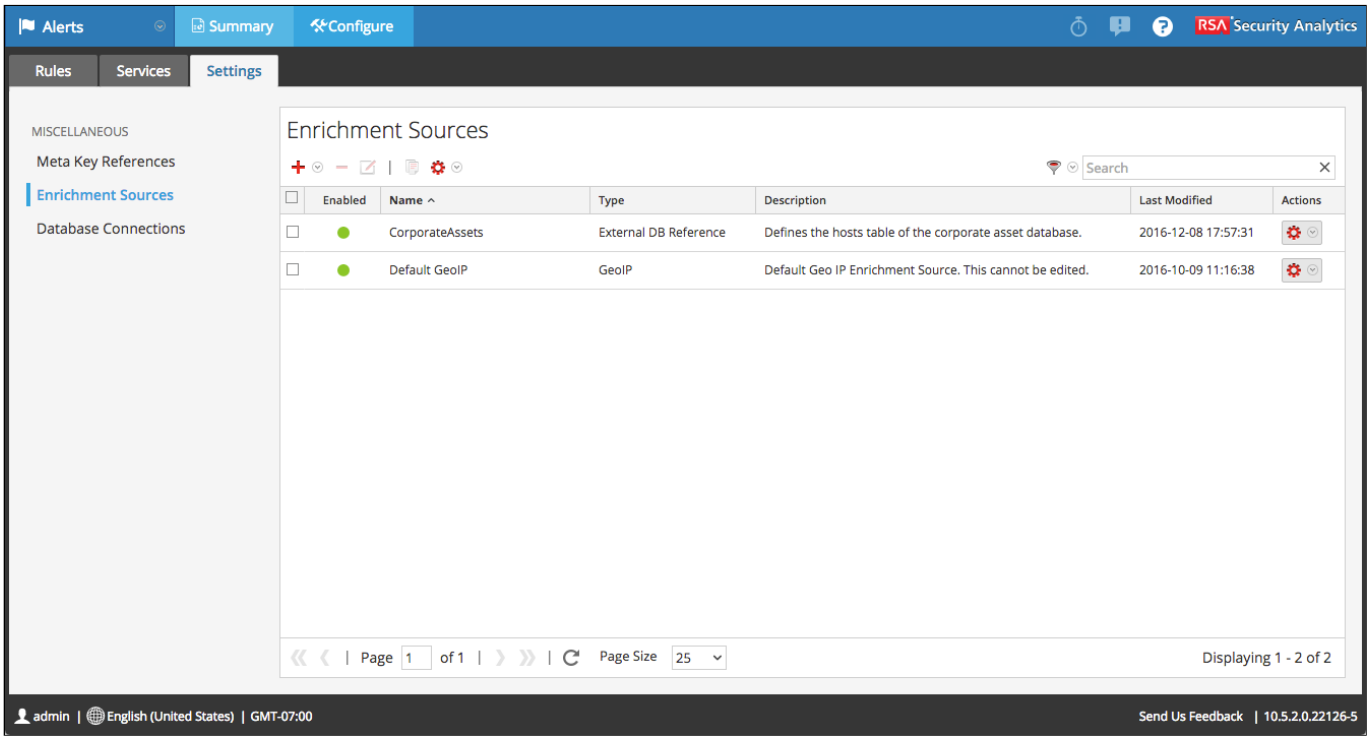


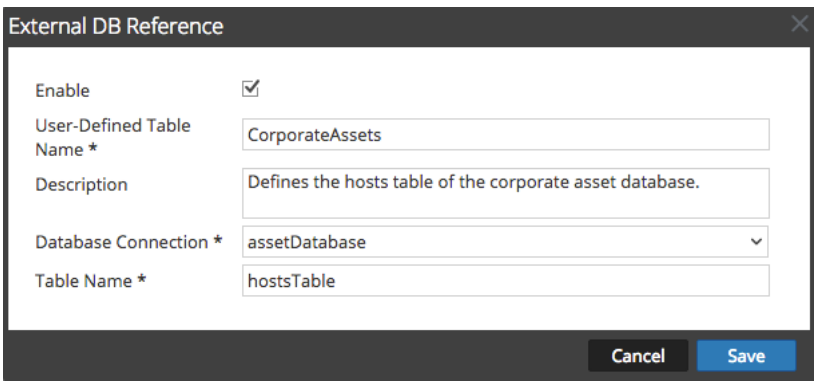


Enrichment Sources

Clicking up to the **Enrichment Sources** tab, it is here where we can make the link between a database and ESA. Whereas the Database Connections tab defines how to connect to a Database, the Enrichment Sources tab allows us to put the database into use by defining a table.

Enrichment sources can be an External DB Reference (actual database tables) , an **In-Memory Table** (this is really just a CSV uploaded into memory), or a **Warehouse Analytics** connection, which we won't get into here. There is also a GeoIP enrichment source built right into ESA, but it cannot be modified.

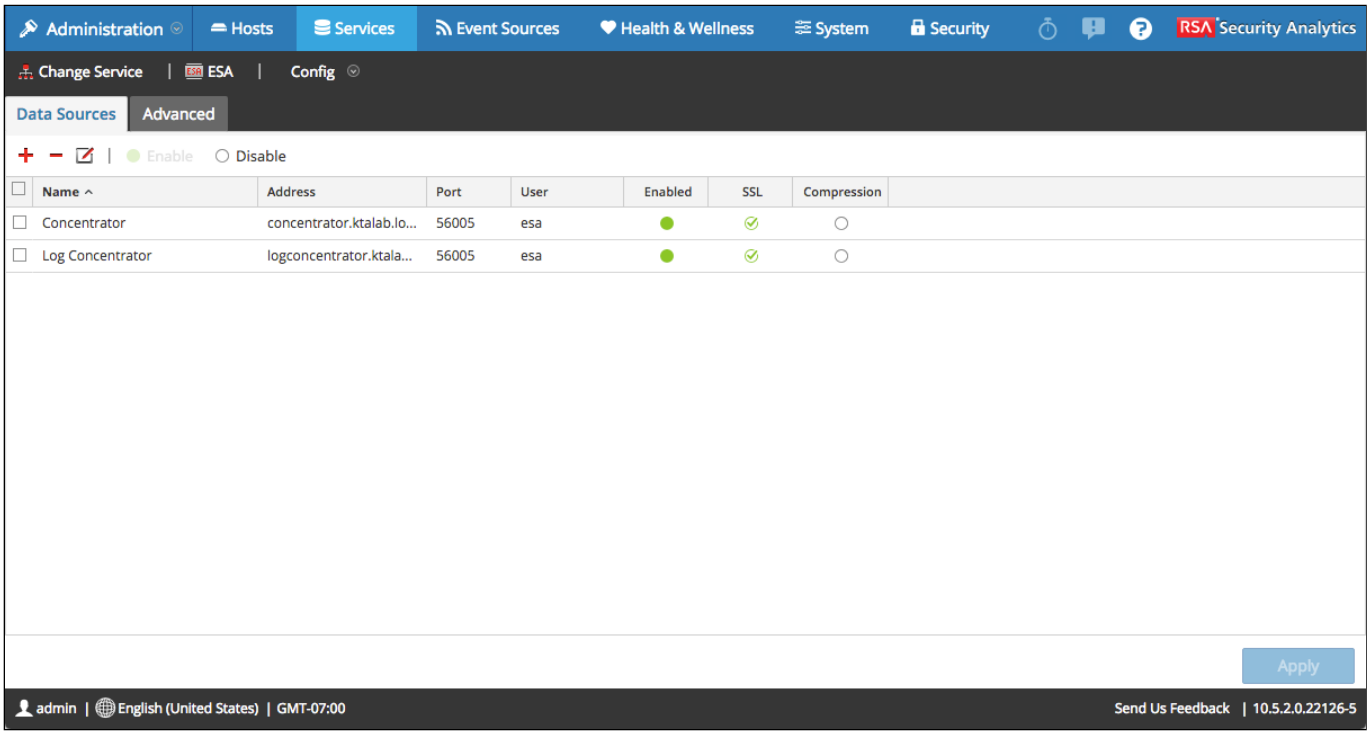




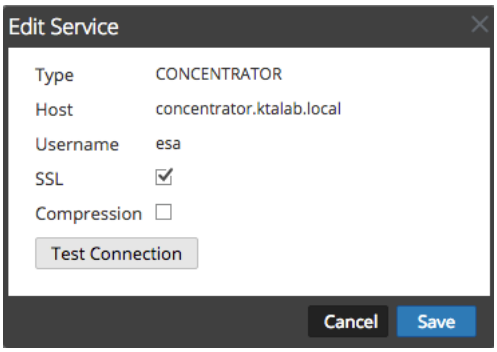
Aggregation

Before ESA will do anything useful, each ESA service must be configured to aggregate meta from upstream Concentrators. For these, we head back to the Admin interface, under **Administration->Services->*<ESA Service>*->Config**.

On the Data Sources tab, we'll add our upstream Concentrators. This is similar to adding them to a broker. We recommend using a dedicated service account for ESA,

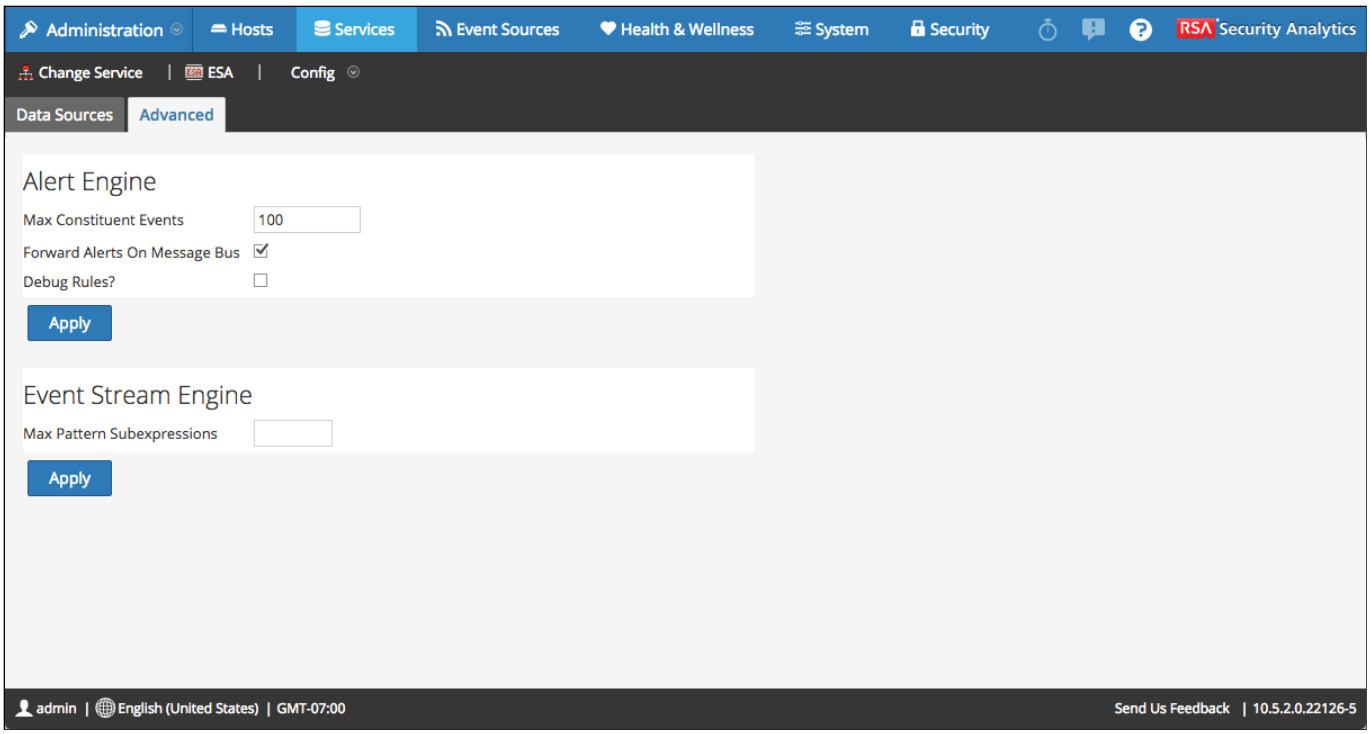


We hope that by now you've learned how to add a device for aggregation. Note that there is a compression setting in the add/edit screen. You might use this in a low-bandwidth scenario, where the ESA host and the Concentrators are connected via a WAN link.



Advanced Settings

Under the **Advanced** tab of **Administration->Services->*<ESA Service>*->Config,** there are a few advanced settings which you should know about.



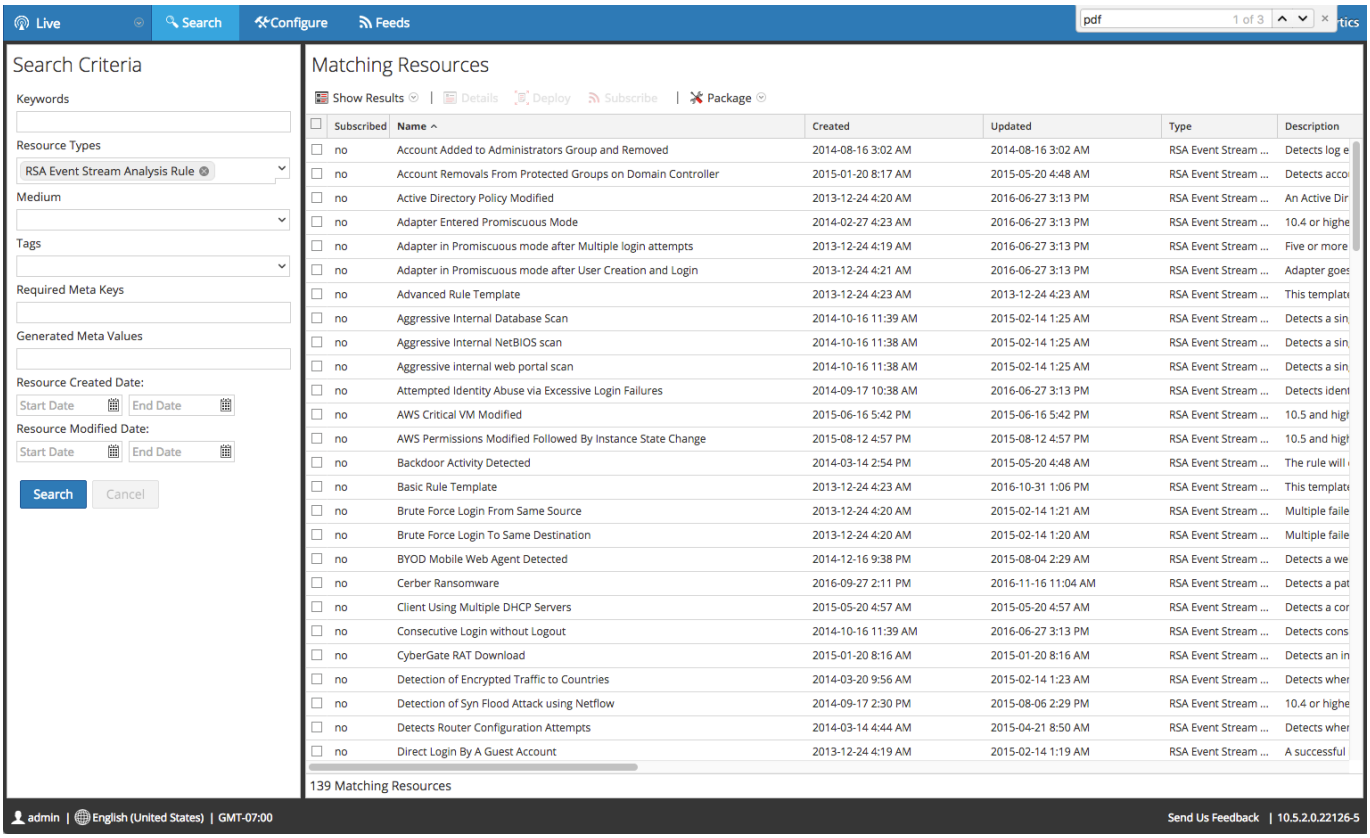
- **Max Constituent Events** - Sets the maximum number of events which an ESA alert may contain. The default is 100.
- **Forward Alerts On Message Bus** - This instructs ESA to send its alerts to SA's Incident Management component. If you're not using IM at all, there's no need for this. The default is to enable this.
- **Debug Rules** - Turns on rule debugging.
- <how is this different from activating debug on a rule?>
- **Max Pattern Subexpressions** - Some rules create multiple sub-expressions in memory, which together aid the Esper engine in deciding to trigger an alert. This a safety feature which limits the number of EPL sub-expressions can be kept in memory for a given rule. 0 is the default value, which disables any sub-expression limits. If you have issues with the stability of your ESA service because of memory utilisation, you'll likely want to set this value.

ESA Rules

ESA Rules are possibly the most complex things to understand in SA. We shan't get too far in depth here, but we'll cover some basics.

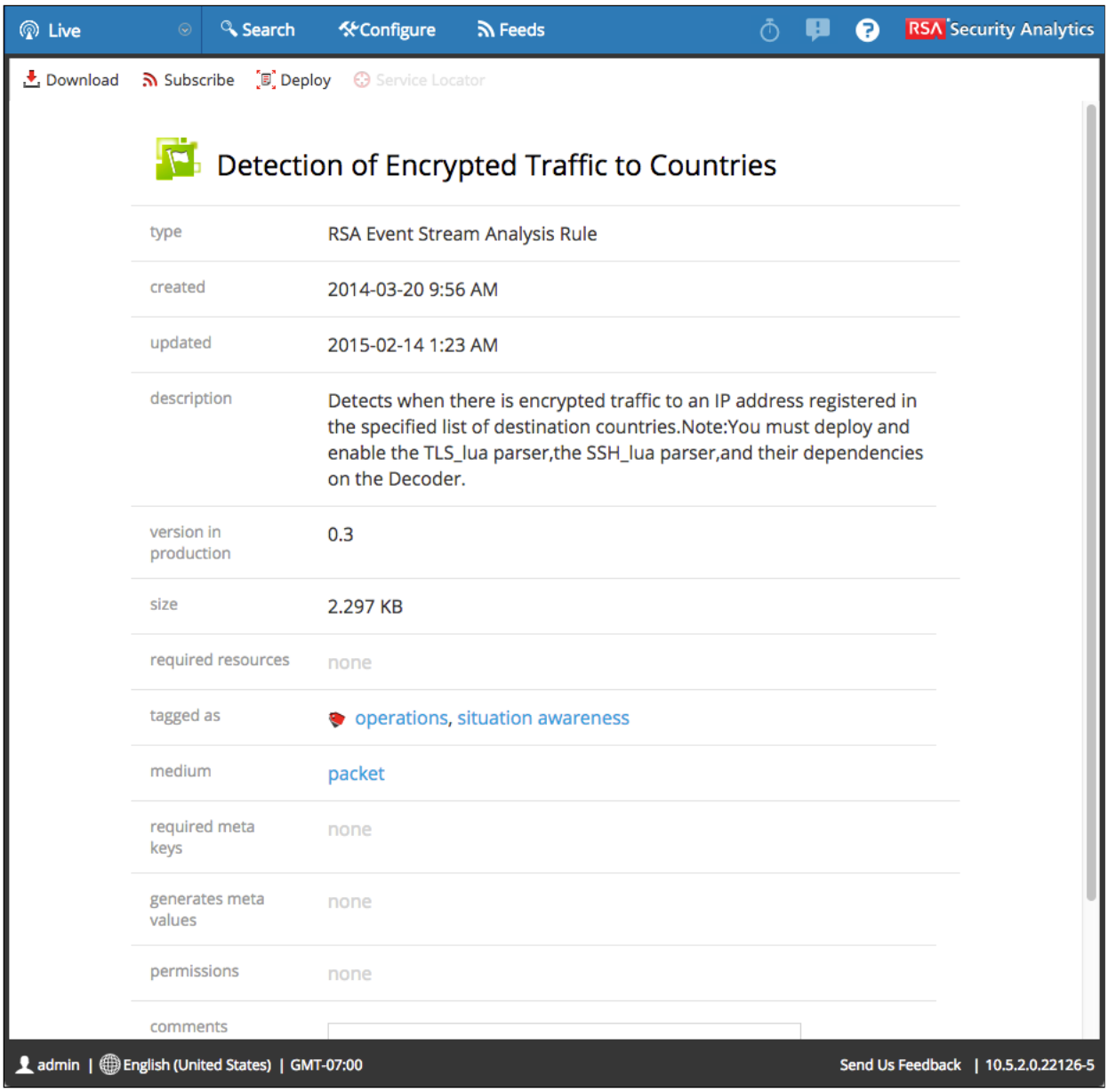
Live Content for ESA

Shall we deploy some ESA content from Live? Choose some content that interests you, and deploy it to ESA. The Resource Type is "**RSA Event Stream Analysis Rule**". Many ESA rules have parser dependencies, for both logs and packets.

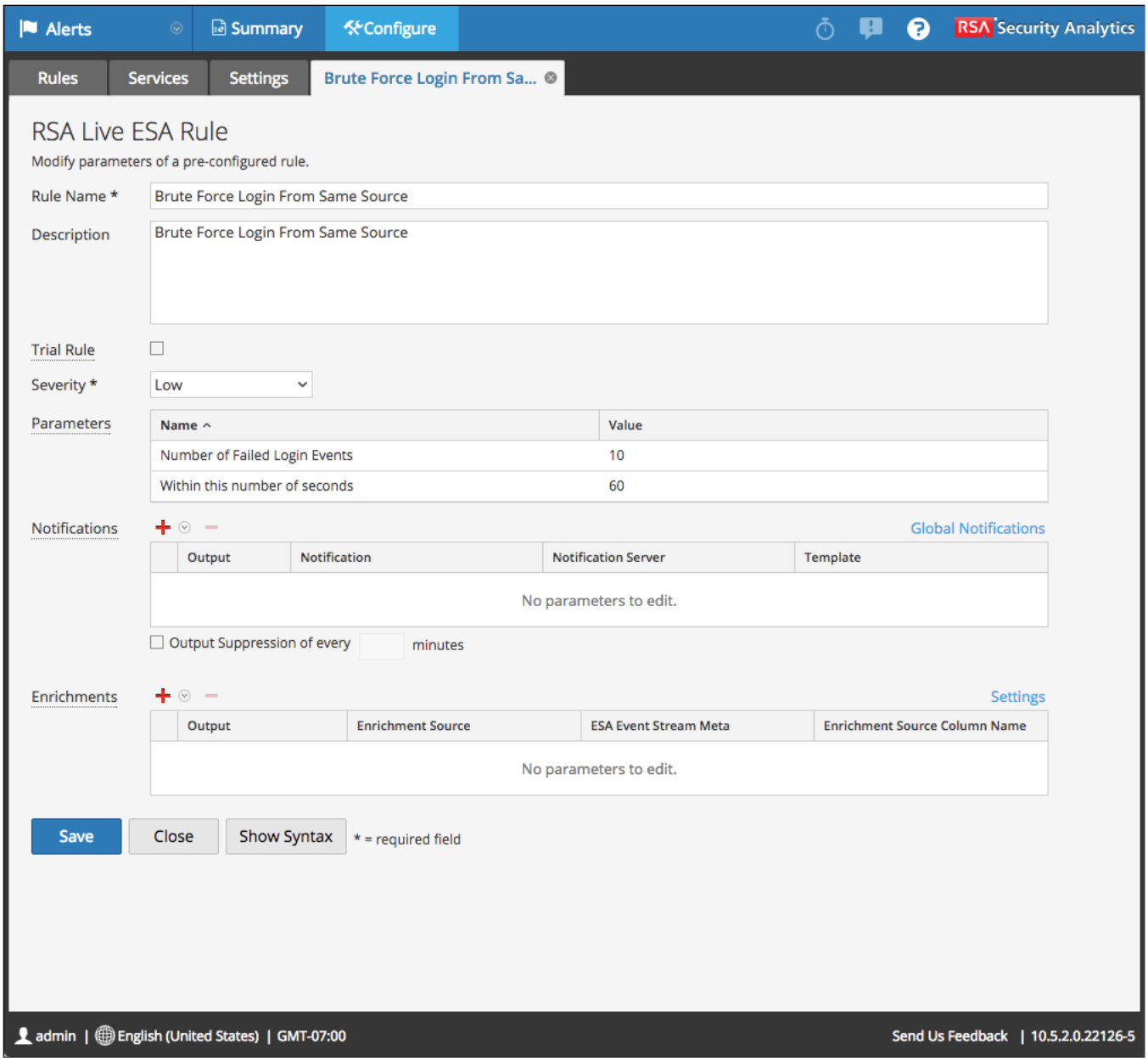


Top Tips:

- Deploying Live content into ESA only deploys it to the ESA rule library (this is called "**SA UI Endpoint"** in the Live Deployment Wizard). To activate it, you must add it into an ESA **Deployment Group** and push the group.
- Using Live deployments with ESA is usually a bad idea. You don't want good rules automatically getting overwritten by a content update that breaks something. So deploy manually instead.
- After you've deployed an ESA rule from Live, be sure to always enable Trial Mode on the rule before deploying it. Only when you know that Live rules are working properly, should you disable trial mode.
- When deploying any ESA content from Live, always be sure to check the rule dependencies in the content details. Any dependent parsers and feeds which aren't already installed must also be deployed after your Live content has.



See here that Live ESA rules have a section not present in normal rules: **Parameters**. Some of these rules have customisable variables which can impact the way a rule behaves. See below:



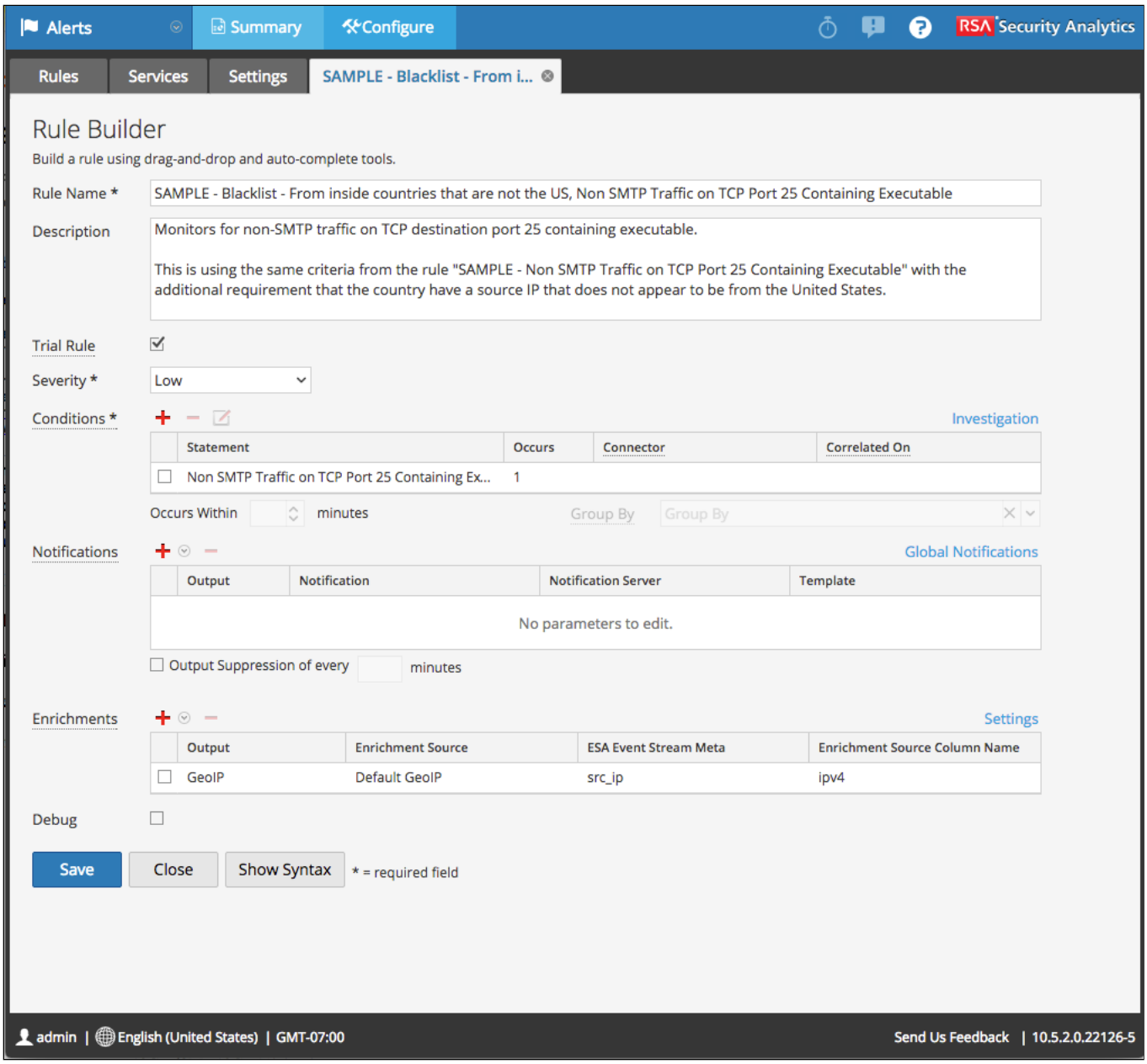
Types of ESA Rules

There are two types of ESA rules:

- **Rule Builder** - These are more basic rules where the alert conditions are built using a UI-driven interface. There are things which you won't be able to do with this, though. When this is the case, there are...
- **Advanced EPL** - These rules are written in raw EPL language. They're more for content authors who are familiar with the EPL language.

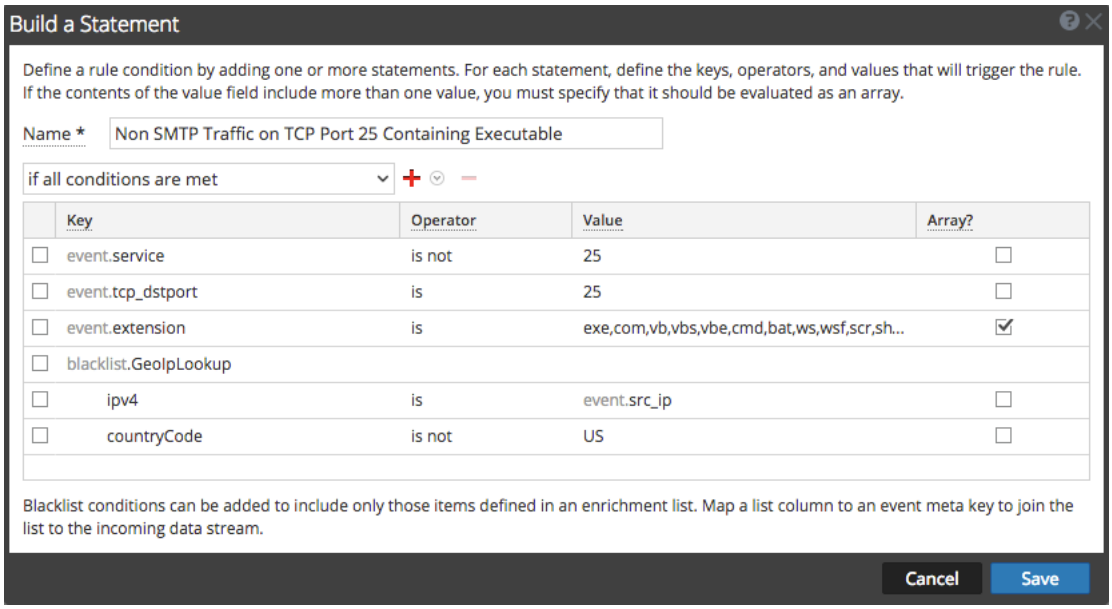
A Look at a Sample Rule - Blacklist - From inside countries that are not the US, Non SMTP Traffic on TCP Port 25 Containing Executable

In the Rule Library, double click on the rule named "**SAMPLE - Blacklist - From inside countries that are not the US, Non SMTP Traffic on TCP Port 25 Containing Executable**".



Here, you can see all the bits that make up a Rule Builder rule.

- **Rule Name -** Self-explanatory
- **Description -** Self-explanatory
- **Trial Rule -** Rule is set to run in trial mode. See section below on **Trial Rules**.
- **Severity -** The severity of the alert which the rule will generate. Options are Low, Medium, High, and Critical.
- **Conditions -** The meat of the rule, which defines when an alert will be triggered. By adding or editing a **Conditions** statement, this opens the rule builder.



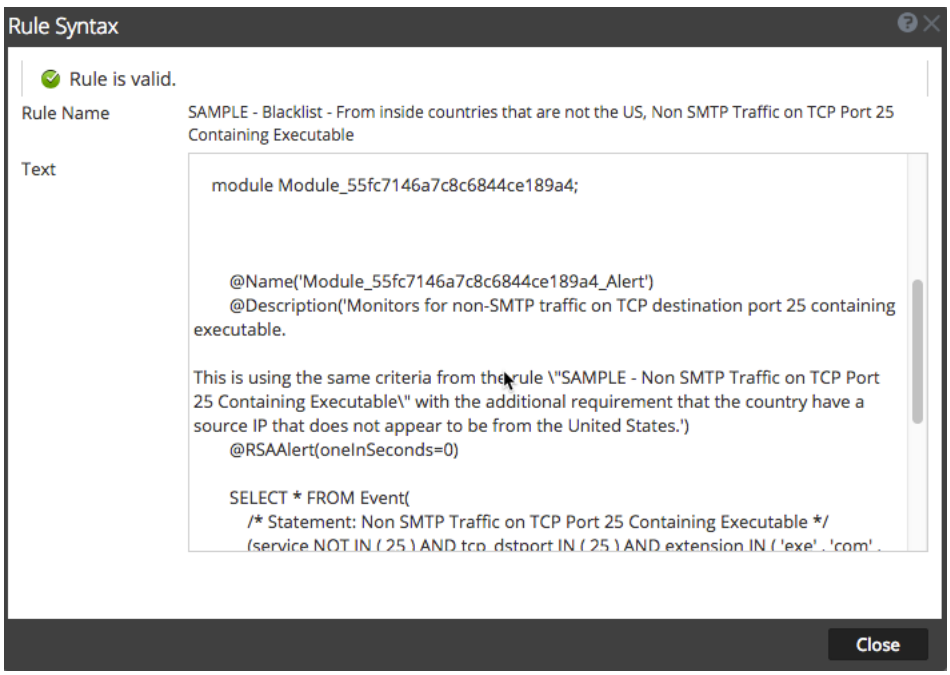
- **Notifications -** Let someone or something know when an alert has occurred. This can be via Email, SNMP trap, Syslog, or by triggering a script.

  Top Tip: Take note that if you're using SA's Incident Management component, you may not want to use ESA Notifications at all, but rather have alerts trigger Incidents instead.

- **Enrichments -** Enrichments are a way of bringing external data into your rules. This is data which isn't contained within a session's metadata, but rather data which can be queried through use of SQL-like rules. By default, there ESA contains a GeoIP database for looking up IP address geographical locations. You can interface rules with SQL database tables, and in-memory tables, as well. More on this later.

- **Debug Mode -** Adds "**@Audit('stream')**" to the rule, which dumps raw event data into the ESA logfile, located at **/opt/rsa/esa/logs/esa.log**. This can aid in debugging EPL rules. Switch this off when done!

- **Show Syntax** - Shows the raw EPL of the rule.



Advanced EPL Rules

Advanced EPL rules are just like rule builder rules, with the exception that all ESPER is written by hand rather than in a UI. You can do more with these rules if you know what you're doing with EPL.

Alerts | Summary | Configure | RSA Security Analytics

Rules | Services | Settings | SAMPLE - Blacklist - From in... | New Advanced EPL Rule

Advanced EPL

Write a rule in Event Processing Language.

Rule Name * Advanced SAMPLE - Blacklist - From inside countries that are not the US, Non SMTP Traffic on TCP Port 25 Containing Executable

Description Monitors for non-SMTP traffic on TCP destination port 25 containing executable.

This is using the same criteria from the rule "SAMPLE - Non SMTP Traffic on TCP Port 25 Containing Executable" with the additional requirement that the country have a source IP that does not appear to be from the United States.

Trial Rule ☑

Severity * Low

Query *

```
@Name('Module_55fc7146a7c8c6844ce189a4_Alert')
@Description('Monitors for non-SMTP traffic on TCP destination port 25 containing executable.

This is using the same criteria from the rule \"SAMPLE - Non SMTP Traffic on TCP Port 25 Containing Executable\" with the additional requirement that the country have a source IP that does not appear to be from the United States.')
@RSAAlert(oneInSeconds=0)

SELECT * FROM Event(
  /* Statement: Non SMTP Traffic on TCP Port 25 Containing Executable */
  (service NOT IN ( 25 ) AND tcp_dstport IN ( 25 ) AND extension IN ( 'exe' , 'com' , 'vb' , 'vbs' , 'vbe' , 'cmd' , 'bat' , 'ws' , 'wsf' ,
```

Notifications Global Notifications

Output	Notification	Notification Server	Template
No parameters to edit.			

☐ Output Suppression of every minutes

Enrichments Settings

Output	Enrichment Source	ESA Event Stream Meta	Enrichment Source Column Name
☑ GeoIP	Default GeoIP	src_ip	ipv4

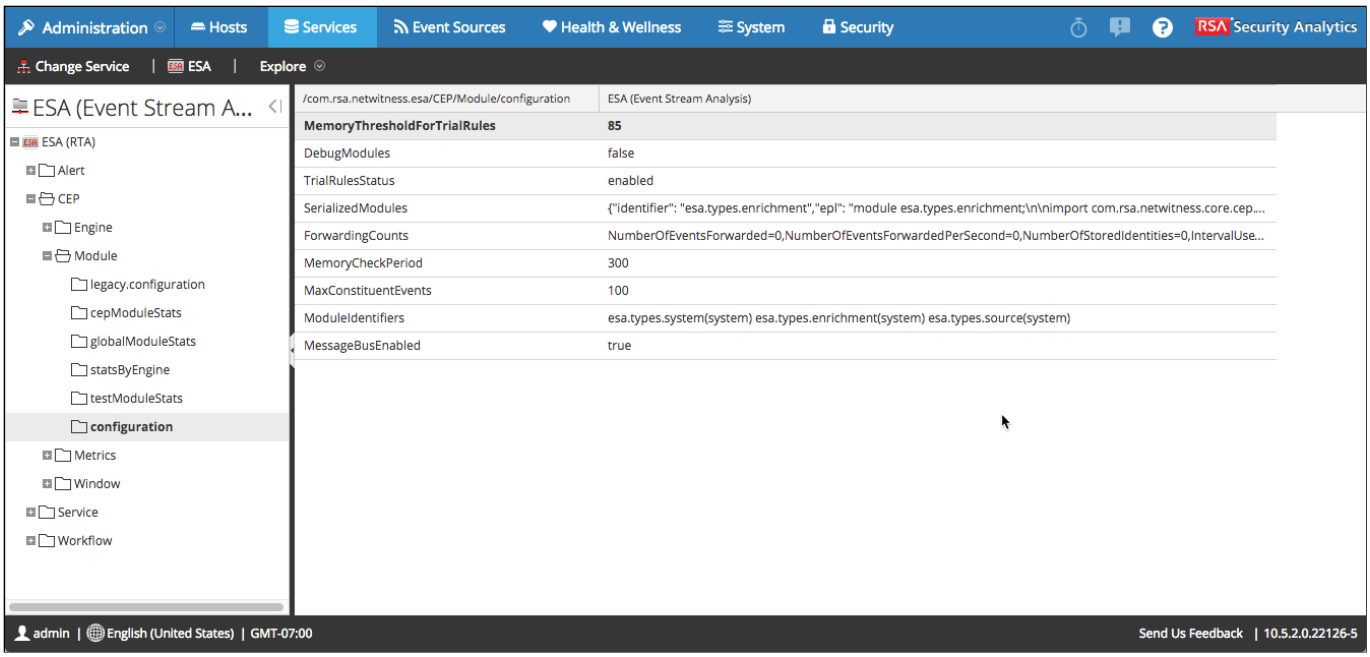
Save | Close | Show Syntax | * = required field

admin | English (United States) | GMT-07:00 | Send Us Feedback | 10.5.2.0.22126-5

Trial Rules

One of the first things you should know about ESA rules is that every new rule has the potential to bring down the entire ESA service, by consuming too much memory (this includes rules from Live). This can happen if rules are poorly written, e.g. too broad. Trial mode is ESA's primary safeguard against this. Trial mode works this way: when combined trial rule memory usage reaches 85% (configurable) of JVM memory, ALL trial rules will be disabled. Should this occur, you will need to go to the **Alerts->Configure->Services** tab to re-enable the trial rules.

The memory threshold is configurable in **Administration->Services-><YourESAservice>->Explore**, then navigate to **/CEP/Module/Configuration**. The the node to configure is **MemoryThresholdForTrialRules**.



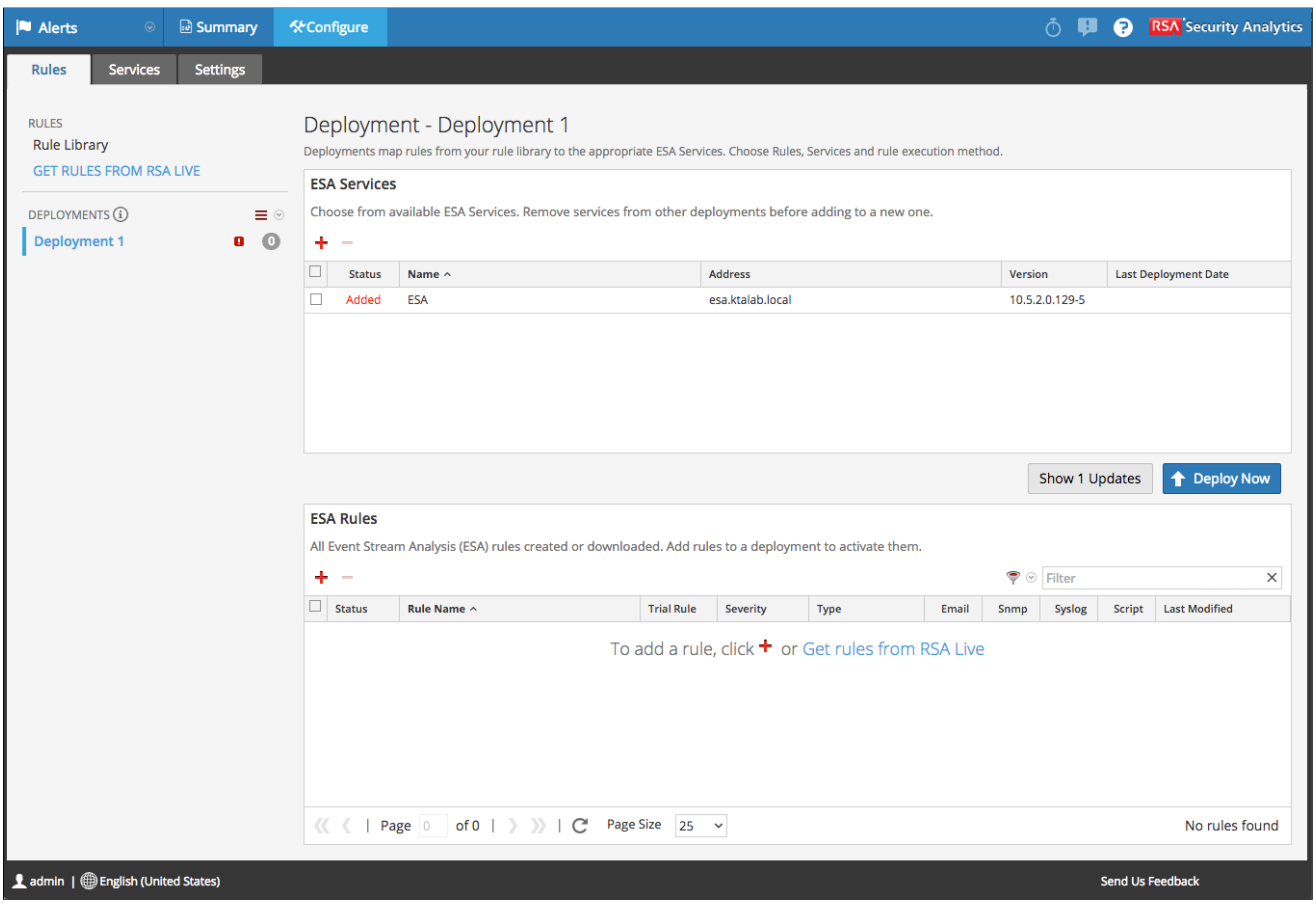
Deploying ESA Rules

To deploy ESA rules to an ESA engine:

1. Navigate to **Alerts->Configure->Rules.**

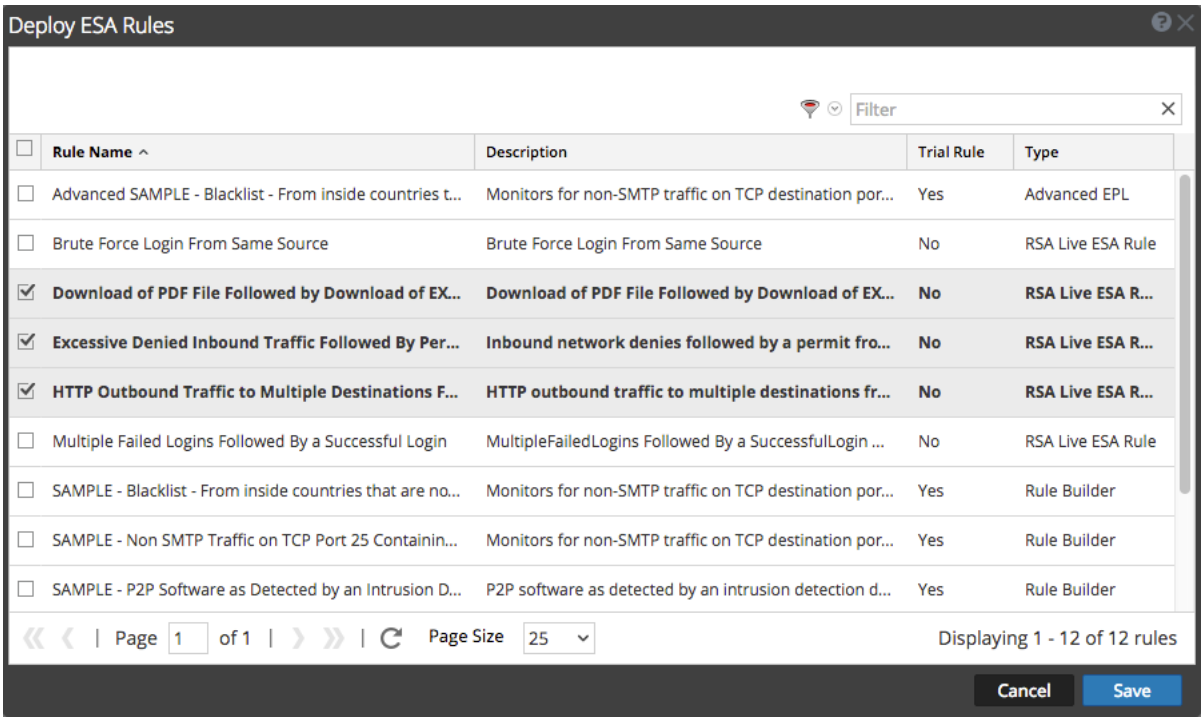
2. Click on a Deployment Group.

3. Ensure you've at least one service in the ESA Services area.



4. Under ESA Rules, click the + button

5. Choose the rules you wish to add to the Deployment Group and click **Save**.



6. If you click on the icon next to the Deployment Group, or on the **Show Updates** button you'll now see the rules you just added, which indicates that the group is ready to be pushed.

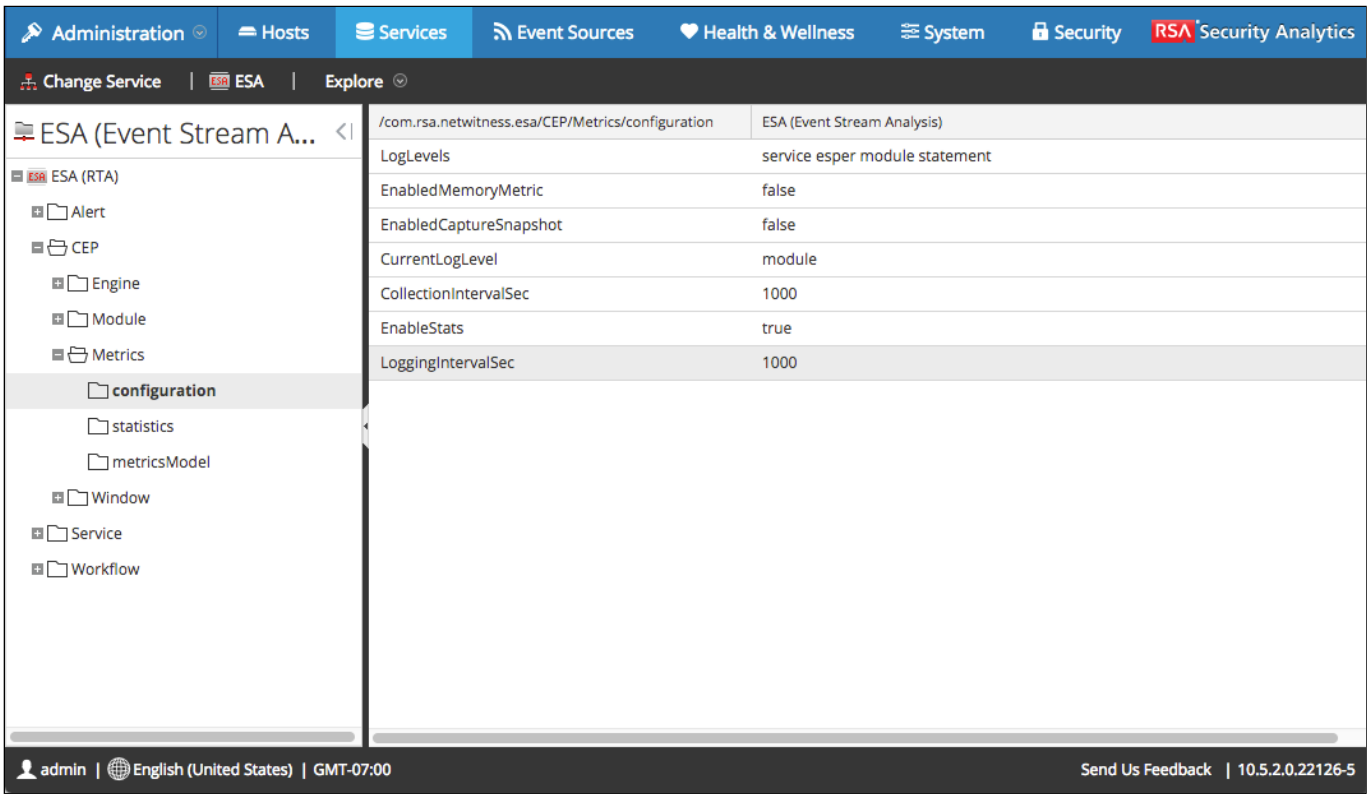
7. Click the Deploy Now button to push the policy. It should now be active, and assuming there's matching traffic, alerts should begin firing.

Memory Statistics for ESA Rules

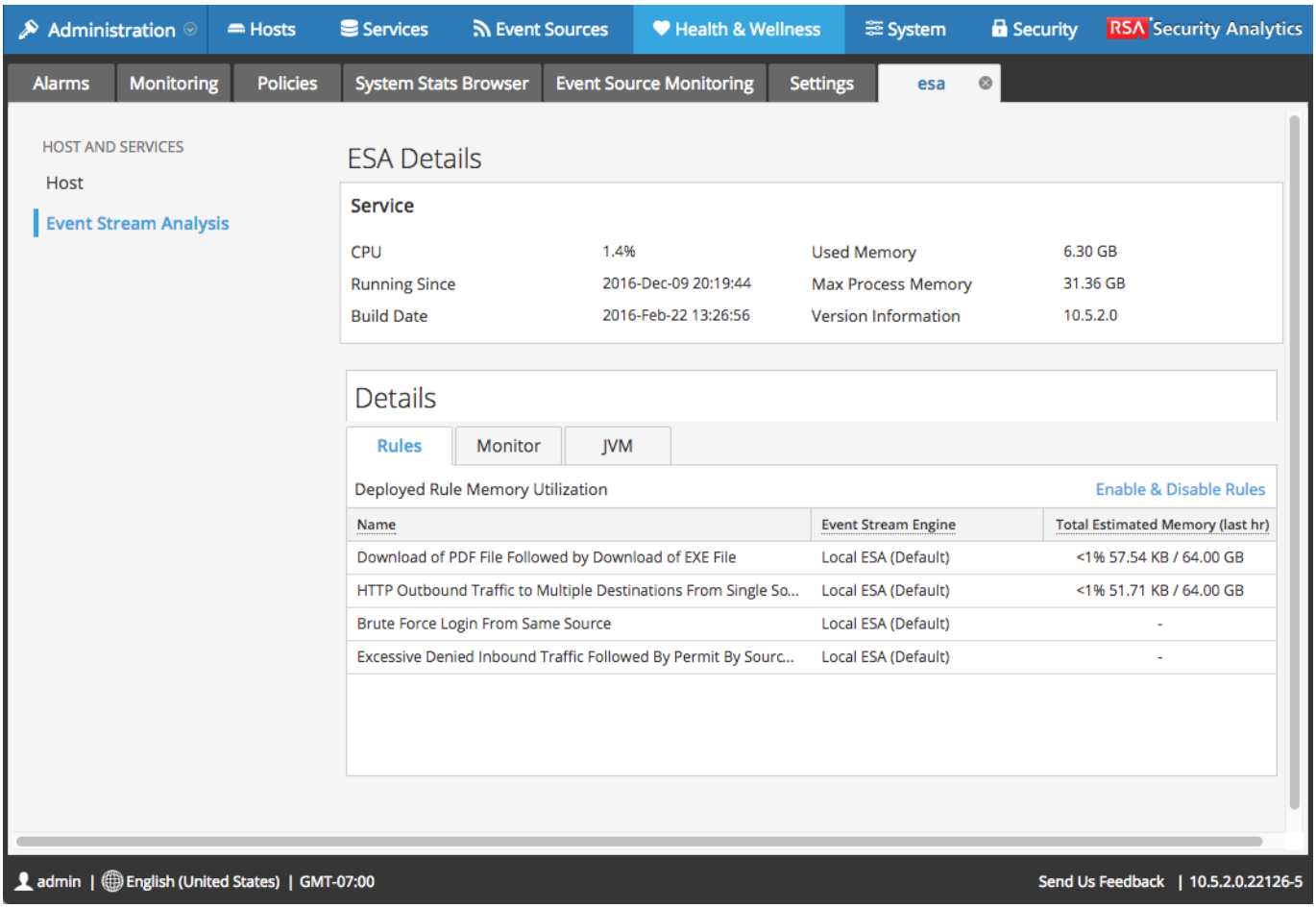
If your ESA service has gotten into the very bad habit of failing, the reason almost surely is that one or more rules is consuming too much memory. This could be because you don't have enough memory, but this shouldn't be the case with RSA appliances. The most common cause is a poorly-written rule, or a rule is storing too many events in memory. To aid in the debugging of this, we can activate memory statistics for ESA rules.

Top Tip: This isn't something to be taken lightly - use it only when you have to! It will negatively impact ESA performance.

Getting on with it, go to **Administration->Services->*<YourESAService>*->Explore**. Now navigate to **/CEP/Metrics/Configuration**. Change the value of **EnableStats** to **true**.



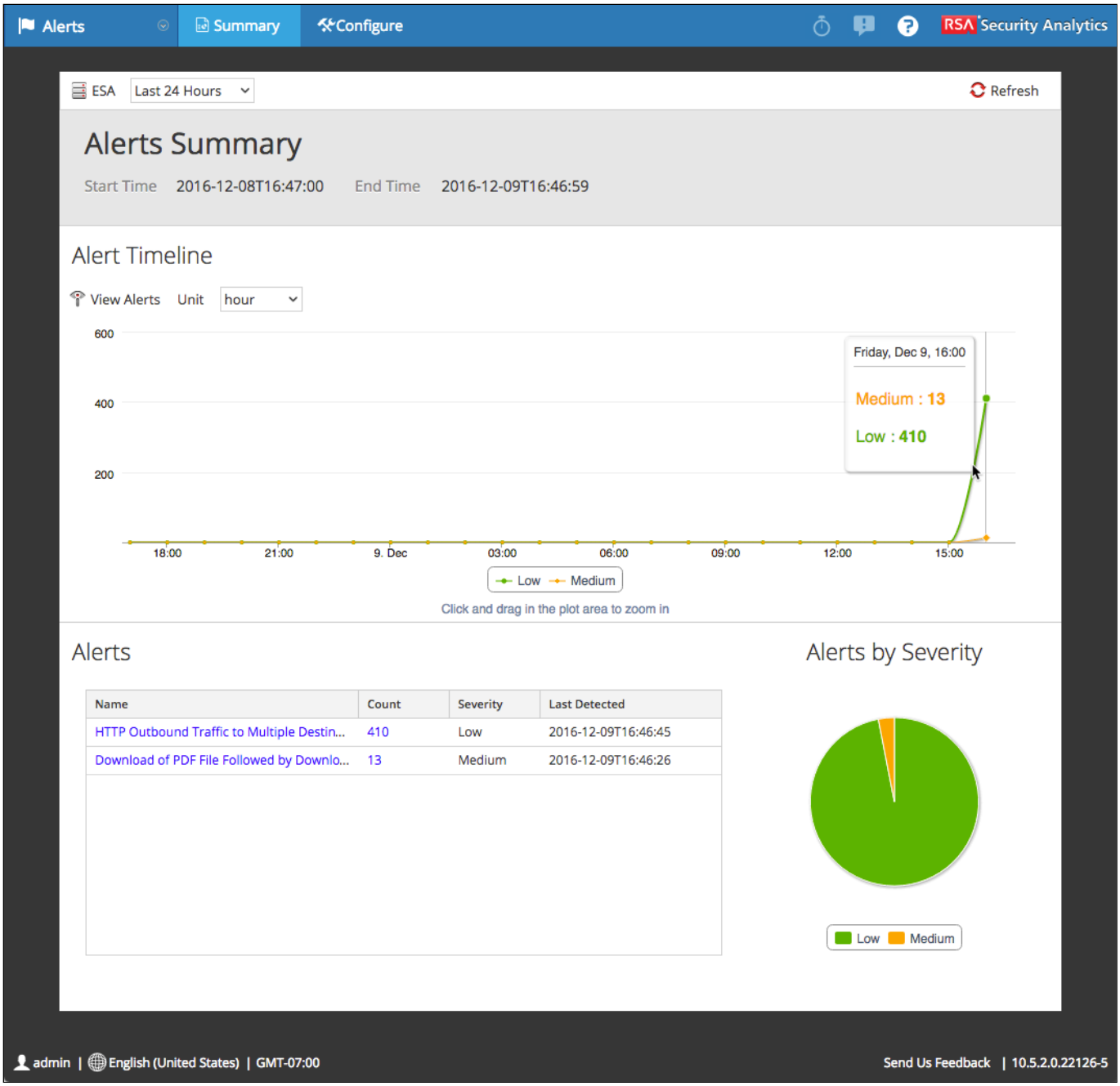
Now Navigate to **Administration->Health & Wellness->Monitoring**. Click on the ESA service link. Under the Rules tab, every deployed rule will be listed, as well as how much memory it is consuming.



Et voila!

ESA Alerts

Now that we've got configuration out of the way, let's have a look at the alerts which it produces. We'll navigate now to **Alerts->Summary**, selecting our lone ESA service in the selector dialog.



This screen shows the alerts that ESA has generated within a selectable timeframe, by default 24 hours. We can drill in on parts of the chart by drag-clicking on the chart.

We can also access alert data by clicking on an alert name.

Alerts | Summary | Configure | RSA Security Analytics

Back to Summary

Date	Name	Severity	# of Events
2016-12-09T16:46:26	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:46:17	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:46:08	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:45:36	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:45:30	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:45:14	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:44:48	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:44:43	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:44:30	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:44:02	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:43:59	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:43:38	Download of PDF File Followed by Download of EXE File	Medium	2
2016-12-09T16:43:13	Download of PDF File Followed by Download of EXE File	Medium	2

Page 1 of 1 | 25 | Displaying 1 - 13 of 13

admin | English (United States) | GMT-07:00 | Send Us Feedback | 10.5.2.0.22126-5

We can then further drill into an individual alert by double-clicking on one. We can see the meta of the sessions in the **Event Meta** tab, and see raw logs and link to Investigation in the **Events** tab :

Download of PDF File Followed by Download of EXE File

Description	Download of PDF File Followed by Download of EXE File
Time	2016-12-09T18:35:00
Severity	Medium
# Of Events	2

Event Meta | Events

Date	Source	Destination	Username	Alias Host
2016-12-09T18:34:36	192.168.0.51	2.23.144.54		partnerdirect.dell.com, partnerdirect.dell.com, partner...
2016-12-09T18:34:37	192.168.0.51	80.239.217.19		downloadmirror.intel.com

Additional Meta

action	GET
alert	large session size
alert.id	nw32505
asn.dst	1299
client	Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:33.0) Gecko/20100101 Firefox/33.0
content	spectrum.filter
country.dst	Sweden
countrycode.dst	EU
did	decoder
directory	/18717/eng/
domain.dst	teliacarrier.com
esa.time	2016-12-10T01:35:00.212Z
eth.dst	78:54:2e:9f:10:28

Close

Download of PDF File Followed by Download of EXE File

Description	Download of PDF File Followed by Download of EXE File
Time	2016-12-09T16:46:17
Severity	Medium
# Of Events	2

Event Meta | Events

Export Logs

Date	Id	Raw Content
2016-12-09T00:47:41	**24613224**	
2016-12-09T00:47:42	24613416	

Close

By clicking on the Events tab, we can access the event's raw payload in Investigation's Event Reconstruction. Remember that in ESA parlance, an **Event** is another name for **Session**. We can also export the log data of the constituent events, if the events are log events.

Investigation | Navigate | Events | Malware Analysis | RSA Security Analytics

Event Reconstruction

service	id	type	source	destination	service	first packet time	last packet time
Concentrator	24613224	Network Session	192.168.0.51 : 40038	2.23.144.54 : 80	80	2016-12-08T17:47:41.627	2016-12-08T17:47:59.169

Request & Response | Top To Bottom | View Text | Actions | Cancel

Request

```
GET /sites/channel/Documents/Dell-Precision-M4800-Spec-Sheet.pdf HTTP/1.1
Host: partnerdirect.dell.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:33.0) Gecko/20100101 Firefox/33.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: GAFed_Identity=9dphbC5OzAb9/BqDJj8tcLxPjPIl2UDexU5lSXoF5TctQfpOUMj6cAOuL28JX4aldGMV8LN4u8gTDSu2V48UCA==;
eSupId=SID=f50fe564-bb34-40c1-9959-440c9ca74883; lwp=c=us&l=en&cs=19&s=dhs; eSupportSegment=c=us&l=en&cs=19&s=dhs;
mbox=PC#1425644660198-289324.26_18#1426858608|session#1425648304782-806791#1425650868|check#true#1425649068;
s_fid=0932BE4999EC60E3-1170688E880115F0; s_c49=c%3Dus%26l%3Den%26s%3Ddhs%26cs%3D19%26servicetag%3D6wh2h12;
s_ppv=us%257Cen%257Cdhs%257C19%257Cesupport-home%257Cdriversdetails%257Cindex%2C45%2C45%2C929; cidlid=%3A%3A;
s_channelstack=%5B%5B%27Direct%2520Load%27%2C%271425644660998%27%5D%2C%5B%27Direct%2520Load%27%2C%271425648306181%27%5D%5D;
sessionTime=2015%2C2%2C6%2C13%2C24%2C20%2C998;
s_hwp=19%7C%7Cnull%7C%7C6%3A3%3A2015%3A14%3A44%7C%7CN%7C%7CN%7C%7Cnull%7C%7C81%7C%7Cnull%7C%7Cnull%7C%7CN%7C%7Cnull%7C%7Cnull%7C
%7C6wh2h12; s_vnum=1457180660998%26vn%3D2; s_cc=true; s_vi=[CS]v1|2A7CCE3A05308E97-40000301A0156DF6[CE]; s_sq=%5B%5BB%5D%5D;
OLRProduct=OLRProduct=6wh2h12|; gpv_pn=us%7Cen%7Cdhs%7C19%7Cesupport-home%7Cdriversdetails%7Cindex; s_depth=4; s_dl=1;
s_invisit=true
Connection: keep-alive
```

Response

```
HTTP/1.1 200 OK
Content-Length: 470704
Content-Type: application/pdf
Last-Modified: Tue, 03 Mar 2015 10:11:39 GMT
ETag: "{AE9EAD84-E608-4978-A888-F1B7BB8D1EB0},147"
Server: Microsoft-IIS/8.0
X-SharePointHealthScore: 0
ResourceTag: rt:AE9EAD84-E608-4978-A888-F1B7BB8D1EB0@00000000147
Public-Extension: http://schemas.microsoft.com/repl-2
SPRequestGuid: 41b4ef9c-e356-70f2-1009-b99ac4bf2339
request-id: 41b4ef9c-e356-70f2-1009-b99ac4bf2339
X-FRAME-OPTIONS: SAMEORIGIN
SPRequestDuration: 5936
SPIisLatency: 160
X-Powered-By: ASP.NET
X-Content-Type-Options: nosniff
X-MS-InvokeApp: 1; RequireReadOnly
Cache-Control: private, max-age=0
Expires: Fri, 06 Mar 2015 13:47:03 GMT
Date: Fri, 06 Mar 2015 13:47:03 GMT
Connection: keep-alive


%PDF-1.4
%....
442 0 obj
<</Linearized 1/L 470704/O 444/E 262228/N 2/T 461743/H [ 3567 481]>>
endobj
```

processed **100** of 659 packets; 1 new event(s) | Show Reconstruction Log

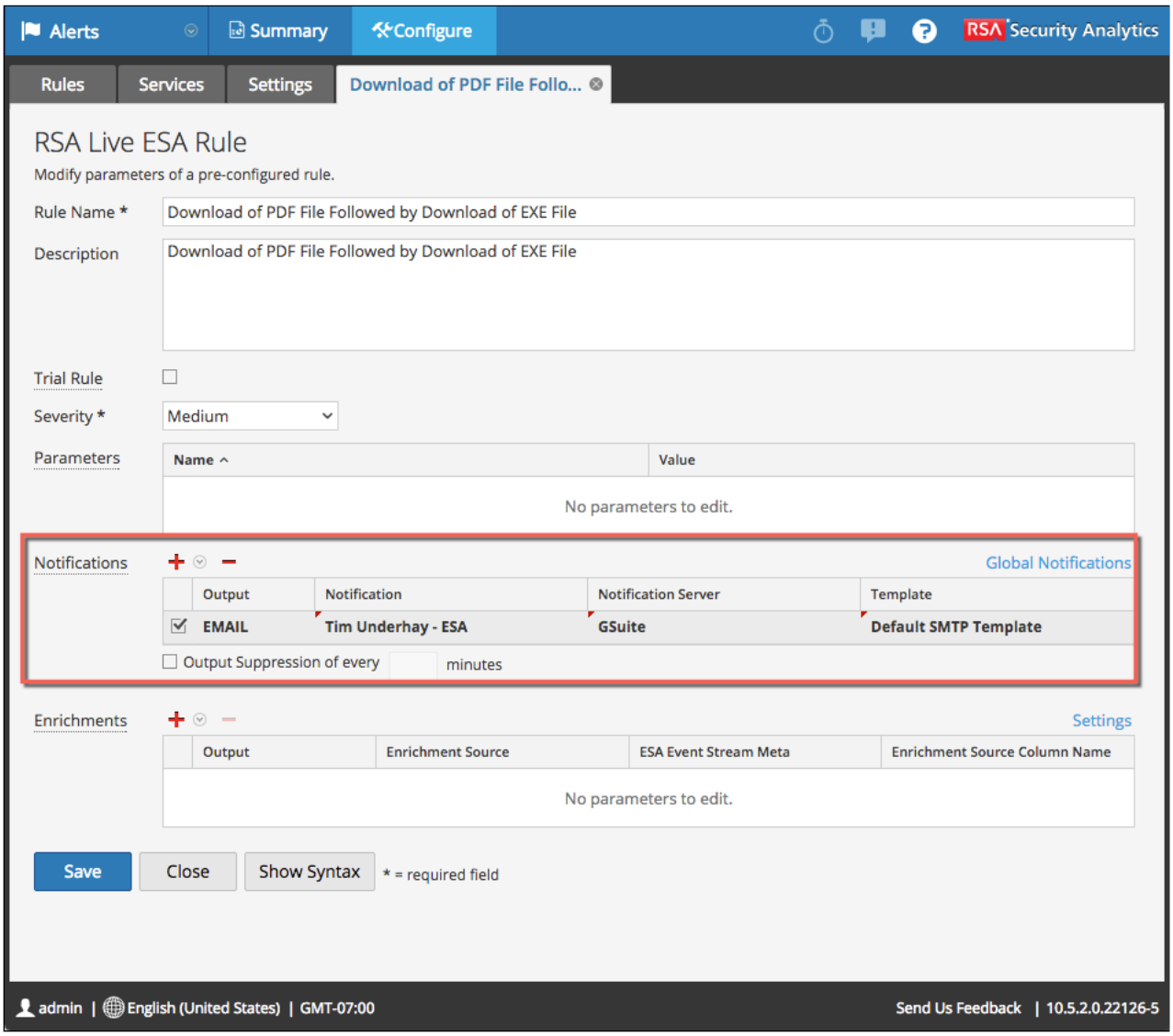
admin | English (United States) | GMT-07:00 | Send Us Feedback | 10.5.2.0.22126-5

Email, SNMP Trap, and Syslog Alerts

ESA's Email, SNMP Trap, and Syslog alerts work in almost exactly the same manner as those for Health & Wellness and Event Source Monitoring (using Global Notifications), with one glaring exception: the inability, out-of-the-box, to set the subject line in alert emails to anything other than a simple string, such as "ESA Alert". Do not fret this, though, as Ken-Tech has already dispatched this bother.

To configure an ESA to alert with one of these mechanism, simply set the notification parameters within an ESA rule. For some reason, ESA's templates do not say "ESA" in their names, but they all start with "Default". E.g. **Default SMTP Template**, **Default Syslog Template**, **Default SNMP Template**, and **Default Script Template**.

Don't forget to save the rule, and push the change to your Deployment Groups!

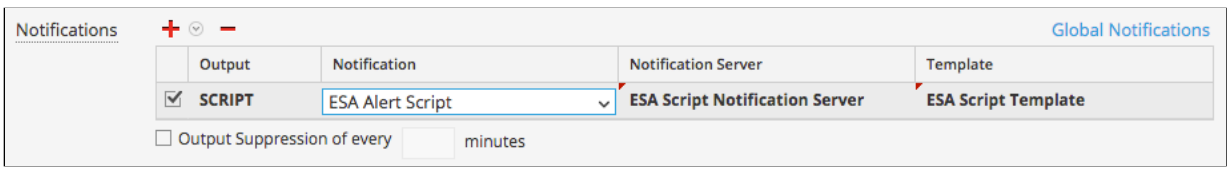


Output Suppression

If an ESA rule fires a lot, you can get pummelled with emails. To stop this from happening, tick the box for **Output Suppression of every *N* minutes**, and type a value to the left of *minutes*. ESA will only send an alert notification once during this time window.

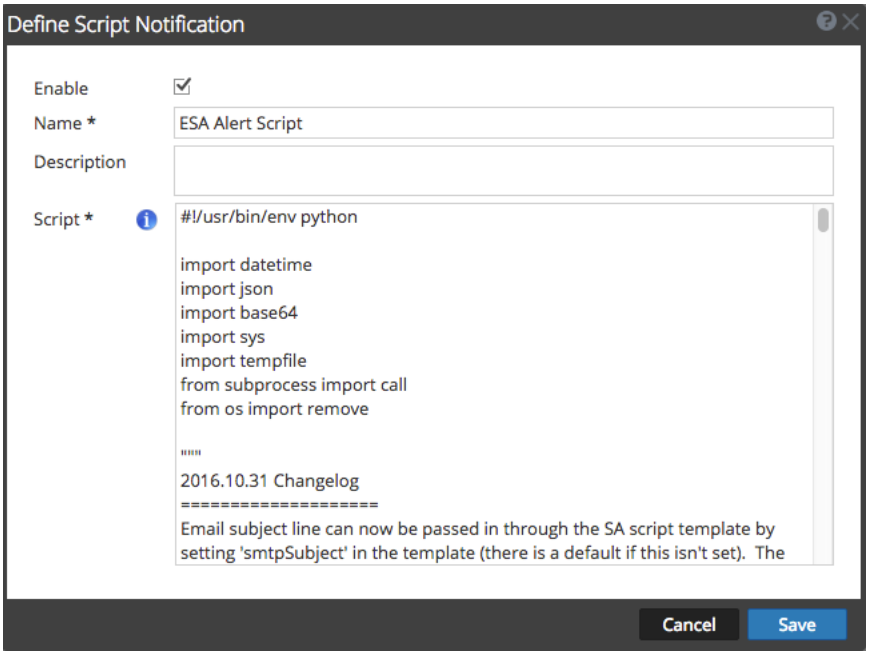
Fixing the Subject Line Issue - Script Notifications

ESA alerts have one very cool feature which H&W and ESM do not, though: the ability to trigger a custom script when an alert is fired. In this example, we'll teach how script notifications work using Ken-Tech's bespoke solution, available here.



1. To see how this works, let's have another gander at **Administration->System->Global Notifications**.

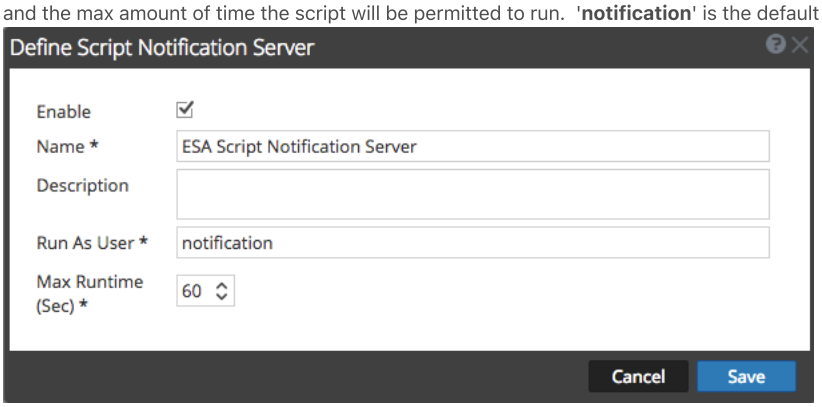
2. In the **Outputs** tab, this is where we define our script. Create a new output of type **Script**, and paste in our script.



Top Tip: Your script can be in any scripting language, as long as it begins with a standard UNIX *shebang* line (**#!**) indicating which interpreter should be used. E.g.

#!/usr/bin/env python
#!/usr/bin/perl
#!/bin/bash

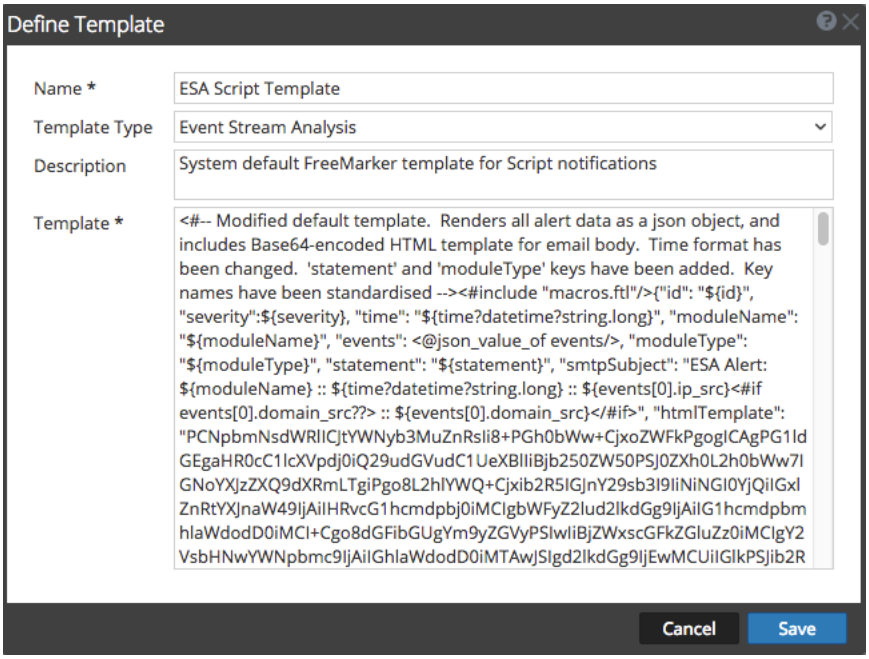
3. On the Servers tab, define a Server of type **Script**. All this really does is specify the local Linux user that the script will run under on the ESA host, and the max amount of time the script will be permitted to run. '**notification**' is the default user, which is pre-configured on the ESA host.



4. Moving over to the Templates tab, define a template for ESA, of type **Event Stream Analysis**. ESA templates use Apache FreeMarker template notation. This template will be passed to your script as a command line parameter. Notice that there are a lot of FreeMarker variables in the template, defined as **${*varName*}**. Also, FreeMarker templates support if-then-else logic, and arrays. In our solution, we are passing alert and event parameters as JSON data, which the script then parses. Here, we pass the email subject line in as JSON attribute **smtpSubject**, and because it's FreeMarker, you can make the subject as flexible as you want, such as including the rule name!

   In addition to that, we also pass in the email body itself as a Base64-encoded JSON attribute called **htmlTemplate**, and the email body is itself a FreeMarker template!. The reason we do that is because we want to pass the email body to the script rather than hard-code it in the script. This is the beauty of our solution - it's flexible - you pass in nearly everything as JSON data, and the only thing which gets hard-coded in the script is your email server config.

   Top Tip: Note that our solution requires a third-party tool called FMPP to process the body of the email, so you can't just do this OOTB with your own script. Have a look at the solution if you have any questions.



The full template here:

```
<#-- Modified default template.  Renders all alert data as a json object, and includes Base64-encoded HTML template for email body.
 Time format has been changed.  'statement' and 'moduleType' keys have been added.  Key names have been standardised --><#include
"macros.ftl"/>{"id": "${id}", "severity":${severity}, "time": "${time?datetime?string.long}", "moduleName": "${moduleName}",
"events": <@json_value_of events/>, "moduleType": "${moduleType}", "statement": "${statement}", "smtpSubject": "ESA Alert:
${moduleName} :: ${time?datetime?string.long} :: ${events[0].ip_src}<#if events[0].domain_src??> :: ${events[0].domain_src}</#if>",
"htmlTemplate":

--extra space added for readability---

"PCNpbmNsdWRlICJtYWNyb3MuZnRsIi8+PGh0bWw+CjxoZWFkPgogICAgPG1ldGEgaHR0cC1lcXVpdj0iQ29udGVudC1UeXBlIiBjb250ZW50PSJ0ZXh0L2h0bWw7IGNoYXJz
```

Stopping ESA

```
service rsa-esa stop
```

Starting ESA

```
service rsa-esa start
```

Key File Locations

- **/opt/rsa/esa** - ESA Directory
- **/opt/rsa/esa/logs/esa.log** - ESA service logfile
- **/opt/rsa/esa/logs/audit/audit.log** - ESA audit log

25. Incident Management

Incident Management, or **IM** for short, is a very cool and relatively new component of SA which helps security teams track and manage security incidents, by taking them through detection through to remediation. It does this by grouping one or more security **alerts** into an '**incident**'. This lesson focuses on how IM fits together with the rest of SA, and the mechanics of its use. We won't delve very much into how an security organisation should operationalise IM, in terms of personnel, processes and procedures.

Incident Management is a service which runs on the SA head server, but its database lives on an ESA server. RSA uses a commercial implementation of MongoDB called TokuMX.

Prerequisites

As the above implies, making use of IM also requires ESA, as the IM database lives on the ESA server.

User Personas

- **Incident Responder** - This is an Analyst who investigates security alerts and incidents.
- **Incident Remediator** - Someone whose job is to handle actions to take action in the environment in response to an incident.
- **SOC Manager** - This persona oversees Analysts, and should be kept informed of incident activity.

Initial Setup of IM

The IM service is installed by default on the SA Head server. Before IM can be utilised, the IM service must first be configured to connect to the IM database that lives on the ESA server. To do this:

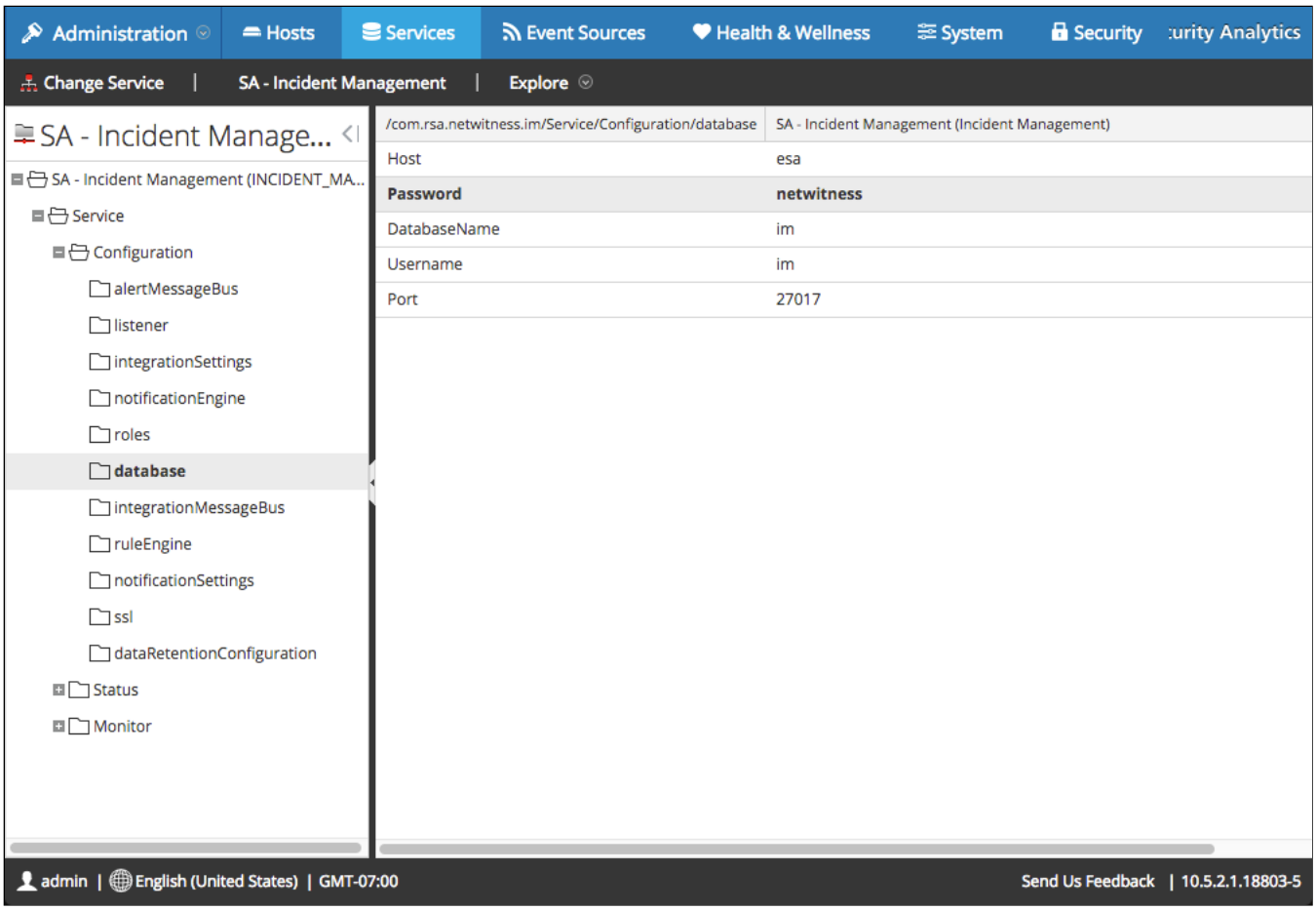
1. Set an initial password for your IM database. SSH to your ESA server and run:

```
# mongo im -u admin -p netwitness --authenticationDatabase admin
```

...and then enter:

```
db.changeUserPassword('im','<new_password>')
exit
```

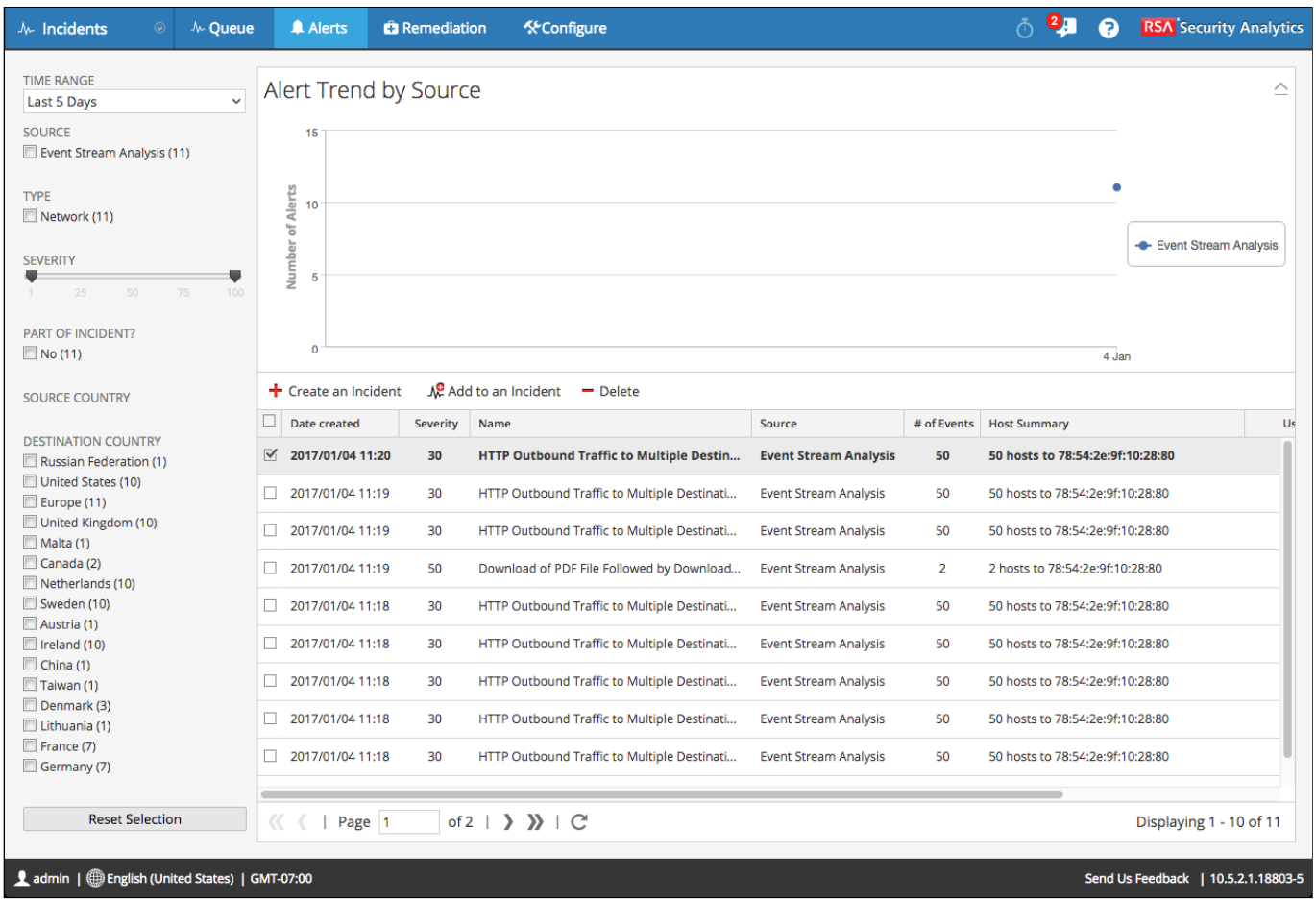
2. Configure the IM service to connect to the database by navigating to **Administration->Services-><YourIMService>->Explore**. In the API tree, navigate to **/Service/Configuration/database**. Set the **Host**, **Username** ('**im**'), and **Password** (whatever you set it to) parameters. The DatabaseName parameter should remain '**im**'.



3. Restart the IM service. SSH to your SA server, and type:

```
# service rsa-im restart
```

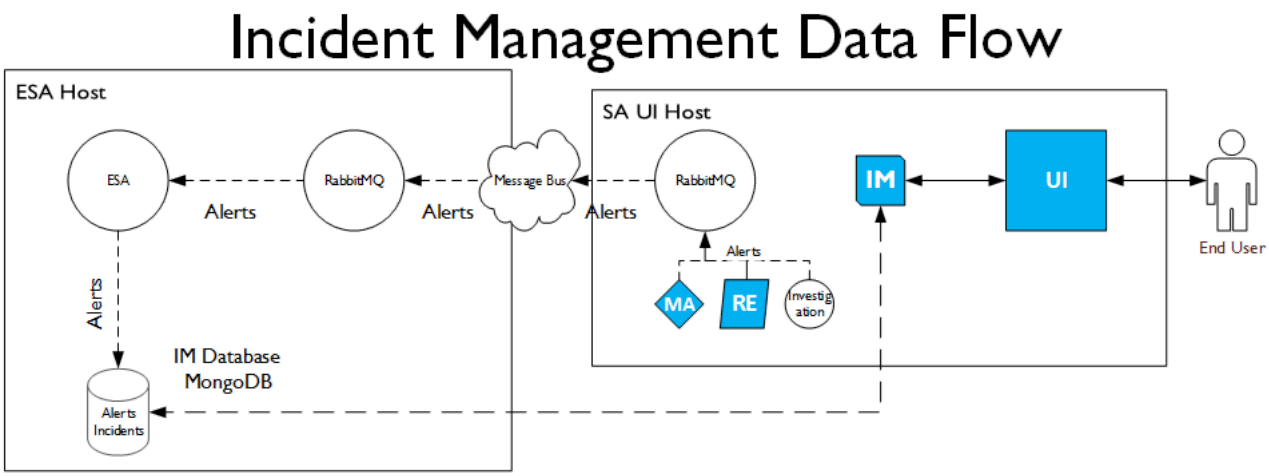
Because we've previously configured ESA to generate alerts, you should now see some alerts in **Incidents->Alerts.**



Feeding Alerts to IM

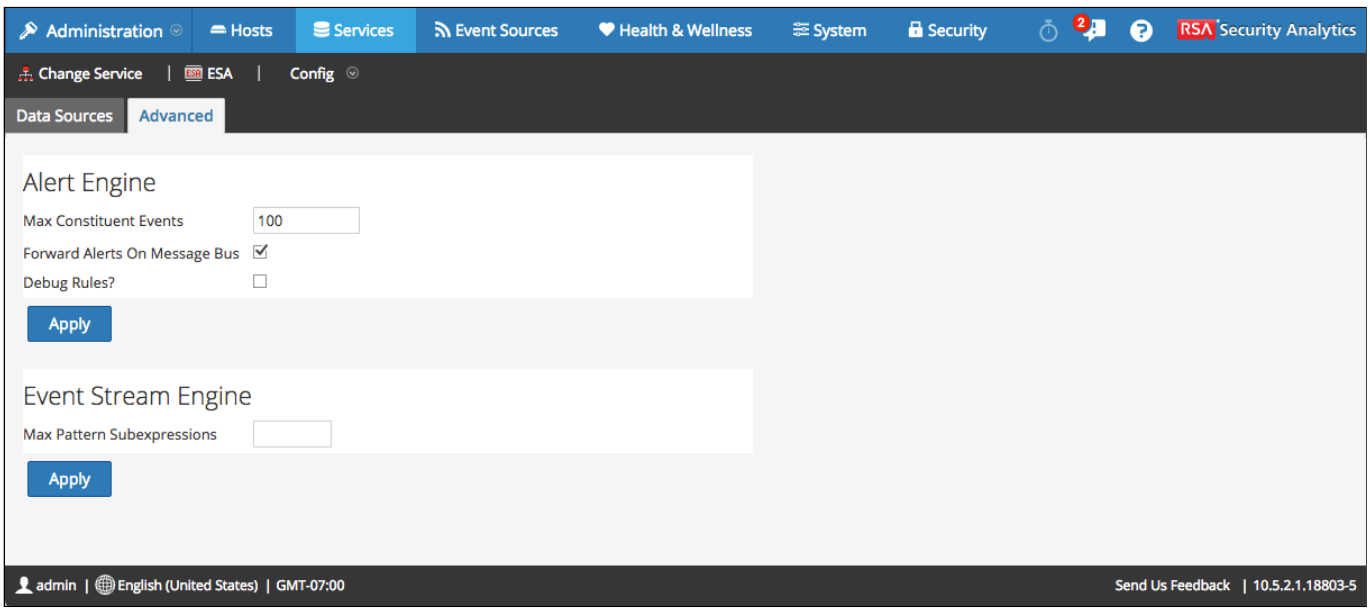
Alerts are fed into IM via the SA Message Bus (RabbitMQ). They are then stored by IM in its MongoDB database on the ESA server. Supported alert sources are:

- ESA
- Investigation
- Reporting Engine
- Malware Analysis

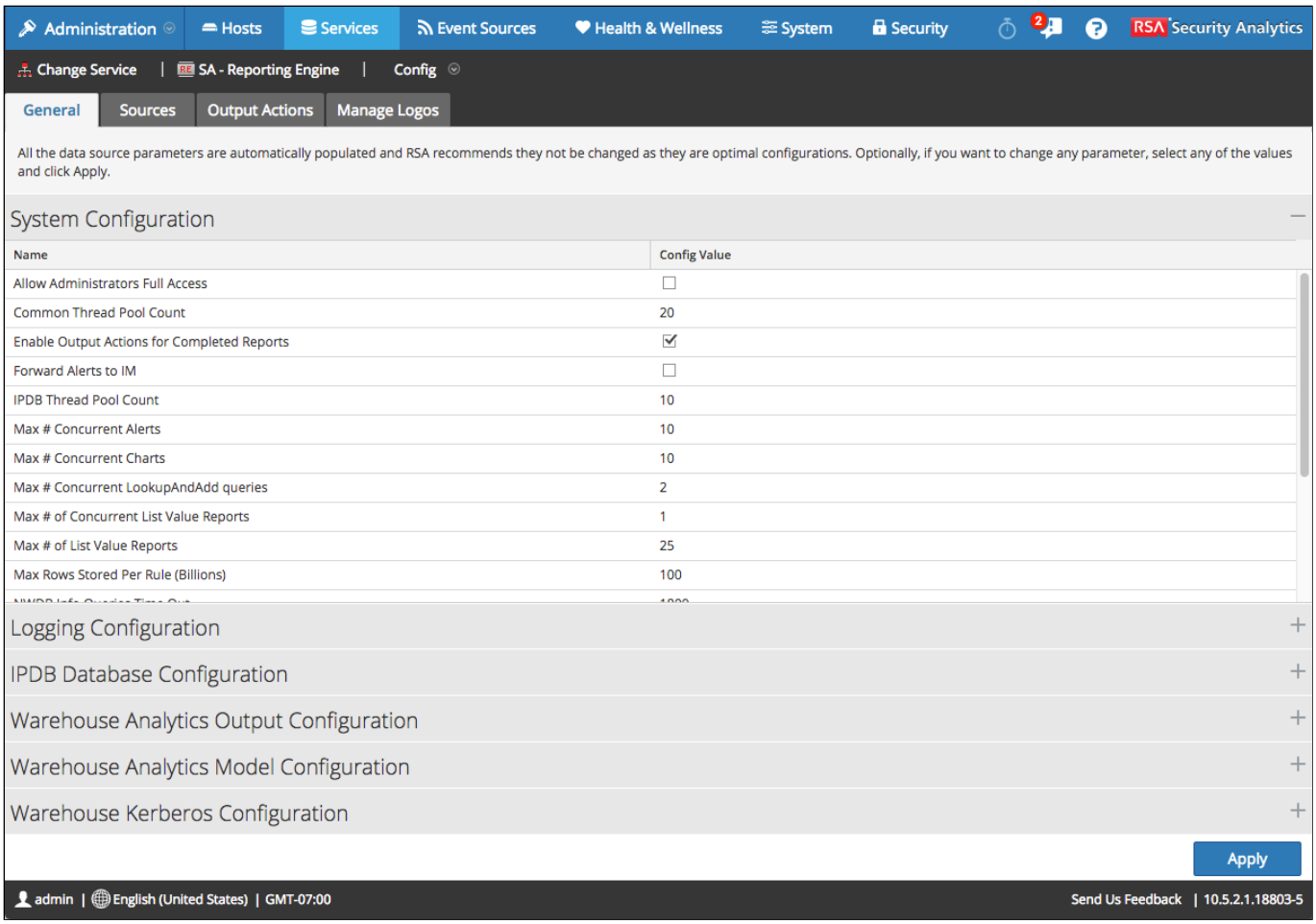


Except for ESA and Investigation, these services are not by default configured to send their alert data to IM. Here's how to enable forwarding of alerts to IM. Without this, alerts from the RE and MA won't show up in IM.

- **ESA -** Navigate to **Administration->Services-><YourESAService>->Config->Advanced.** Tick the box for **Forward Alerts On Message Bus**. This is ticked by default.



- **Investigation** - No action is necessary to enable this.
- **Reporting Engine** - Navigate to **Administration->Services-><YourREService>->Config->Advanced.** Tick the box for **Forward Alerts to IM**. This is not ticked by default.



- **Malware Analysis** - MA is not covered by this course.

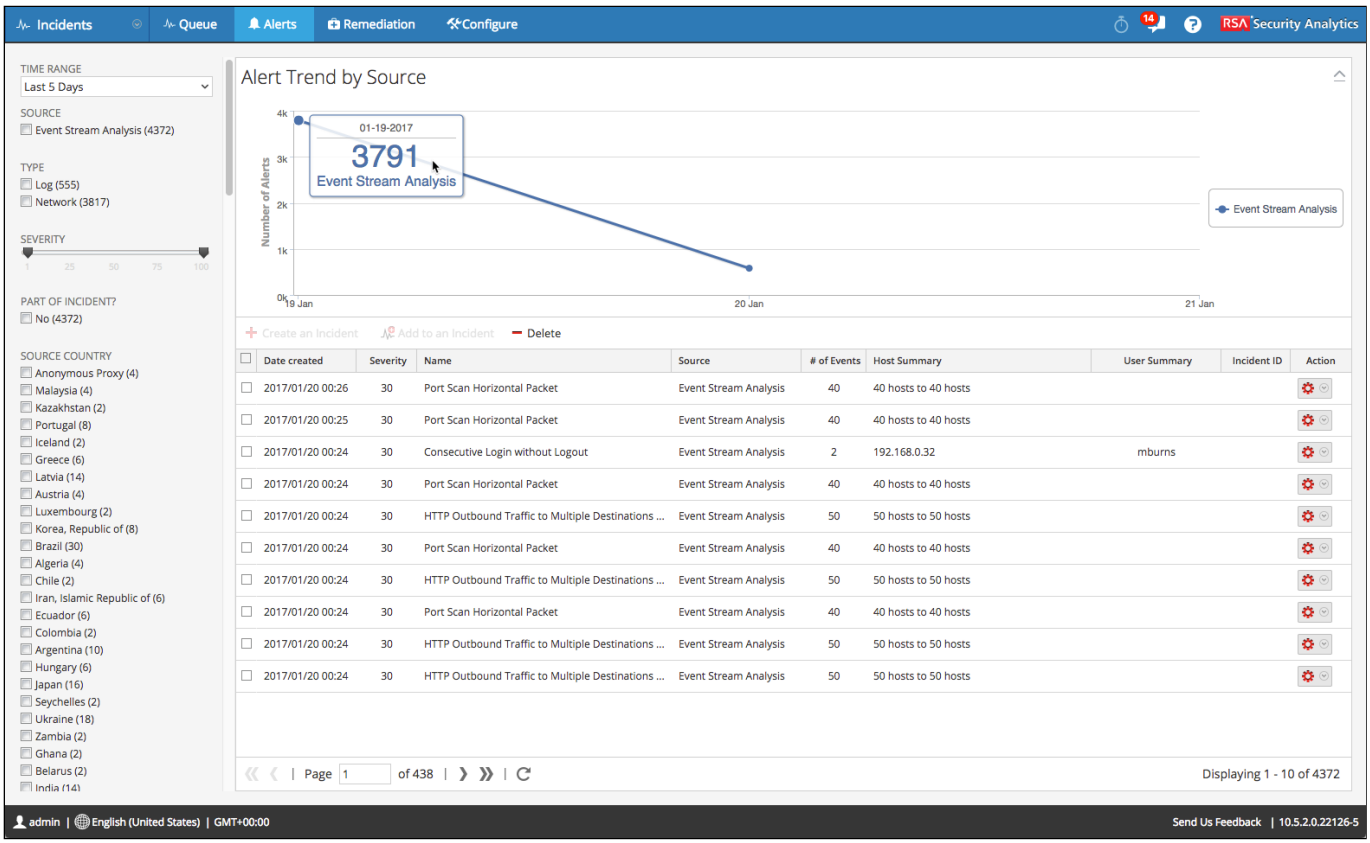
Alerts

What's a Security Alert?

A security alert is a slightly abstract concept. An alert is generated by either ESA, the Reporting Engine, Malware Analysis, or SA Investigation. An alert is essentially a titled grouping of one or more security **events**, which is to say captured sessions. The reliability of each alert is dependant on whether the quality of the underlying rules or the human that created it. The presence of an alert does not necessarily mean that something negative has occurred, but it may warrant further investigation.

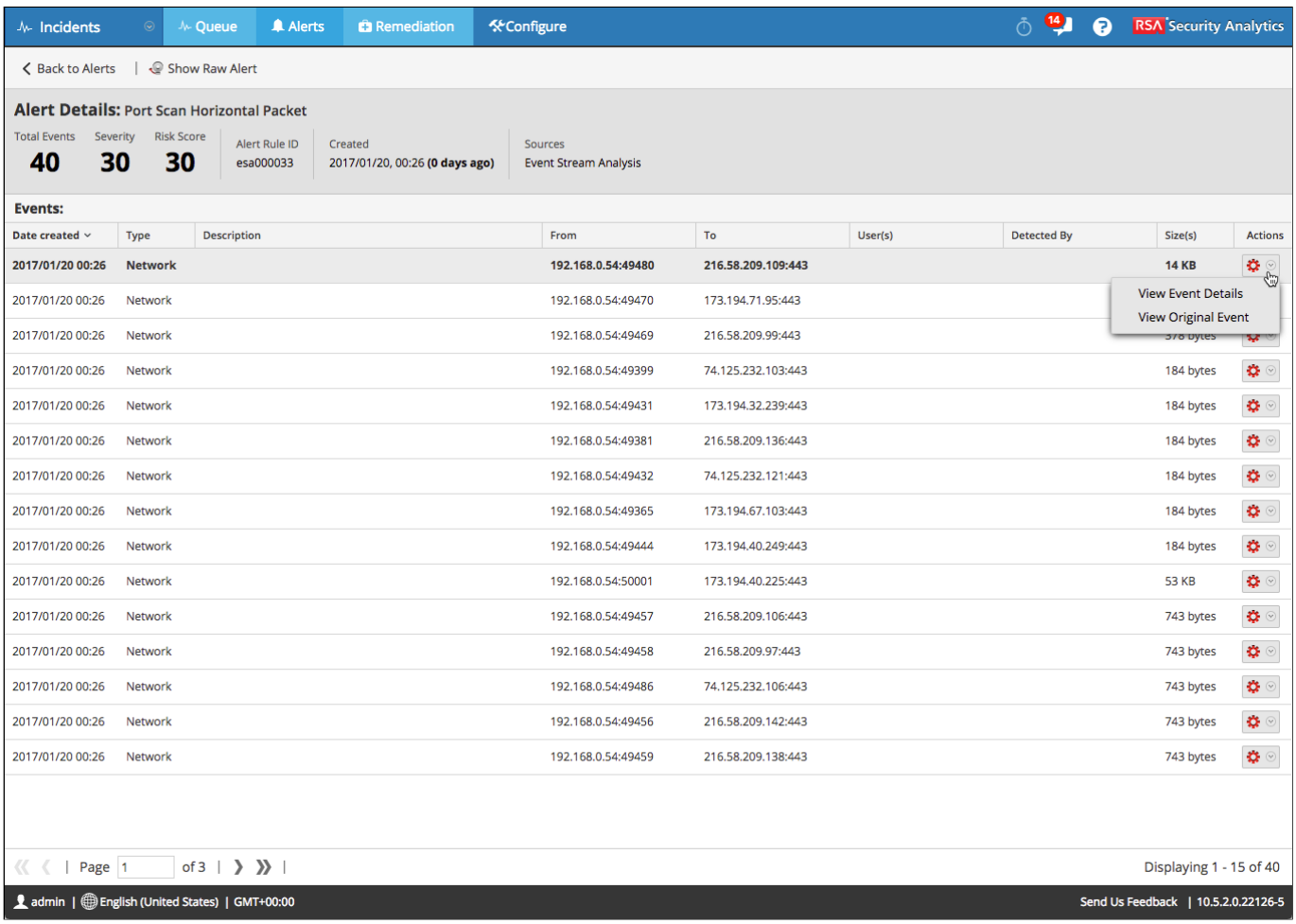
Looking at Alerts

The UI page for viewing Alerts is **Incidents->Alerts**. Here, you'll see a list of alerts, and also filtering options. You can filter by Source type (i.e. ESA, RE, or MA), Log or Packet Alerts, the alert severity, whether the alerts are part of an incident, and also by source and destination country.



There's a graph which shows alert trends and which type of alert it is (ESA, RE, MA).

From **Incidents->Alerts**, double-click on an alert to open it. The view will switch to reveal the individual alert.



Click **Back to Alerts** to return to the list of alerts.

Investigating Alert Event Data

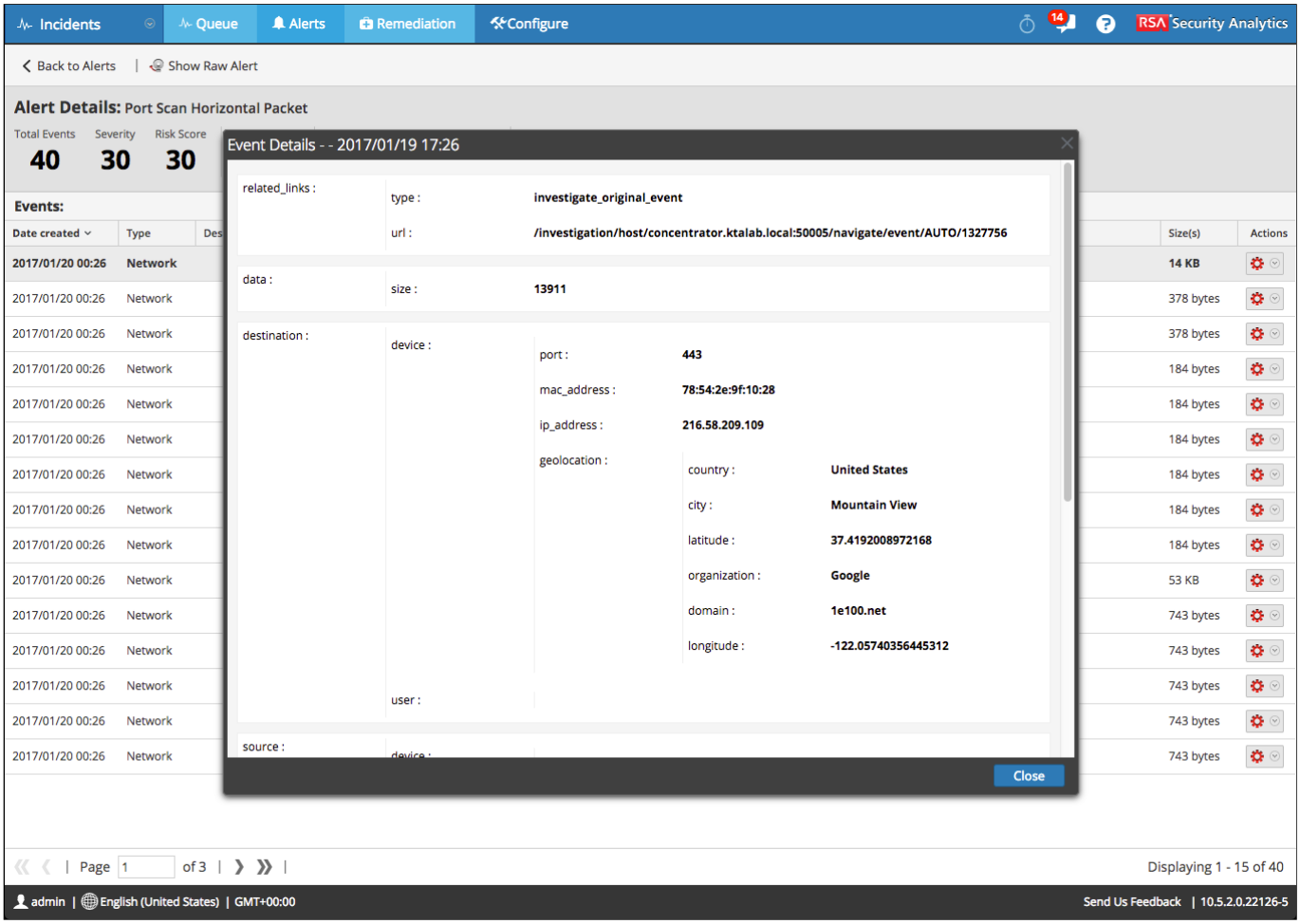
From the Alerts screen, one can pivot into **Investigation** to view an Alert's constituent events. Clicking on icons under the **Action** column of the **Incidents->Alerts** page, you will see options to:

- **Investigate Events** - Shows constituent events in the **Investigation->Navigate** view. It simply executes a query containing individual session ID's. E.g. **'sessionid=1293391||sessionid=1293392||sessionid=1293393||sessionid=1293394||sessionid=1293398||sessionid=1326493||sessionid=13265**

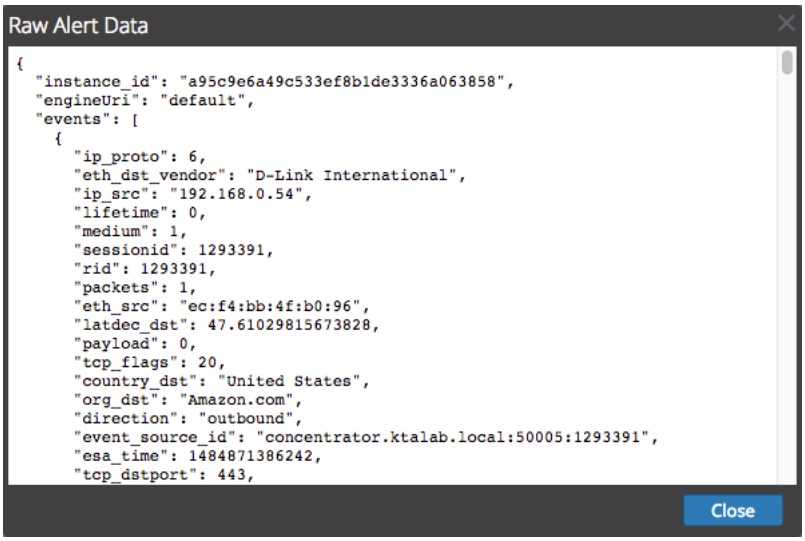
- **Investigate Source IP Address** - Shows all meta for all of the alert's event source IP's in the **Investigation->Navigate** view. The query looks like **'ip.src=54.192.98.138,54.230.97.226,108.160.165.138,54.194.2.86,137.116.224.167,8.36.120.226,65.52.98.231,191.237.171.29,204.79.197.203'**

- **Investigate Destination IP Address** - Shows all meta for all of the alert's event destination IP's in the **Investigation->Navigate** view. The query looks like **'ip.dst=54.192.98.138,54.230.97.226,108.160.165.138,54.194.2.86,137.116.224.167,8.36.120.226,65.52.98.231,191.237.171.29,204.79.197.203'**

By clicking on one of the icons under the **Actions** column in an individual Alert, you can view the original alert data by clicking **View Event Details**. **View Original Event** will take you into Investigation to view the event data.

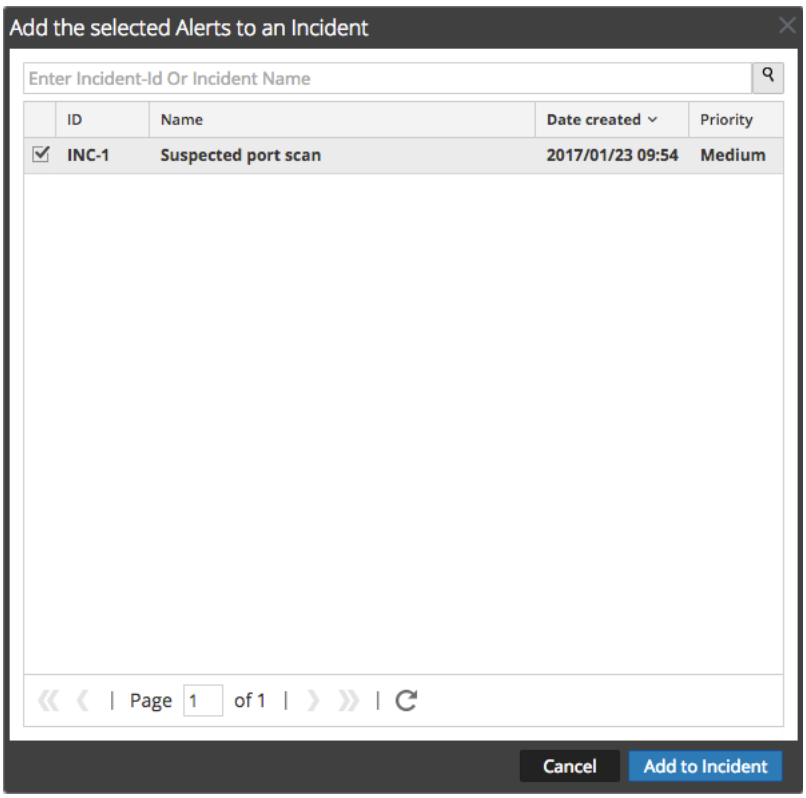


You can view the raw alert data, in JSON notation, by clicking on **Show Raw Alert**.



Adding Alerts to an Existing Incident Manually

Existing alerts can be added to a case directly from **Incidents->Alerts**. Simply select the alerts you want to add to the existing incident, and click Add to an Incident.



Deleting Alerts

Alerts can also be deleted from **Incidents->Alerts**, by use of the Delete button.

Incidents

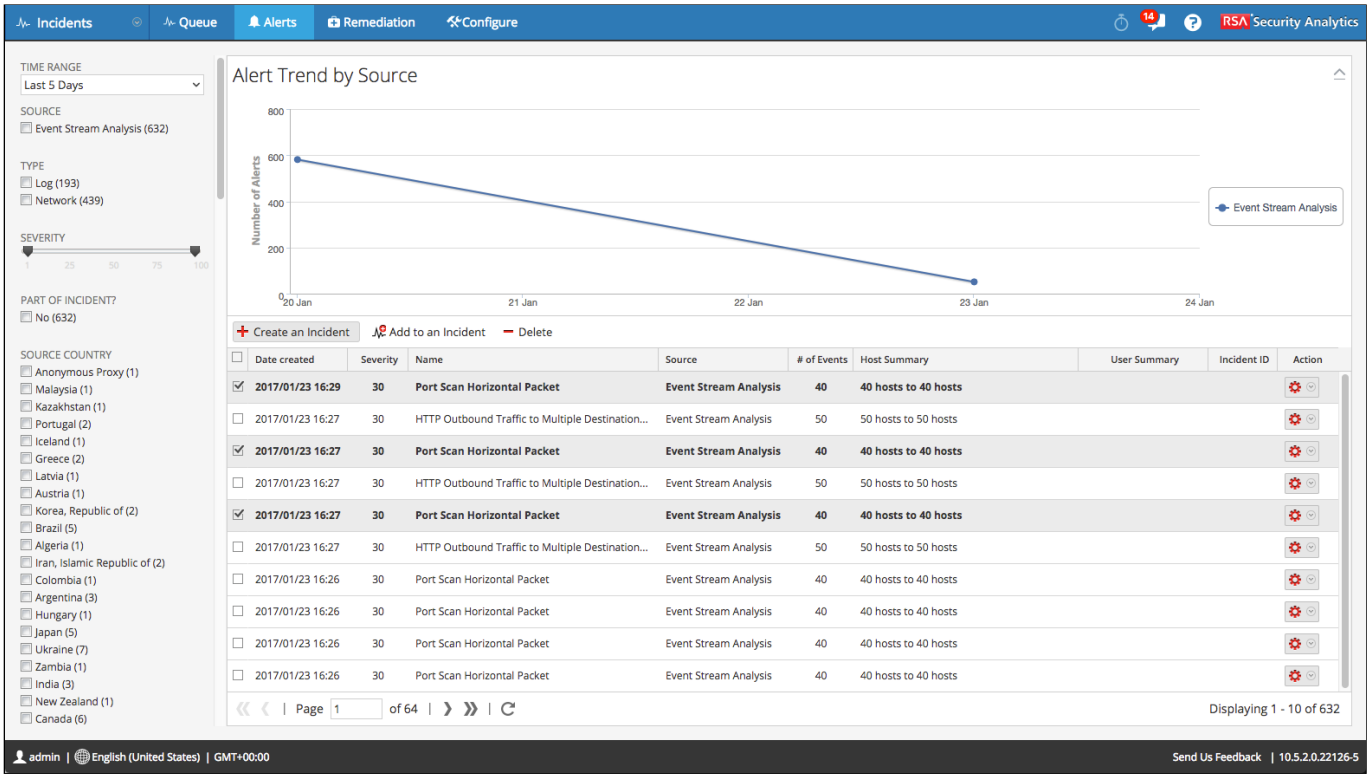
What's an Incident?

An incident is a case record, or 'ticket', which is created when one or more alerts, when viewed together, seem serious enough to warrant further investigation by an Analyst.

Creating an Incident Manually from Alerts

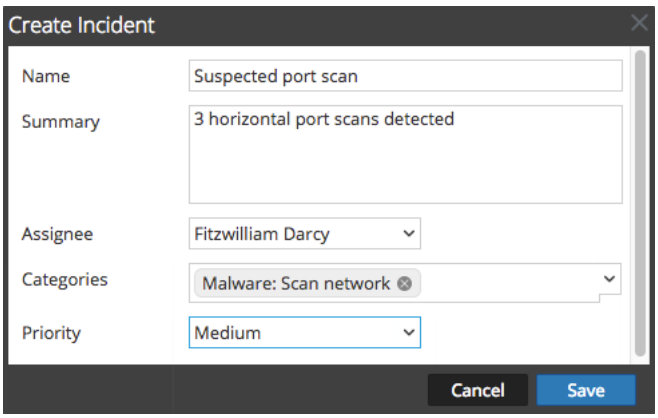
From the **Incidents->Alerts** page of the UI, we can create an incident out of alerts.

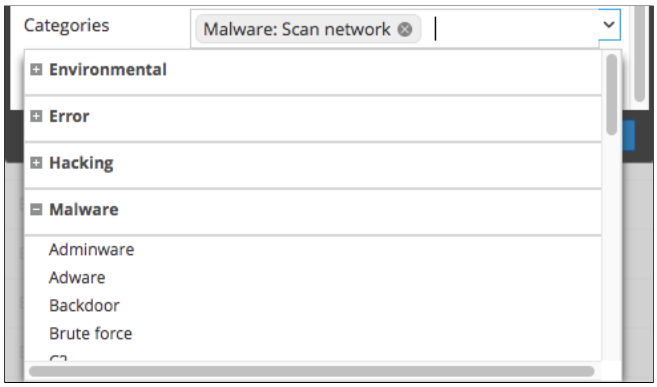
1. Simply select one or more alerts from the list and click **Create an Incident**.



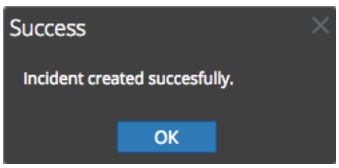
2. Then give it a name, a summary, assignee, one or more categories, and a priority.

Categories are predefined elements that describe the observed or suspected activity. They're broken into headings like Environmental, Hacking, Malware, etc. There are many different categories to choose from, and you are not limited to only one.



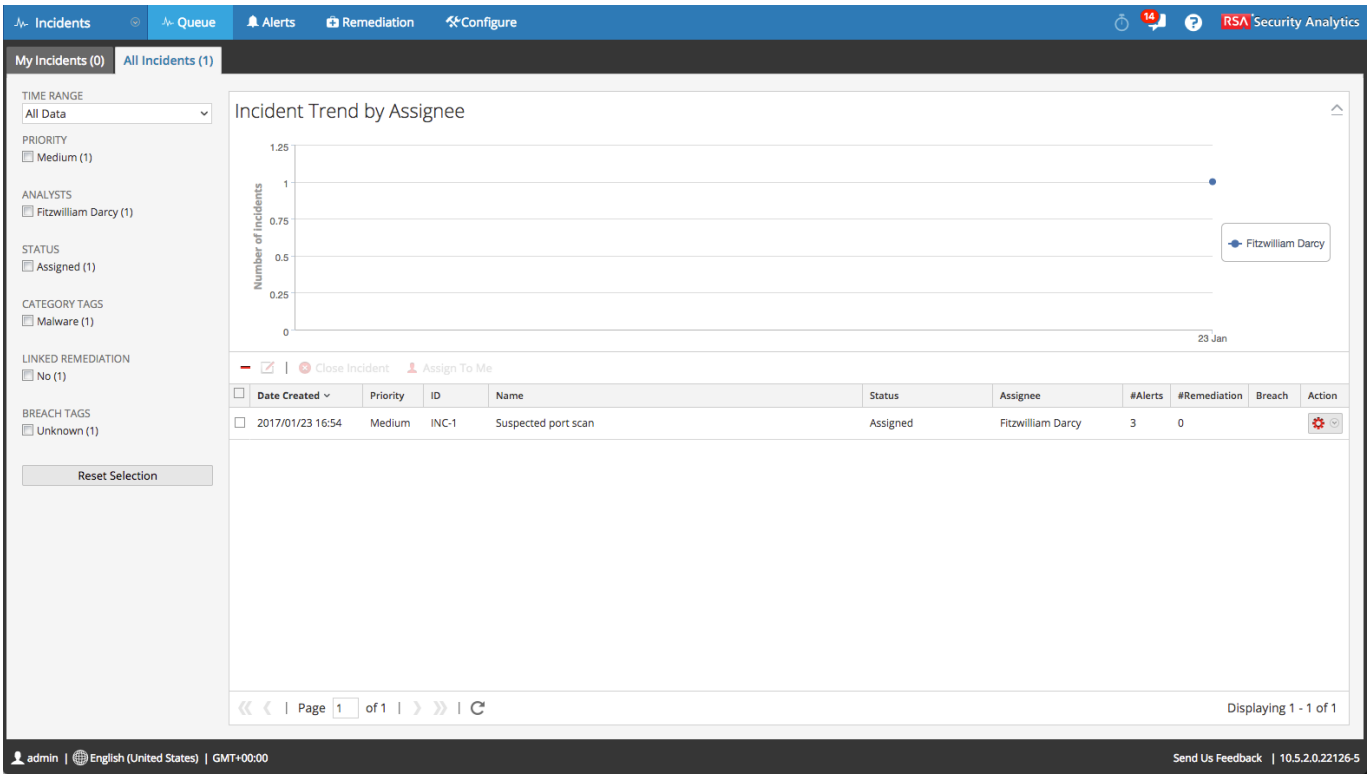


3. Click **Save**. You should see a pop up which says "Incident created successfully."

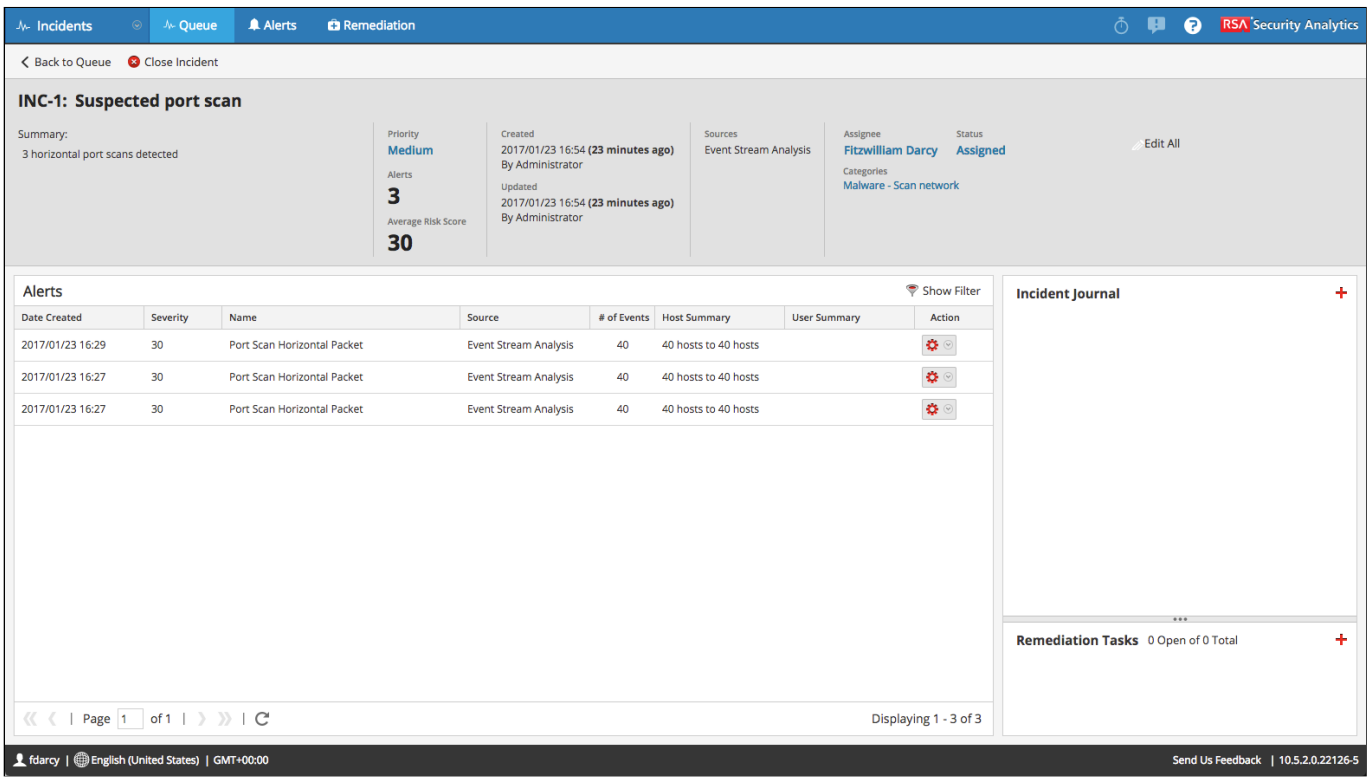


The Incidents Queue

Once you've created an incident, it should be visible on the **Incidents->Queue** page. This page is split into two tabs: My Incidents (the ones assigned to the logged-in user) and All Incidents. Just as in Alerts, Incidents can be filtered. Filtering options include Priority, the analysts, status, category tags, linked remediation tasks, and breach tags.



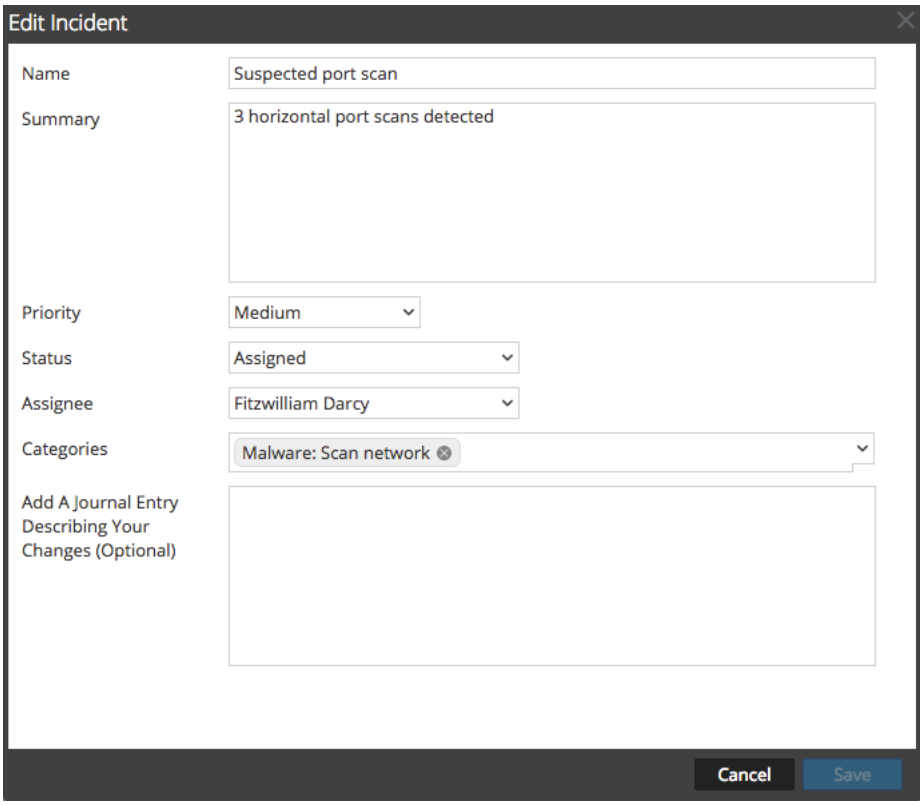
To open an Incident, double-click on it from the **Incidents->Queue** page.



An incident is a space in which to track a security investigation. It contains things like:

- A case priority
- Average risk score, numeric
- An Assignee - an SA user who owns the Incident
- A case status
- Individual alerts
- An Incident Journal - a 'notebook' or log of actions and findings in an Incident.
- Remediation Tasks - These are task records which can be assigned to a queue whereby someone will do something in response to an incident, like change a firewall rule, take a computer offline, or re-image a computer.

Top Tip: The 'edit' ( ) button on the **Incidents->Queue** page does not open a full view of an Incident. It opens a smaller data entry-style window. Make sure you double-click an incident to get the full view.

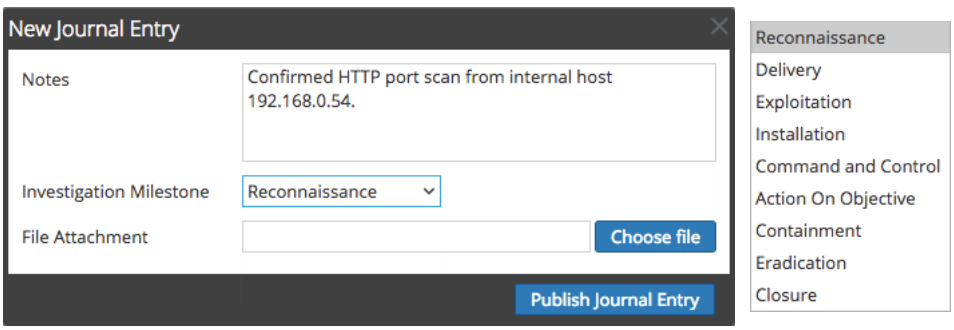


You can access this edit view from within an incident by clicking the Edit All button.

Creating a Journal Entry

You can create a new entry in the Incident Journal by clicking the + button next to Incident Journal. In here, you can add a note, choose an Investigation Milestone, which maps to the cyber kill chain, with three additions for remediation (containment, eradication, and closure).

You can also attach files to a journal entry to document and corroborate your findings.



Journal Entries can also be edited ( ) and deleted ( ) from the main incident page.

Cyber Kill Chain

The cyber kill chain is an industry concept which categorises different phases of cyber intrusions. From the "Kill chain" Wikipedia page:

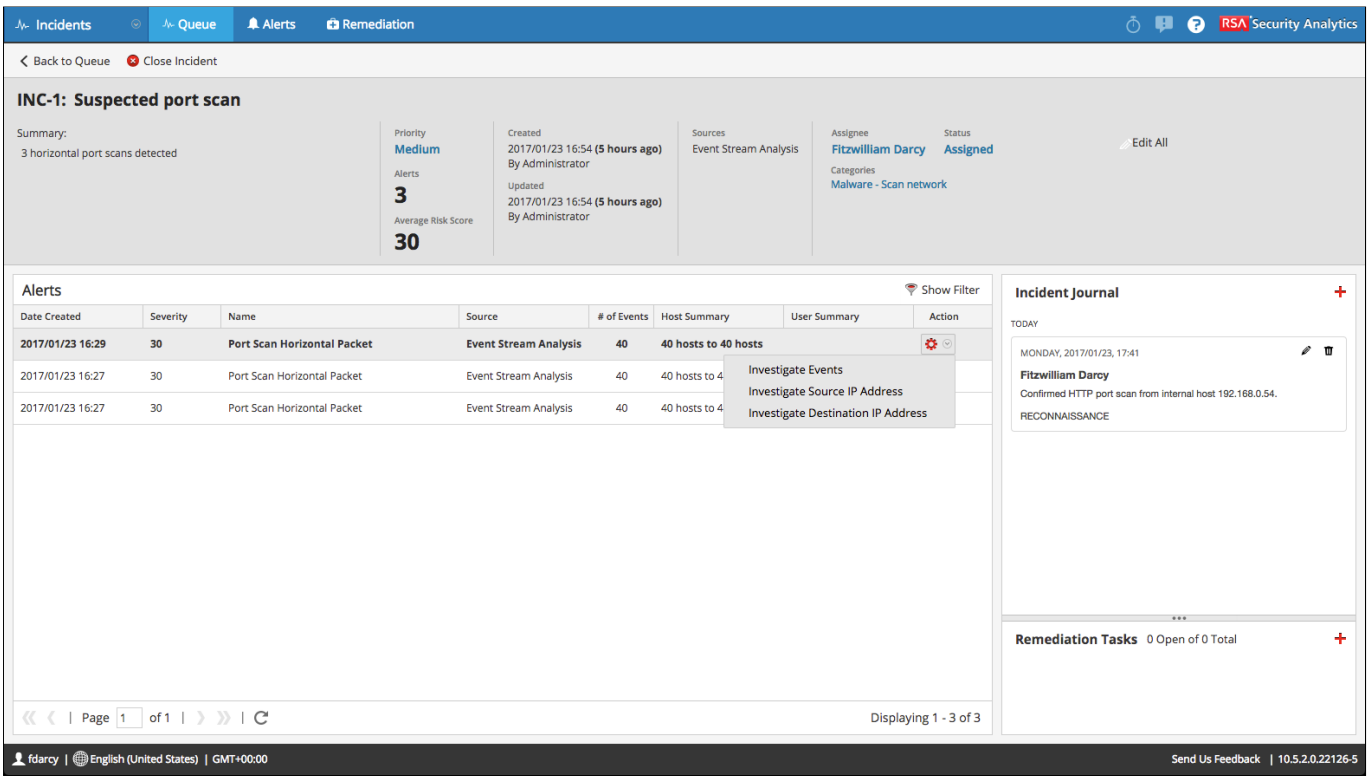
Computer scientists at Lockheed-Martin corporation described a new "intrusion kill chain" framework or model to defend computer networks in 2011.[6] They wrote that attacks may occur in stages and can be disrupted through controls established at each stage. Since then, the "cyber kill chain" has been adopted by data security organizations to define stages of cyber-attacks.[8]

A cyber kill chain reveals the stages of a cyberattack: from early reconnaissance to the goal of data exfiltration.[9] The kill chain can also be used as a management tool to help continuously improve network defense. Threats must progress through several stages in the model, including:

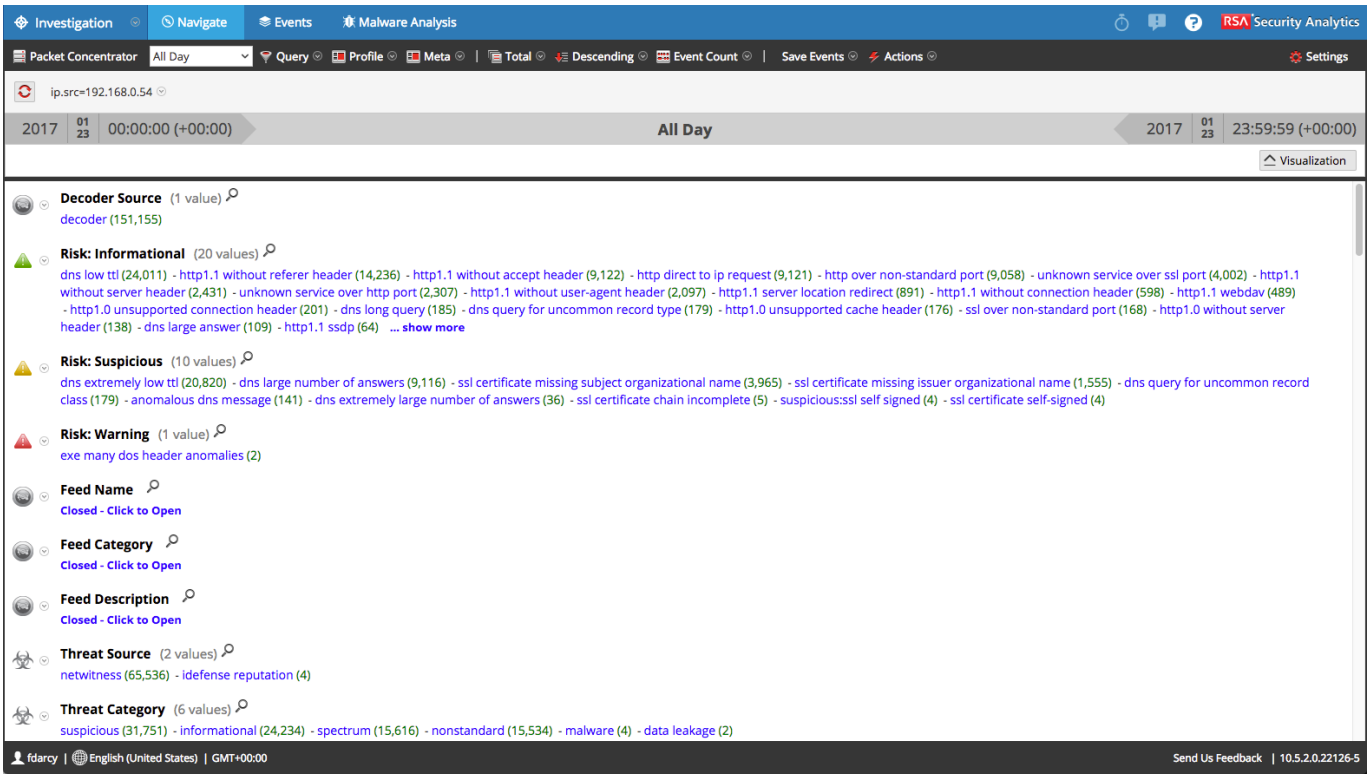
- **Reconnaissance**: Intruder selects target, researches it, and attempts to identify vulnerabilities in the target network.
- **Weaponization**: Intruder creates remote access malware weapon, such as a virus or worm, tailored to one or more vulnerabilities.
- **Delivery**: Intruder transmits weapon to target (e.g., via e-mail attachments, websites or USB drives)
- **Exploitation**: Malware weapon's program code triggers, which takes action on target network to exploit vulnerability.
- **Installation**: Malware weapon installs access point (e.g., "backdoor") usable by intruder.
- **Command and Control**: Malware enables intruder to have "hands on the keyboard" persistent access to target network.
- **Actions on Objective**: Intruder takes action to achieve their goals, such as data exfiltration, data destruction, or encryption for ransom.

Accessing an Incident's Alerts

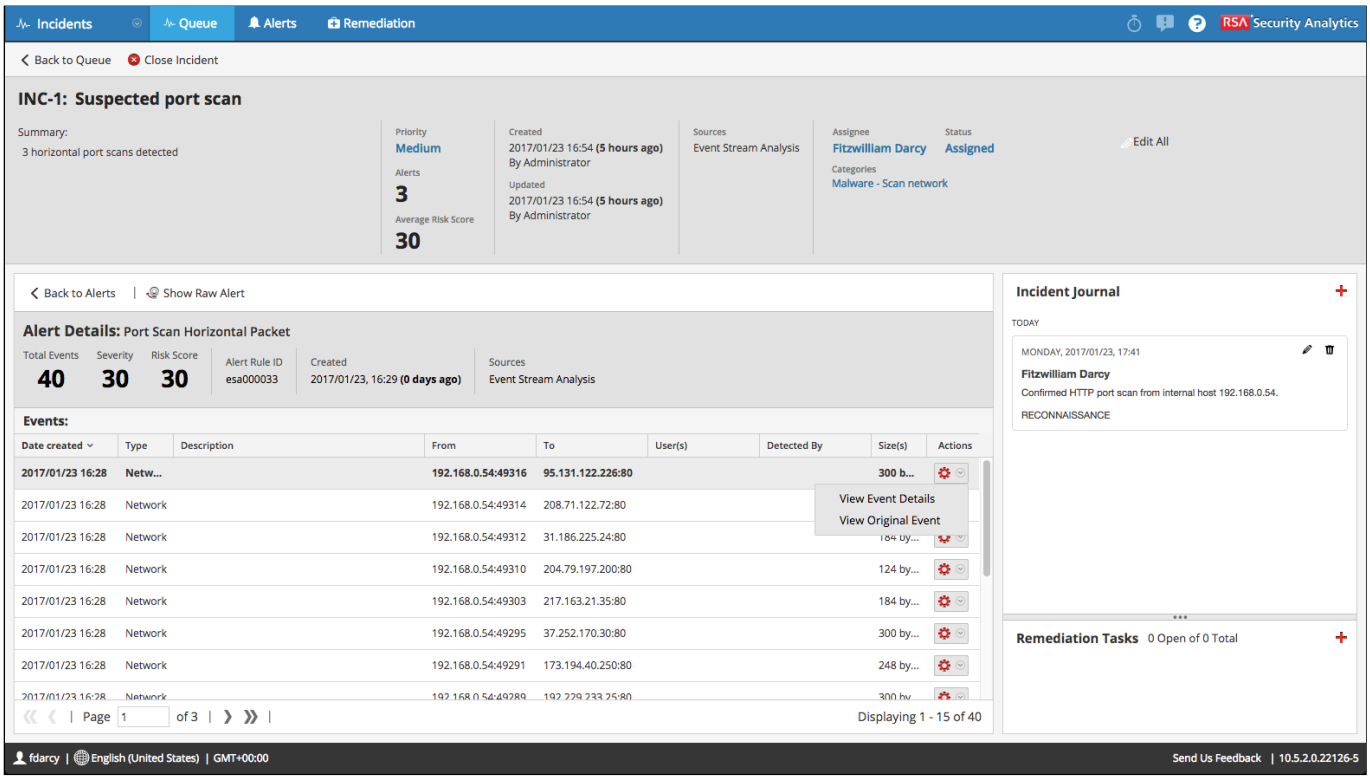
Double-click on an alert inside an incident to pull up its events. You can also use the button to drill into the events in **Investigation**.



In the case of the port scan incident we're investigating, if we investigate the source IP address, we can get a more complete picture of what's going on with this particular host 192.168.0.54.



Once you've double-clicked on an alarm, the lower-left hand pane will then display the selected alert's event data.



Click on Back to Alerts to restore the alarms view for the Incident.

Click on Show Raw Alert to show the events in JSON format, just as in the Alarms view.

Assigning an Incident

To take ownership of an incident from the **Incidents->Queue**, select one or more incident and click Assign To Me. You can also edit the Assignee field directly from within an incident.

Incident Status and Closing an Incident

Valid incident statuses are:

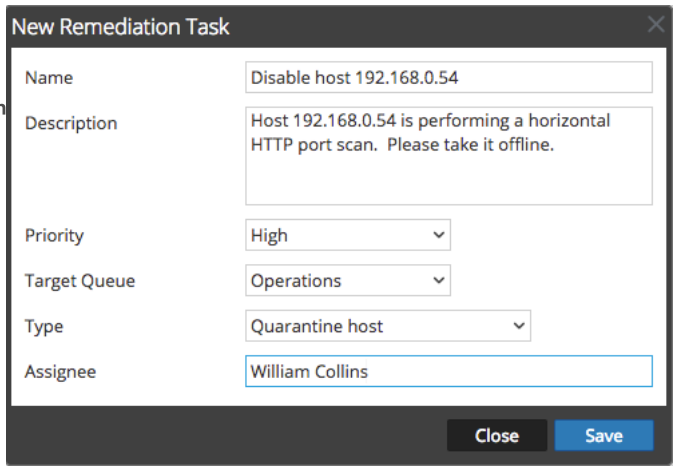
- **New** - Initial state
- **Assigned** - When the case has been assigned to an analyst but not worked
- **In Progress** - The analyst is actively working the case
- **Remediation Requested** - A Remediation Task has been opened
- **Remediation Completed** - The remediation task has been closed

To close an incident, click the Close Incident button. This will simply set the status to **Closed**. To re-open an incident, just change the Status back to something other than Closed.

Remediation Tasks

Opening Remediation Tasks

One other type of 'ticket' within the IM framework is a **Remediation Task**. This type of ticket is a request for someone to do something about a security incident, such as change a firewall rule, re-image a PC, and/or take a PC off of the network. To open a Remediation Task, click the + icon under **Remediation Tasks**, and then fill in the name, description, priority, target queue type, and Assignee. Take note that the assignee of a Remediation Task is different than an Incident assignee. Whereas an Incident assignee is an actual SA user, a Remediation Task assignee can be any string of text.



Valid Types are:

- Quarantine host
- Quarantine Network Device
- Block IP/Port
- Block External Access to DMZ
- Block VPN Access
- Reimage host
- Update Firewall Policy
- Update IDS/IPS Policy
- Update Web Proxy Policy
- Update Access Policy
- Update VPN Policy
- Custom

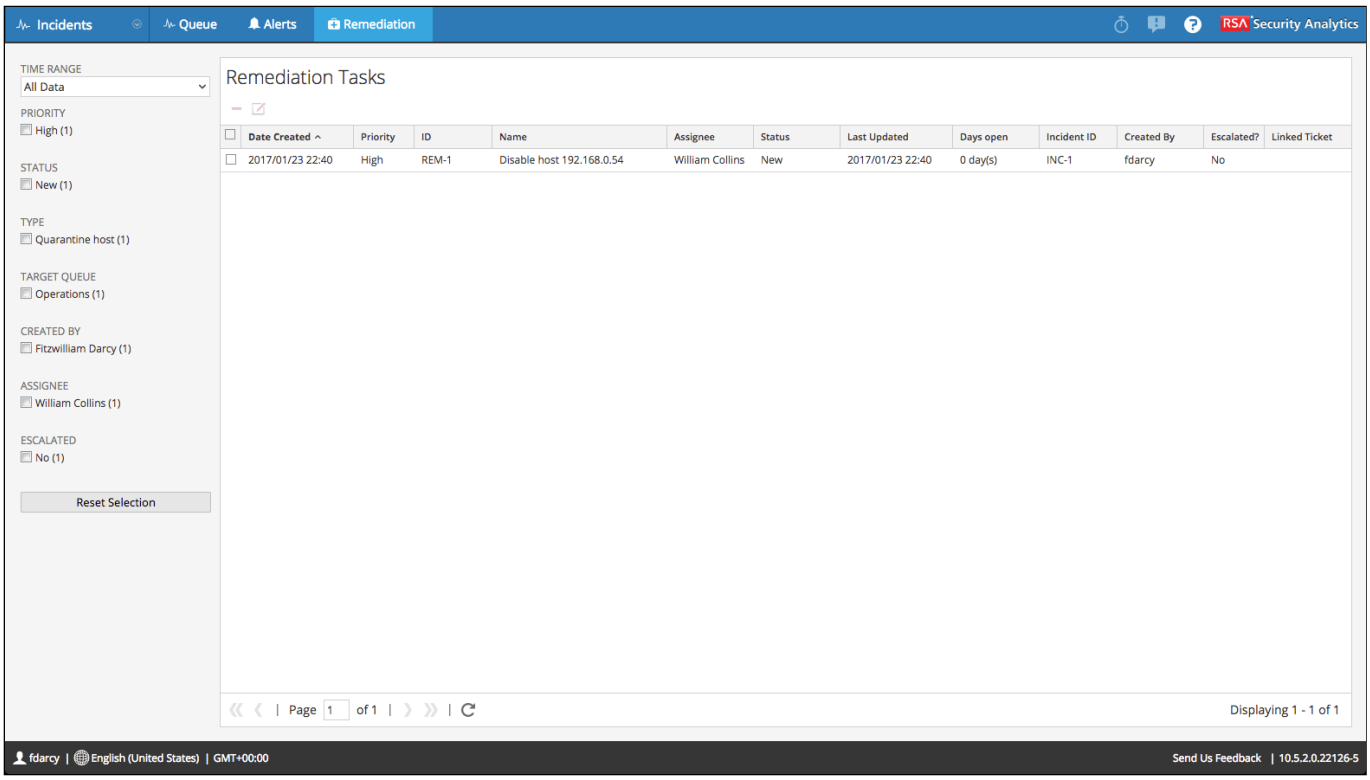
Valid **Target Queues** are:

- Operations - For operations personnel, usually for operations involving a quarantine operation.
- GRC - For Governance, Risk and Compliance Personnel. This often has to do with account access and policy updates.
- Content Improvement - Use this queue if you wish to create a feedback loop to add information to an SA feed or the like. This could be useful for whitelisting or blacklisting a host.

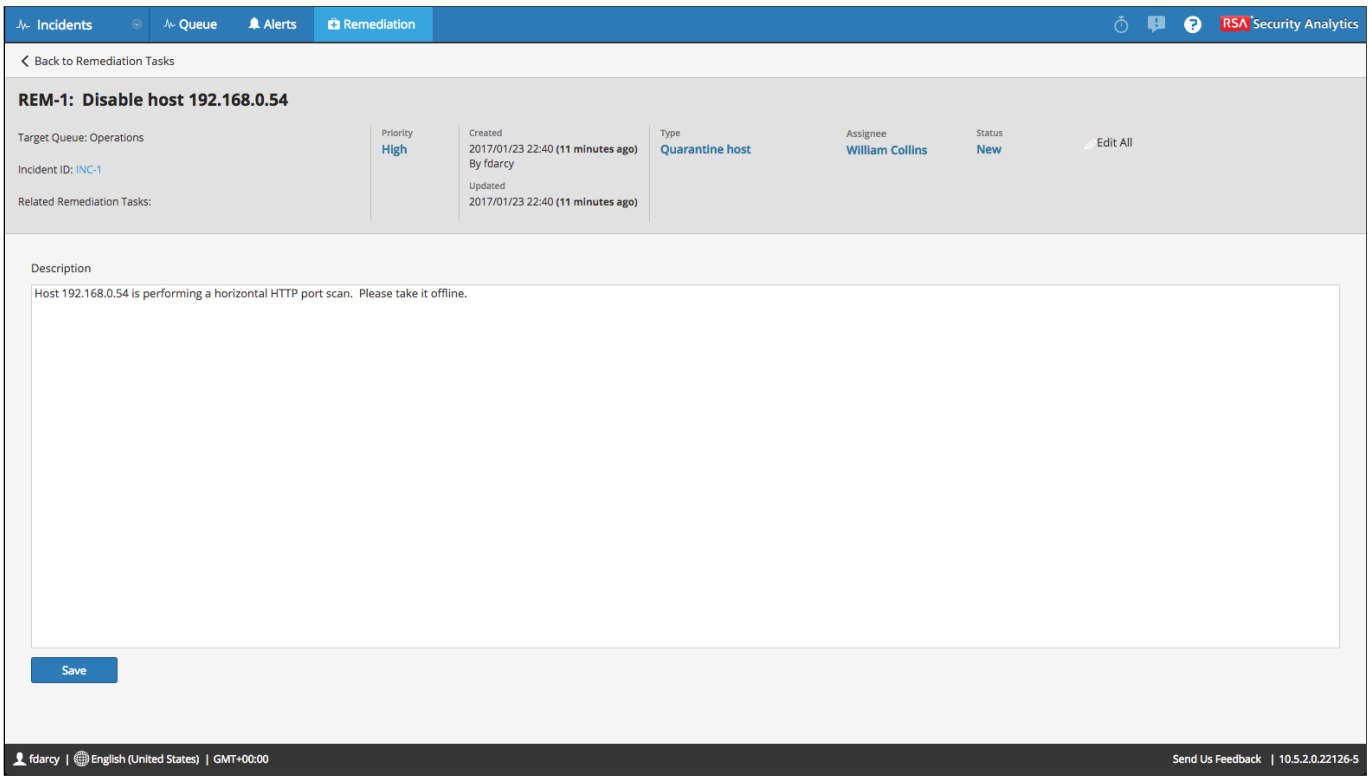
Remediation Tasks are not as well integrated as other parts of SA. They are also poorly documented in the SA documentation.

The Remediation Queue

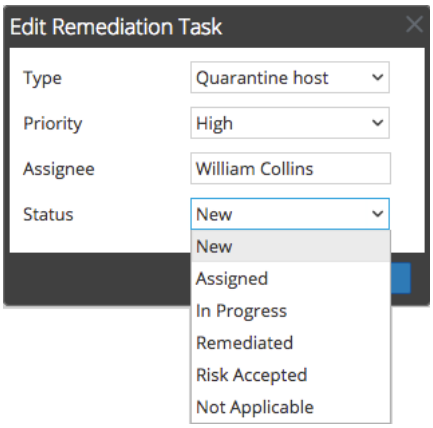
Remediation Tasks can be accessed from **Incidents->Remediation**.



Double-click on one to view the full task...



Or click Edit All to edit Type, Priority, Assignee, or Status.



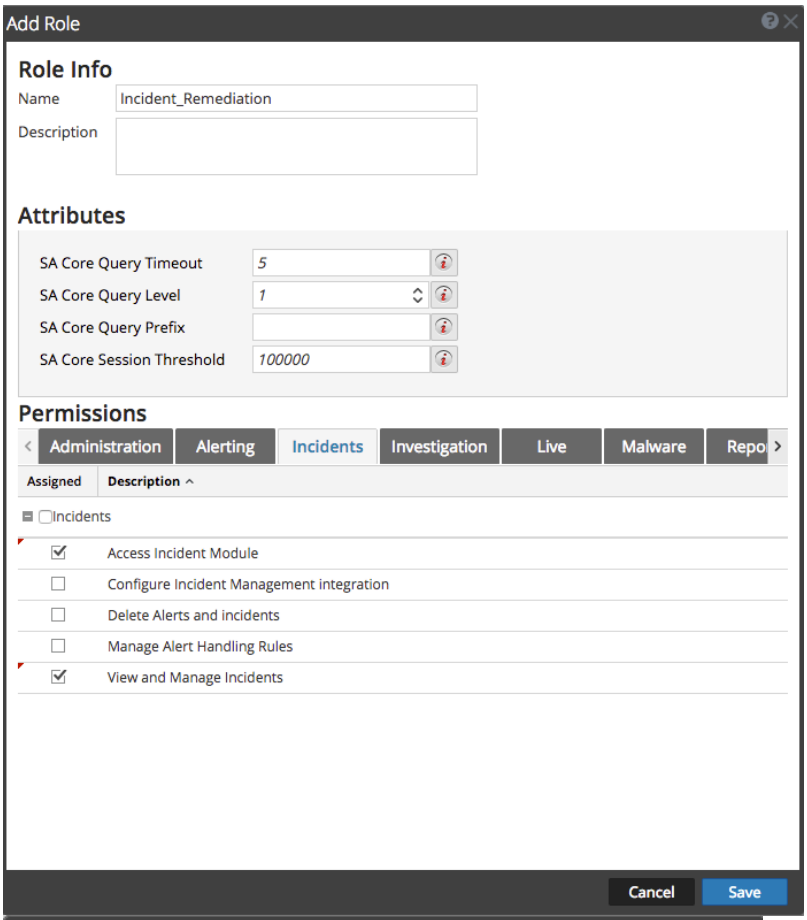
The Status is fairly self-explanatory, except for possibly **Risk Accepted**. This means that a decision has been made to do nothing about it, and the risk is being accepted by the organisation.

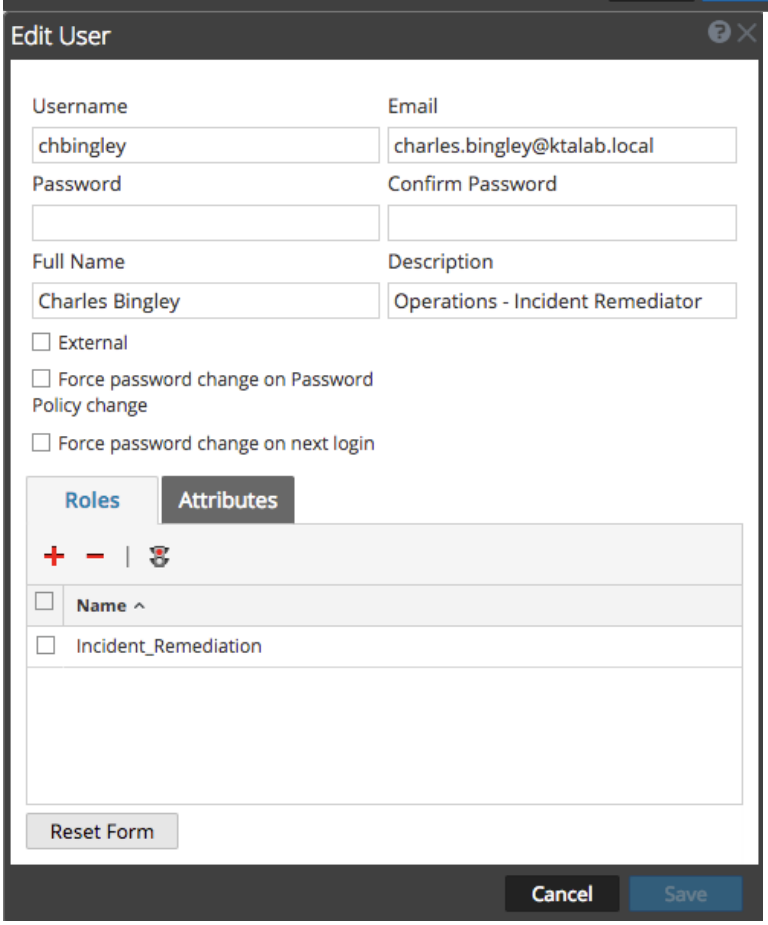
Remediation User Security

Because Remediation Tasks are not bound to any actual SA user, personnel whose role is solely to act on Remediation Tasks should only have enough access to view Remediation Tasks. For this, I suggest creating a **System Role** with the following permissions on the **Incidents** tab:

- Access Incident Module
- View and Manage Incidents

Then create users for your remediation personnel who belong to this Role. In the below screenshot, we call the role **Incident_Remediation**.

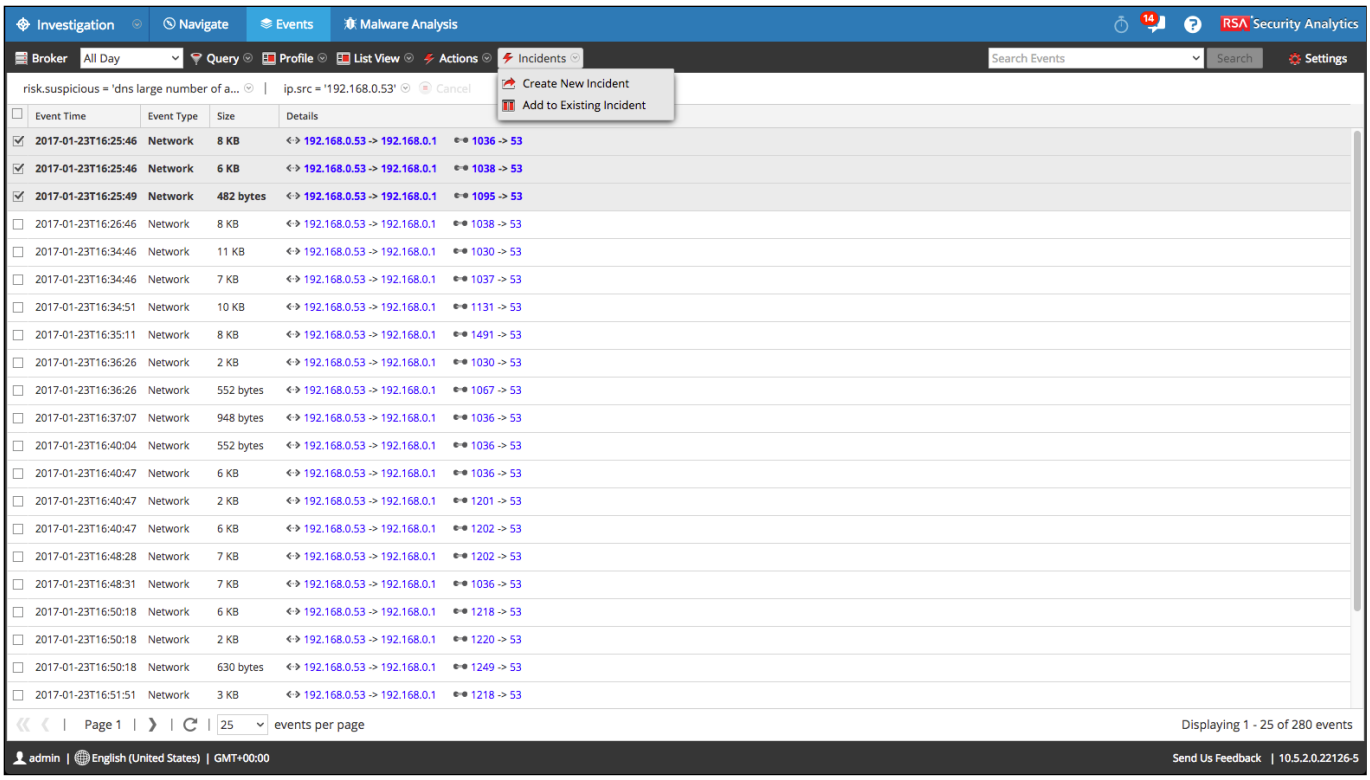


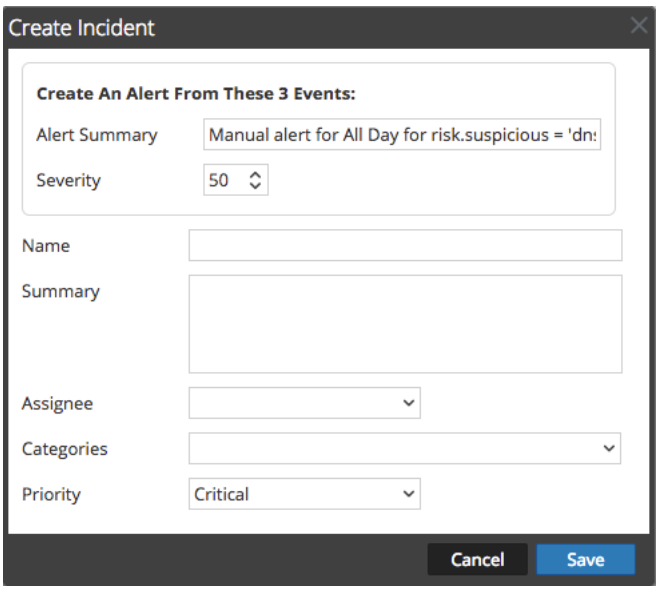


Other Ways of Creating Incidents

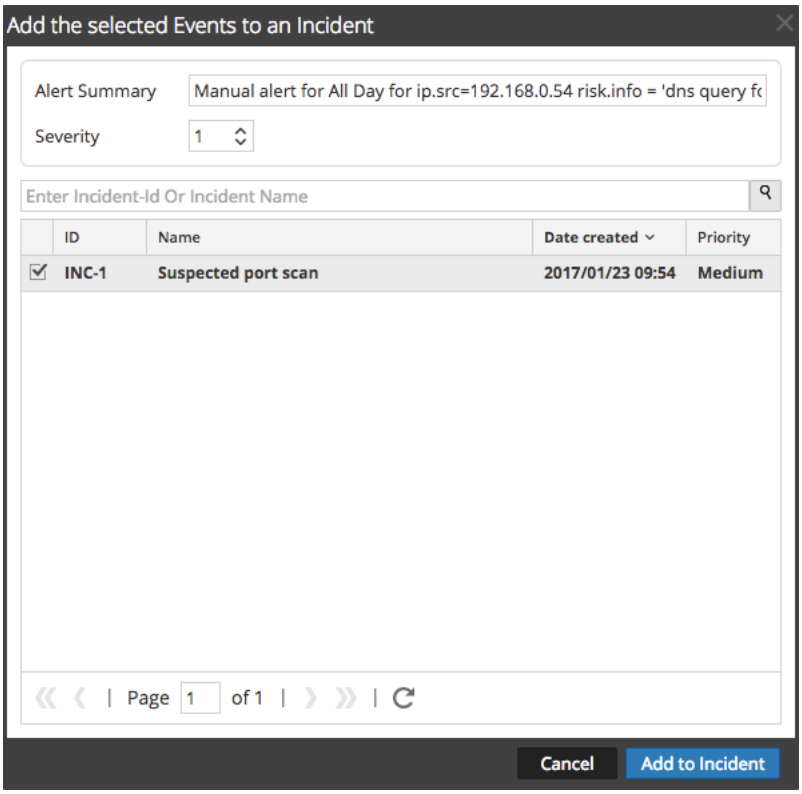
Creating an Incident Manually from Investigation

Another way of creating an Incident is to do so from the **Investigation->Events** view. Select one or more events in the Events view and click the Incidents dropdown, and then either **Create New Incident**, or **Add to Existing Incident**.





Adding to an existing incident...



All selected events will now be continued in a single alert with the incident. The name of the alert will be what you chose in the Alert Summary above, and the alarm severity will also match what you entered. The default is 1 (the lowest) and the maximum is 100. The source column will show as Security Analytics Investigator.

Top Tip: When adding Investigation events to an existing incident, the incident must be assigned to the logged-in user before the incident will appear in the Add window.

Creating Incidents Automatically - Aggregation Rules

Having to scour multitudes of alert data to find threats isn't a very scalable process. One needs a way to create incidents automatically when certain alert criteria are met. Enter **Aggregation Rules**. We get here from the **Incidents->Configure->Aggregation Rules** screen. By double-clicking on one of the predefined rules, you should see this rule:

Incidents | Queue | Alerts | Remediation | Configure | RSA Security Analytics

Aggregation Rules | Notifications | Integration | Retention Scheduler | [Edit Rule] High Risk Alerts...

Enabled

Name* High Risk Alerts: ESA

Description This incident rule captures alerts generated by the RSA ESA platform as having a Risk Score of "High" or "Critical".

Match Conditions: Query Builder / Advanced

All of these | Add Condition | Add Group

Source | is equal to | Event Stream Analysis

Risk Score | is equal or greater than | 50

Action: Group into an Incident / Suppress the Alert

Grouping Options: Group By: Risk Score; Time Window: 1 Hours

Incident Options: Title: ${ruleName} for ${groupByValue1}; Summary; Categories; Assignee

Priority: Use the following to set the priority for incident:

- Average of Risk Score across all of the Alerts
- Highest Risk Score available across all of the Alerts
- Number of Alerts in the time window

Critical 90, High 50, Medium 20, Low 1

Move slider to adjust scale: 1, 25, 50, 75, 100

Notifications: Notify These Users When Incidents Are Created By This Rule:

Enter comma separated list of email addresses

Save | Close

admin | English (United States) | GMT-07:00 | Send Us Feedback | 10.5.2.1.18803-5

Taking a look at the Aggregation Rule above, we can see that there are three main parts to the rule, namely:

- **The conditions to satisfy in order to trigger - these mainly act upon alert data. We can use things like the:**

Match Conditions: Query Builder / Advanced

All of these | Add Condition | Add Group

Source | is equal to | Event Stream Analysis

Risk Score | is equal or greater than | 50

None of these | Add Condition | Remove Group

Alert Name | is equal to | High Risk PDF

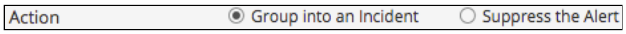
Group added as an example

- Alert Rule ID - The UUID of the alert. The is not typically used in favour of the Alert Name
- Alert Name

- Alert Source - The type of service which generated the alert - 'Event Stream Analysis', 'Security Analytics Investigator', etc
- Alert Type - log, packets ('network session'), etc
- Alert Severity
- Alert Risk Score
- Date Created
- Source IP
- Source Country
- Destination IP
- Destination Port
- Destination Country
- Source Username
- Detector IP Address
- Filename
- File MD5 Hash

Conditions can also be grouped together, so that all conditions, any conditions, or no conditions must be met. Multiple groupings can be in a rule.

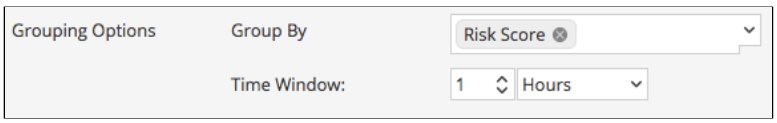
- **The action to be taken** - to either create an incident, or it can also be used suppress matching alerts.



- The grouping options

How Alerts Get Allocated to Incidents

We choose how alerts are allocated to individual events using **Grouping Options**. All alerts with identical grouping values, within the time window will automatically be grouped to the newly-created incident.

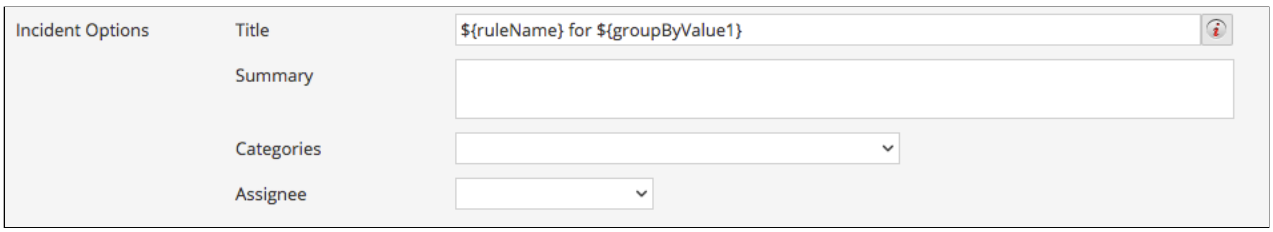


This means that for the rule in the above screenshot, all ESA alerts with a risk score greater than fifty that occur within a one hour time window will be grouped into incidents. And because we're grouping by **Risk Score**, all alerts with a risk score of 50 will be one incident, all those with a value of 70 into another incident, a value of 71 into yet another incident.

You can select multiple group by fields, where all selected group by fields must match in order to be grouped together in the same incident.

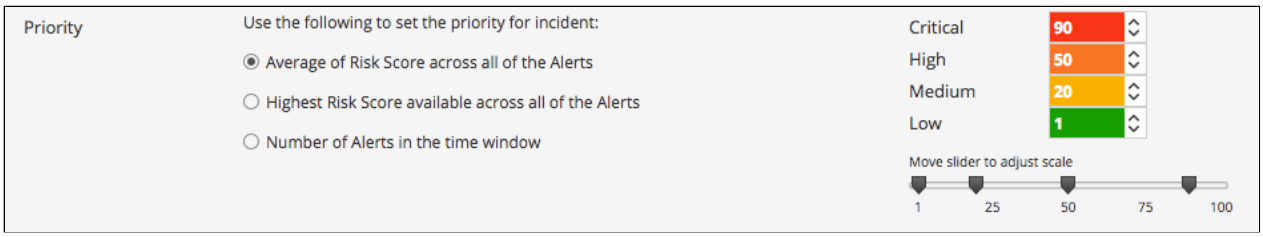
Rule Options

Everything after this is essentially details:



- We set the incident title using a simple template, where **${ruleName}** is the name of the current aggregation rule. **${groupByValue1}** is the first group by key chosen, **${groupByValue2}** the second, etc.
- The summary is text you can optionally pre-fill into the incident
- One or more categories can be automatically assigned (Malware: C2, Misuse: Embezzlement, etc)
- The incident can automatically be assigned to an analyst

Setting the Incident's Priority



Incidents require a priority (low, medium, high, or critical), so we set it in this section. This can be can be determined by:

- The average of the risk scores of all alerts in the incident

- The highest of all risk scores in all of the alerts in the incident
- Or the raw number of alerts in the time window.

Once we know the value of this operation, we know whether it's low, medium, high, or critical from the coloured values to the right. Simply change the numbers or drag the sliders to adjust. In this case, if the average risk score across all alerts in the incident is 70, then the incident's priority would be 70.

Notifying When Incidents are Created by an Aggregation Rule

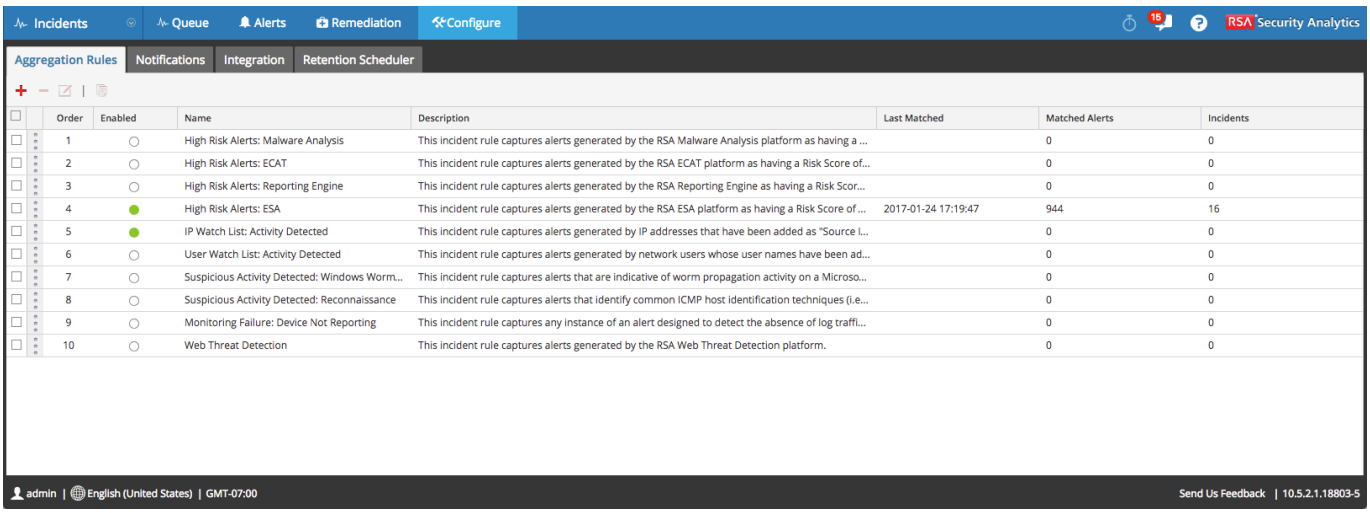
This is self-explanatory.

Lastly

Remember tick the **Enabled** box for the rule and click **Save.**

Really, One Last Thing

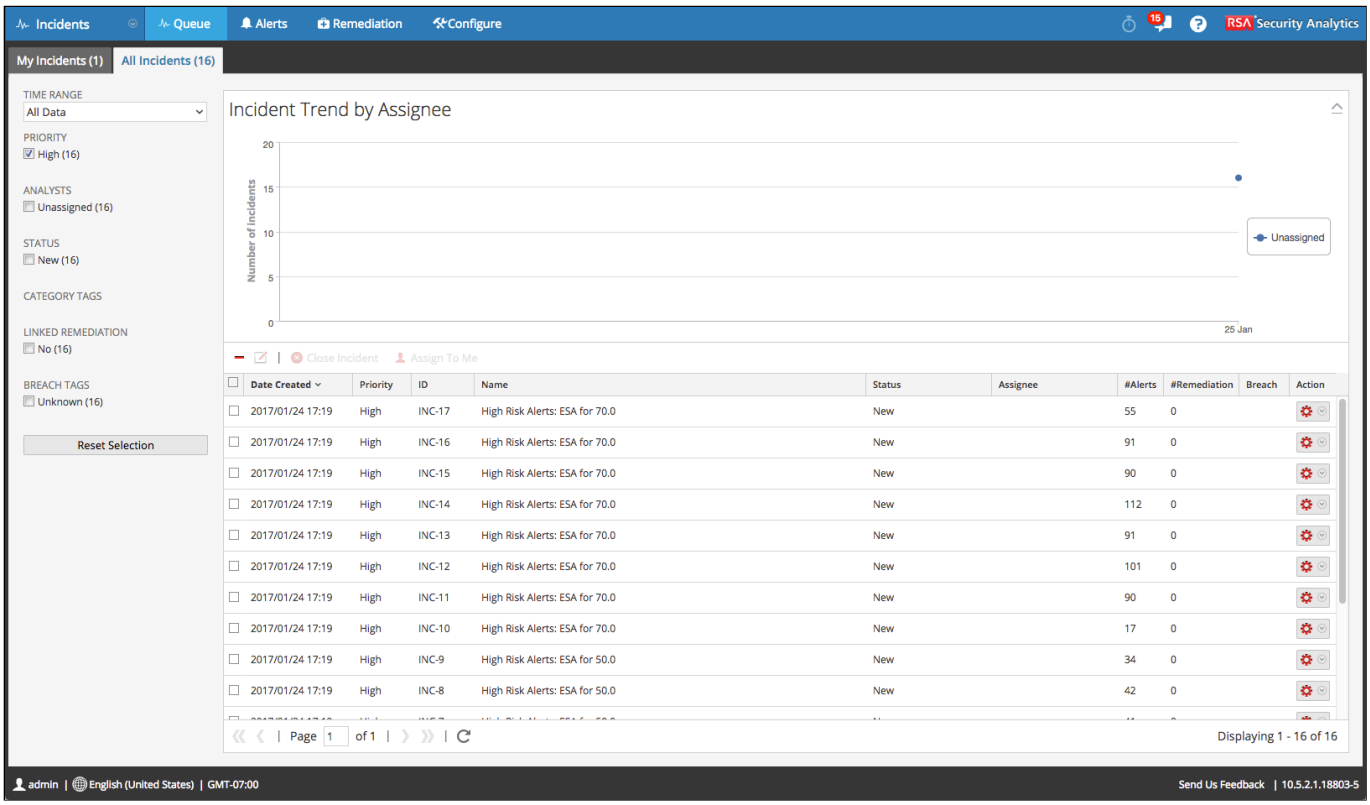
The order of your aggregation rules matters. When an alert matches two rules, the higher rule will take precedence. It's like a first-match firewall policy. Drag a rule up or down to change its ordering.



Order	Enabled	Name	Description	Last Matched	Matched Alerts	Incidents
1	○	High Risk Alerts: Malware Analysis	This incident rule captures alerts generated by the RSA Malware Analysis platform as having a ...		0	0
2	○	High Risk Alerts: ECAT	This incident rule captures alerts generated by the RSA ECAT platform as having a Risk Score of...		0	0
3	○	High Risk Alerts: Reporting Engine	This incident rule captures alerts generated by the RSA Reporting Engine as having a Risk Scor...		0	0
4	●	High Risk Alerts: ESA	This incident rule captures alerts generated by the RSA ESA platform as having a Risk Score of ...	2017-01-24 17:19:47	944	16
5	●	IP Watch List: Activity Detected	This incident rule captures alerts generated by IP addresses that have been added as "Source I...		0	0
6	○	User Watch List: Activity Detected	This incident rule captures alerts generated by network users whose user names have been ad...		0	0
7	○	Suspicious Activity Detected: Windows Worm...	This incident rule captures alerts that are indicative of worm propagation activity on a Microso...		0	0
8	○	Suspicious Activity Detected: Reconnaissance	This incident rule captures alerts that identify common ICMP host identification techniques (i.e...		0	0
9	○	Monitoring Failure: Device Not Reporting	This incident rule captures any instance of an alert designed to detect the absence of log traffi...		0	0
10	○	Web Threat Detection	This incident rule captures alerts generated by the RSA Web Threat Detection platform.		0	0

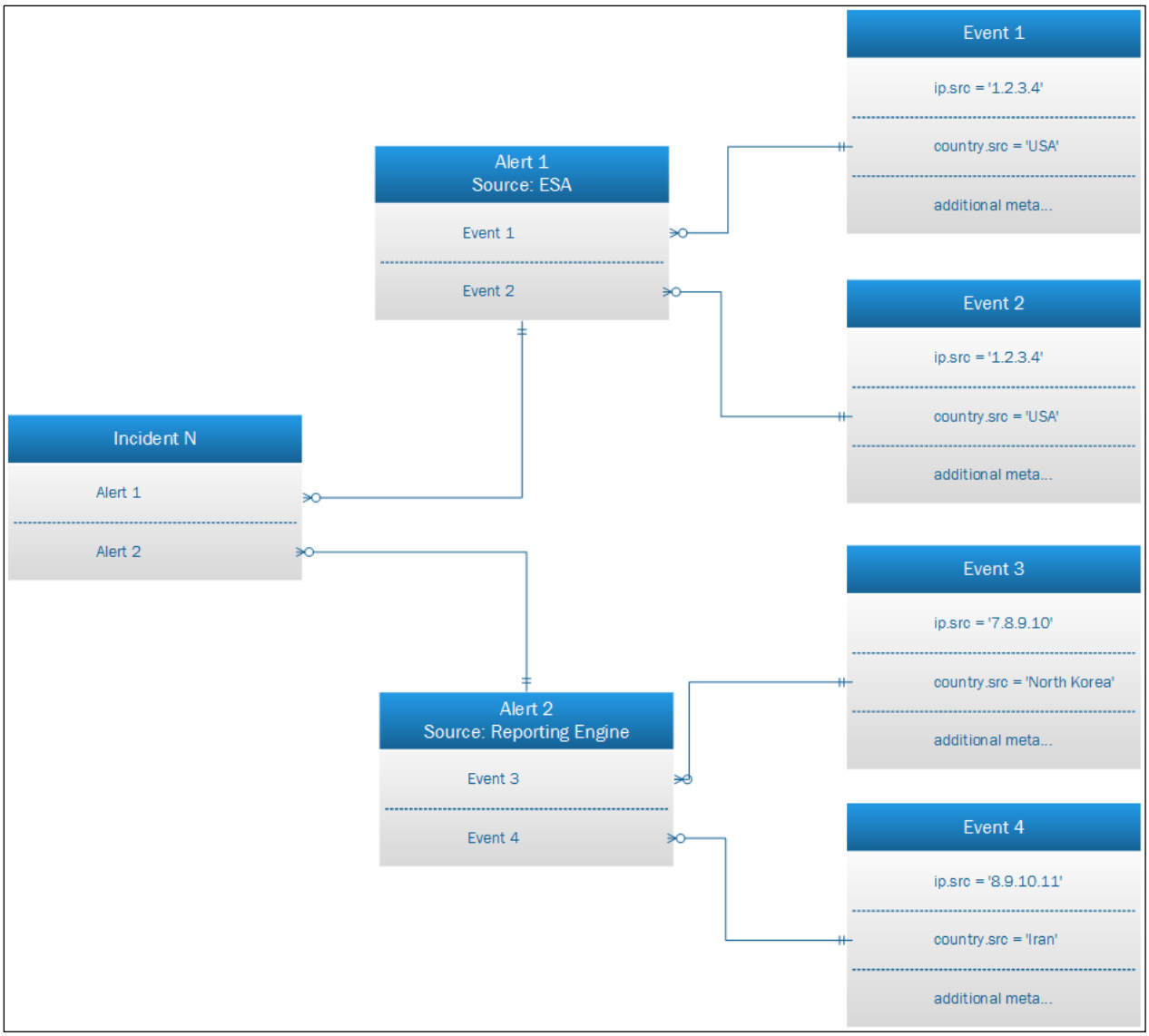
The Proof is in the Pudding

By turning on the above rule, we should see incidents get created almost immediately. That's right, aggregation rules are retroactive as well. They don't work only on fresh data.



How Does it All Fit Together?

- Incidents are comprised of alerts. Incidents can be created manually, or by Aggregation Rules.
- Alerts are comprised of events. They are generated by either ESA, the Reporting Engine, or Malware Analysis.
- Events are contained within alerts, and are in fact sessions generated by Decoders and Log Decoders.
- Remediation Actions are generated manually by an Analyst from an Incident.



Getting Notified of Incident Activity

IM can notify SOC Managers and the Incident Assignee via email when things happen in IM. This is done in **Incidents->Configure->Notifications**. The user experience here is a bit different than for other parts of the product. To do this, you must:

1. Choose an email server which has already been defined in **Administration->Settings->Global Notifications**.

2. Define the email addresses of your SOC Managers in the SOC Managers field

3. Choose when you want your Assignee and/or SOC Manager to be notified. When:

- An incident is created
- An incident record is updated
  - when the status changes
  - when the assignee is changed
  - when the priority is changed
  - when the category is changed
- Remediation task is created
- Remediation task is updated
- Remediation task is closed

Incidents | Queue | Alerts | Remediation | Configure | RSA Security Analytics

Aggregation Rules | Notifications | Integration | Retention Scheduler

Notification Settings

Email Server: GSuite

Email Server Settings

SOC Managers: socmanager@kensingtontechlimited.co.uk,fred@kensingtontechlimited.co.uk

Workflow	Incident Assignee?	SOC Managers?	Additional Addresses	Template
Incident is created	☑	☐		Edit
Incident record is updated	☑	☐		Edit
Incident status is changed	☐	☐		
Incident assignee is changed	☑	☑		
Incident priority is changed	☐	☑		
Incident category is changed	☐	☐		
Remediation task is created	☐	☑		Edit
Remediation task is updated	☐	☐		Edit
Remediation task is closed	☐	☑		Edit

Apply

admin | English (United States) | GMT-07:00 Send Us Feedback | 10.5.2.1.18803-5

Top Tip: You can also designate additional email addresses to send a notification to in the **Additional Addresses** column.

HTML Templates

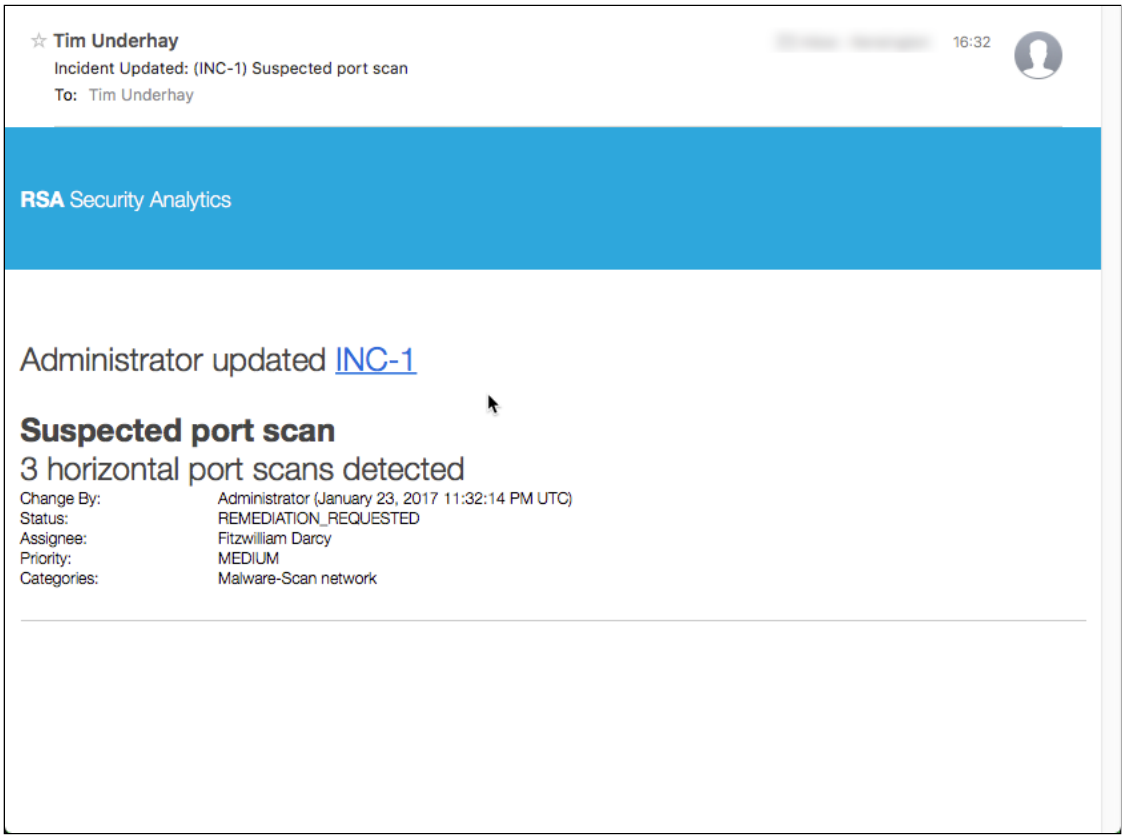
It is also possible to edit the HTML template used to send an email. Click the Edit button next to the workflow type in question.

Template forIncident is created

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <style type="text/css">
    <!--
    body {
      background-color: #fff;
      font-family: HelveticaNeue-Light, "Helvetica Neue Light", "Helvetia", sans-serif;
      margin: 0
    }

    h1 {
      font-family: HelveticaNeue-Light, "Helvetica Neue Light", "Helvetia", sans-serif;
      font-weight: 300;
      font-size: 40px;
      line-height: 40px;
      margin: 0 0
    }

    hr {
      background: #ccc;
      border: 0;
      height: 1px;
```

Save

Here's what our email template for updates looks like:

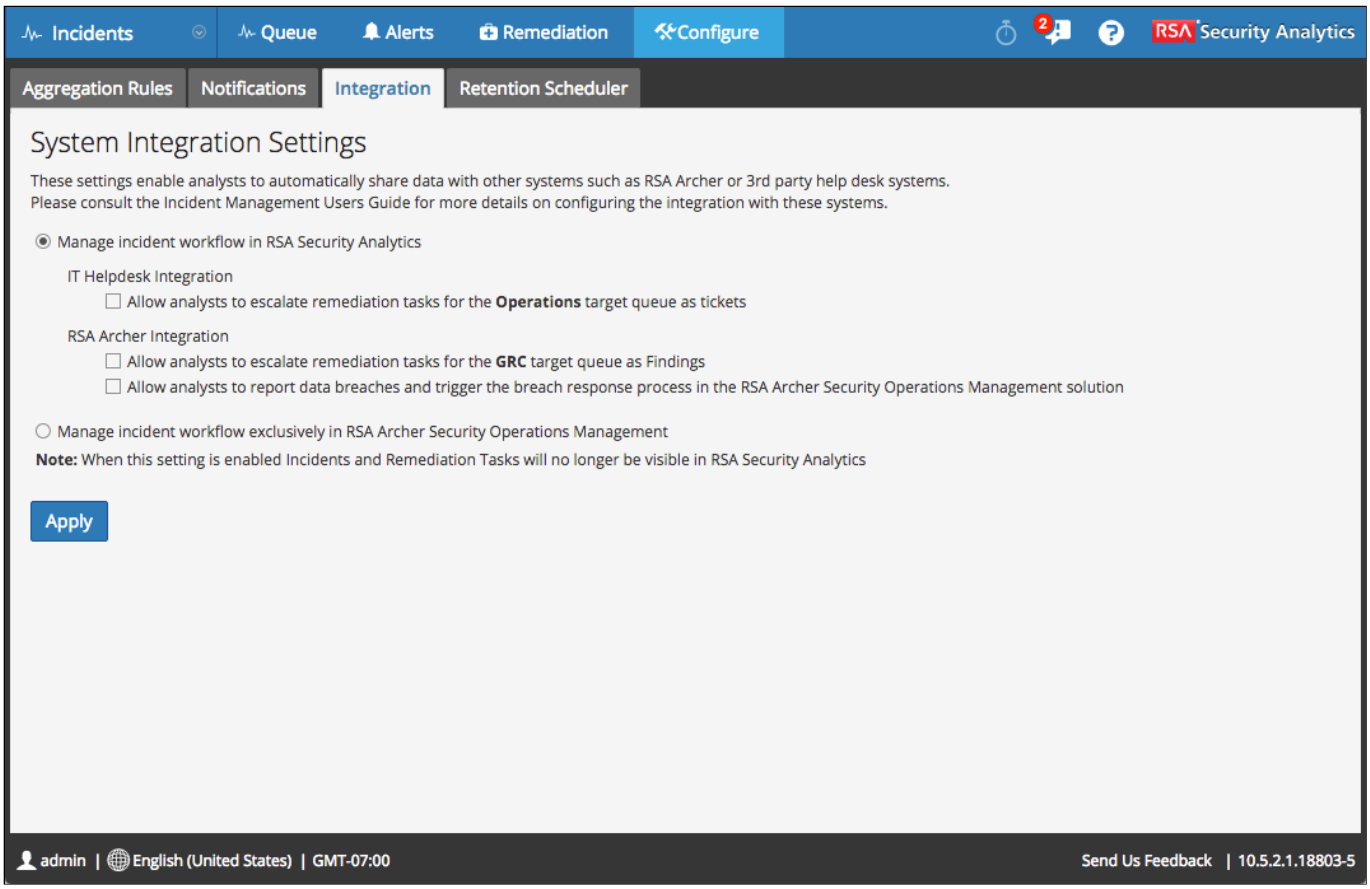


Integrating IM With Other Products

IM can also be used to:

- Push/escalate remediation tasks to a third-party help desk ticketing system, so they can be tracked to closure.
- Push/escalate remediation tasks to RSA Archer GRC.
- Report data breaches and cause the breach response process to be initiated in RSA Archer SECOPS (Security Operations Management).
- Act as a conduit to push all alerts to RSA Archer SECOPS, where SECOPS will be used exclusively for all incident and remediation management. Using this option will disable the incident and remediation capabilities of IM, and it is mutually exclusive of the above three options.

These can all be switched on in **Incidents->Configure->Integration**.

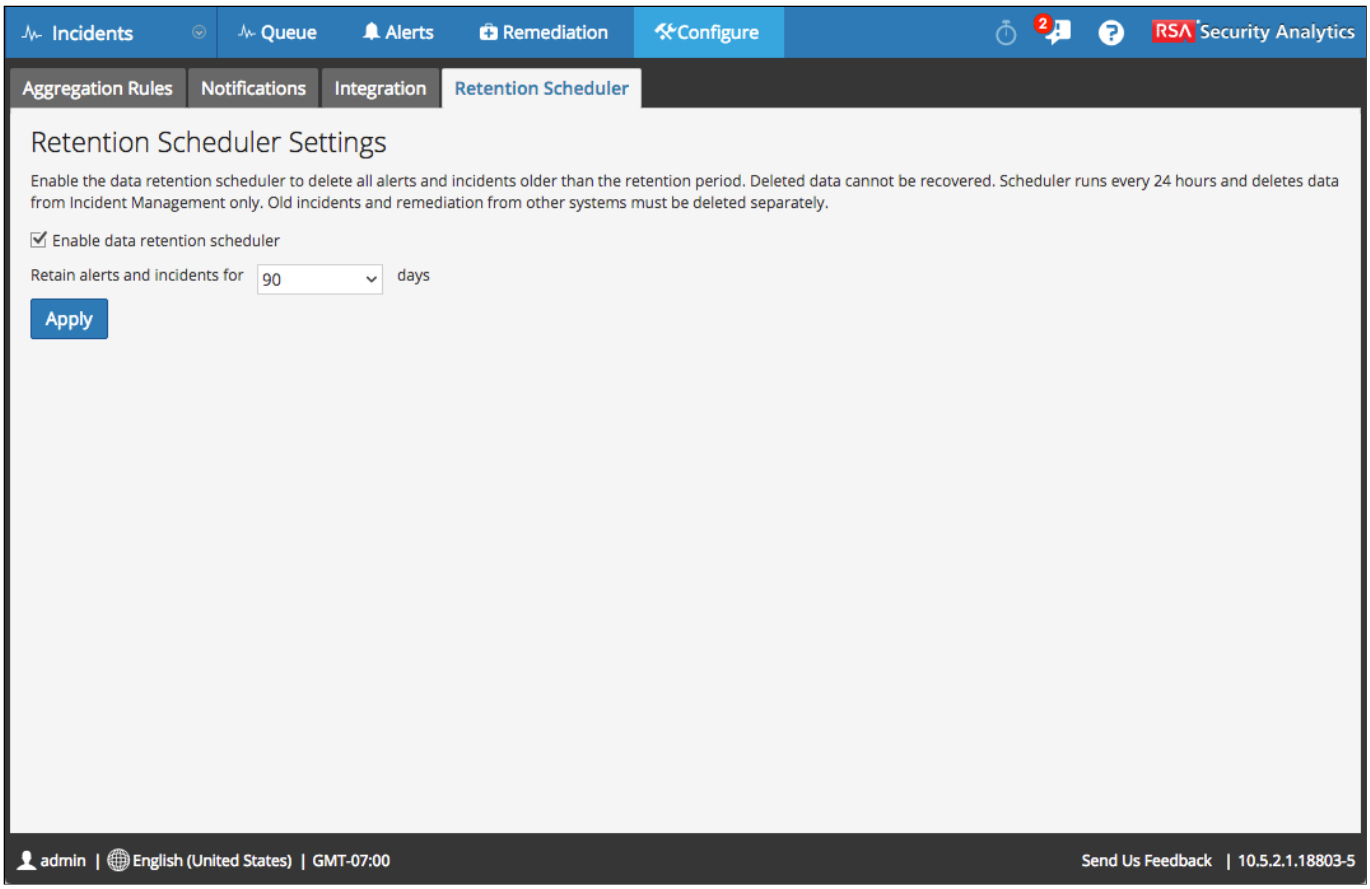


Purging Old Alerts and Incidents - The Retention Scheduler

The more data that's in the IM database, the slower IM's responses will be. The retention scheduler, available in **Incidents->Configure->Retention Scheduler**, allows all alerts and incidents older than N days to be purged from the database. The retention scheduler is disabled by default.

Warning: Before enabling the retention scheduler, see that your company's policies do not require the storage of incident and alert data for longer than the number of days you set here.

Top Tip: If you have a very noisy or misconfigured ESA rule which generates too many alerts, it can obliterate the performance of IM by causing the IM database to fill up. In this situation, it could take IM many minutes to display alert data, thus destroying your workflow. The easiest way to remedy this is to disable the offending ESA rule, and enable the retention scheduler to purge all the old data. This will also purge data you want, though.


Incidents
Queue
Alerts
Remediation
Configure
RSA Security Analytics
Aggregation Rules
Notifications
Integration
Retention Scheduler
Retention Scheduler Settings
Enable the data retention scheduler to delete all alerts and incidents older than the retention period. Deleted data cannot be recovered. Scheduler runs every 24 hours and deletes data from Incident Management only. Old incidents and remediation from other systems must be deleted separately.
Enable data retention scheduler
Retain alerts and incidents for 90 days
Apply
admin | English (United States) | GMT-07:00
Send Us Feedback | 10.5.2.1.18803-5

26. Reporting

The Reporting Engine, or RE, is SA's general source for all things reporting. It is a java-based service which runs on the SA host. It provides functionality of these kinds:

- **Reporting** - reports are based on SQL-like queries on SA data sets. The can be presented in HTML, PDF, Text, and CSV formats
- **Charting** - SA is capable of charting data as well.
- **Alerting** - Alerts are simple queries run against an SA data source. Alerts are run every minute against a data source, on the previous minute of data, so they are lightweight.

Reporting Use Cases

The Reporting Engine supports two basic use cases: **compliance** and **analytical**.

Setting Up the Reporting Engine

Before you can start using the RE, you must take a few steps to configure it:

1. Configure Data Sources
2. Activate alert forwarding to Incident Management
3. Set a Hostname in the RE Configuration
4. Configure RE notification settings (SMTP, SNMP, Syslog, SFTP, URL, and NetworkShare)
5. Upload any custom logos

RE Data Sources

The Reporting Engine is capable of running reports against three types of data sets:

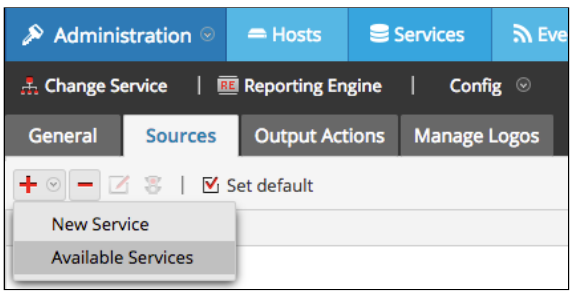
- **NWDB** - NetWitness database. This is any data stored by an SA core service
- **IPDB** - This is RSA's
- **RSA Warehouse** - The RE is capable of running HIVE queries against a Hadoop cluster called the RSA Warehouse.

In our lesson, we shall only cover NWDB, as IPDB and Warehouse aren't part of the SA training course.

Also, know that the RE presently neither supports Health & Wellness data nor Event Source Monitoring data. Unfortunately reporting is not available for these data types as of yet.

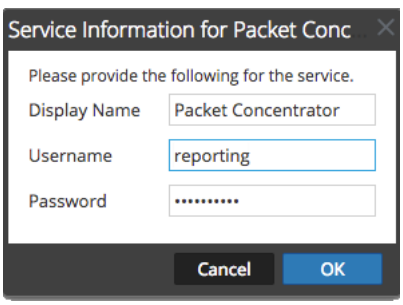
Configuring Data Sources

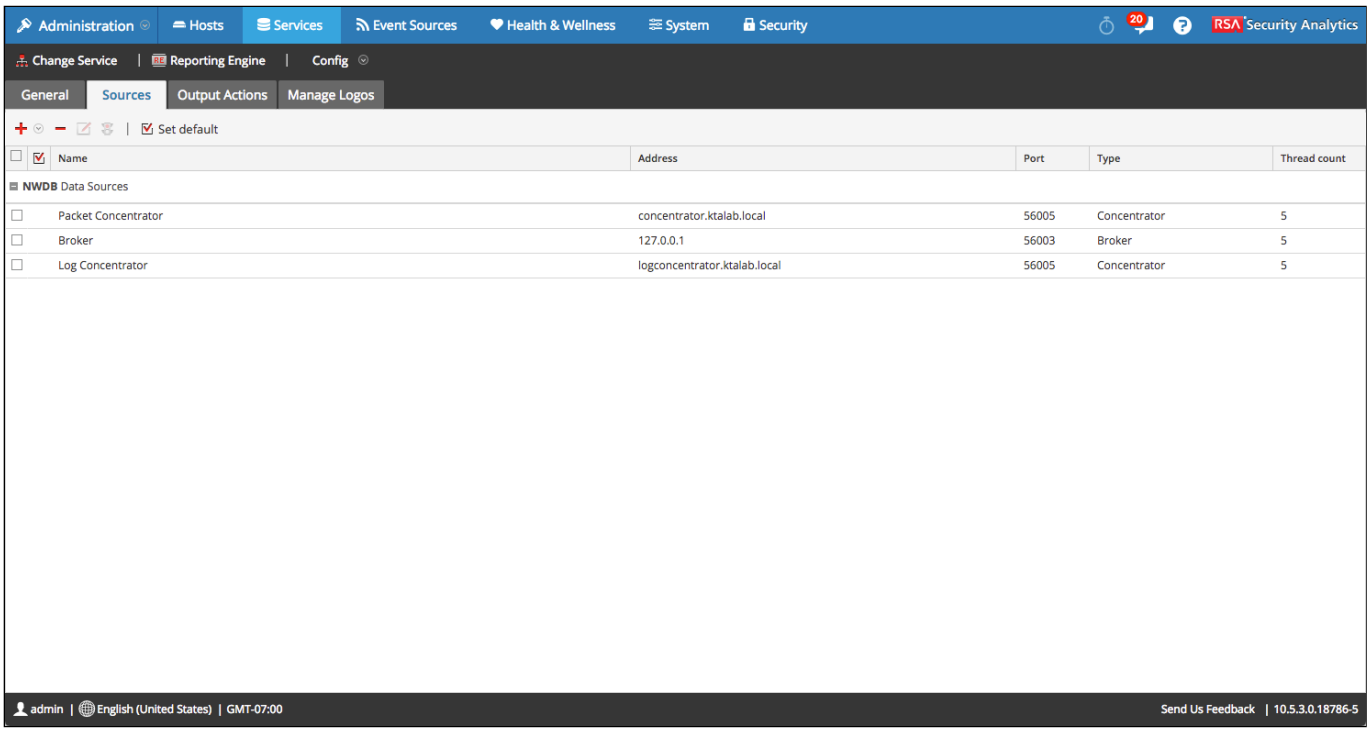
Before one can begin querying the SA stack, data sources must be configured in the RE. Navigate to **Administration->Services->Reporting Engine->Config**, and select the **Sources** tab. Click the + dropdown and choose **Available Services**.



Choose the services you want to add as a data source to the Reporting Engine. You should probably choose your Broker and Concentrators. You must add the services one at a time.

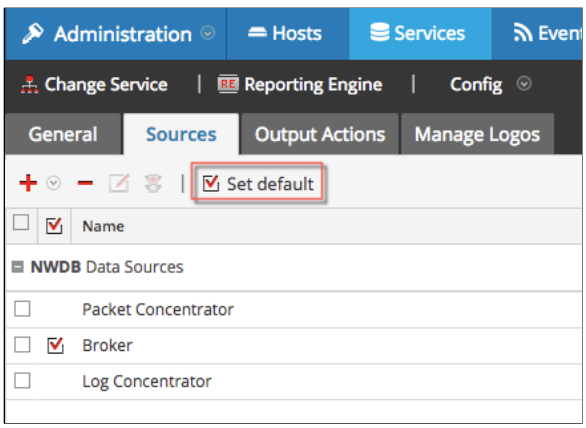
Top Tip: We strongly recommend creating a dedicated service account for Reporting Engine connectivity to your SA Core services.





Set the Default Data Source

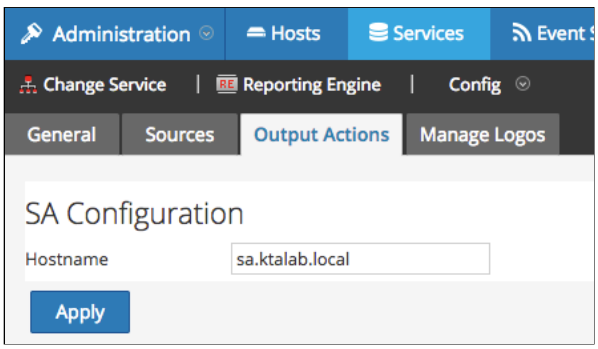
You should also set one of your data sources to be the default. We recommend it be your top-level Broker, so that it has visibility across your entire organisation.



Setting a Hostname in the RE Configuration

You'll want to set the hostname of your SA host in your RE configuration. The reason for this is so that you can click on Investigation links from RE-generated reports, alerts and charts, and have them open in your browser. To this end, make sure it's a hostname that can be resolved from your SA client workstations. To do this:

1. Navigate to the **Output Actions** tab of the RE configuration.
2. Type the hostname in the **hostname** field.
3. Click **Apply**.



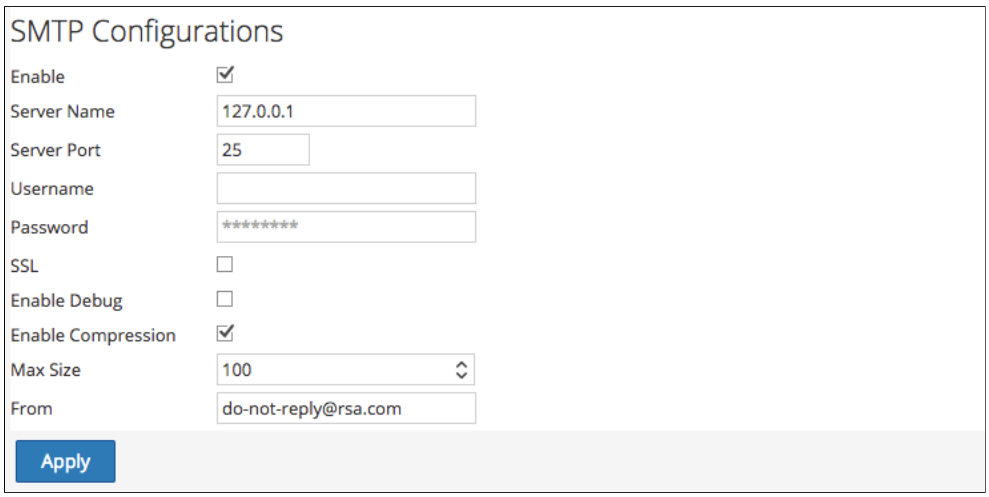
Configuring Reporting Engine Notification Settings

The Reporting Engine unfortunately doesn't use SA's Global Notifications framework, but rather has its own one-off configuration experience. These are accessed in **Administration->Services->Reporting Engine->Config**, on the **Output Actions** tab.

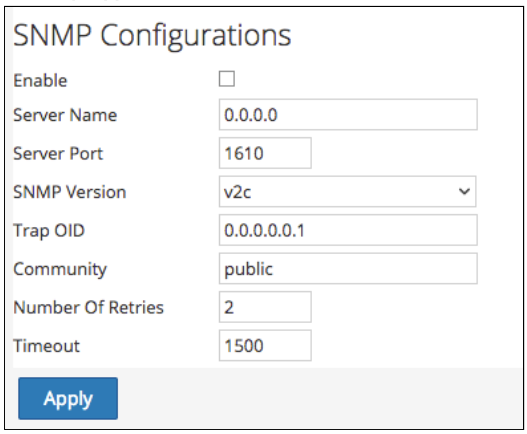
Some settings on this page are used to notify someone that an execution has completed, and some are file destinations where completed reports can be saved to.

Notification Destinations - SA sends a notification when an RE execution has completed. These should all be self-explanatory:

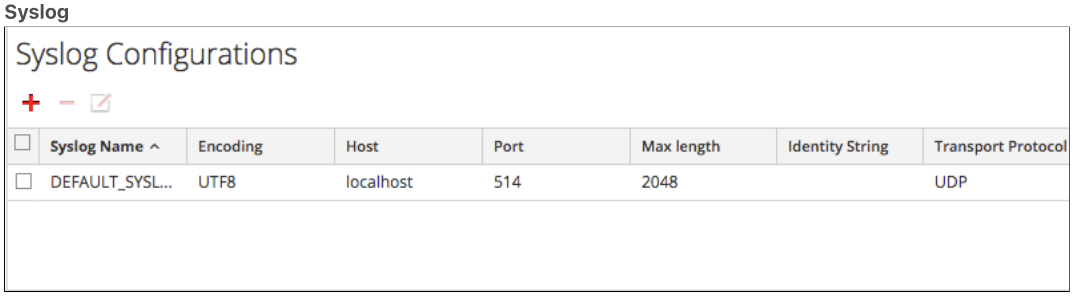
- **SMTP**



- **SNMP (trap)**

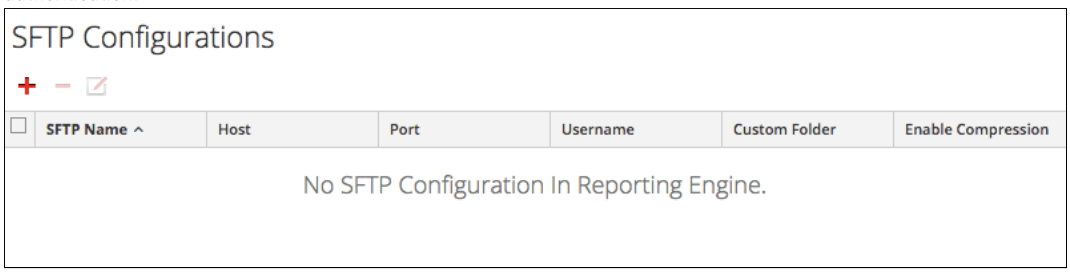


- **Syslog**

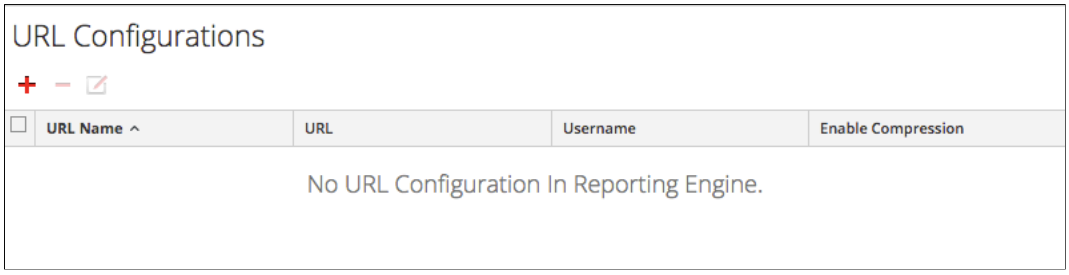


File Destinations - SA writes the output file to these destinations:

- **SFTP** - The RE can write the output file to an SFTP share. Note that SSH keys do not appear to be supported here, so you must use password authentication.



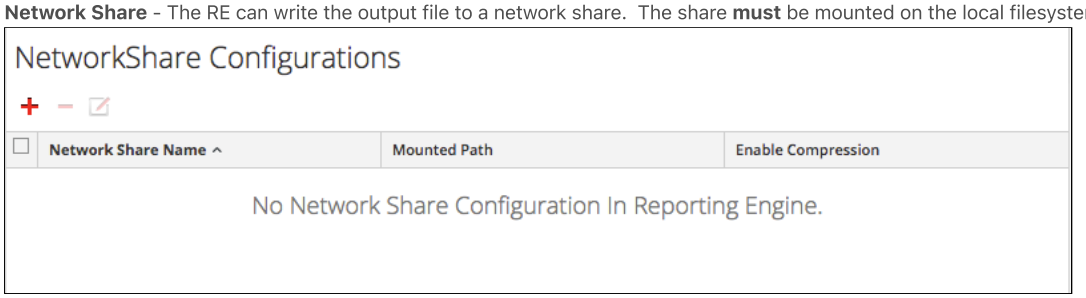
- **URL** - The RE can write the output file to an HTTP URL.



HTTP parameters will be also be sent:

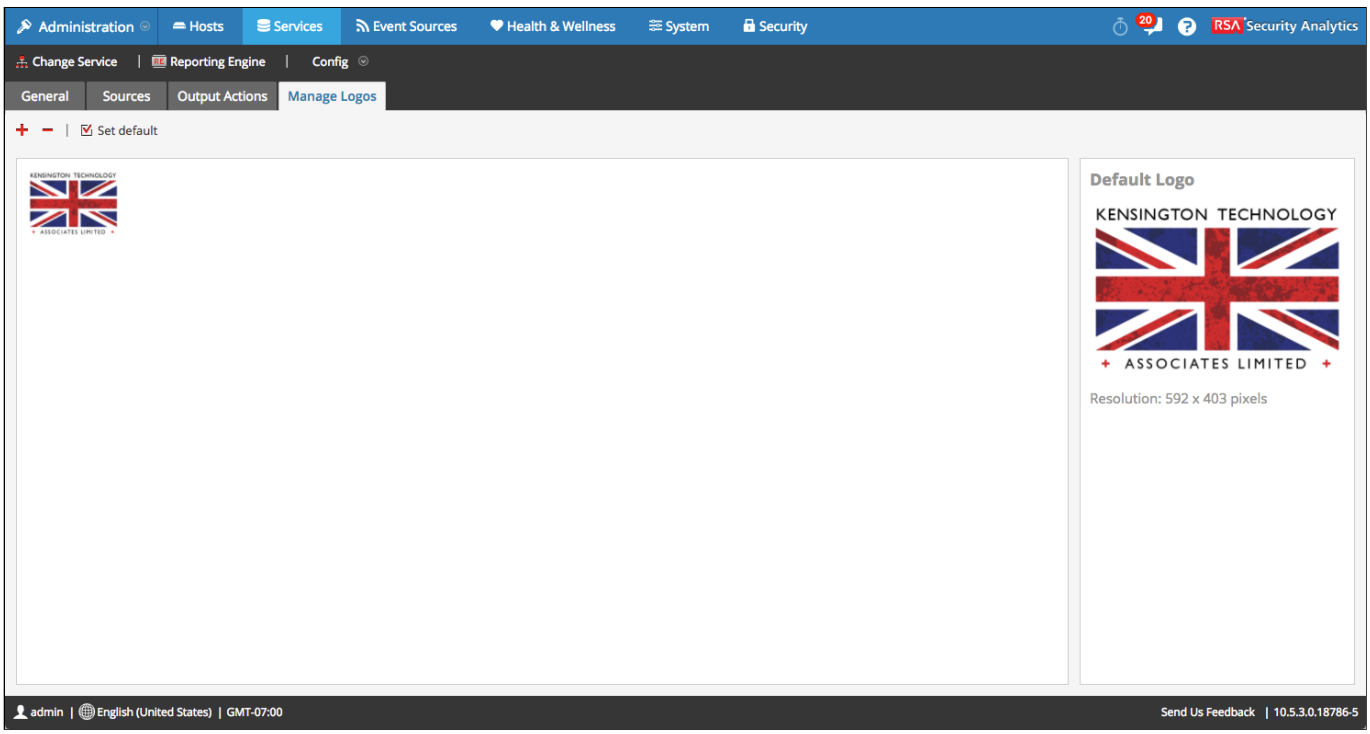
- executionid
- filename
- filesize
- filetype
- filechecksum
- hashingalgorithm
- reportexecutionstarttime
- reportname
- status
- status description

- **Network Share** - The RE can write the output file to a network share. The share **must** be mounted on the local filesystem of the RE/SA host.



Configuring a Custom Logo

From the **Manage Logos** page of RE Configuration, we can upload custom logos which can be used in Reports. The supported file types are jpg, png, and gid. The maximum file size is 500 kilobytes.



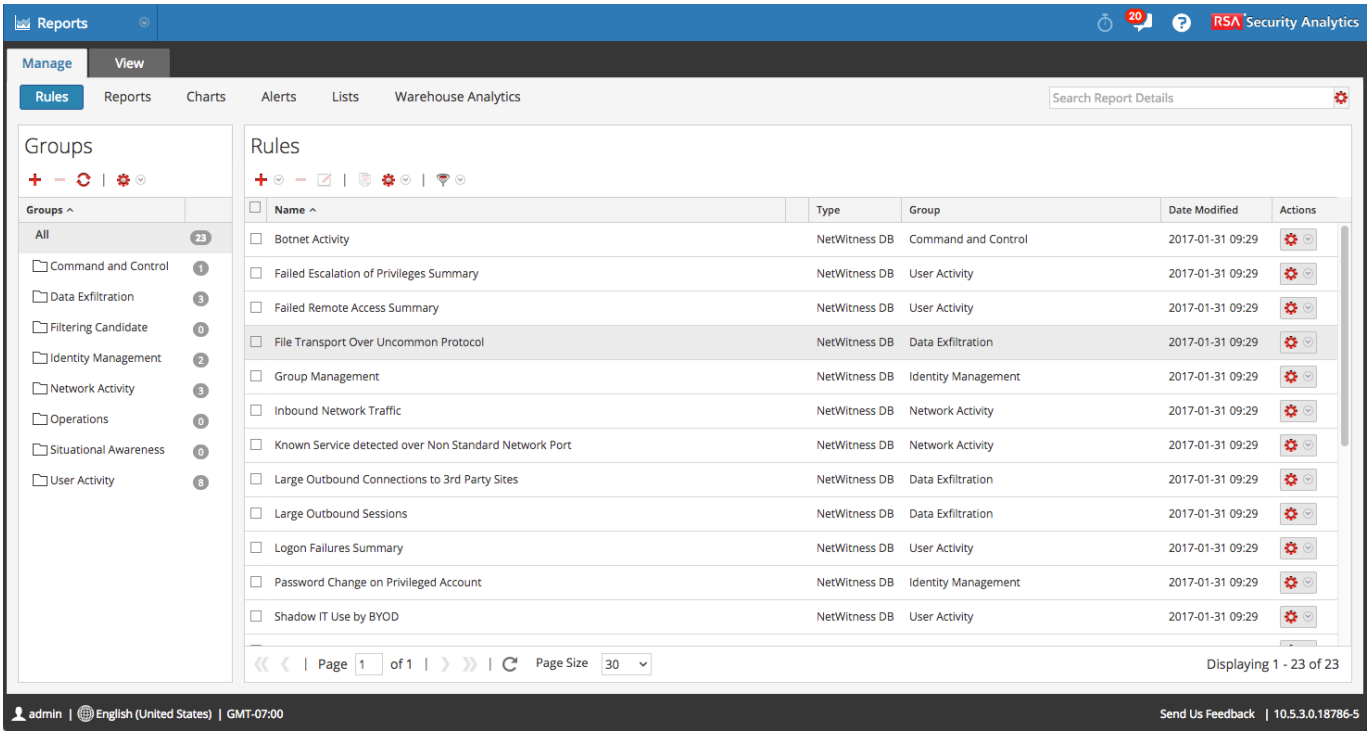
The Good Stuff

Now that we're through the initial configuration of the RE service, we can crack on with looking at the **Reporting** component of the UI.

Organisation of 'Reports' in the UI

Reporting in the UI is all under the **Start Menu** item

. Let's visit here now.



We can see that **Reports** is organised into two top-level views: **Manage** and **View**. **Manage** is where one goes to configure RE content - rules, reports, charts, alerts, and lists. **View** is a place where generated Reports, Charts, and Alerts can be viewed. We'll look at **View** later.

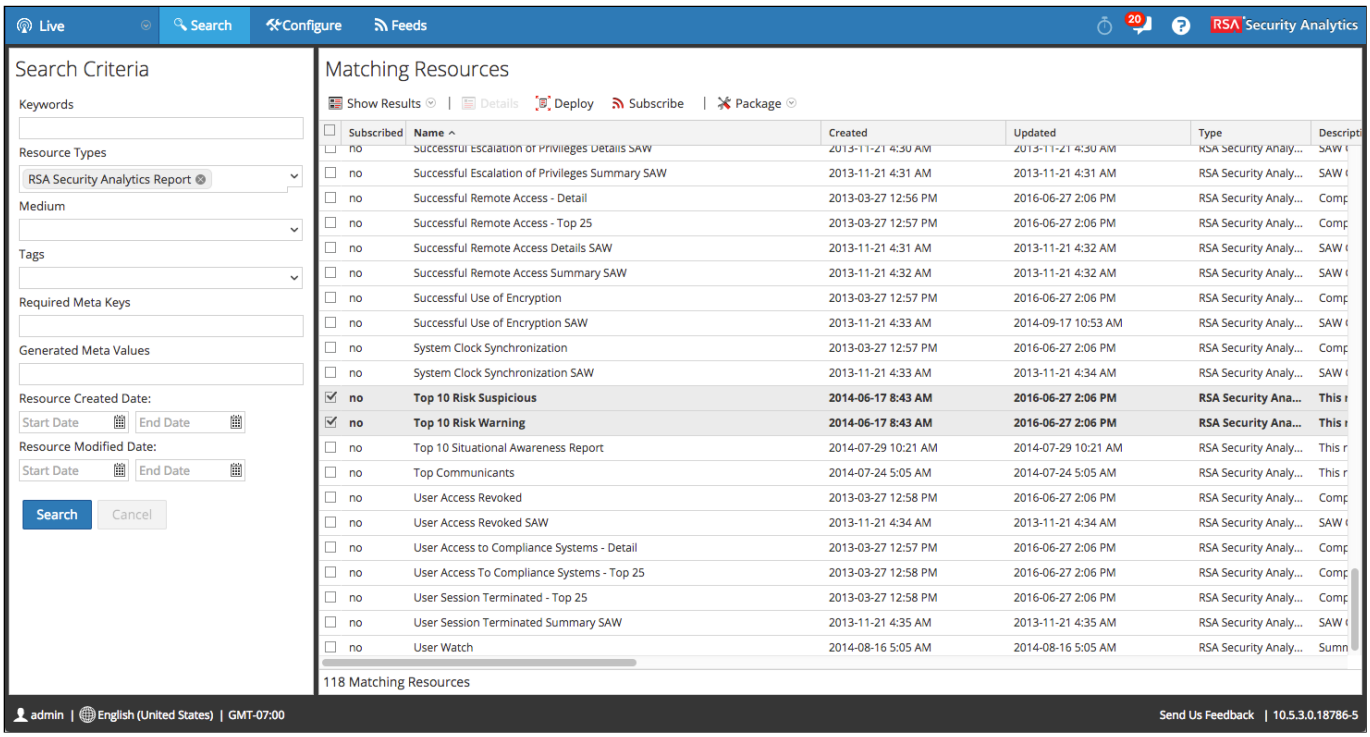
From the above screenshot, we can also see that the RE permits its content to be organised into **Groups**, or a folder structure. This is purely for organisational purposes. The exception to this is **Alerts**, which cannot be similarly categorised.

Live Content for Reporting

Before we delve into rules, let's get some example content into our Reporting Engine. It so happens that RSA Live contains quite a bit of content for reporting. Go ahead and deploy some of it now to your Reporting Engine. The resource types you want to deploy are:

- Security Analytics List
- Security Analytics Report - This will also deploy rule and list dependencies
- Security Analytics Rule

From **Live->Search**, deploy the **"Top 10 Risk Suspicious"** rule, which we will use as our first example.

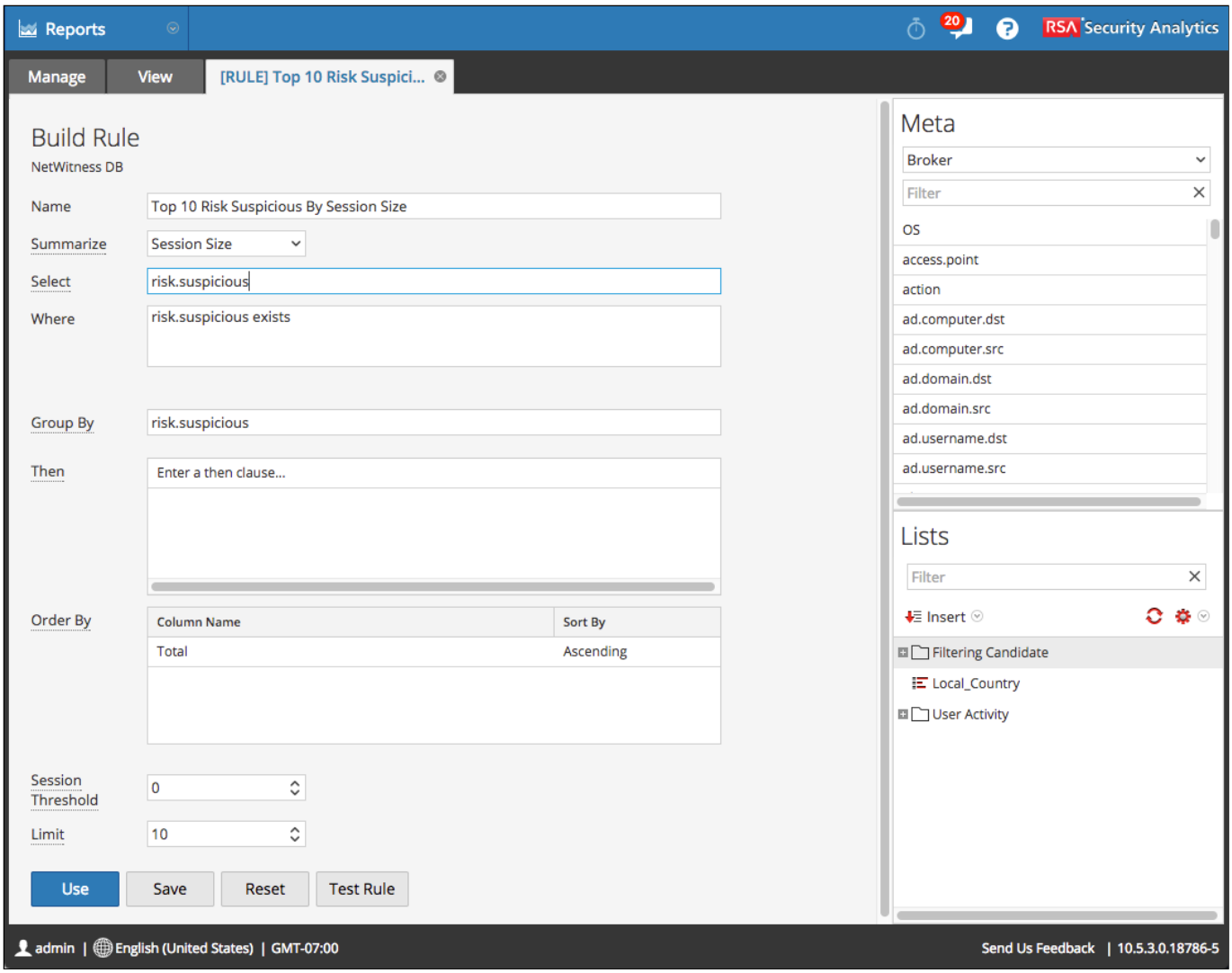


Top Tip: The first time you deploy reporting content from Live, Live may say that there were errors. Just click the **Retry** button and it should work the second time.

Reporting Rules

Now that we've covered some of the configuration fundamentals, let's move into the meat of the RE. Nearly everything in the RE is based on the concept of '**rules**'. A rule is the basic reporting unit that undergirds reports, charts, and alerts.

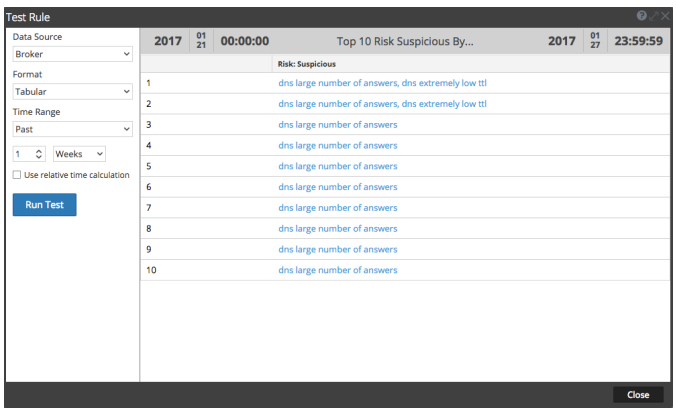
Open the rule "**Top 10 Risk Warning By Session Size**".



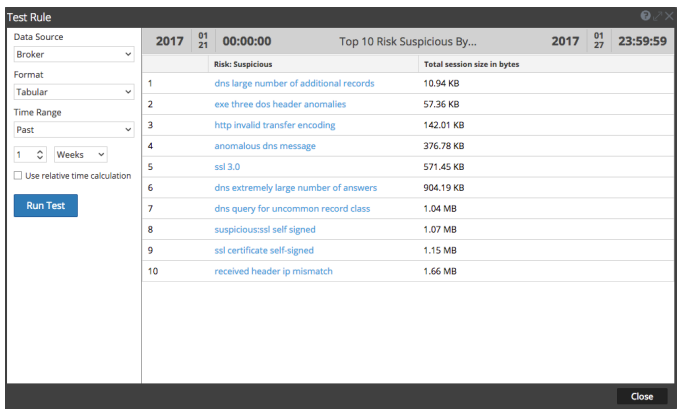
A rule consists of these elements:

Rule Name - Self-explanatory

Summarize By - This is an aggregation function which causes the results of the NWDV query to be summarised by either **Event Count**, **Packet Count**, **Session Size**, or a **Custom** value. You can also choose not to summarise results at all by choosing **None**. What this means is that if a query returns more than one row with the same value, then it will combine the identical rows into a single row, with an additional column added for either the combined event count of all sessions (a row represents an NWDB session), packet count, session size, or custom value. **Summarize By** interacts with, or is qualified by, the **Group By** field.



None chosen for Summarize By. Notice that we can only see two unique values under **risk.suspicious**.



Summarized by **Session Size**. Notice the additional column, and the fact that every **risk.suspicious** value is unique.

SELECT Clause - This tells the RE which meta keys' values to show in the report. Every key listed here will be in its own **column** in a report. Often a single key is used here, but you are not limited to one. Multiple keys should be comma separated. Selecting multiple keys may have adverse effects on how report data is displayed, however. The **SELECT** clause is mandatory.

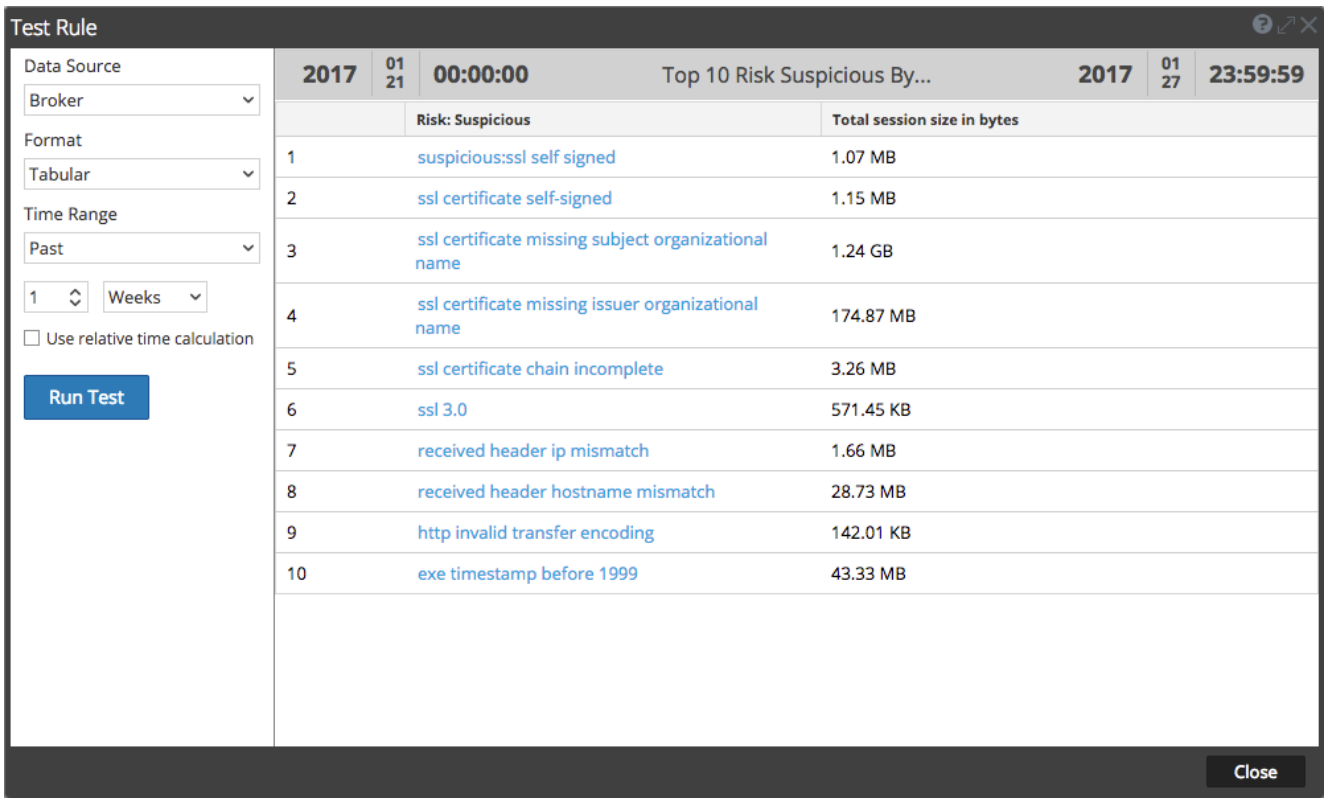
WHERE Clause - This is the query statement to execute against the NWDB data source. This serves to narrow down or filter the data set so that only desired data is returned. The **WHERE** clause is optional. The **WHERE** clause uses the exact same syntax as **Investigation queries**. For instance, perhaps we want to SELECT **ip.src**, so that the first column of our report is ip.src. But we don't want every row which defines ip.src, which would be

every single packet session. Instead, we could put "**risk.suspicious exists**", so that the query only returns rows, or sessions, where the **risk.suspicious** key is defined.

Group By - The **Group By** field works together with the **Summarize By** option, by instructing the **Summarize By** function which key, or column, it should use to find aggregation matches. In our example rule, we can see that we have **risk.suspicious** in the SELECT clause, so it follows that we want to summarise by grouping rows together that have the same **risk.suspicious** value. The purpose of the **Group By** function is really only variable if SELECTing multiple keys. Otherwise, just use the same key you used in your **SELECT** clause. This field will not be displayed if **None** is chosen for **Summarize By**.

THEN Clause - Additional actions to perform on the result of a query. One does this by performing specified functions against the returned dataset. Examples are further filtering the data set, and looking up additional data based on the result and adding it to the report in new columns or child rows. We'll cover these functions in a little while.

Order By - This instructs the RE how to sort the NWDB query results. **Total** means to sort the result by the Summarize By column values. So if you're sorting by Session Size in Descending order, the results will have the row with the largest aggregate session size at the top, with every row having lower session size than the last. **Value** means to sort by the first SELECTed column. If this is text, it would be in alphabetical order. You can also insert addition.



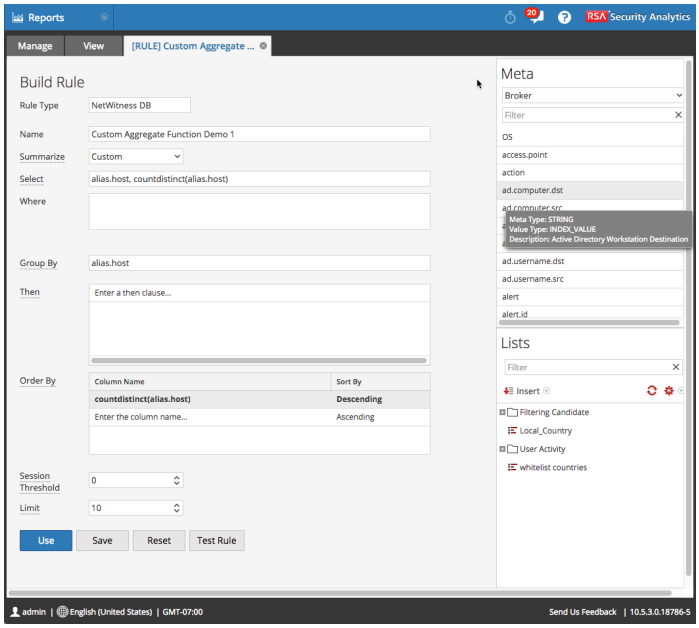
Order By Descending Value (reverse alphabetical order)

Session Threshold - This limits the number of rows, or sessions to include in the **Summarize By**, or aggregation, function. A value of zero includes all sessions in the aggregation results. For example, setting the **Session Threshold** to zero will display a truly accurate **Event Count** column, but setting the threshold to 10 will limit the Event Count to a maximum of 10. Also using a Session Threshold of 10, if summarising by **Session Size**, the report would display an inaccurate session size value if the NWDB query returns more than 10 sessions with the same **Group By** value.

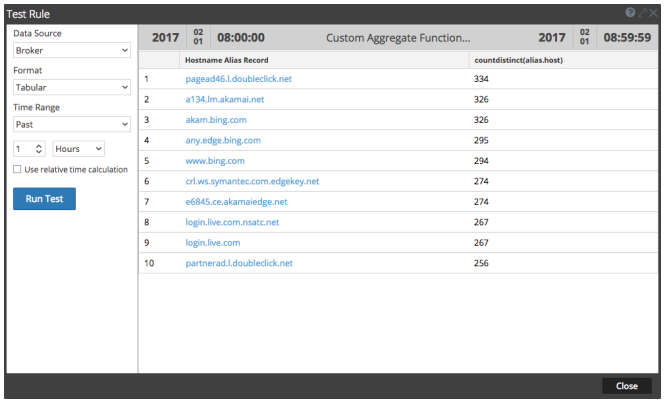
Limit - Limits the number of rows displayed in the report to this value. Clear the value entirely to see all results.

Meta Aggregate Functions

Choosing '**Custom**' from the **Summarize** dropdown gives one more options when it comes to summarising output. One isn't limited to only Event Count, Packet Count, and Session Size. By inserting a Meta Aggregate Function in a rule's **SELECT** clause, we have access to additional methods, using the syntax **function*(key)***. For example, if we were to use the aggregate function **countdistinct()** in a rule, the rule would look like:



Custom Meta Aggregate Rule

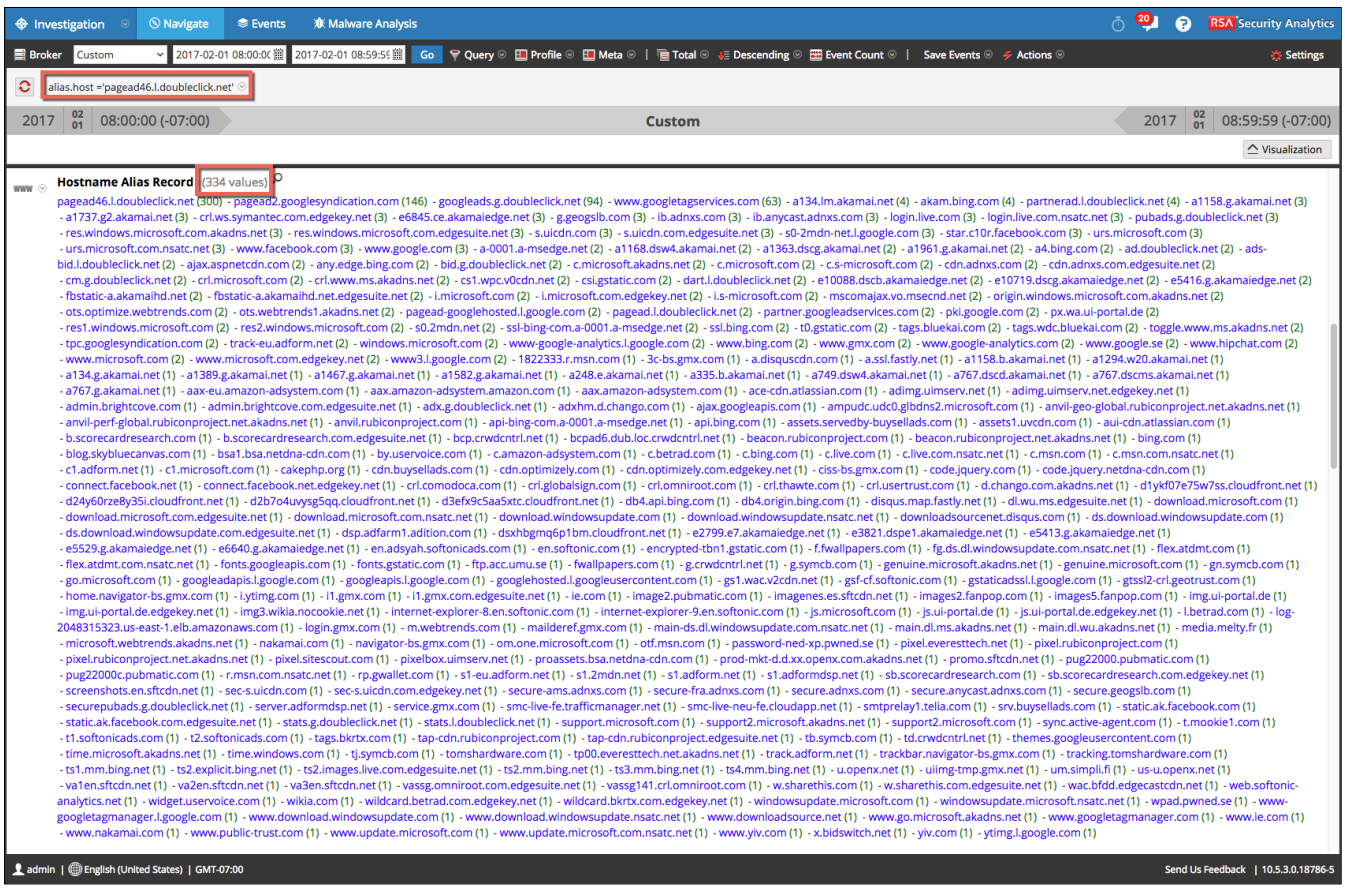


Rule Output

- **sum(*key*)** - Adds all meta values together and returns result. Works only on meta keys of numeric formats (not Text, Date, IP Address, etc). E.g. "**sum(payload)**"

- **count(*key*)** - Returns how many total meta values were returned from a query containing a given value X. Note that this does not mean how many sessions it was seen in, but how many values in total were returned for sessions containing a given value. This includes values which are repeated. So if the given meta value is contained in a session more than once, every time it is listed would count as one.

- **countdistinct(*key*)** - This means that for all sessions containing a given value X, how many distinct meta values are there, including X? E.g. if I were to query in **Investigation** for "**alias.host ='pagead46.l.doubleclick.net'**" on the past hour, I would see that there are 334 distinct values under the **alias.host** key.

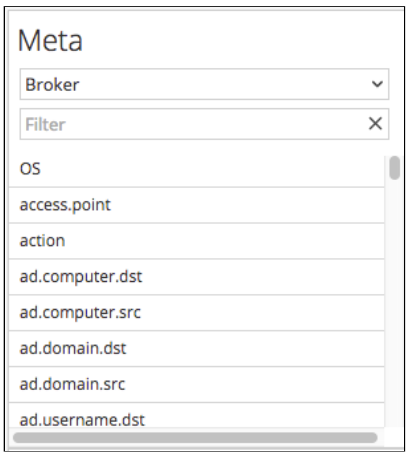


- **min(*key*)** - Returns the lowest value for the given key. If numeric, this is the lowest number. If string, it uses alphabetical order (i.e. 'a' is lower than 'b').

- **max(*key*)** - Returns the highest value for the given key. If numeric, this is the highest number. If string, it uses alphabetical order (i.e. 'b' is greater than 'a').
- **avg(*key*)** - Returns the average of all values for the given key. Works only on meta keys of numeric formats (not Text, Date, IP Address, etc).
- **first(*key*)** - Returns the first occurrence of the meta value returned for a given key, in terms of time.
- **last(*key*)** - Returns the first last occurrence of the meta value returned for a given key, in terms of time.
- **len(*key*)** - Returns the length, in bytes, of the meta values returned for a given key. For a string, this means the number of characters. For any IPv4 address, it will always return 4. When using this, it may be necessary to adjust the **Session Threshold** to limit the result set.
- **distinct(*key*)** - Returns all distinct meta values for the given key.

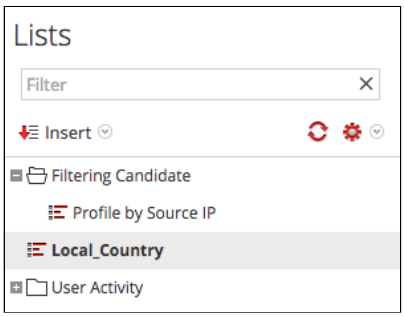
The Meta Panel

There is a Meta panel on the right-hand side of the Rules window which shows which meta keys are available to use in your rule.



The Lists Panel

Lists can be used in the **WHERE** and **THEN** clauses in NWDB rules. To reference a list, use the syntax "**$[/*FolderName*/*ListName*]**". Or you can use the **Lists** panel by selecting the list you want to reference, then the Insert button, and either **Where** (to insert into the **Where** clause) or **Then** (to insert into the **Then** Clause). Don't worry that we haven't explained what **Lists** are; they'll be covered later in the lesson.



Testing a Rule

Before using a rule in a report, chart, or alert, you want to ensure that it functions properly. To do this, click the **Test Rule** button at the bottom of the rule. This pops up a modal dialog with a few fields:

- **Data Source** - Which NWDB source, or SA Core service, to run the query against.
- **Time Range** - Same as Investigation. Chooses the time window of indexed data to query.
- **Use relative time calculation** - Ticking this box makes the timeframe relative to the current time. If you do not check this box, it means the last N fully elapsed hours/days/weeks/months/years. For example, if the current time is 21:19, then if the **Relative** box is ticked, the query timeframe is 21:20:00 through 22:19:59. If the **Relative** box is not ticked, then the query timeframe is 21:00:00 - 21:59:59.
- **Time Range** - Either **Past** or **Range**. Self-explanatory.
- **Format** - How to display the results. Tabular means to display the results in report format, with rows and columns. The rest are all chart values

Tabular

Test Rule

Data Source: Broker
Format: Tabular
Time Range: Past
1 Weeks
Use relative time calculation
Run Test

2017 01 21 00:00:00 — Top 10 Risk Suspicious By... — 2017 01 27 23:59:59

	Risk: Suspicious	Total session size in bytes
1	ssl certificate missing subject organizational name	1.24 GB
2	ssl certificate missing issuer organizational name	174.87 MB
3	dns extremely low ttl	111.79 MB
4	dns large number of answers	70.02 MB
5	exe timestamp before 1999	43.33 MB
6	exe dos loader missing	34.51 MB
7	received header hostname mismatch	28.73 MB
8	ssl certificate chain incomplete	3.26 MB
9	received header ip mismatch	1.66 MB
10	ssl certificate self-signed	1.15 MB

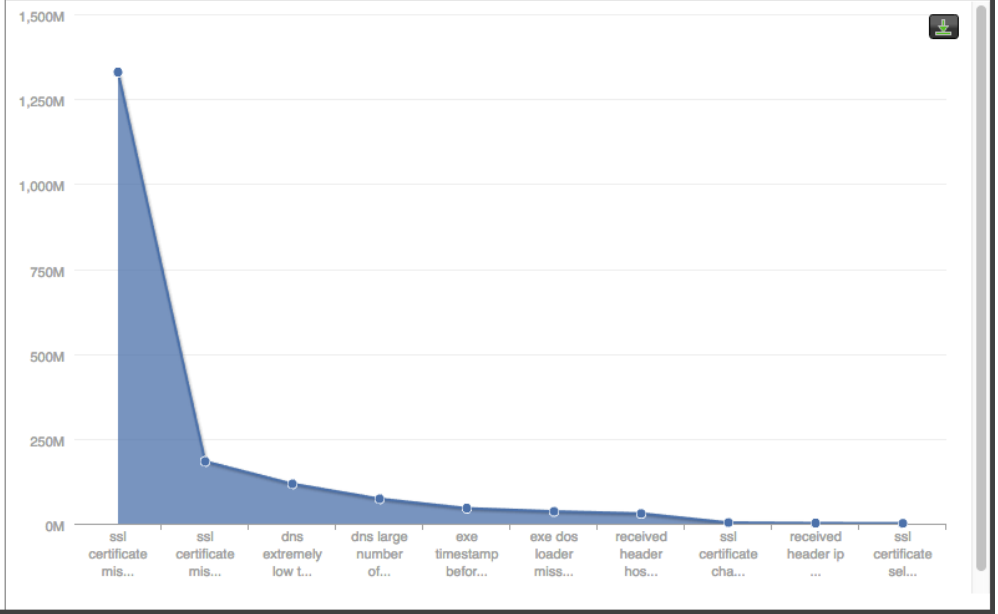
Close

Area

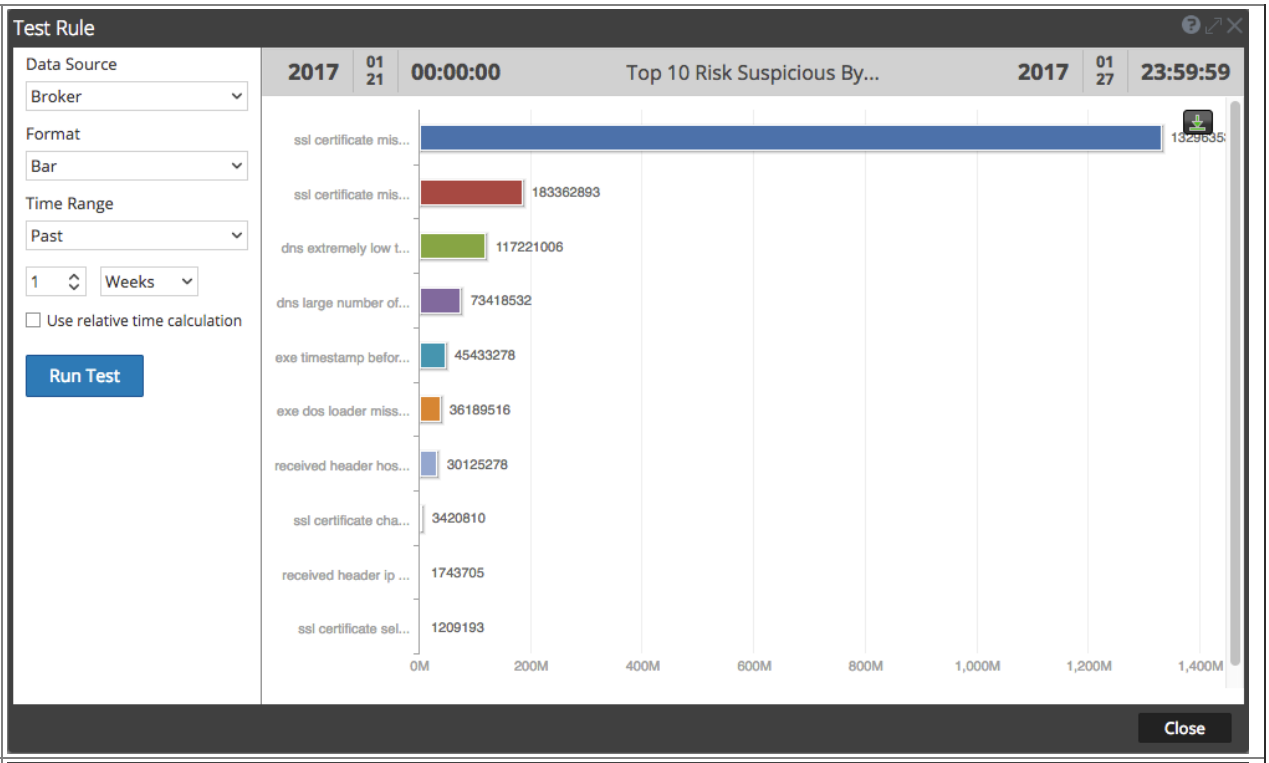
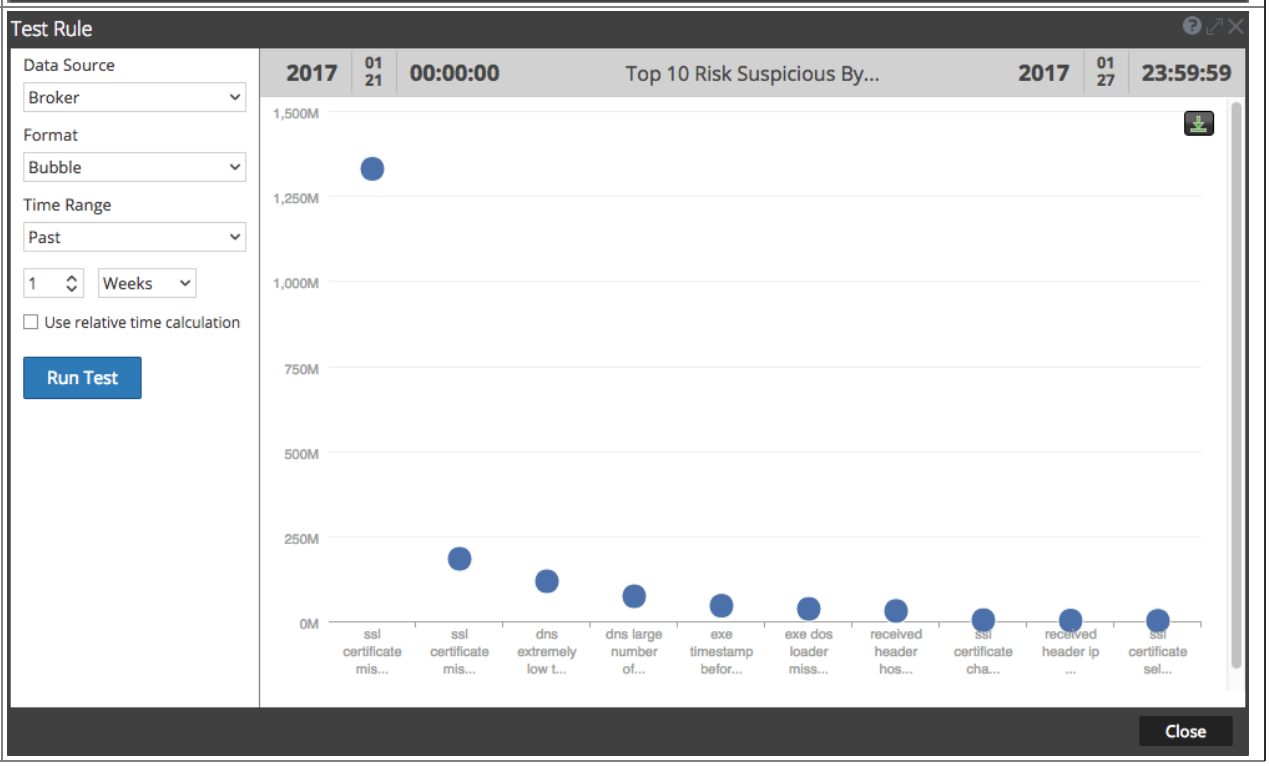
Test Rule

Data Source: Broker
Format: Area
Time Range: Past
1 Weeks
Use relative time calculation
Run Test

2017 01 21 00:00:00 — Top 10 Risk Suspicious By... — 2017 01 27 23:59:59

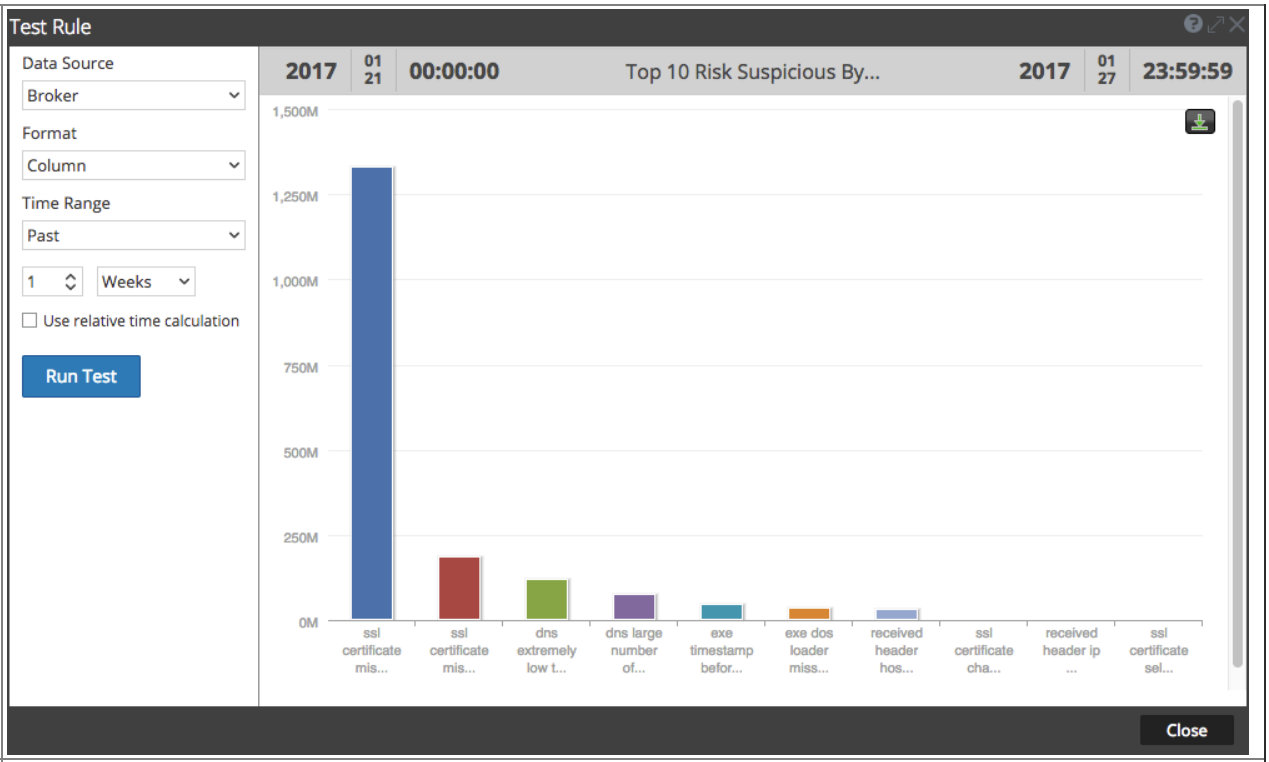
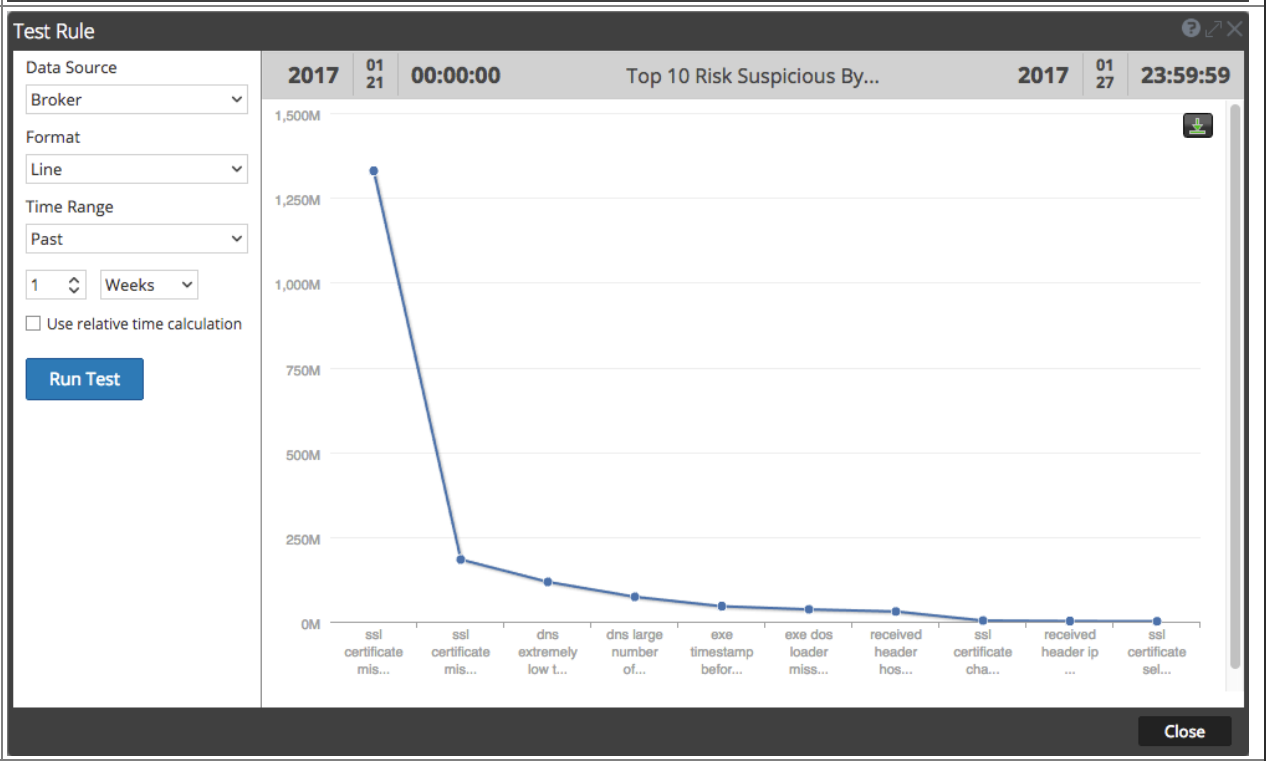


Close

Bar	
Bubble	

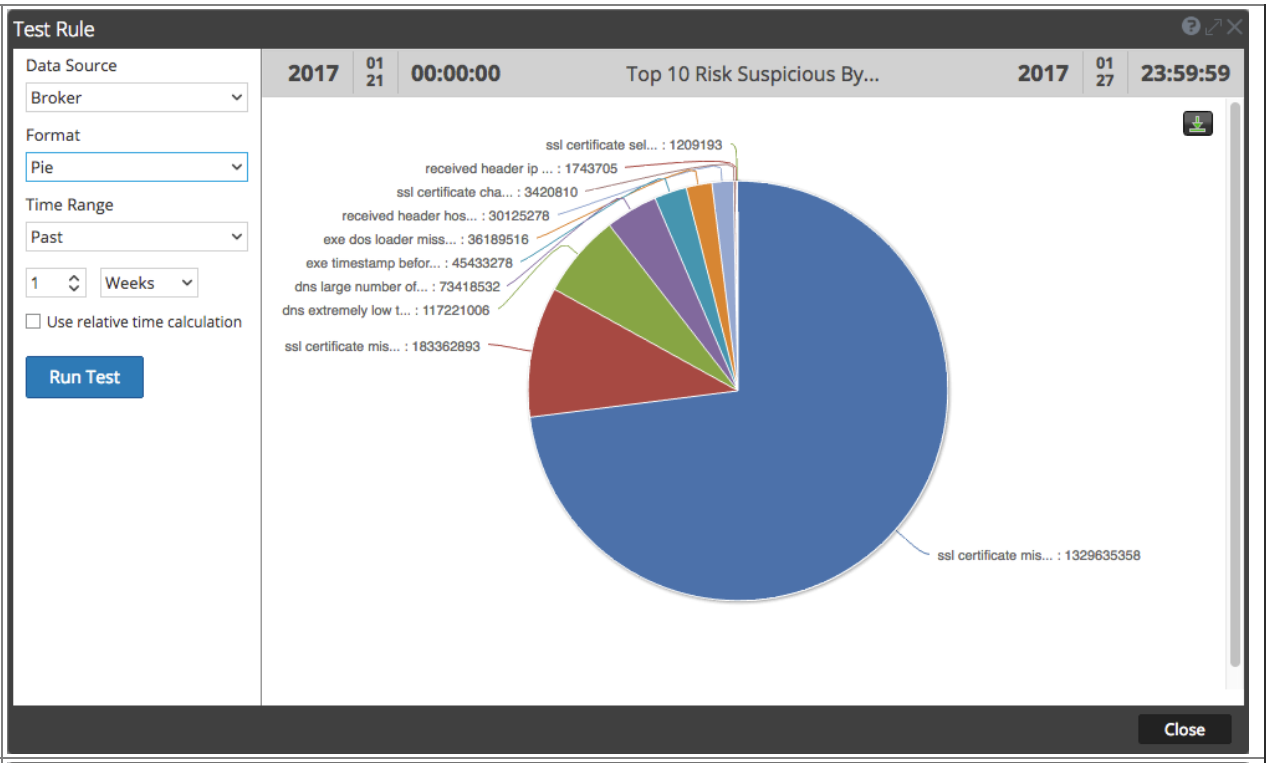
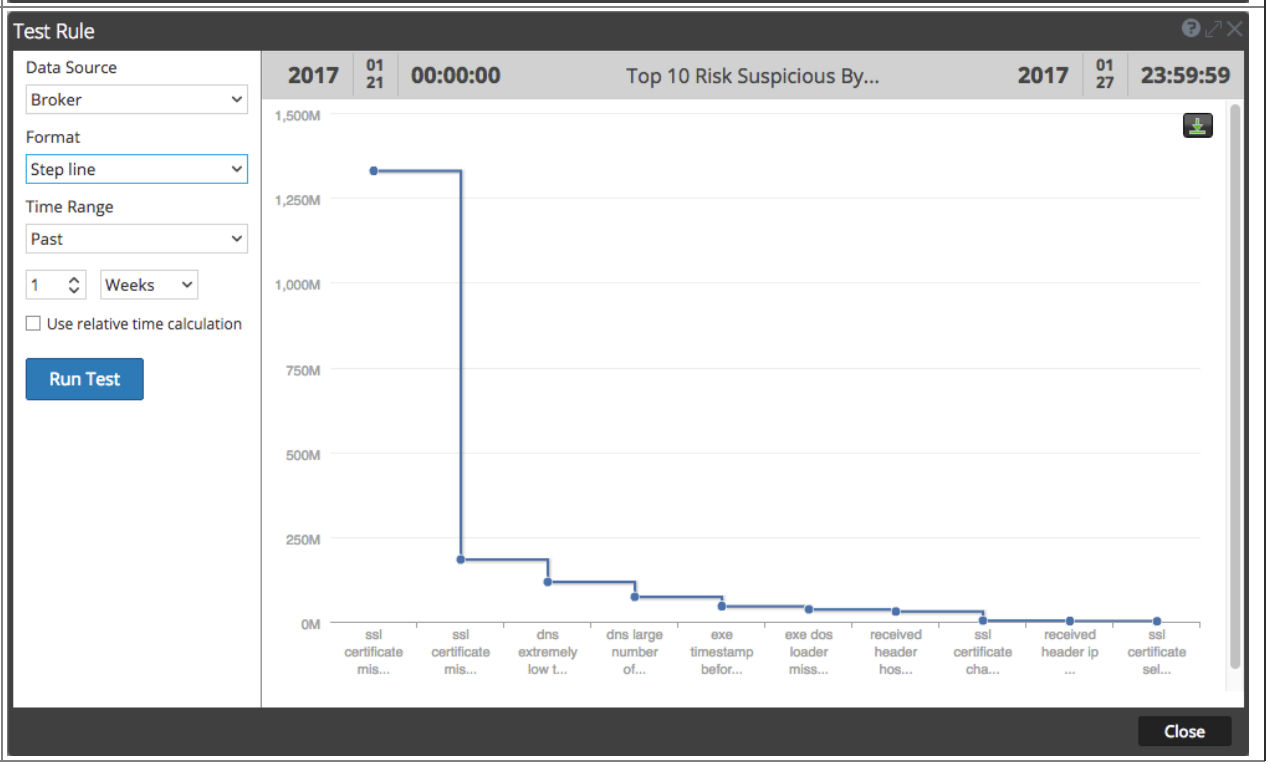
Test Rule
Data Source
Broker
Format
Bar
Time Range
Past
1
Weeks
Use relative time calculation
Run Test
2017 01 21 00:00:00
Top 10 Risk Suspicious By...
2017 01 27 23:59:59
ssl certificate mis...
ssl certificate mis...
dns extremely low t...
dns large number of...
exe timestamp befor...
exe dos loader miss...
received header hos...
ssl certificate cha...
received header ip ...
ssl certificate sel...
183362893
117221006
73418532
45433278
36189516
30125278
3420810
1743705
1209193
0M
200M
400M
600M
800M
1,000M
1,200M
1,400M
Close

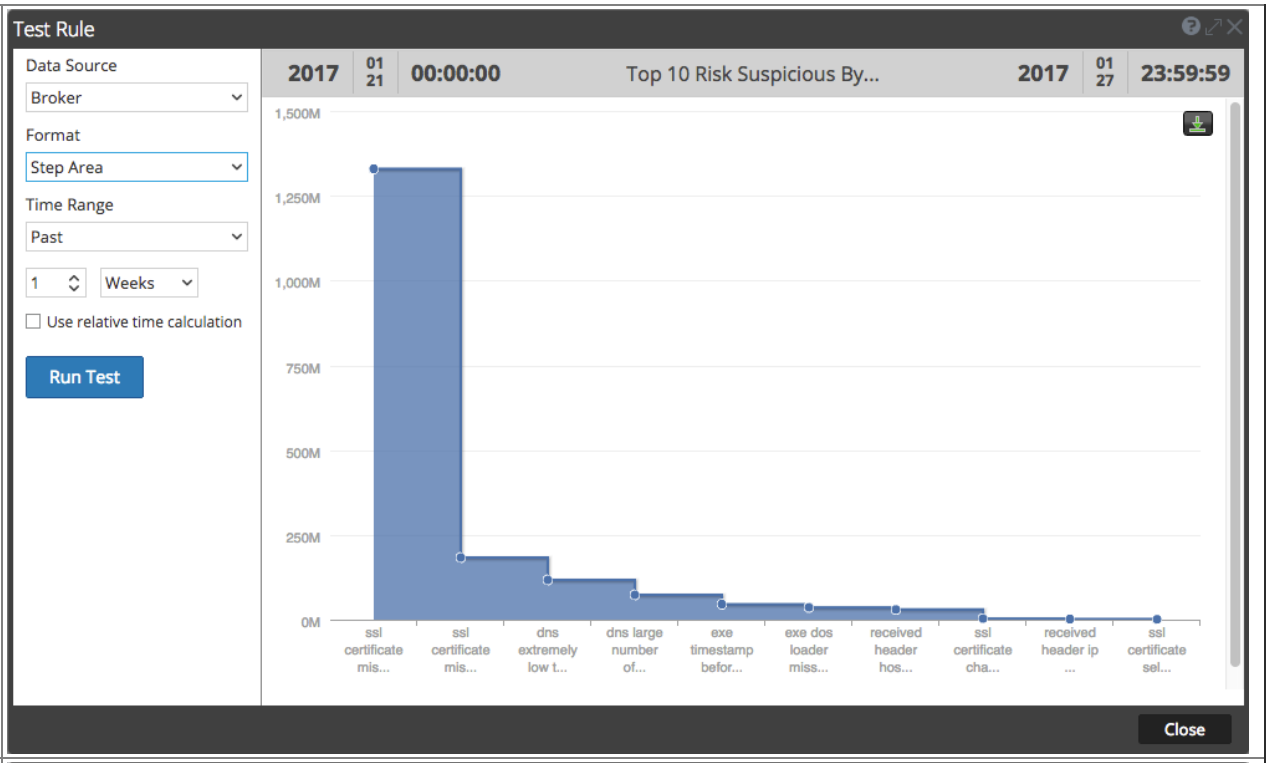
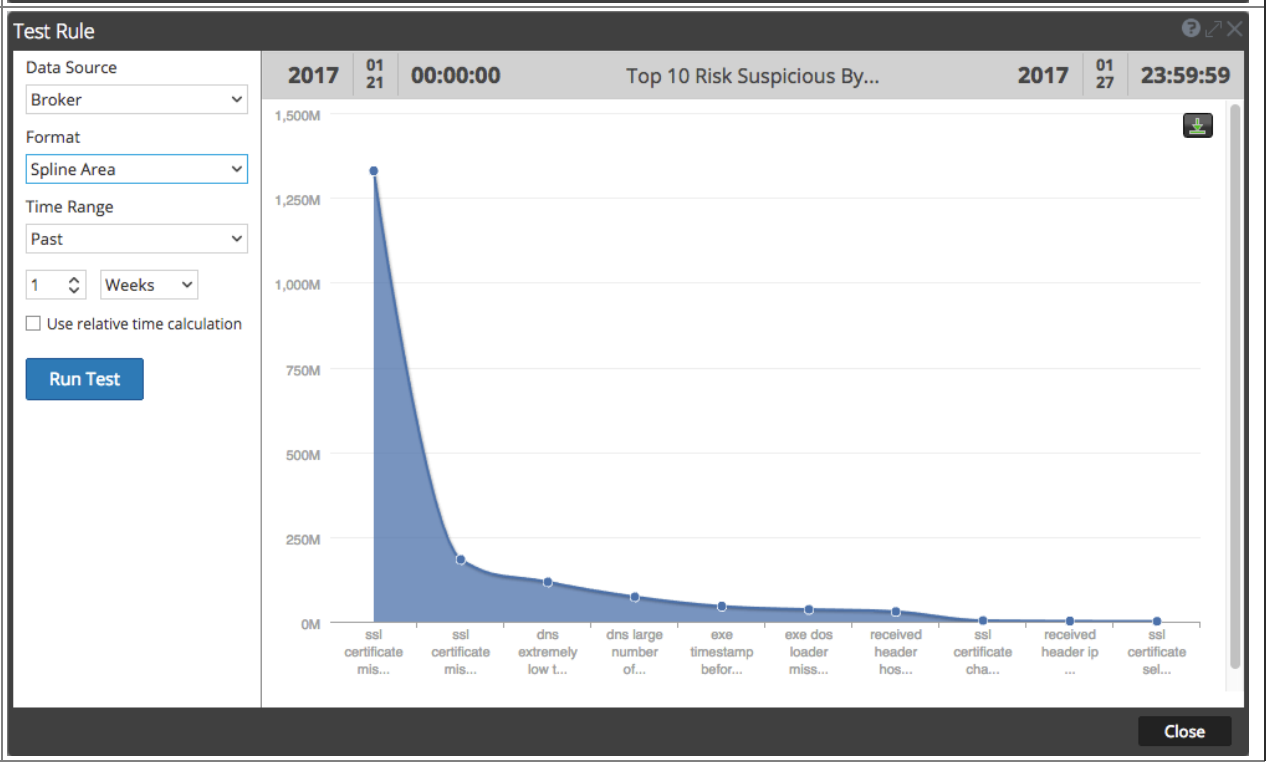
Test Rule
Data Source
Broker
Format
Bubble
Time Range
Past
1
Weeks
Use relative time calculation
Run Test
2017 01 21 00:00:00
Top 10 Risk Suspicious By...
2017 01 27 23:59:59
1,500M
1,250M
1,000M
750M
500M
250M
0M
ssl certificate mis...
ssl certificate mis...
dns extremely low t...
dns large number of...
exe timestamp befor...
exe dos loader miss...
received header hos...
ssl certificate cha...
received header ip ...
ssl certificate sel...
Close

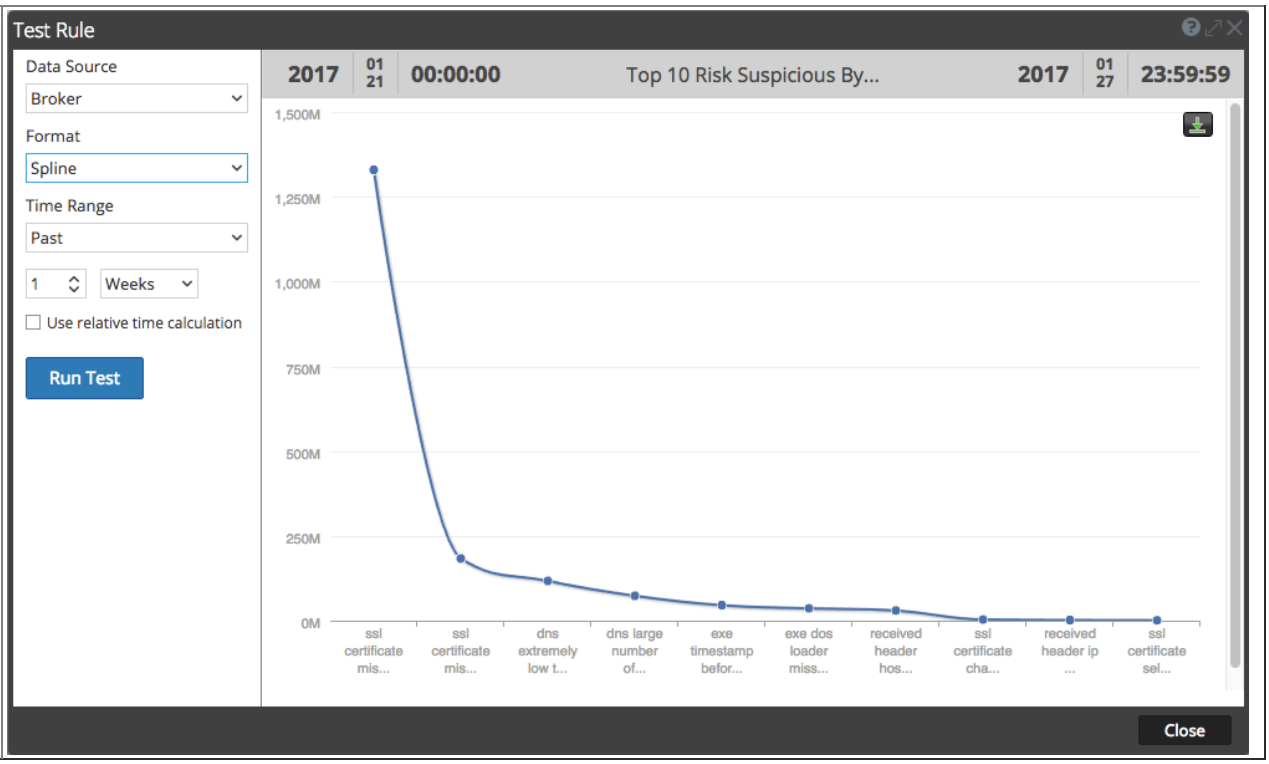
Column	
Line	

Test Rule
Data Source
Broker
Format
Column
Time Range
Past
1
Weeks
Use relative time calculation
Run Test
2017 01 21 00:00:00
Top 10 Risk Suspicious By...
2017 01 27 23:59:59
1,500M
1,250M
1,000M
750M
500M
250M
0M
ssl certificate mis...
ssl certificate mis...
dns extremely low t...
dns large number of...
exe timestamp befor...
exe dos loader miss...
received header hos...
ssl certificate cha...
received header ip ...
ssl certificate sel...
Close

Test Rule
Data Source
Broker
Format
Line
Time Range
Past
1
Weeks
Use relative time calculation
Run Test
2017 01 21 00:00:00
Top 10 Risk Suspicious By...
2017 01 27 23:59:59
1,500M
1,250M
1,000M
750M
500M
250M
0M
ssl certificate mis...
ssl certificate mis...
dns extremely low t...
dns large number of...
exe timestamp befor...
exe dos loader miss...
received header hos...
ssl certificate cha...
received header ip ...
ssl certificate sel...
Close

Pie	
Step Line	

Step Area	
Spline Area	

Spline	

The THEN Clause

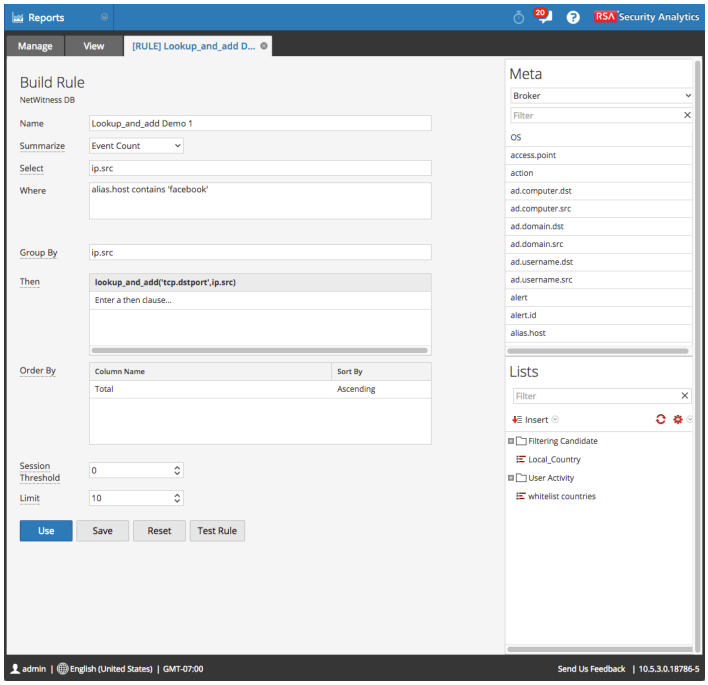
The **THEN** clause is a way to do additional things with a data set returned from a rule's **SELECT** and **WHERE** clauses. **THEN** statements use a fairly standard function syntax from the programming world: **function_name(parameter1, parameter2, parameter3, etc)**. **THEN** statements can be combined to customise the result set; use one statement per line. These are the functions which can be used in a THEN clause. Parameters in **green** are optional.

- **dedup(*field*)** - De-duplication function. Removes duplicate values from a result set which is unsorted, meaning that the **Summarize** field must be set to **None** to use **dedup()** - i.e. it cannot be used in an aggregation rule. ***field*** is the value set (i.e. column) to de-duplicate. E.g. " **dedup(alias.host)** ".

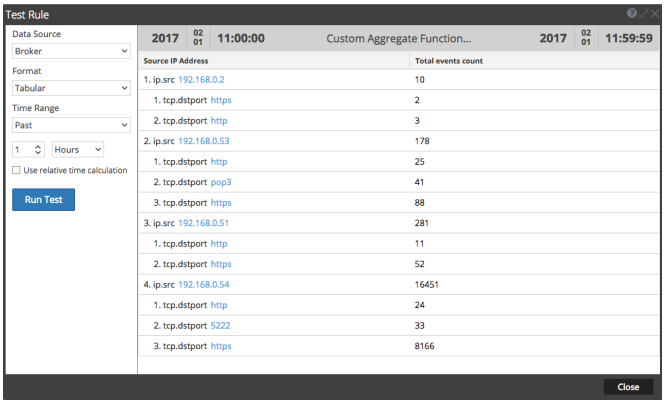
- **filter_on(*filterText, field, boolean exactMatch*)** - Removes values from the result set where ***filterText*** is not in the ***field*** key. ***field*** is which result field to search in for ***filterText***. ***exactMatch*** is either **true** or **false**, defaulting to **true.** Setting ***exactMatch*** to **true** means that ***filterText*** must match the result exactly, and a value of **false** means that the ***filterText*** may only be contained within the result, or a subset of the result. E.g. " **filter_on('google', alias.host, false)** ". If not specified, ***matchExact*** defaults to **true**. ***field*** must be present in the rule's **SELECT** statement, which can make this of limited utility.

- **filter_out(*filterText, field, boolean exactMatch*)** - Removes values from the result set that contain ***filterText*** within the ***field*** key. ***exactMatch*** is the same as for **filterOn**. ***field*** must be present in the rule's **SELECT** statement, which can make this of limited utility.

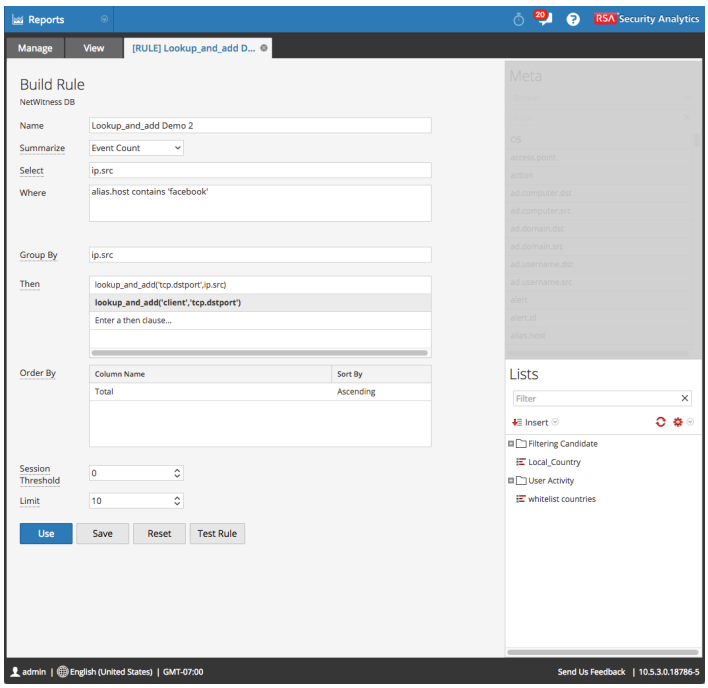
- **lookup_and_add(*field, column*, int *limit*, boolean *inherit*, *extraWhereClauseString*, boolean *aggregate*)** - Can **only** be used with an aggregate rule, if the **SELECT** clause only has one key and one aggregate function. Adds one or more child rows to every returned row by looking up additional meta and displaying the top N values in the output.
  - ***field*** is the meta key to add to the output
  - ***column*** is the field to attach the new child rows to. This can be a SELECTed field, or it can be an a higher lookup_and_add row.
  - ***limit*** (optional) is the number of child rows to add to the output for each value.
  - ***inherit*** is whether to apply the **WHERE** clause of the parent rule to the **lookup_and_add** query. Defaults to '**true**'.
  - ***extraWhereClauseString*** is used in conjunction with inherit='**false**'. This allows one to specify a new WHERE clause for the **lookup_and_add** query.
  - ***aggregate*** is by default set to **true**. Setting this to false sets **threshold=1, Sort by='value'** and **Order=Ascending** on the new queries. This should be set to false to speed up execution if the rule is run on a long time window or the rule uses aggregate functions.



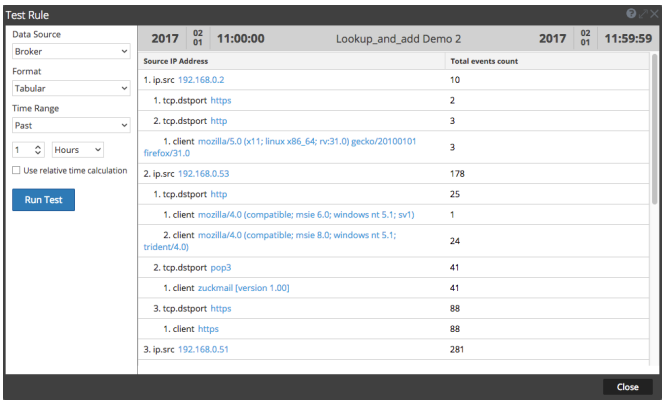
Simple lookup_and_add()



Rule Results



Nested lookup_and_add()



Rule Results

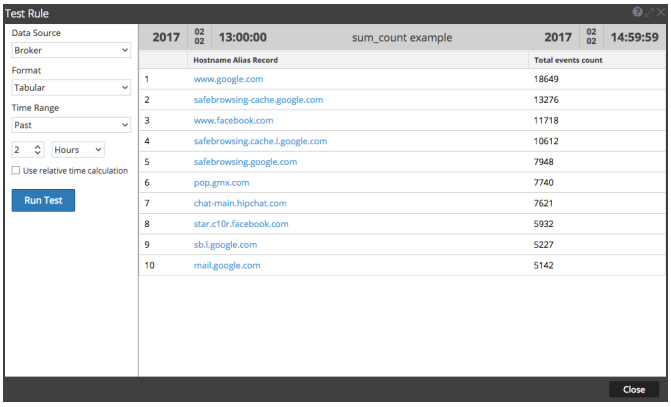
- **max_threshold(*quantityString*, *fieldString*)** - Cleanses the result set of any results with a value higher than ***quantityString***. Without ***fieldString***, it can be used to filter on aggregate values **Event Count**, **Packet Count**, and **Session Size**. E.g. "**max_threshold('200')**" could remove results where the event count exceeds 200. If filtering on Session Size, you can specify a unit - KB, MB, GB, and TB. E.g. **max_threshold('25MB')**

  If you specify ***fieldString***, you can filter the values of custom aggregate functions, or any meta key. If filtering the value of a custom aggregate function, then ***fieldString*** should copy the custom aggregate function name from your SELECT clause. For instance, if your rule has a custom aggregate function '**count(attachment)**' in its SELECT clause, then you should use "**max_threshold(2, count(attachment))**" to filter results where the count of the attachment key exceeds 2.

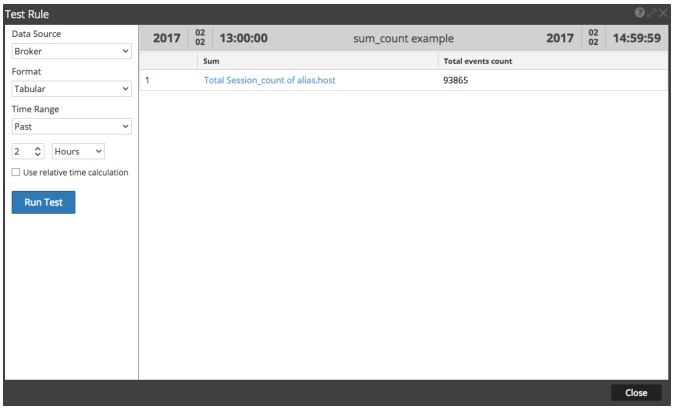
- **min_threshold(*quantityString*, *fieldString*)** - Works the same as **max_threshold()**, except that it means that it sets a minimum threshold rather than a maximum threshold.

- **regex(*regexString*, *field*)** - Applies a Regular Expression, or regex string to the result set. Regular Expressions are a vast topic, and beyond the scope of this course. This is a regex match function, rather than a substitute function. It uses the Java 7 regex library. For supported patterns, visit http://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html. For example: **regex('^mozilla.*', client)**

- **sum_count()** - Returns the sum total of the quantifiers of all results, and replaces all query results with a single row containing this number. Does not work properly with non-aggregation rules. Given a result set:

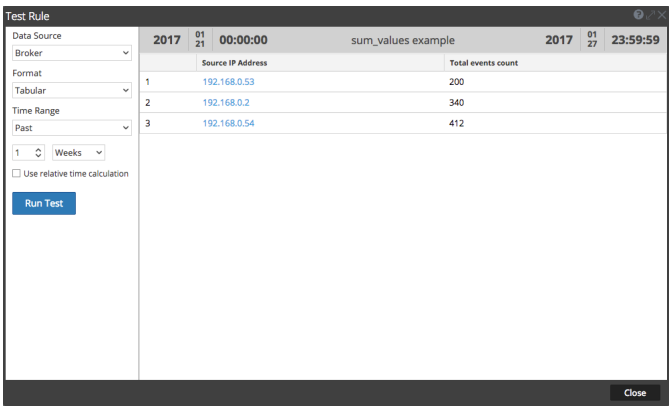


Original Result Set, Without sum_count()

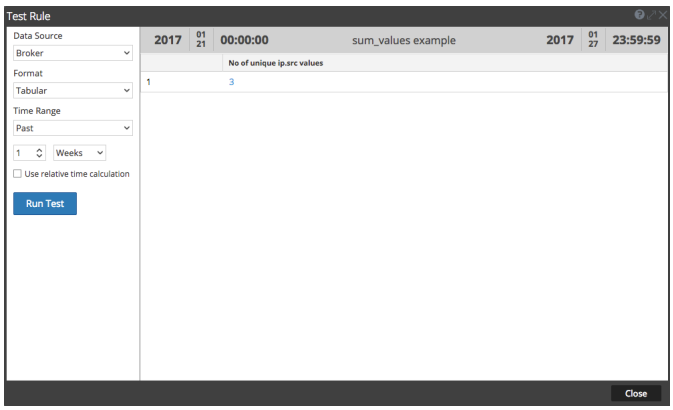


With sum_count() Applied

- **sum_values()** - Returns a count of the number of unique values for a result set, and replaces all query results with a single row containing this number. Does not work properly with non-aggregation rules. Given a result set:



Original Result Set, Without sum_values()



With sum_values() Applied

- **show_whats_new() -** Shows only values that are new to the NWDB. It can only be used with aggregate rules.

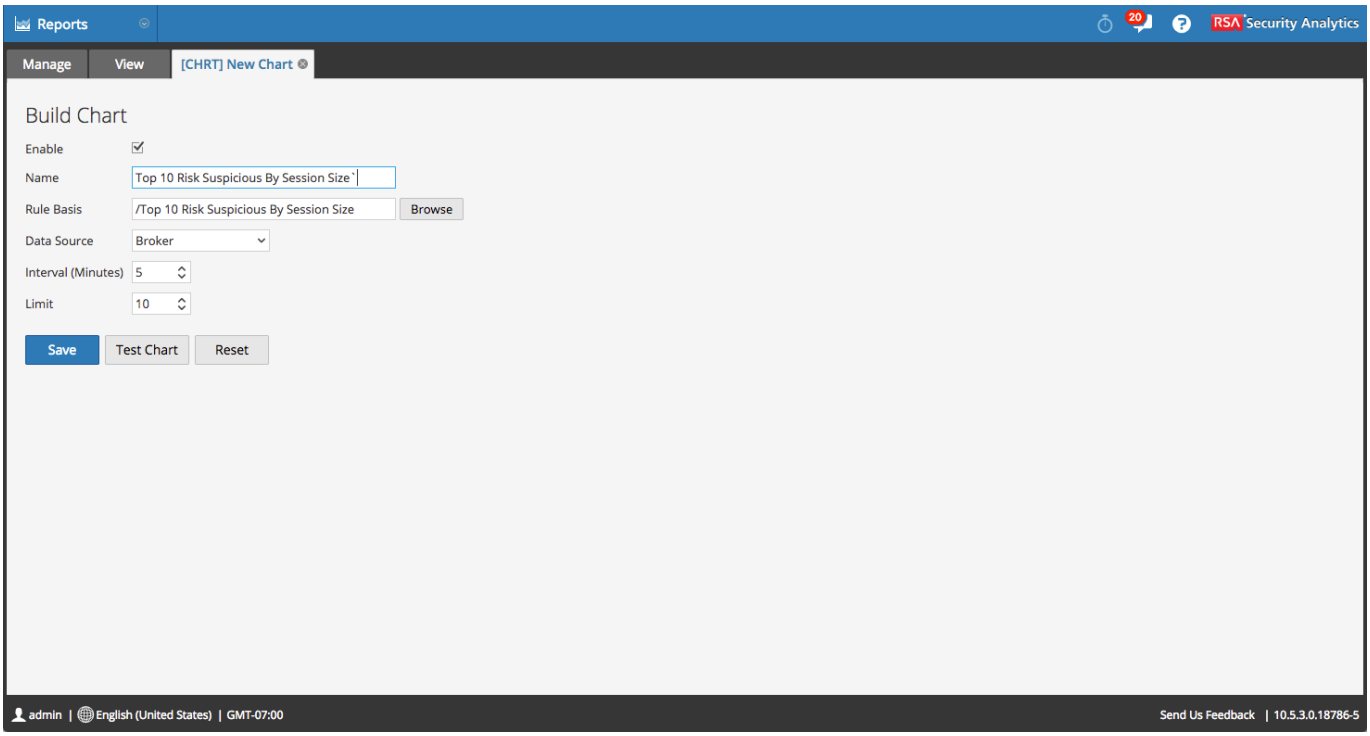
Charts

Reporting charts have two possible uses:

- For use in an SA **Dashboard**.
- For observation in the **Reports->View->Chart**.

It probably doesn't need saying, but you can find reporting Charts under the **Charts** menu of the **Reports->Manag**e.

All a chart really consists of is a name, a **Rule Basis** (a reporting rule), a **Data Source**, an **Interval**, and a row **Limit**. It doesn't even define which format to display the chart in. A chart's format is defined at the point in which it is rendered - within Reports or SA Dashboards. The chart's rule will be run at the Interval defined in the chart. When a chart is rendered, the chart data is pulled from these pre-fetched results.



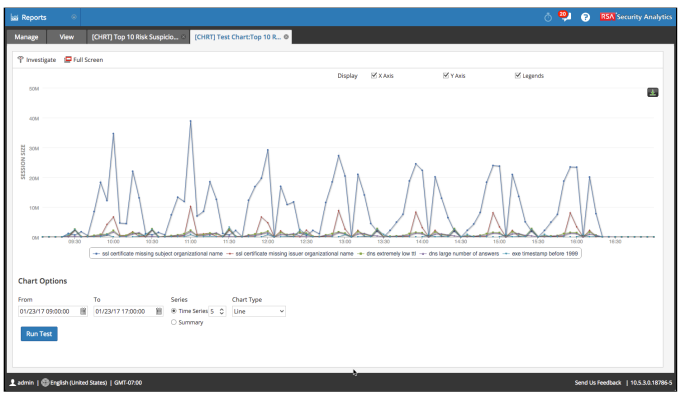
Testing a Chart

Click the **Test Chart** button to open the chart in a tab where it can be rendered. You choose:

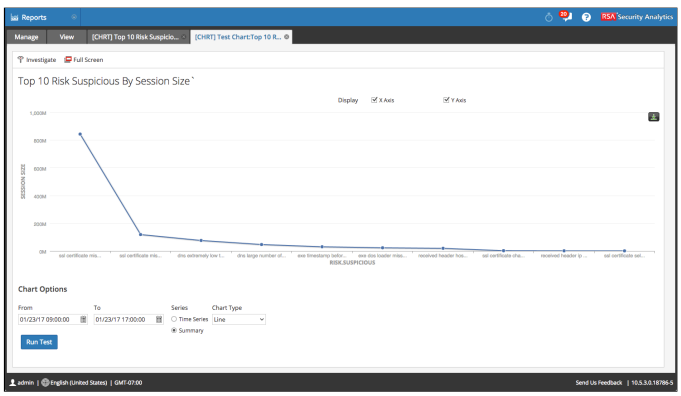
Date Range - Self-explanatory

Chart Type - The same ten chart types (formats) are available as are available from the **Test Rule** dialog (Area, Bar, Bubble, Column, Line, Pie, Step Line, Step Area, Spline Area, Spline).

Series - **Series** defines what the X Axis of the chart will be. **Time Series** means that the selected timeline will be the X Axis, meaning that the values will be plotted over the chosen timeframe, with each value being its own line. **Summary** means that the returned queries' values from the underlying rule will be the X Axis, meaning that there will be a single series, or line for each SELECTed key.



Time Series X Axis



Summary X Axis

When using Time Series, the integer value next to Time Series refers to the plot the top N values.

Top Tip: You can mouse over the chart to see a pop-up displaying the actual value for a given plot point.

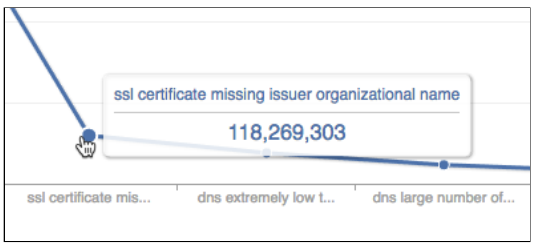
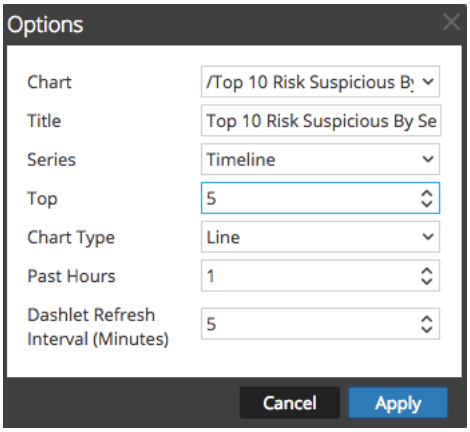


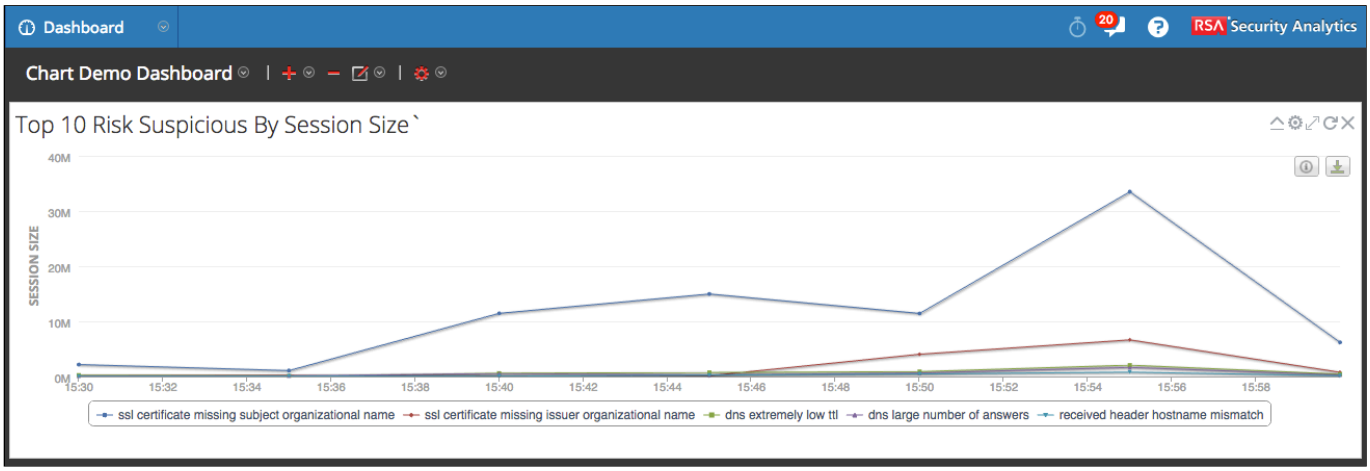
Chart Dashlets

Go ahead and create a new Chart for our example "**Top 10 Risk Suspicious By Session Size**", with the default interval of 5 minutes and a limit of 10. Don't forget to save it!

Once you've done this, create an SA Dashlet which uses this chart. The **Dashlet** type is "**Reports Realtime Chart**". For dashlets, you are limited to only **Pie** and **Column** chart types.

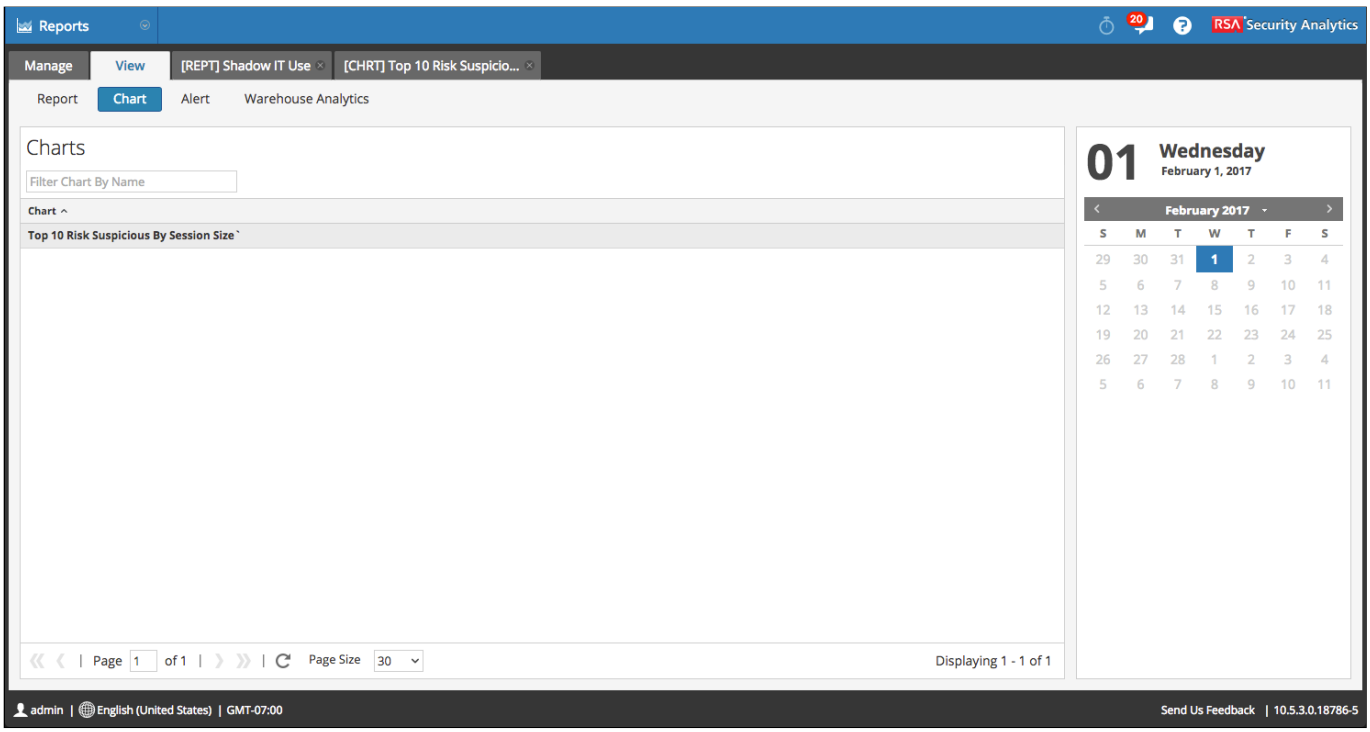


The dashlet options are fairly self-explanatory. **Series** is the same thing as for testing charts. Dashlets can only run on the past N hours of data, so set the number of hours under **Past Hours**. The **Refresh Interval** defines how often the dashlet data gets updated.

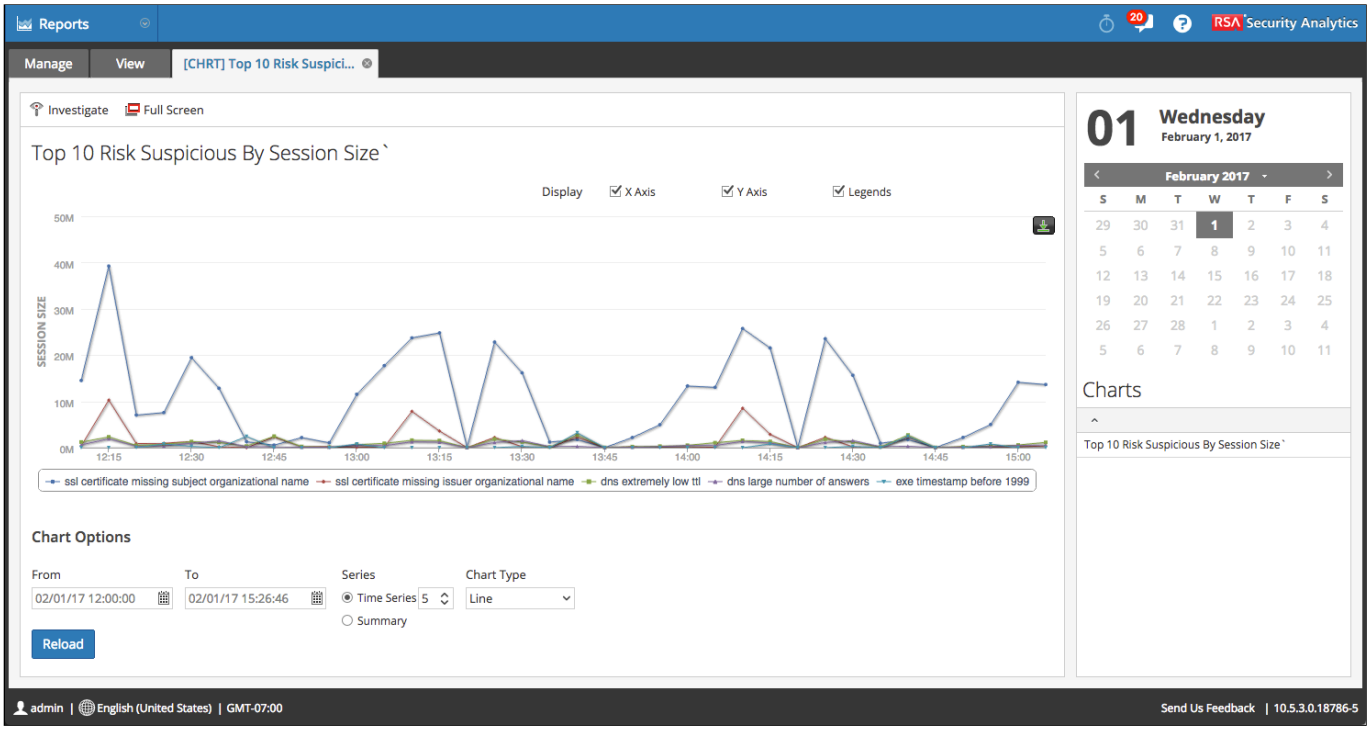


Viewing a Chart

Navigate to **Reports->View** and click the **Chart** tab.

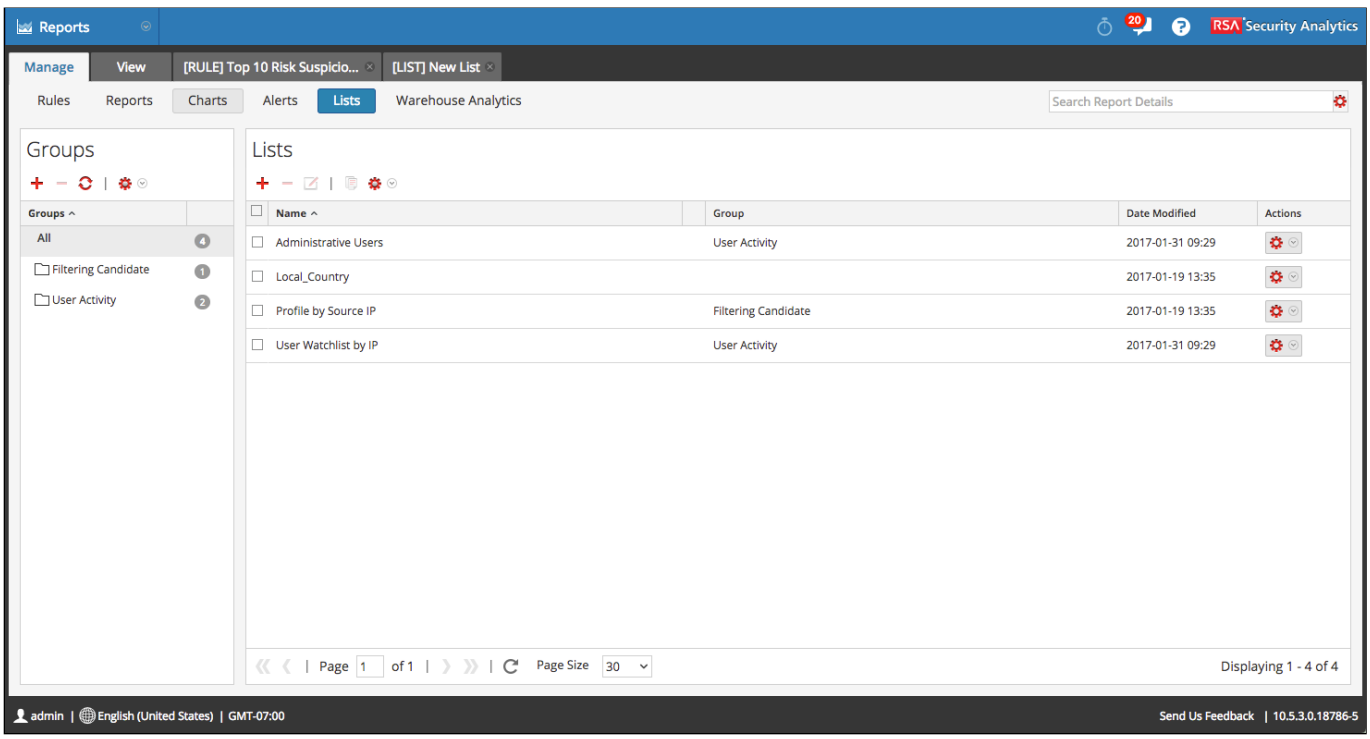


Double-click on a Chart to view it.



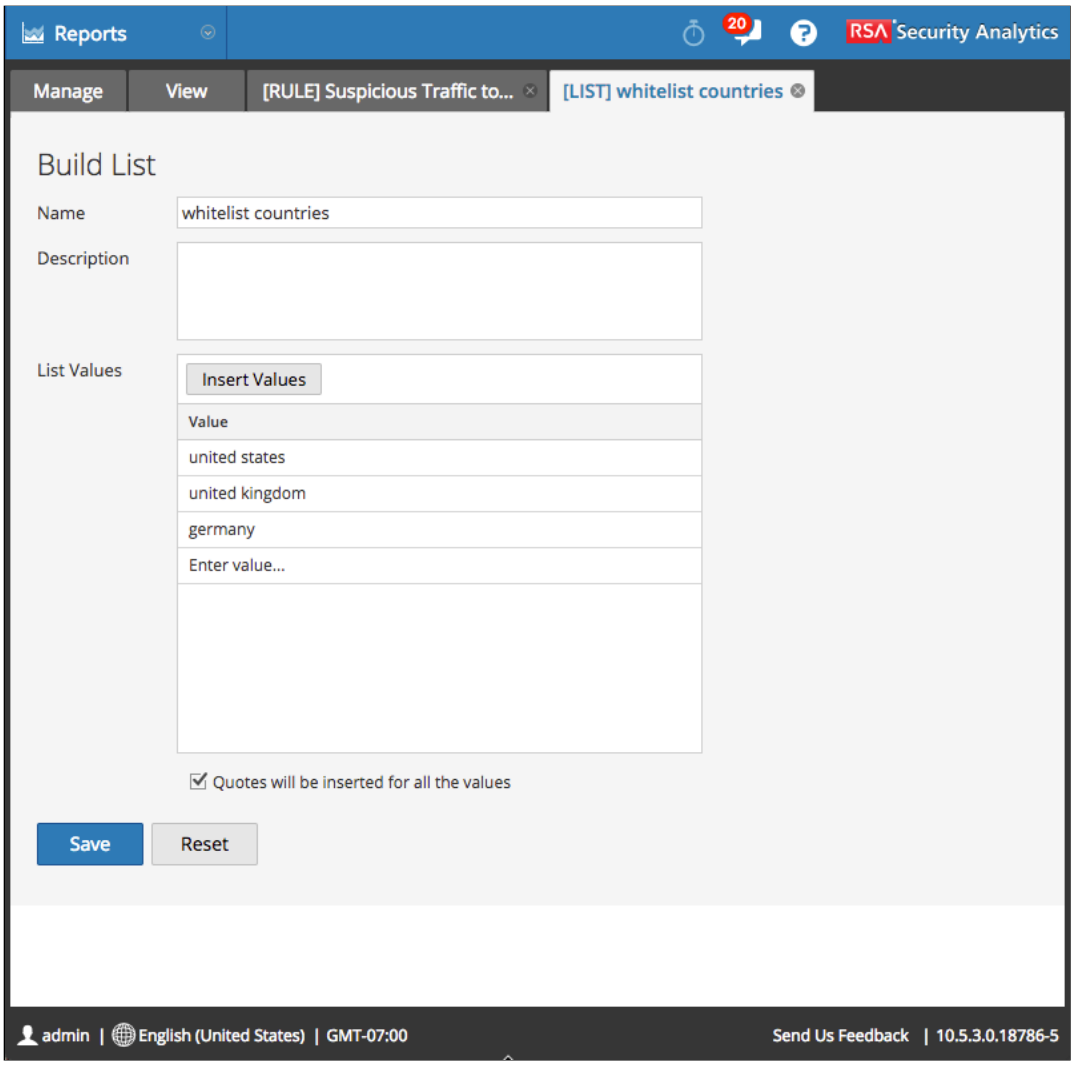
Reporting Lists

You've seen the concept of **Lists** mentioned in the lesson. A list is essentially an array of values which can be referenced in a rule's **WHERE** or **THEN** clauses. Lists are accessed from the Lists tab of **Reports->Manage**.



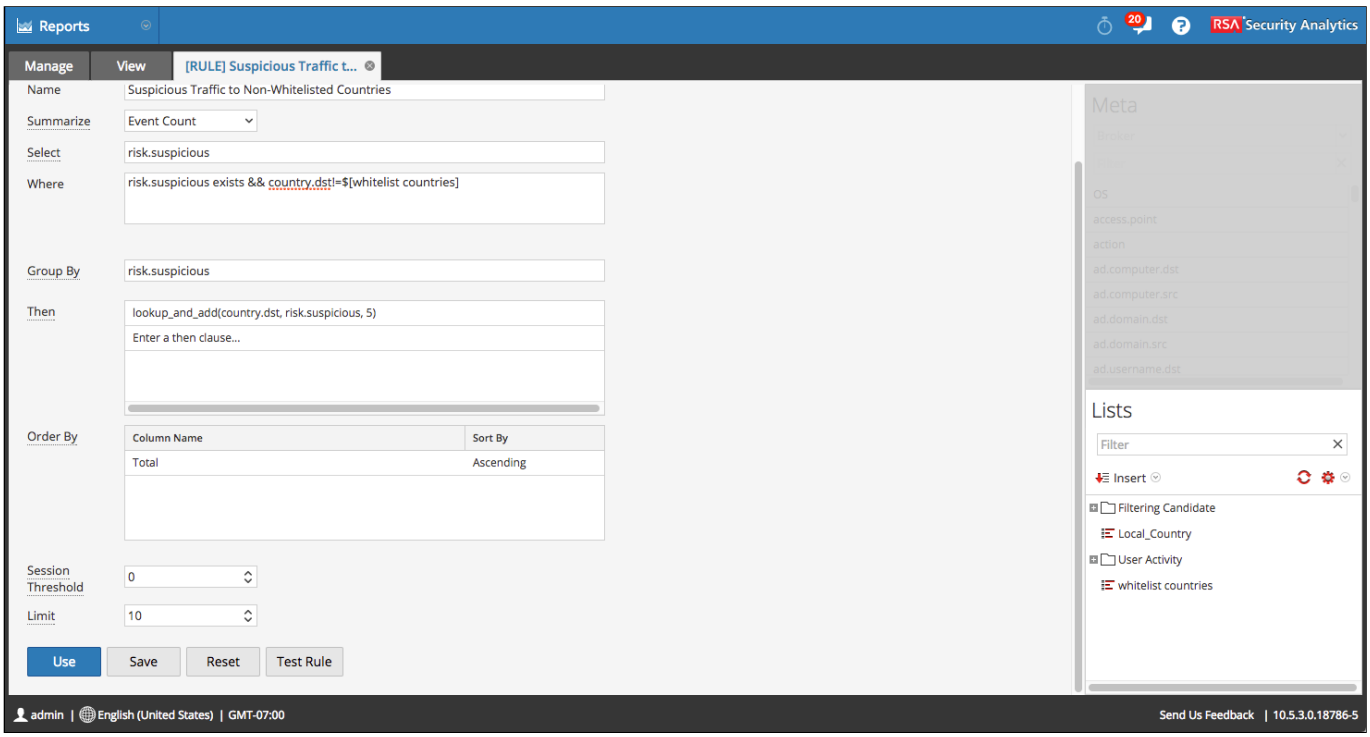
Let's create a list called "whitelist countries". Simply:

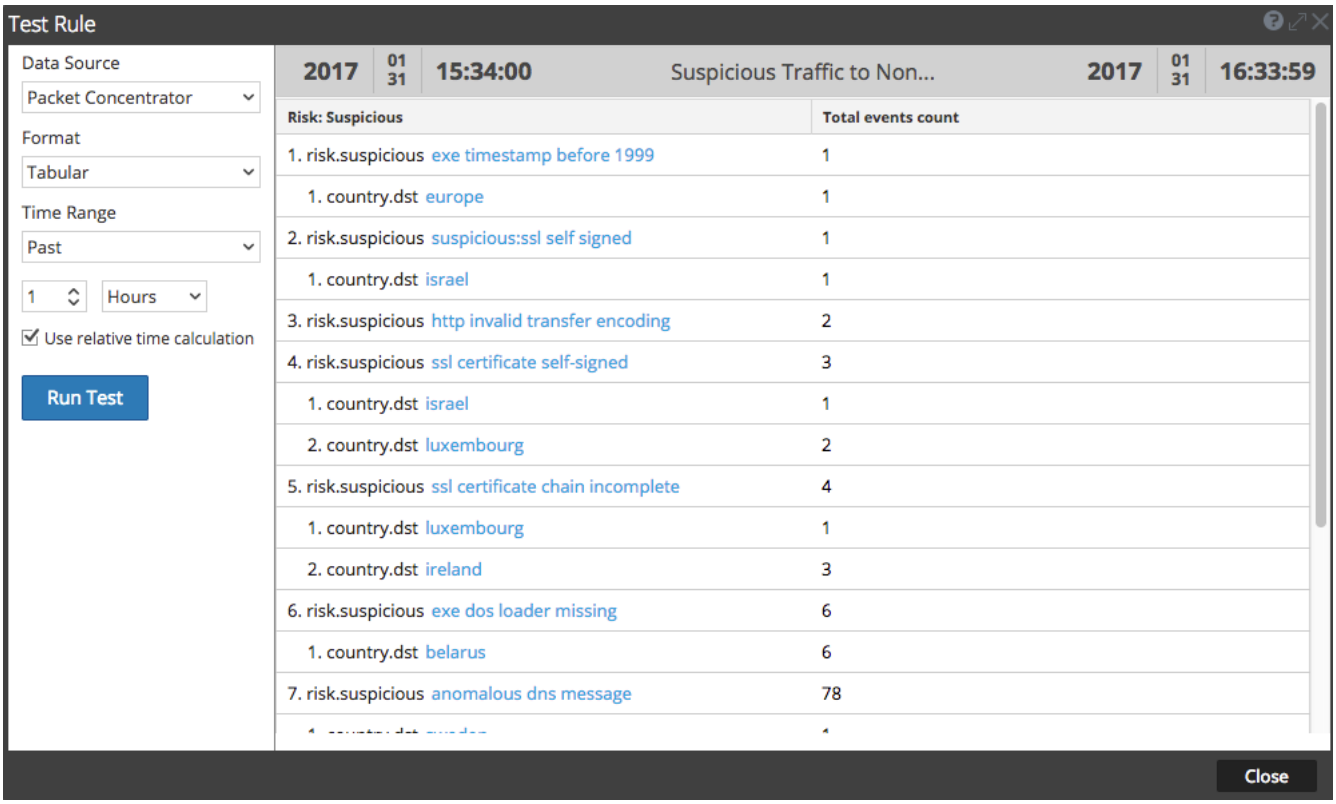
1. Use the + button to create the list.
2. Enter a Name.
3. Use the **Insert Values** button to add new values. Add the values "united states", "united kingdom", and "germany".
4. Tick the box for "**Quotes will be inserted for all the values**". By doing this, when referencing this list in a rule, quotes will automatically be added around each list value, so that the query doesn't break. If you don't do this, your Concentrator logs will show a message like "**Jan 31 23:12:24 concentrator NwConcentrator[1739]: [RulesEngine] [warning] Rule '' in deprecated format. Please make sure all text values are surrounded with quotes.**"
5. Then click **Save.**



Using Lists in the WHERE Clause

Using the above list, in the **WHERE** clause of a reporting **Rule**, one can search for whether a value is a member of our "**whitelist countries**" list. Can reference our List named "whitelist countries". In the rule's **WHERE** clause, I can write "**country.dst!=$[/whitelist countries]**". The rule will return results if a rule has a **country.dst** value that is not **united states**, **united kingdom**, or **germany**.





If we look at the Concentrator logfile, we can see that the above rule is a series of queries, but the WHERE clause of the queries looks like:

```
Jan 31 23:28:33 concentrator NwConcentrator[1739]: [SDK-Values] [audit] User reporting (session 165931, 10.0.0.3:52134) has finished
values (channel 165941, queued 00:00:00, execute 00:00:00): id1=1 id2=208463188 size=10 fieldName=risk.suspicious where="
(time='2017-Jan-31 22:28:00'-'2017-Jan-31 23:27:59') && (risk.suspicious exists && country.dst='united states','united
kingdom','germany')" flags=sessions,sort-total,order-ascending threshold=0
```

We can see how the RE has expanded the $[list] syntax into " **country.dst='united states','united kingdom','germany'** "

Using Lists in the THEN Clause

Lists can also be used in the **THEN** clause of rules as well. For instance, in a filter_on rule, one filter using the values contained in a list, using the syntax '***$[/groupName/listName]***':

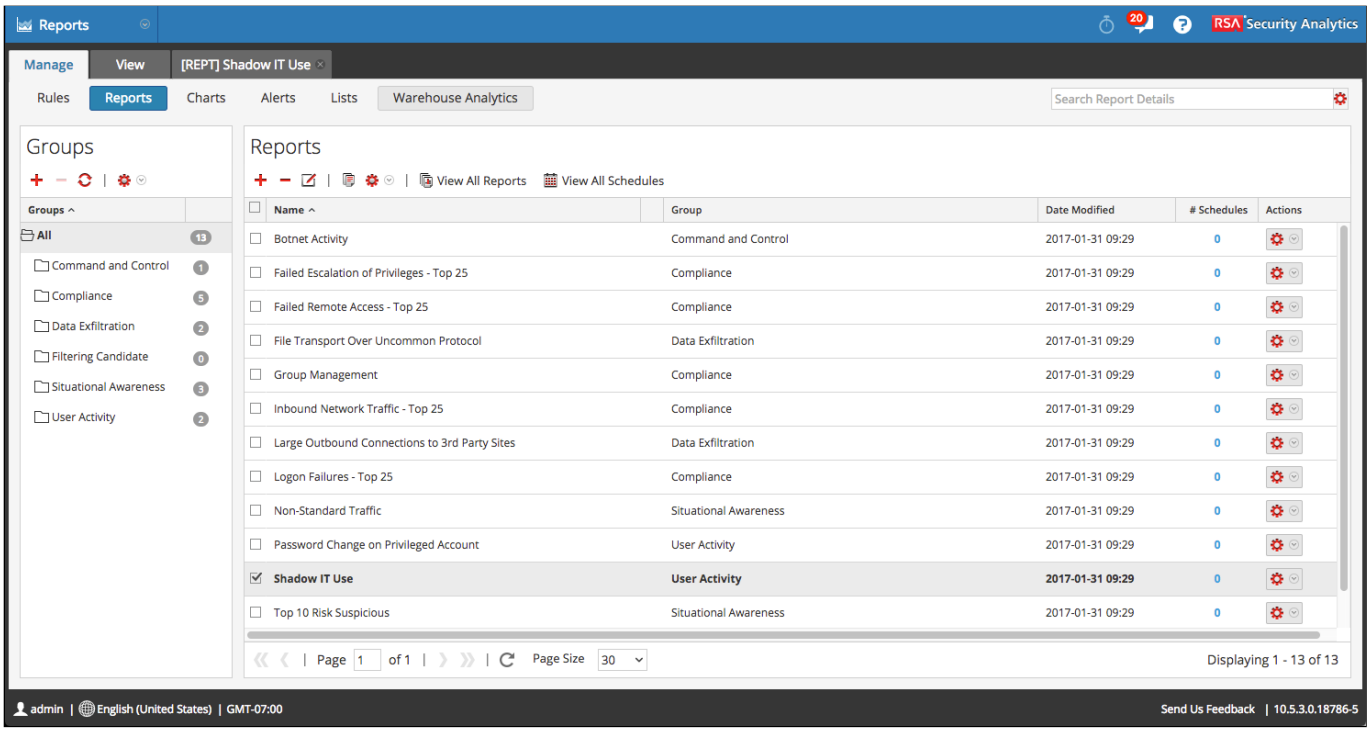
```
filter_out('$[/whitelist countries]','country.dst','false')
```

Creating Lists Dynamically

Lists can also be created dynamically, through scheduled reports. More on this shortly.

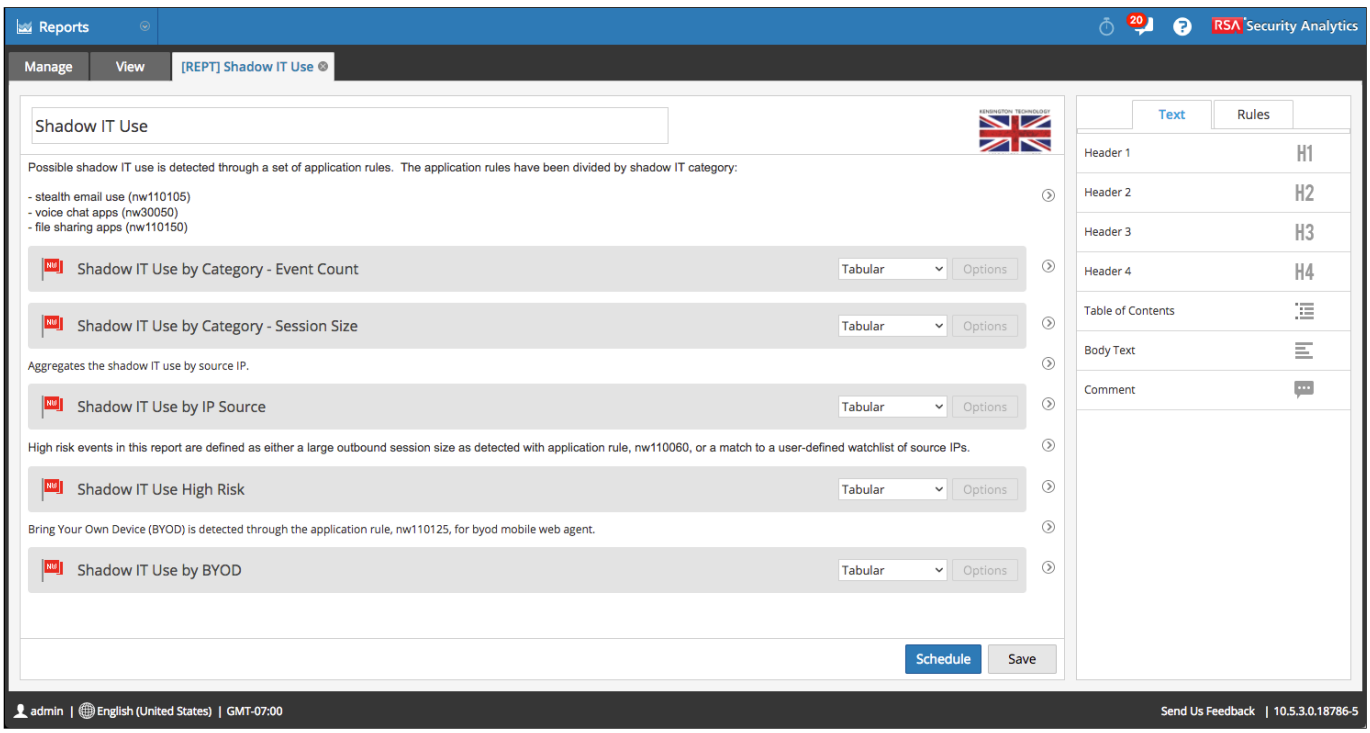
Reports

As you might have surmised, Reports are available from the Reports tab of **Reports->Manage.**



Let's open the "**Shadow IT Use**" Report, available from Live.

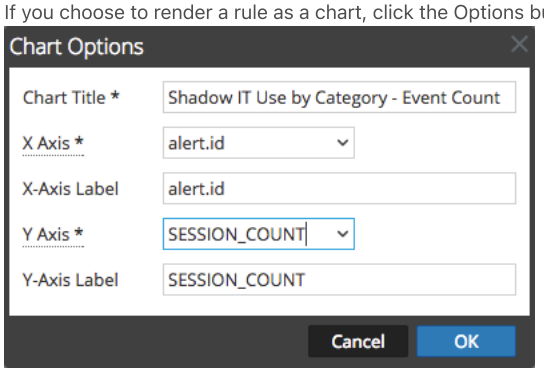
Building a Report



Looking at the the **Shadow IT Use Report**, we can see that building a report is incredibly simple.

- You simply drag elements and rules from the right-hand pane.
- Elements consist of Header 1, Header 2, Header 3, Header 4, Table of Contents, Body Text, and Comment
- When you drag a rule into a report, click the dropdown which says **Tabular** to choose whether the rule will be rendered as tabular data or as a chart

- If you choose to render a rule as a chart, click the Options button to configure the axis options:



- Once in the report, drag an element up and down to change the order.
- To edit a text element in a report, double-click on it.
- To delete an element, click on the icon to the right of it and then click Remove.
- Make sure you hit **Save**

Saving a Report doesn't do anything until it is run.

Running a Report

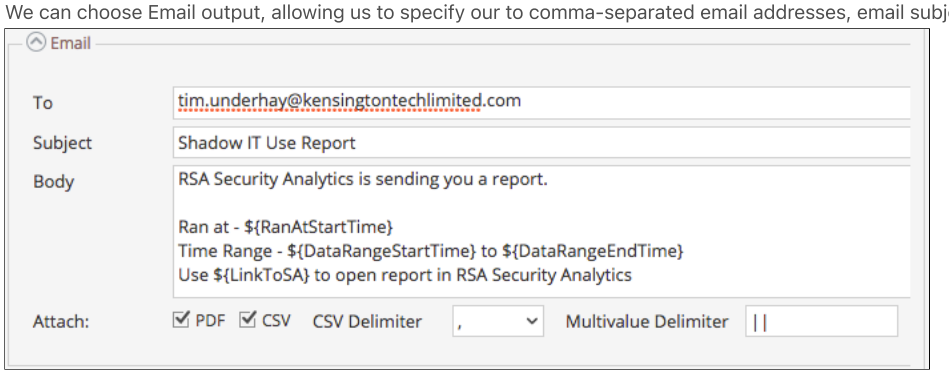
There are two ways to run a report: **Ad Hoc** and **Scheduled**.

Running an Ad Hoc Report

There's almost no difference between **Ad Hoc** and **Scheduled Reports**. Ad Hoc effectively means "this one time". You run them in the exact same manner. Look at your **Reports** in **Reports->Manage**. Next to the **Shadow IT Use** Report, click on the icon, and then **Schedule Report**.

Now we're looking at a **Schedule Report** tab. We need provide:

- **A Schedule Name**
- **A Data Source**
- **Time Zone** - Since the reader of the report could potentially be anywhere in the world, we must choose what time zone the Report's values will be output in.
- **When to Run** - Both **Now** or **Later** mean run the Report in an ad hoc manner.
- **Set Default** - Saves the timezone setting for future new Report schedules
- **On** - What time window on the NWDB should the report be run against?
- **Output Actions**:
  - We can choose Email output, allowing us to specify our to comma-separated email addresses, email subject, and email body



Here's a sample report email:

Reports | RSA Security Analytics

Manage | View | [REPT] New Schedule

Schedule Report

Enable ☑

Report Name: Shadow IT Use

Schedule Name: Ad Hoc

NetWitness DB: Broker

Time Zone: UTC (GMT+00:00) ☐ Set Default

Run: Now

On: Past 2 Hours ☐ Use relative time calculation

Variables: No variables defined

Output Actions

Email

To: Use , To Separate The Email IDs

Subject:

Body:
RSA Security Analytics is sending you a report.

Ran at - ${RanAtStartTime}
Time Range - ${DataRangeStartTime} to ${DataRangeEndTime}
Use ${LinkToSA} to open report in RSA Security Analytics

Attach: ☑ PDF ☐ CSV CSV Delimiter , Multivalue Delimiter | |

Other Options

Output	Notification Servers	Send as PDF	Send as CSV

No parameters to edit.

Dynamic List

List Name

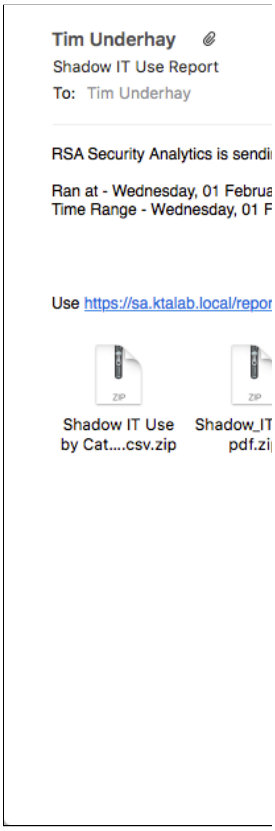
No list is defined

Logo

Change Logo

Schedule | Reset | Configure

admin | English (United States) | GMT-07:00 Send Us Feedback | 10.5.3.0.18786-5



- We can choose to attach the report as a PDF and/or a CSV document
- Other Options:
  - We can use SFTP, URL, and Network Share output, as defined in the Reporting Engine configuration.
- **Dynamic List** - We can output data from the report into an RE List. More on this shortly
- **Custom Logos** - Click Change Logo to switch to a different report logo that you have previously defined in your Reporting Engine config.

Dynamic Lists

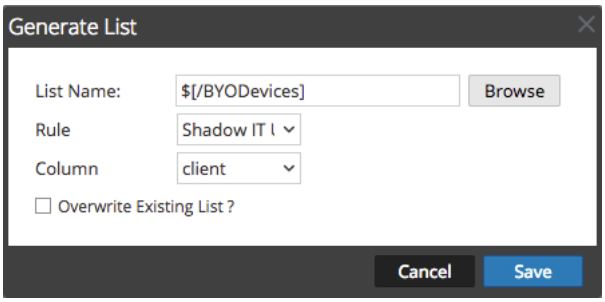
By using the **Dynamic List** section in the **Schedule Report** tab, we can create or add to Reporting **Lists**.

Dynamic List

List Name

No list is defined

Click the + button to create a **Dynamic List** configuration.

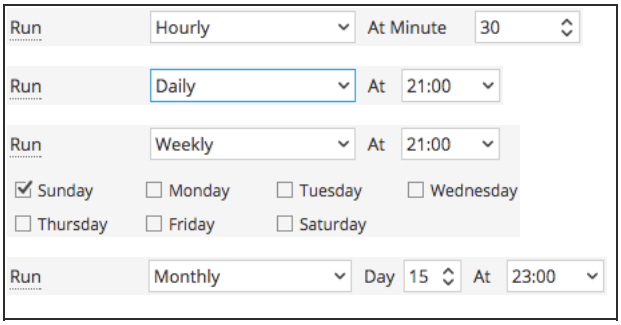


The options are:

- **List Name** - Choose an existing **List** to output data to.
- **Rule Name** - Choose the Rule from the underlying Report to pipe values from
- **Column** - Choose the column in the selected Rule to pipe values from. These are the values which will appear in the List.
- **Overwrite Existing List** - Will cause all values in the existing list to be deleted when the report runs.

Scheduling Reports

To schedule a report, change the Run option to either **Hourly, Daily, Weekly,** or **Monthly,** and choose a time.



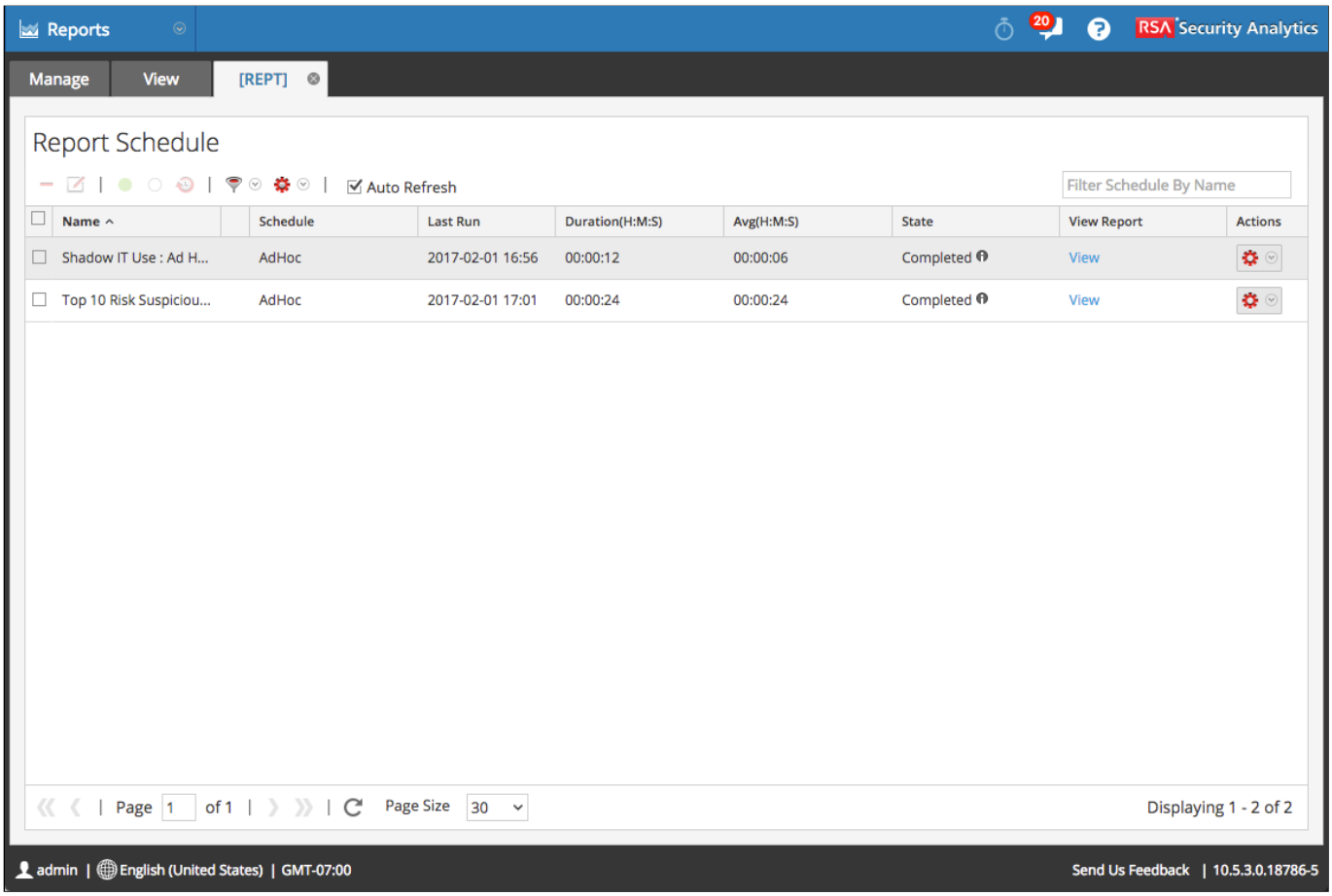
Top Tip: If it can be helped, try to schedule your heavy-duty reports (those run with expensive queries on a long time window) for a time at low system usage, to prevent excess strain on the system with performance penalties.

Committing a Report

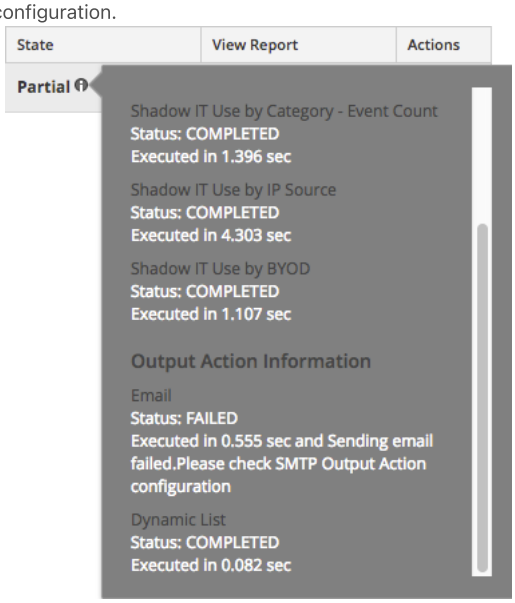
Click **Schedule** to commit the report to run at its scheduled time.

Viewing and Modifying All Report Schedules

From the **Reports->Manage** page under the **Reports** tab, click the **View All Schedules** button to view all scheduled reports.

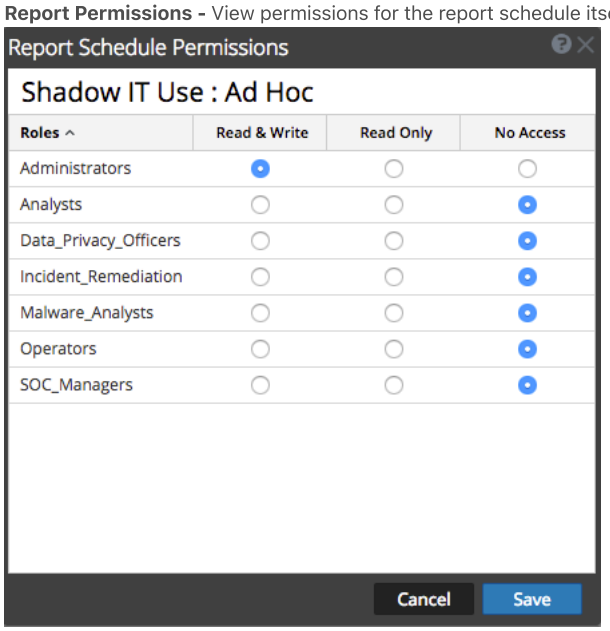


If there is a problem running the report, the state of the job will be **Partial**. Click on the button to see execution information, including what may have gone wrong. In this case to our right, we can see that there was a problem with the RE's SMTP configuration.

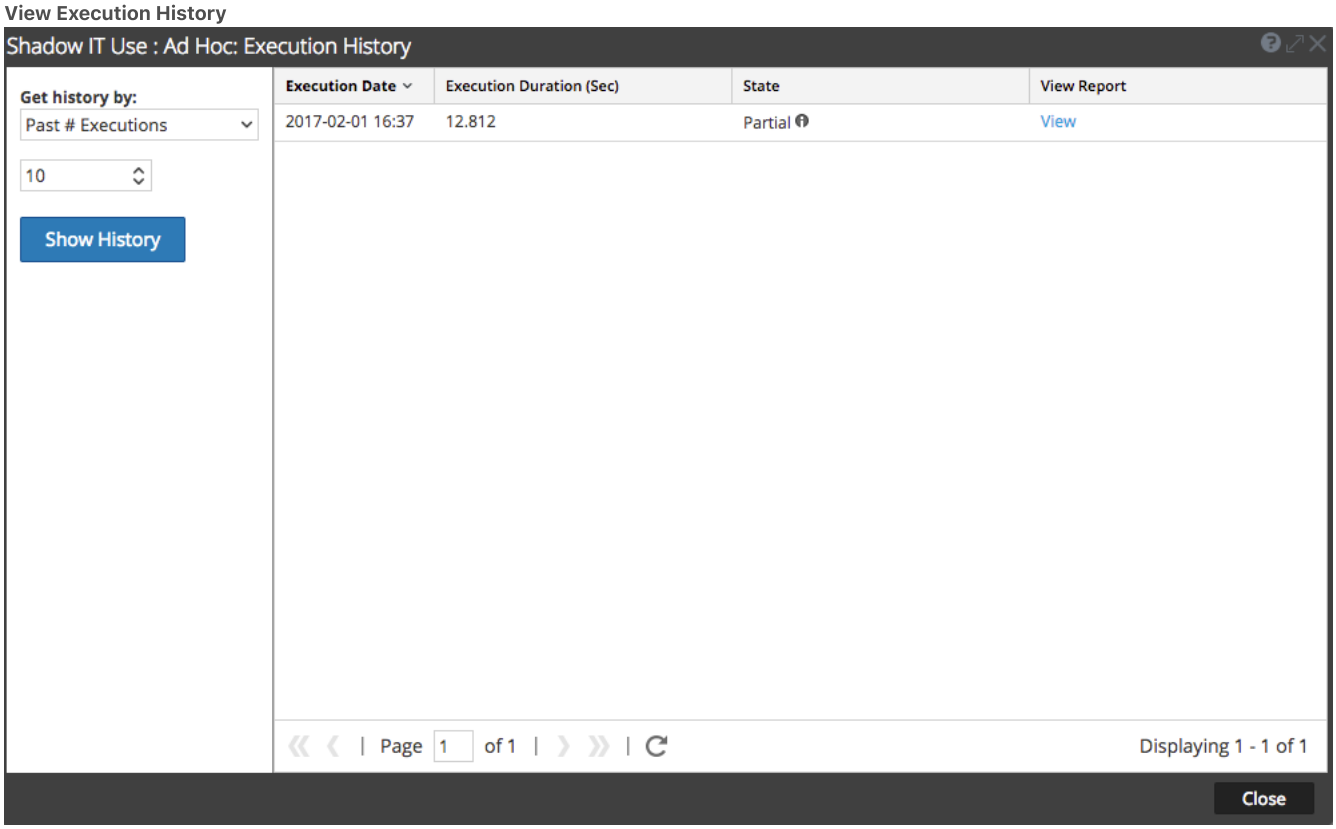


Using the button, we can:

- **Disable the schedule**
- **Start the job (now)**
- **Stop the job**
- **Delete the Schedule**
- **Edit the Schedule** - Takes us back to the Schedule Report tab.
- **Report Permissions -** View permissions for the report schedule itself

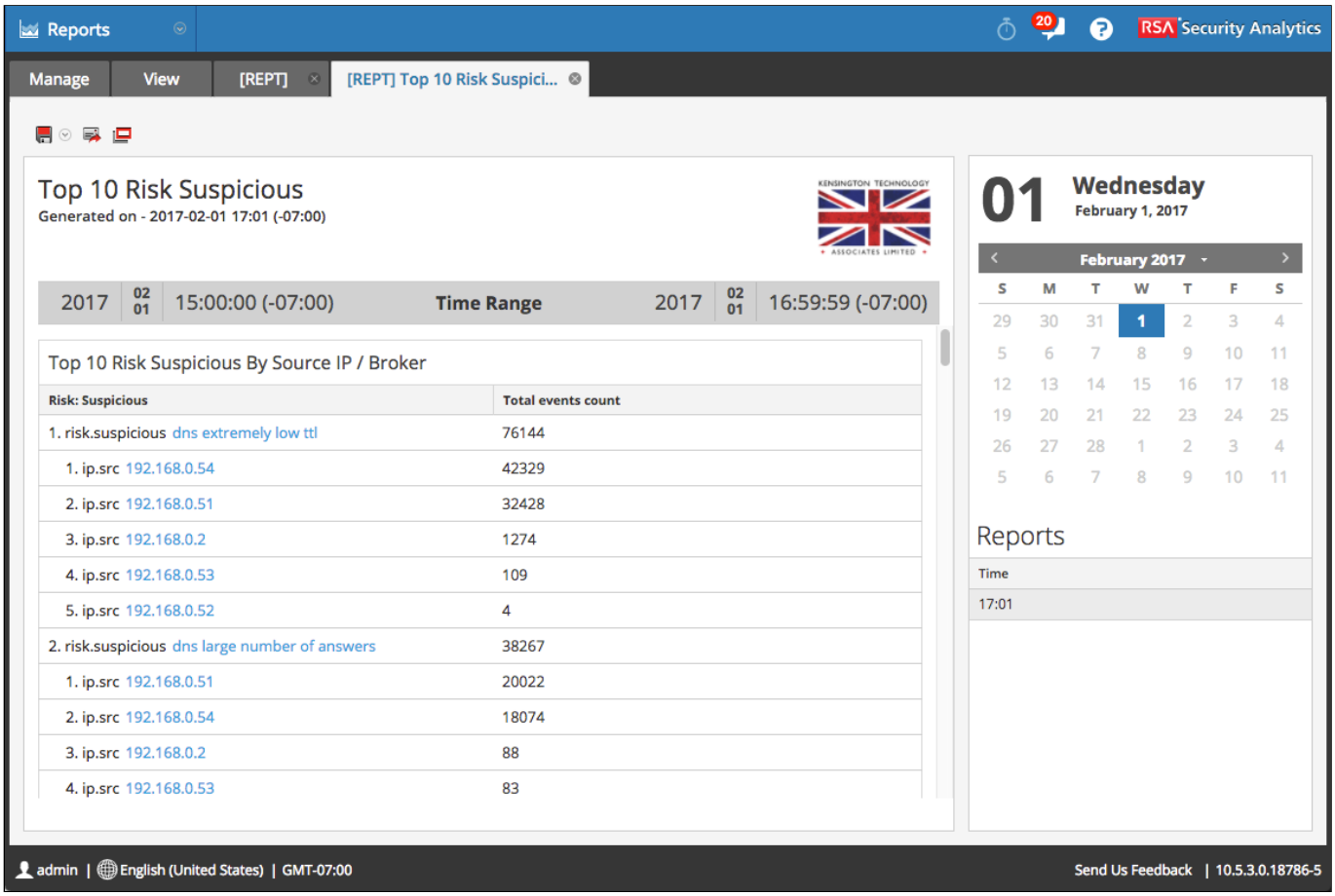


- View Execution History



Viewing a Report - Method 1

By clicking the **View** button on the report from the **View All Schedules** tab, we can see the output of a Report.

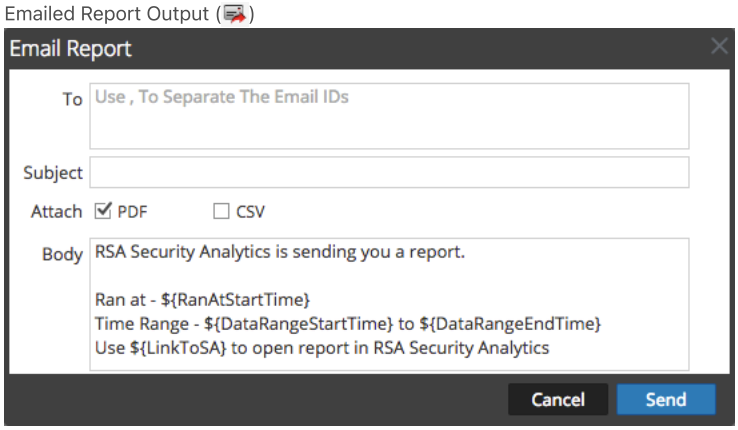


From here, we can use the toolbar to:

- Save as a CSV (

)

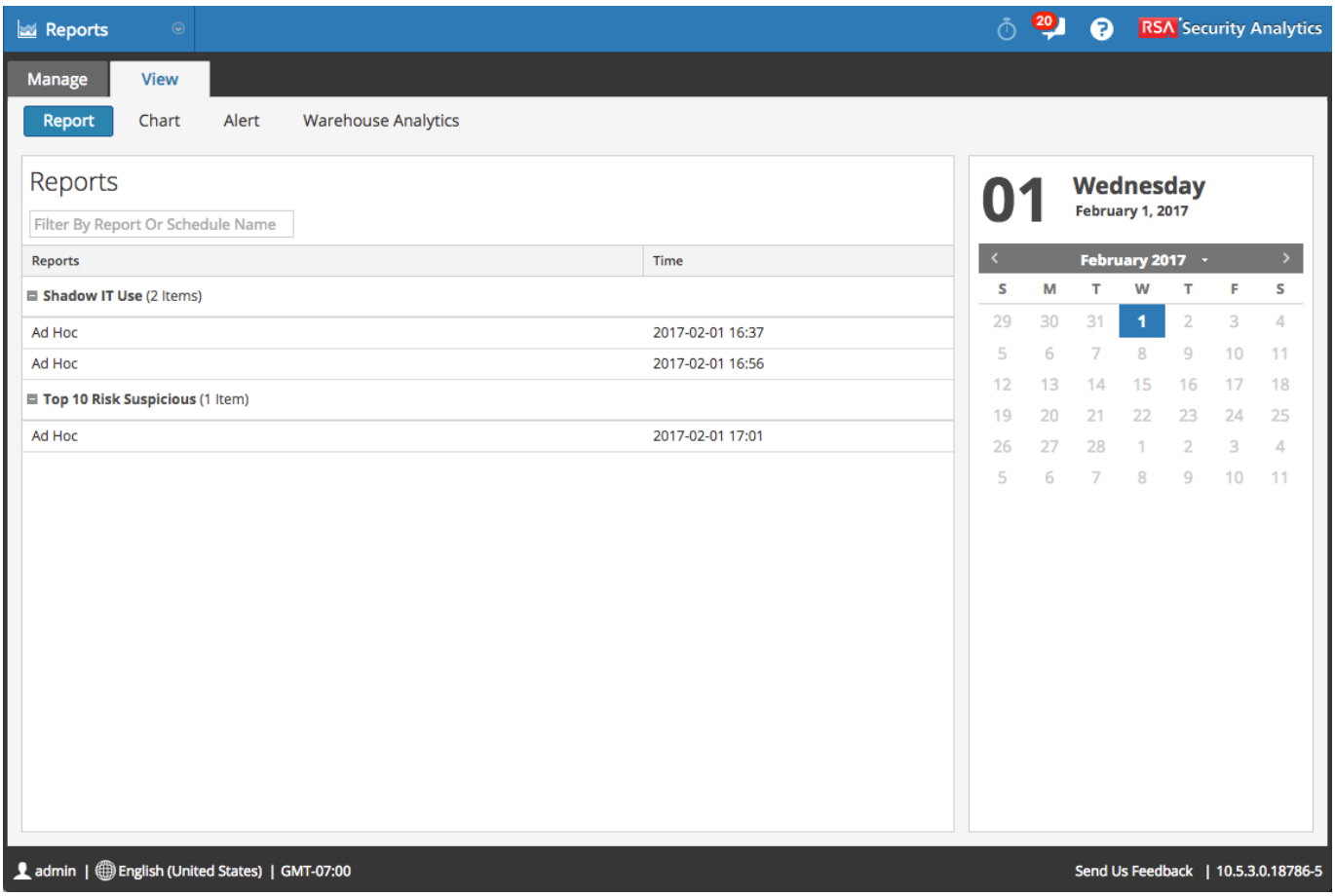
- Save as a PDF (PDF)
- Emailed Report Output ( )



- View Full Screen ( )

Viewing a Report - Method 2

We can also use the Report tab under **Reports->View** to see completed reports. Double-click on one to open it. It by default shows the reports from today.



You've doubtless noticed the calendar to the right-hand side of the page. By choosing a different date, you will see the reports for that date.

From the **Reports** tab under **Reports->Manage**, you can also click the View All Reports button to access the **Reports View** screen.

Exercise - Build a Report

It's time for you to put your knowledge to work and build a report. Build a report and any components it may need according to these specifications:

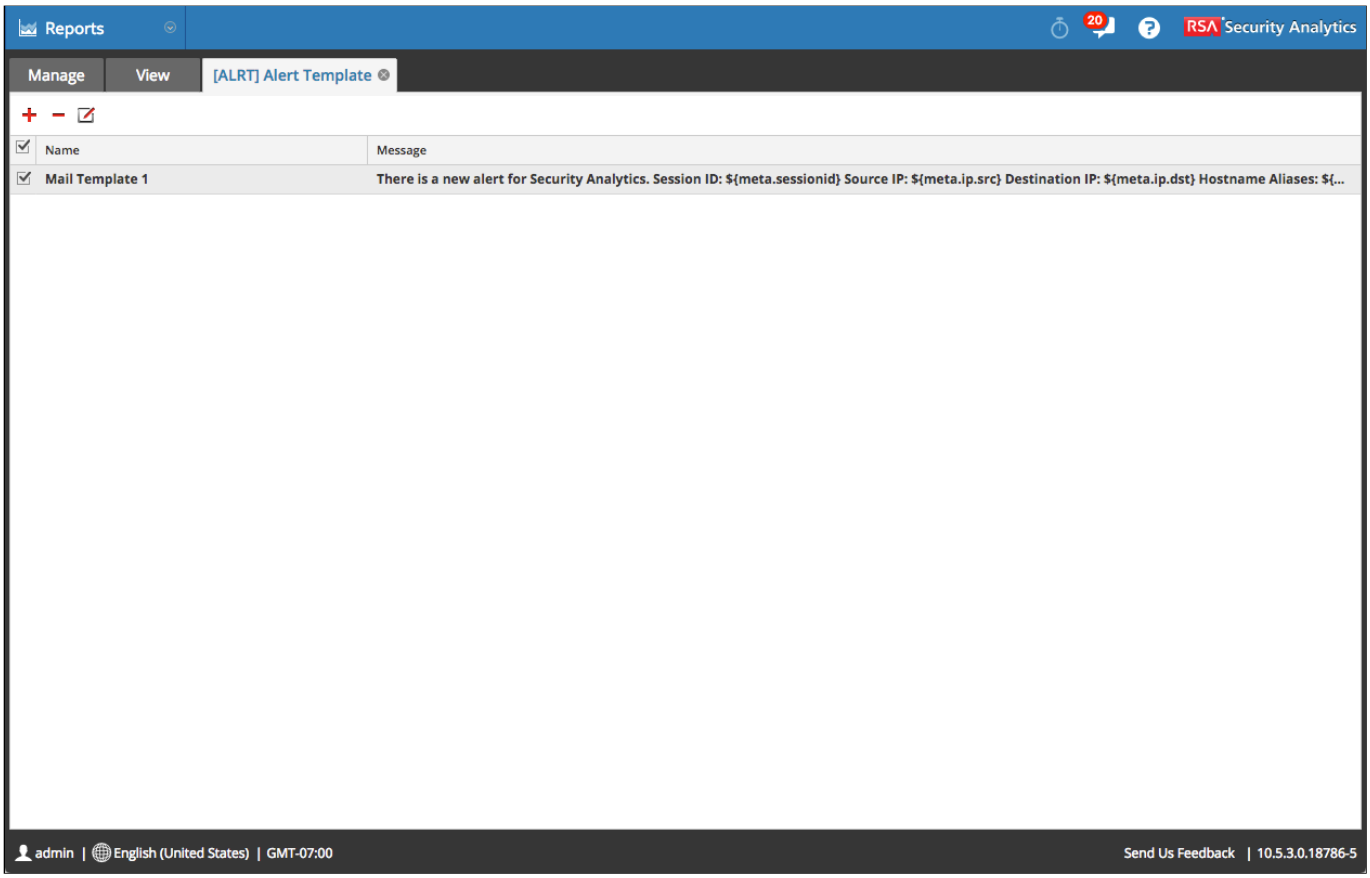
1. It will cover two topics
2. It should inform the report viewer about DNS anomalies, as indicated by the **risk.suspicious** meta key.
3. It should include source IP's and destination IP's
4. It should include a graph of the top service types for traffic matching this profile
5. It should also include text describing the rule.
6. It should include a table of contents

Alerts

The Reporting Engine can be used for forensics and detection purposes by employing the concept of **Alerts**. Alerts are quite simple: using an NWDB rule, a query is run on an NWDB data source every minute, on the last minute of data. Query results can then be sent to notification destinations as defined in the RE configuration, and also to Incident Management, if forwarding to it has been so enabled in the RE configuration.

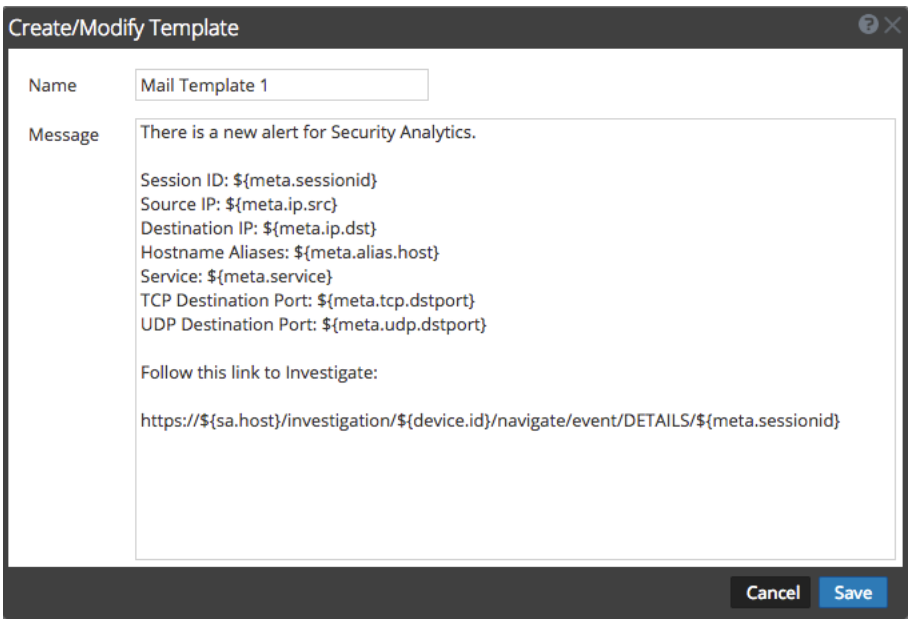
Alert Templates

Before we create an alert, we may want to have a look at alert templates. These are user-definable templates which can later be selected as the body of an alert. Unfortunately, there are no out-of-the-box templates to serve as an example. To access **Alert Templates**, from the **Alerts** tab of **Reports->Manage**, click the Template button.



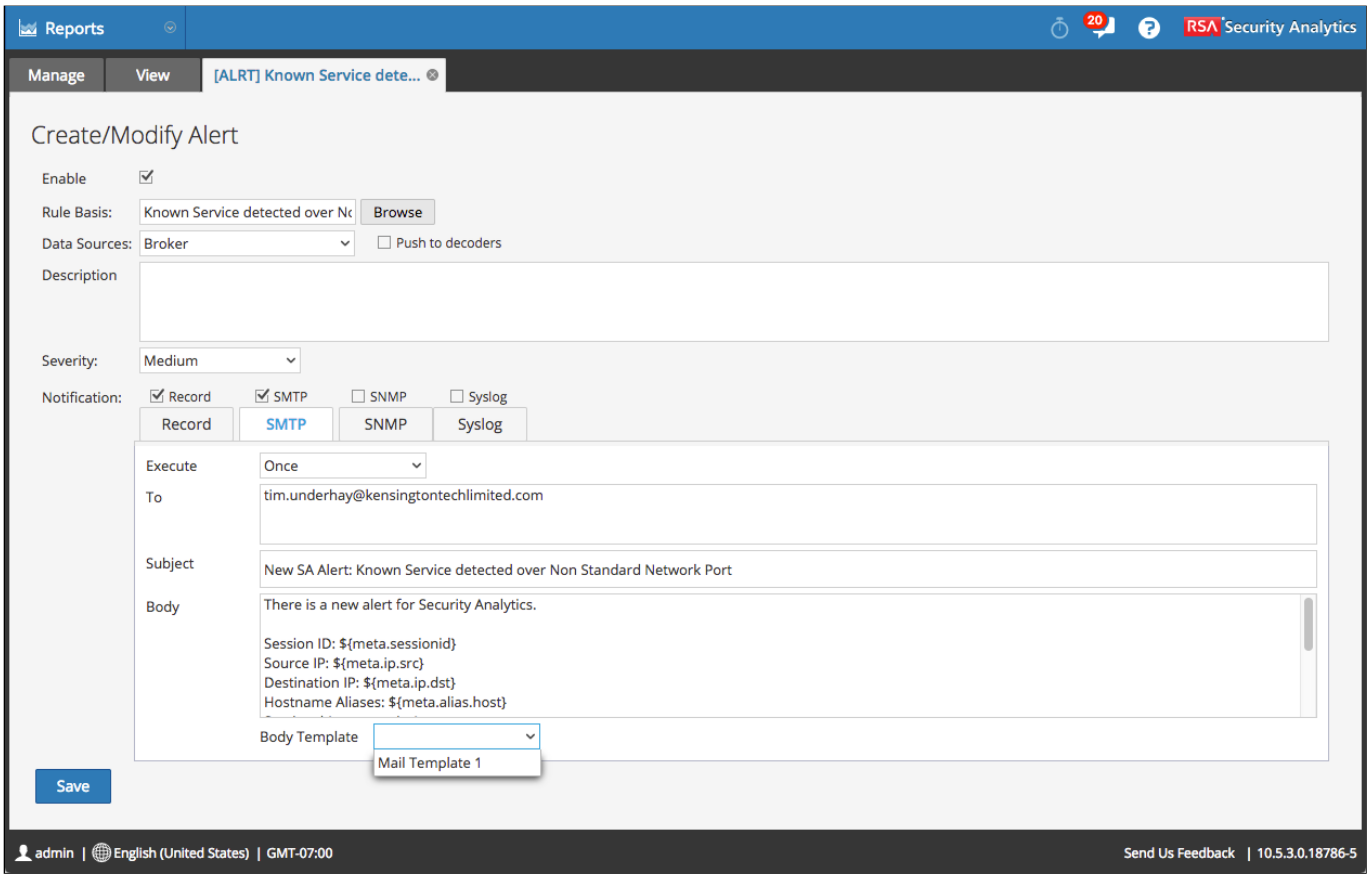
Click the + button and define your template. Click **Create**.

Top Tip: From an alert or template, event meta can be accessed using the syntax **${meta.*keyname*}.** E.g. " **${meta.ip.src}** "



Creating an Alert

To create an alert, navigate to **Reports->Manage->Alerts** and click the + button.



These are the fields to be configured:

- **Rule Basis** - Choose a RE Rule to apply to the alert. The alert's WHERE clause will be executed against the data source configured for the rule.
- **Data Sources** - Choose an NWDB data source to query.
- **Push to Decoders** - This tells the that instead of querying the data source for the underlying **WHERE** clause of the chosen rule, it will query the alert key for the **Rule Name**. This removes some of the query burden for processing a rule from the Concentrator, so that you only need to search for the App Rule name rather than all of the underlying conditions. In theory, this option should create a rule on any upstream Decoders connected to your data source. In practice, this doesn't work at all, and you should instead create your own App rule and RE rule to match it.
- **Description** - A description of the alert.
- **Severity** - **Information, Low, Medium, High,** or **Critical**.
- **Notification**: Tick the box for the output types you want for the alert, and configure the alert body and any parameters relevant to the output type. **Record** means to store the alert locally within the RE. Disabling **Record** output does **not** disable forwarding of the alert to Incident Management. Each output type tab has options specific to that type.

  Syslog Alerts require individual configurations (see below)

- **Execute** option: Choosing **Once** will trigger a single alert output for multiple matches on a rule. Choosing **Each event** would trigger a deluge of emails if a rule triggers many alerts, as every matched session returned would trigger an additional alert.
- **Body Template** - Choose an Alert **template** to import into the **Body** field.

Record

SMTP

SMTP

Execute: Once

To: tim.underhay@kensingtontechlimited.co.uk

Subject: New SA Alert: Unknown Service detected over Standard Network Port

Body:
There is a new alert for Security Analytics.

Session ID: ${meta.sessionid}
Source IP: ${meta.ip.src}
Destination IP: ${meta.ip.dst}
Hostname Aliases: ${meta.alias.host}

Body Template:

SNMP (trap)

SNMP

Execute: Once

Body: https://${sa.host}/investigation/${device.id}/navigate/event/DETAILS/${meta.sessionid}

Body Template:

Syslog

Syslog

Syslog Name	Execute	Severity	Facility
DEFAULT_SYSLOG	ONCE	WARNING	LOCAL7

New Syslog Configuration

Syslog Configs: DEFAULT_SYSLOG

Execute: Once

Facility: Local7 (23)

Severity: Warning

Body: https://${sa.host}/investigation/${device.id}/navigate/event/DETAILS/${meta.sessionid}

Body Template: Choose ...

Cancel Save

Viewing Alerts

Alerts can be viewed by clicking the View Alerts button in the **Reports->Manage->Alerts** tab, or by visiting **Reports->View->Alert**.

Hit the button to open an alert in **Investigation, or** just click on the alert **Name** link.

Reports | RSA Security Analytics

Manage | View

Report | Chart | Alert | Warehouse Analytics

Alert View

All Day | Max No Of Alerts 100

Investigate	Name	Number of hits	Detected	Message
	Known Service detected over Non...	**408**	**2017/02/02 11:04:45**	**There is a new alert for Security Analytics. Session ID: 9745800 Source I...**
	Unknown Service detected over Sta...	358	2017/02/02 11:04:02	There is a new alert for Security Analytics. Session ID: 9651087 Source IP: 19...
	Known Service detected over Non S...	592	2017/02/02 11:03:45	There is a new alert for Security Analytics. Session ID: 9732247 Source IP: 19...
	File Transport Over Uncommon Pro...	1	2017/02/02 11:03:29	There is a new alert for Security Analytics. Session ID: 9732227 Source IP: 19...
	Unknown Service detected over Sta...	413	2017/02/02 11:03:02	There is a new alert for Security Analytics. Session ID: 9635946 Source IP: 70....
	Known Service detected over Non S...	422	2017/02/02 11:02:45	There is a new alert for Security Analytics. Session ID: 9701853 Source IP: 19...
	File Transport Over Uncommon Pro...	1	2017/02/02 11:02:29	There is a new alert for Security Analytics. Session ID: 9732206 Source IP: 19...
	Unknown Service detected over Sta...	303	2017/02/02 11:02:02	There is a new alert for Security Analytics. Session ID: 9623401 Source IP: 18...
	Known Service detected over Non S...	50	2017/02/02 11:01:45	There is a new alert for Security Analytics. Session ID: 9700088 Source IP: 19...
	Unknown Service detected over Sta...	94	2017/02/02 11:01:02	There is a new alert for Security Analytics. Session ID: 9593154 Source IP: 19...
	Known Service detected over Non S...	14	2017/02/02 11:00:45	There is a new alert for Security Analytics. Session ID: 9691706 Source IP: 19...
	Unknown Service detected over Sta...	165	2017/02/02 11:00:02	There is a new alert for Security Analytics. Session ID: 9591421 Source IP: 19...
	Known Service detected over Non S...	169	2017/02/02 10:59:45	There is a new alert for Security Analytics. Session ID: 9680340 Source IP: 19...
	File Transport Over Uncommon Pro...	1	2017/02/02 10:51:29	
	Unknown Service detected over Sta...	86	2017/02/02 10:44:16	
	Known Service detected over Non S...	145	2017/02/02 10:43:58	
	Unknown Service detected over Sta...	253	2017/02/02 10:43:16	

Page 1 of 2 | Displaying 1 - 30 of 51

admin | English (United States) | GMT-07:00 | Send Us Feedback | 10.5.3.0.18786-5

Sample Alert Email

Tim Underhay Inbox - Kensington 10:28

New SA Alert: Unknown Service detected over Standard Network Port

To: Timothy Underhay

There is a new alert for Security Analytics.

Session ID: 9573142
Source IP: 192.168.0.51
Destination IP: 80.239.174.117
Hostname Aliases:
Service: 0
TCP Destination Port: 443
UDP Destination Port:

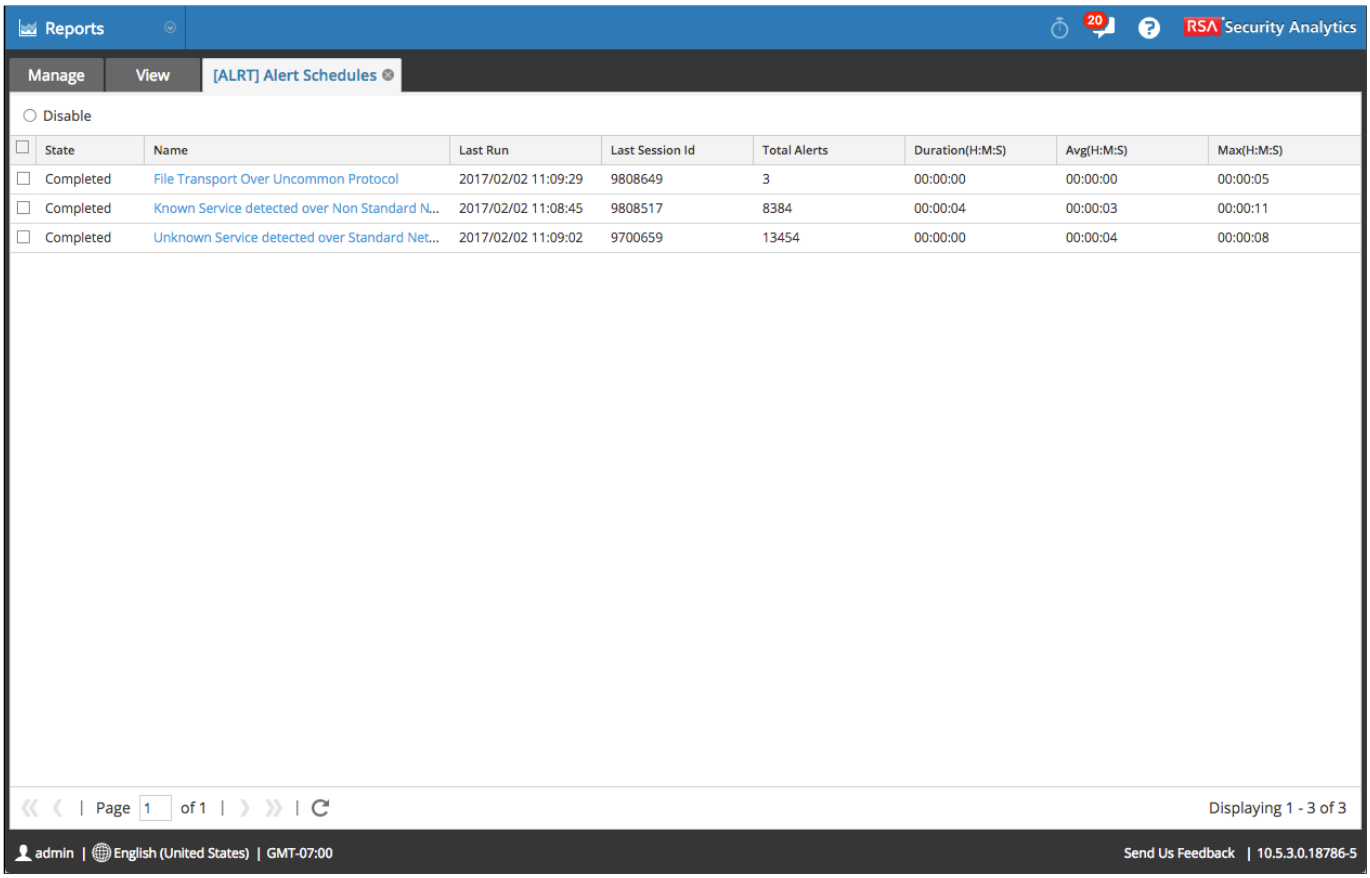
Follow this link to Investigate:

https://sa.ktalab.local/investigation/5/navigate/event/DETAILS/9573142

Viewing Alerts State

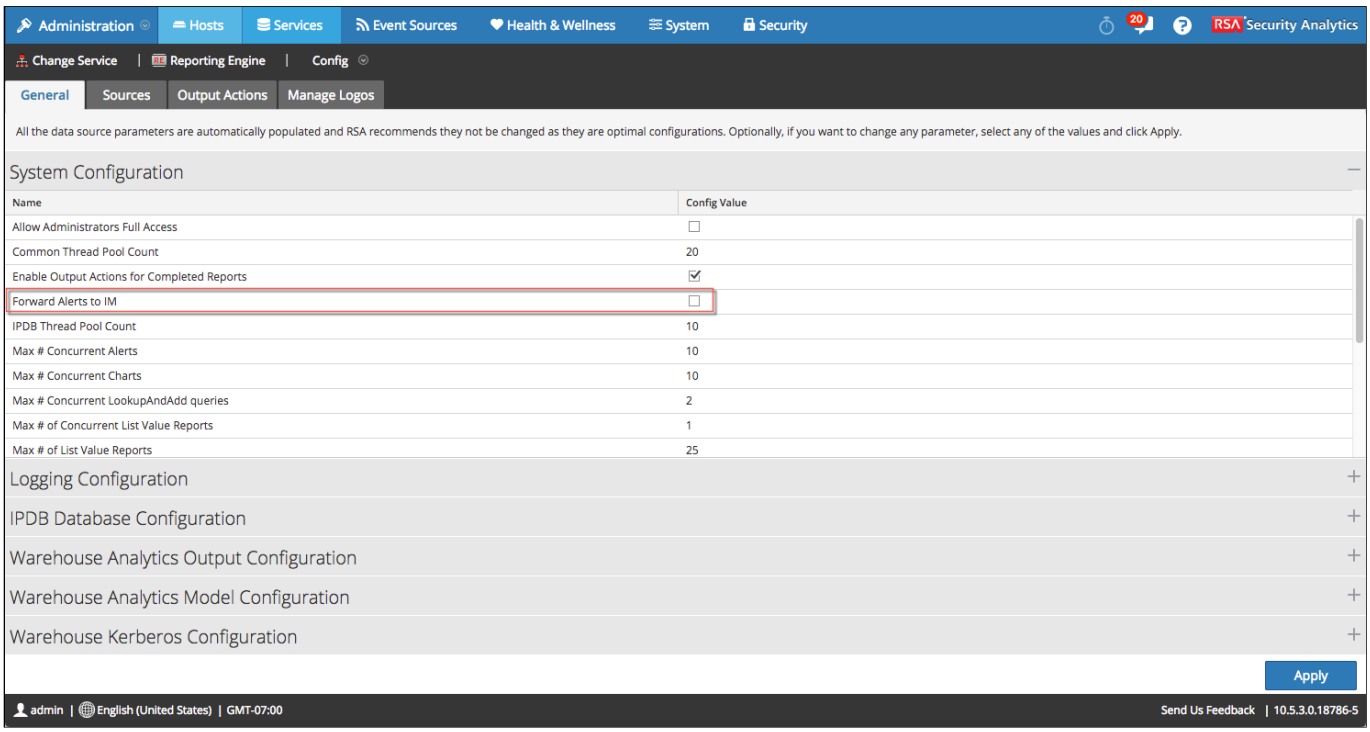
From the **Alerts** tab under **Reports->Manage**, click the View Schedule button. This will take you to the **Alert Schedules** page. 'Alert Schedules' is a bit of a misnomer, as their schedule cannot be modified - they always run every minute. Alerts can only be enabled or disabled.

From this page, you can see when an alert was last run, the last Session ID it matched, and the total number of alerts the rule has triggered.



Sending Alerts to Incident Management

If you'd like to use your RE alerts in Incident Management, the RE must first be configured to send its alerts there. Navigate to **Administration->Services->Reporting Engine->Config**, tick the box for "**Forward Alerts to IM**", and then hit **Apply**.



RE Security and Access Control

The Reporting Engine possesses its own internal security and role based access control (RBAC) mechanisms. Access to RE resources can be restricted to members of certain SA Security Roles. These are roles which are configured in **Administration->Security->Roles**.

RE Permissions can be set on:

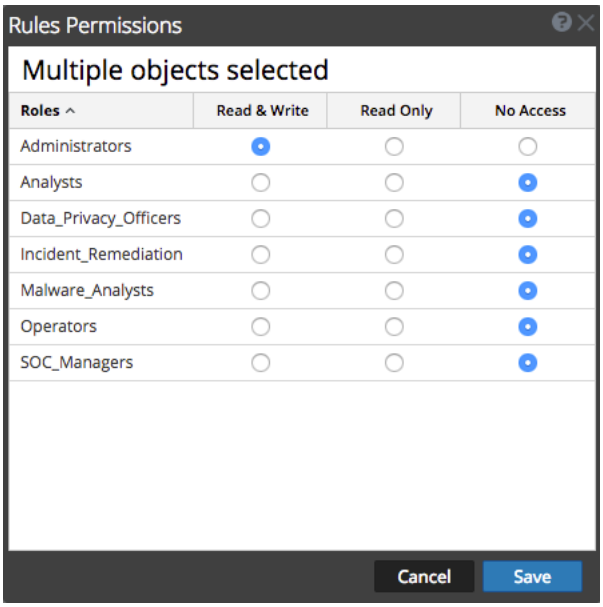
- **Rule Definitions**
- **Rule Groups**

- **Report Definitions**
- **Report Groups**
- **Report Schedules**
- **Chart Definitions**
- **Chart Groups**
- **Alerts**
- **Lists**
- **List Groups**
- **Data Sources**

The user experience is nearly identical across all of these types, so rather than covering them individually, we'll do so in a more generic manner, using **Rules** as an example.

Setting Permissions on Individual Rules

From the **Rules** (or **Reports**, or **Charts**, or **Alerts**, or **Lists**) tab of **Reports->Manage**, select one or more objects (i.e. rules), and click the button and click **Permissions**.



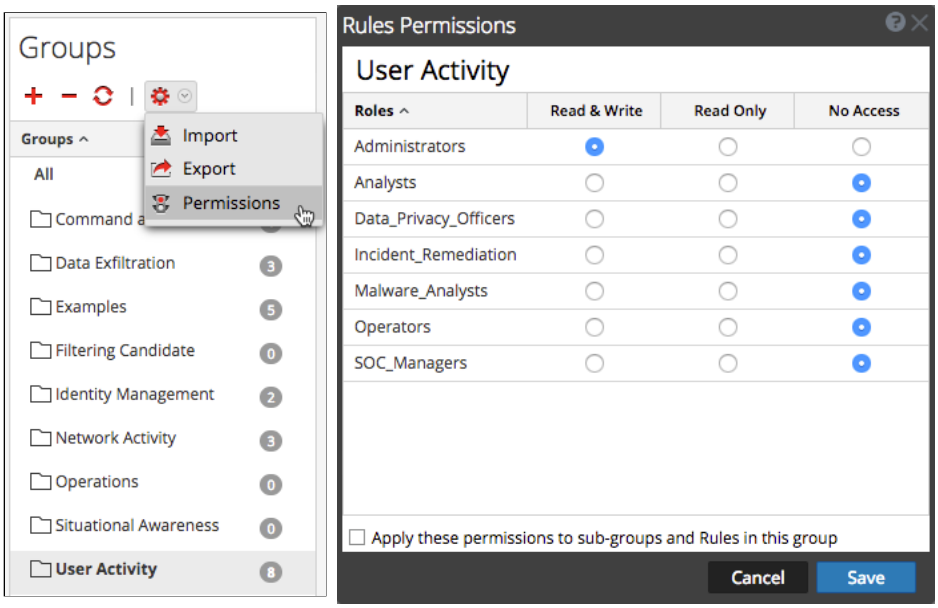
Use the radio buttons to choose the level of access permitted to each SA role. The available permissions are:

- Read & Write
- Read Only
- No Access

Click **Save**.

Setting Permissions on Rule Groups

From the **Rules** (or **Reports**, or **Charts**, or **Alerts**, or **Lists**) tab of **Reports->Manage**, select a Group, and click the button and click **Permissions**.



If you wish to apply the same permissions to sub-groups and rules, tick the box for **Apply these permission to sub-groups and Rules in this group**.

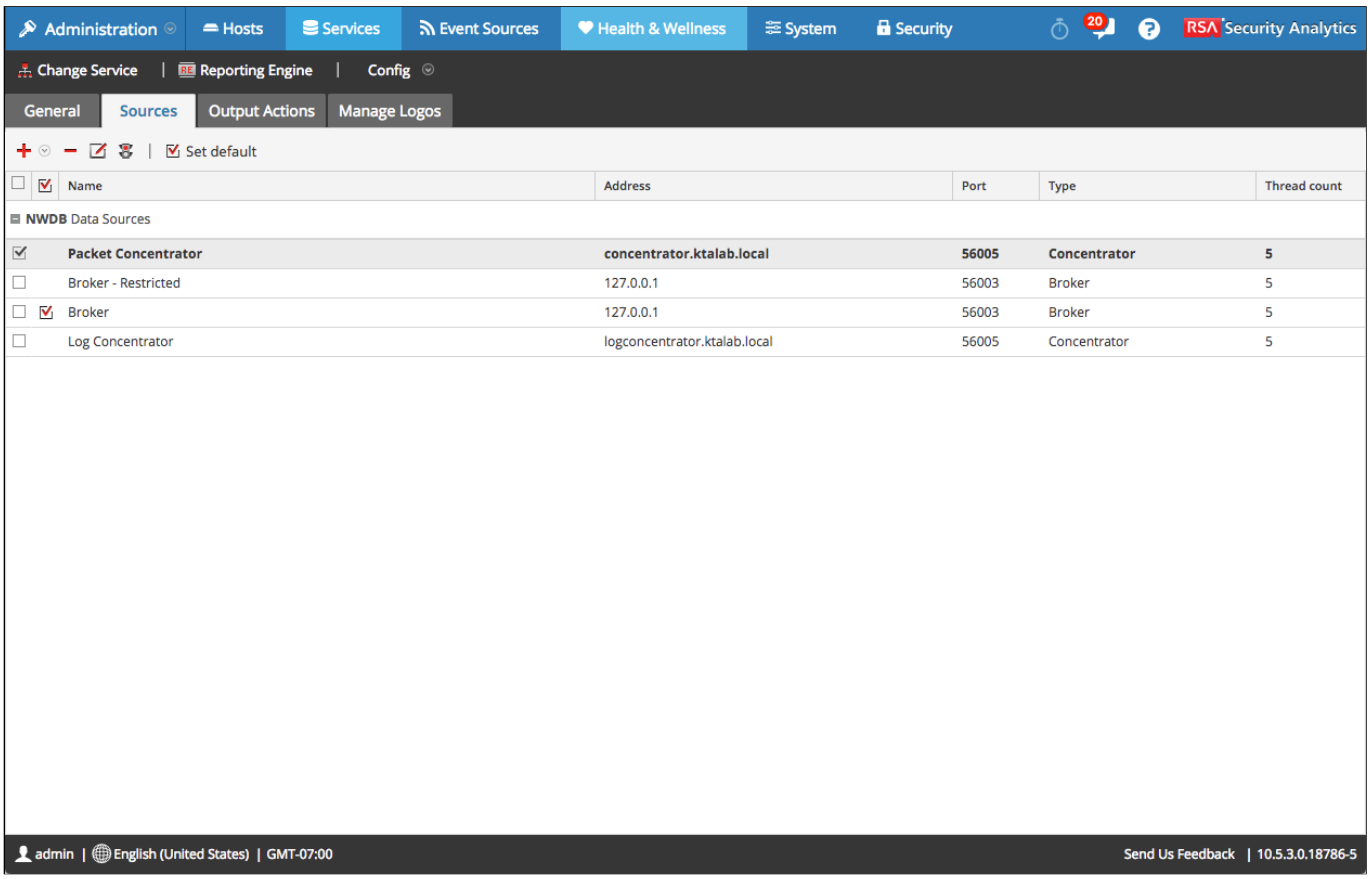
NWDB Query Security

The Reporting Engine uses a single service account for every NWDB data source (each data source can be a different account). All queries run against the data source will run with the SA Core permissions of that service account. If you wish to limit the Reporting Engine's access to meta on a particular data source, you can configure **system.roles** on the data source to restrict access to its meta keys. See the lesson on SA Security for more information.

Data Source Access Control

You can also restrict access to individual data sources by SA Security Role, making it possible to use SA Core's **system.roles** RBAC to have multiple permissions profiles for a single data source.

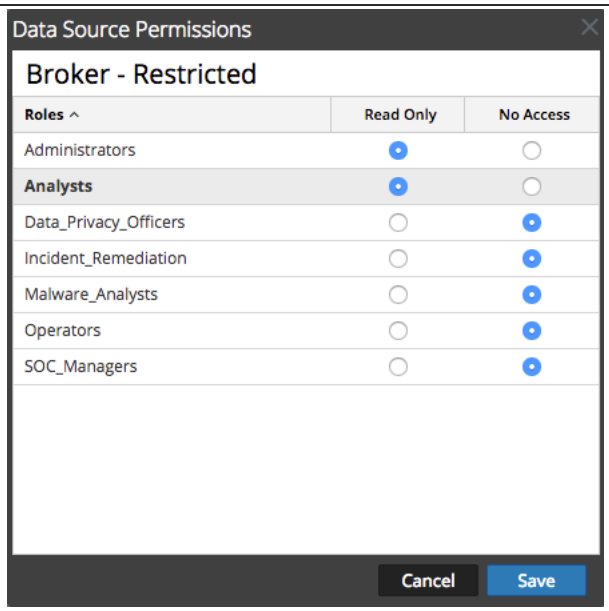
To do this, navigate to **Administration->Services->Reporting Engine->Config,** then click the **Sources** tab.



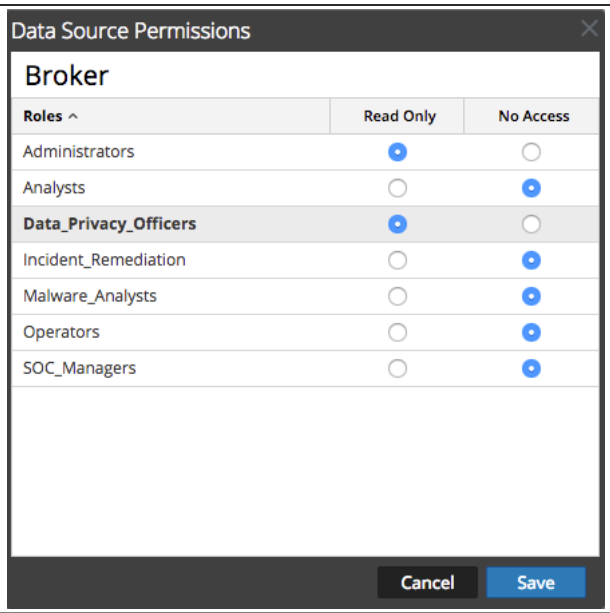
Notice that in the above screenshot, there are two different Broker data sources - one called **Broker**, and one called **Broker - Restricted**. The Broker - Restricted data source uses a different service account than Broker. This is a service account with restricted access to the meta data. Now you can grant access to the

Select a data source and click the button, and choose your permissions as you would for other RE components, except that the permissions available are:

- Read Only
- No Access



Analysts can only access this **Broker - Restricted** data source, which uses a
service account called **reportingRestricted**



Only **Administrators** and **Data Privacy Officers** can access this **Broker** data source,
which is the same physical Broker as Broker - Restricted.

Technical

How Does One Stop and Start the Reporting Engine?

```
# stop rsasoc_re
# start rsasoc_re
```

Where is RE Data Stored?

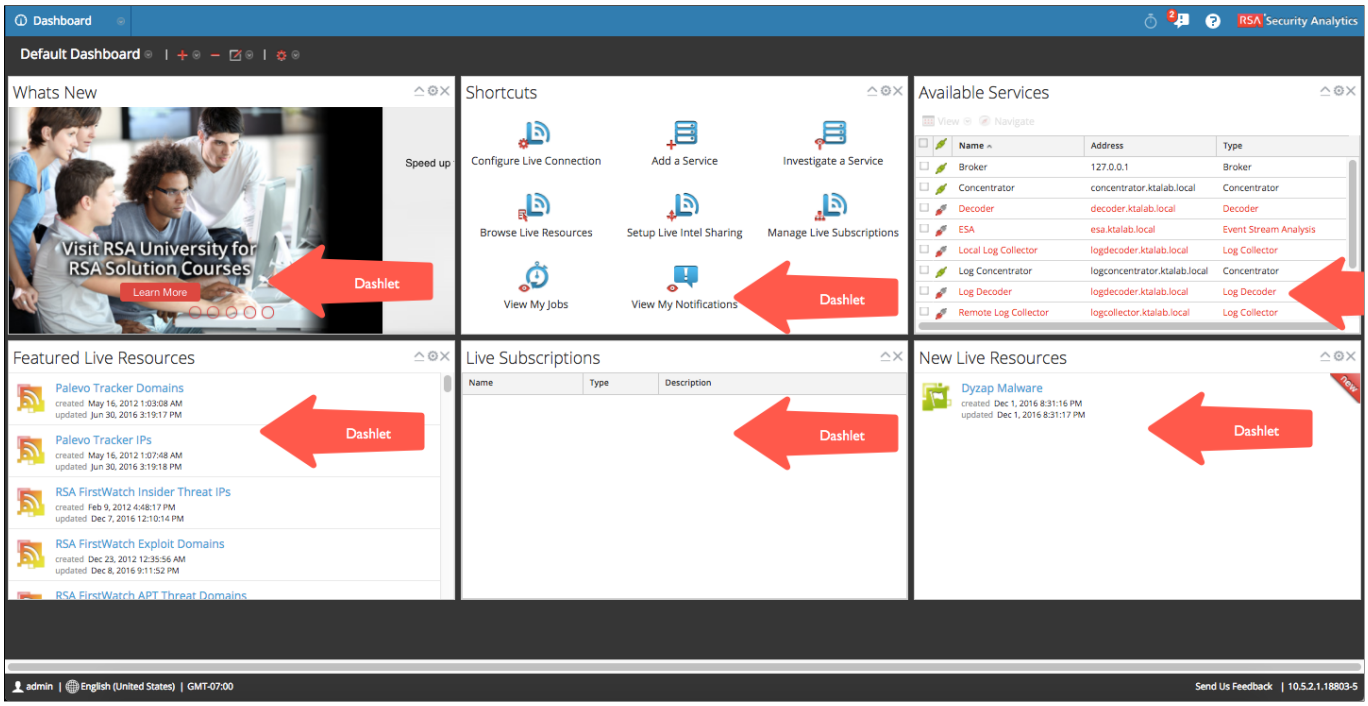
```
/home/rsasoc
```

Where is the RE Logfile Located?

```
/home/rsasoc/rsa/soc/reporting-engine/logs/reporting-engine.log
```

27. Dashboards

As SA's default landing page, the dashboard is the first thing you see when you log into SA. A dashboard is a window of one or more columns, which contain discreet "panes of glass", called dashlets, and they can be used for both administrative purposes and analysis. Dashboards are often overlooked, but when configured smartly, they can be a great launching pad for a security team.

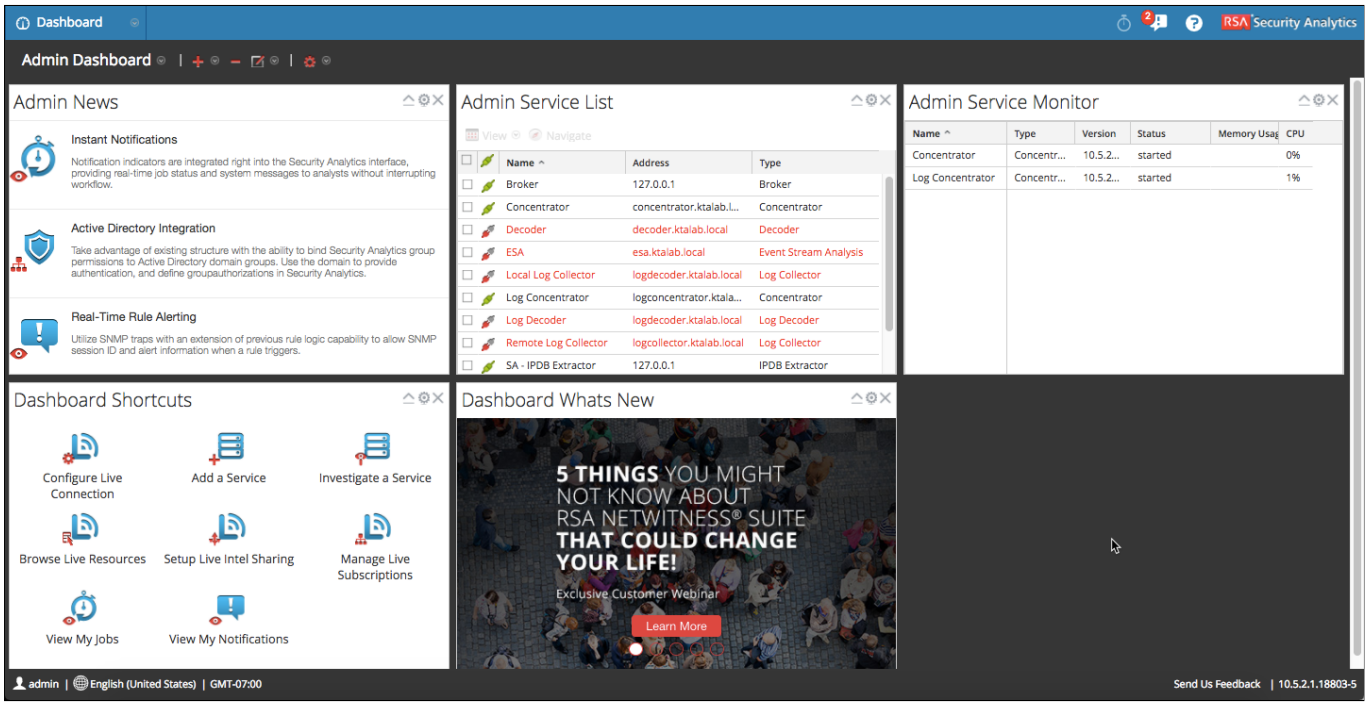


The Default Dashboard

Dashlet Types

Dashlets for the Admin Persona

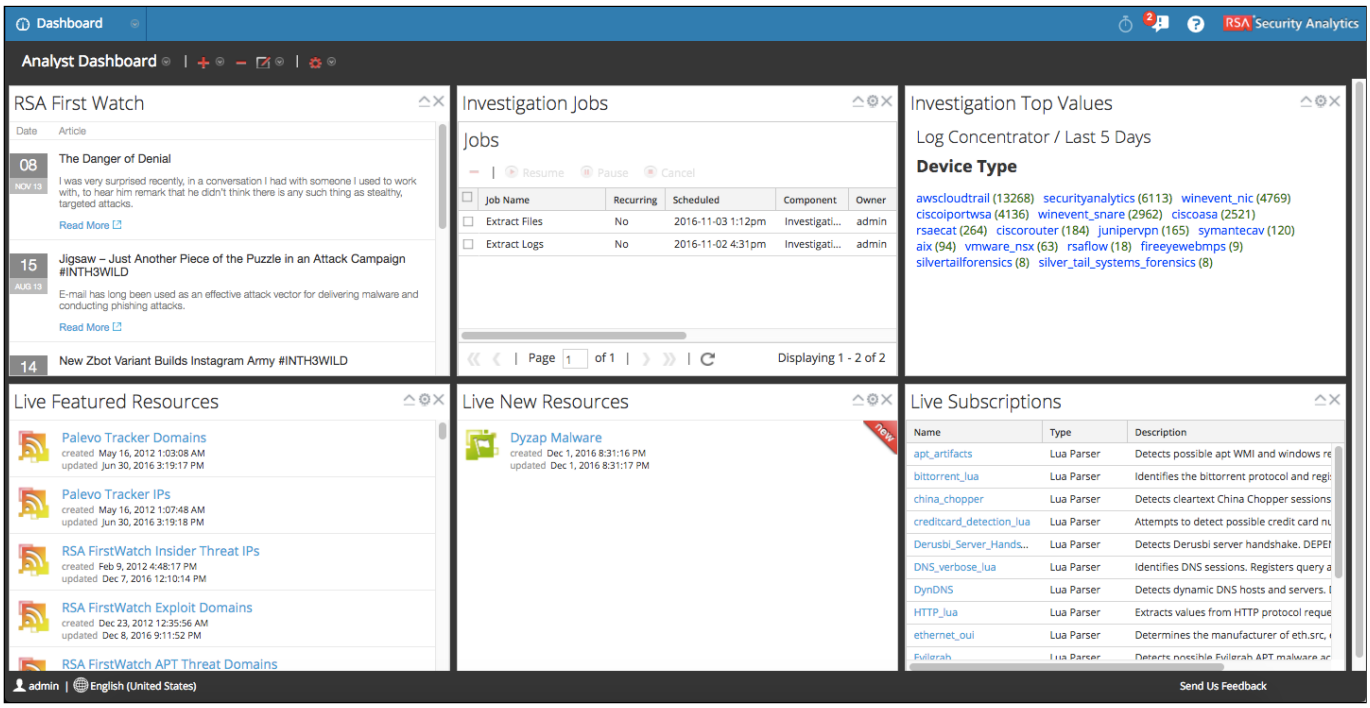
- **Admin News** - Not terribly useful. Shows a very short list of some features.
- **Admin Service List** - Displays a list of SA services, from where you can launch into **Administration** or **Investigation**.
- **Admin Service Monitor** - Fairly useless. Displays only status, memory usage, and CPU usage for a single selected service type.
- **Dashboard Shortcuts** - Some buttons which link to some UI pages. Again, not very useful.
- **Dashboard Whats New** - Displays some RSA news, links to articles, and RSA events. The HTML displayed in this dashlet is fetched directly from your browser, rather than the UI fetching it itself.

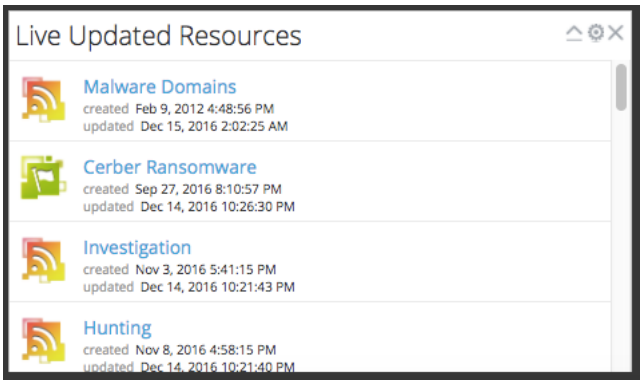


Admin Dashlets

Analyst Persona Dashlets

- **Dashboard RSA First Watch** - Shows articles from the RSA First Watch blog.
- **Investigation Jobs** - Shows jobs which have run inside Investigation, such as file extractions and PCAP exports.
- **Investigation Top Values** - Displays top N values and number of sessions for a selected meta key of a selected service over a selectable timeframe. It can also apply a query to filter the results.
- **Live Featured Resources** - Shows Resources in Live which RSA has chosen to feature
- **Live New Resources** - Shows a list of resources which are new to RSA Live
- **Live Subscriptions** - Shows which Live resources SA is currently subscribed to.
- **Live Updated Resources** - Shows which resources have been updated recently in Live

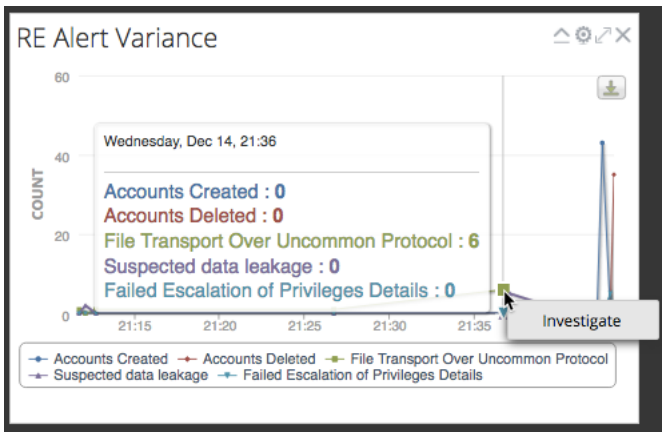




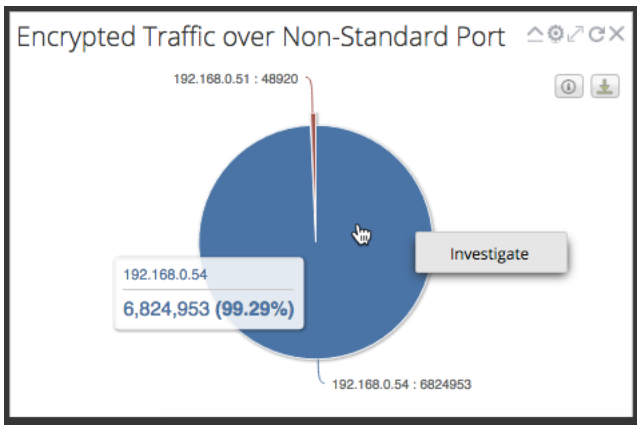
Reporting Dashlets

(These can be either Analyst, Incident Responder, or Administrator persona, case depending)

- **Reports RE Alerts Variance** - Plots top N Reporting Engine (not ESA) alerts in a timeline chart. The type of chart is selectable, as are the number of alerts to plot.

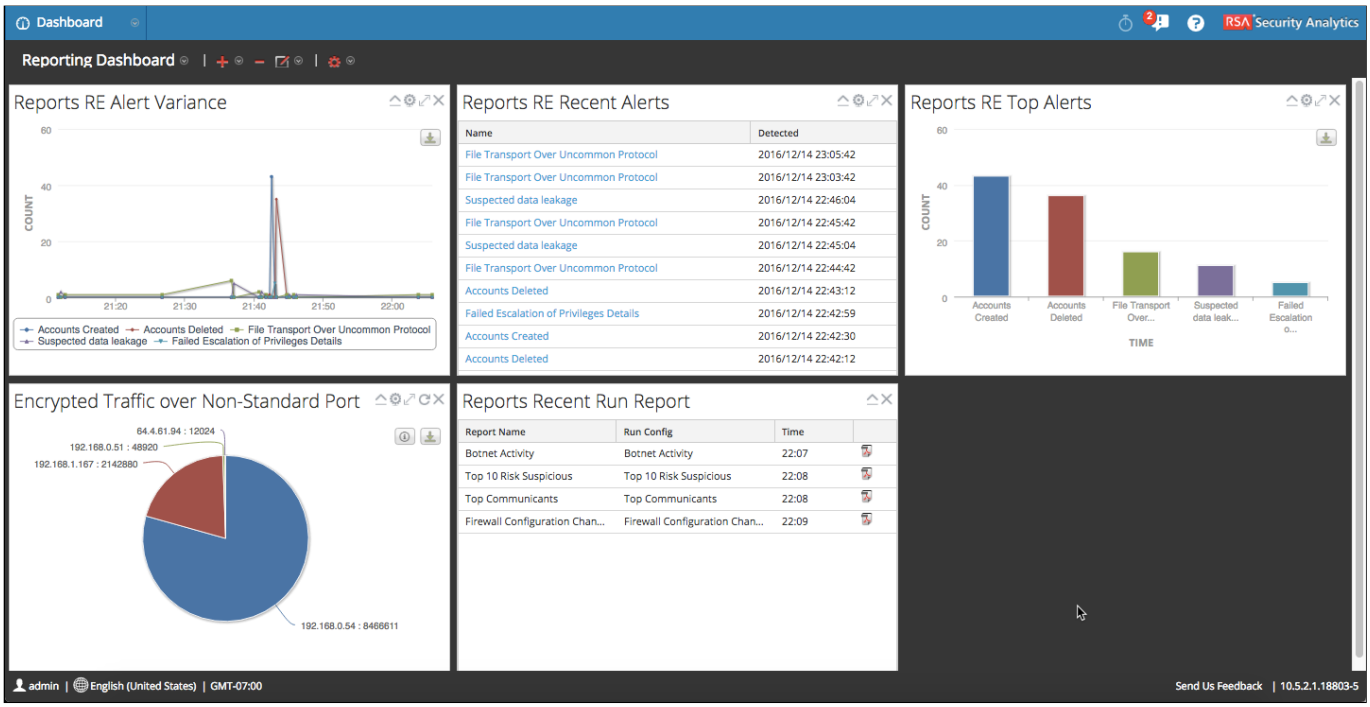


- **Reports RE Recent Alerts** - Shows a list of the most recent N triggered RE alerts over a configurable timeframe.
- **Reports RE Top Alerts** - Shows top N alerts over a configurable timeframe in a chart.
- **Reports Realtime Chart** - Displays a preconfigured RE chart in the dashlet over a configurable period of time. The type of chart is configurable within the dashlet. This can be immensely useful in displaying key analytical / event statistics.



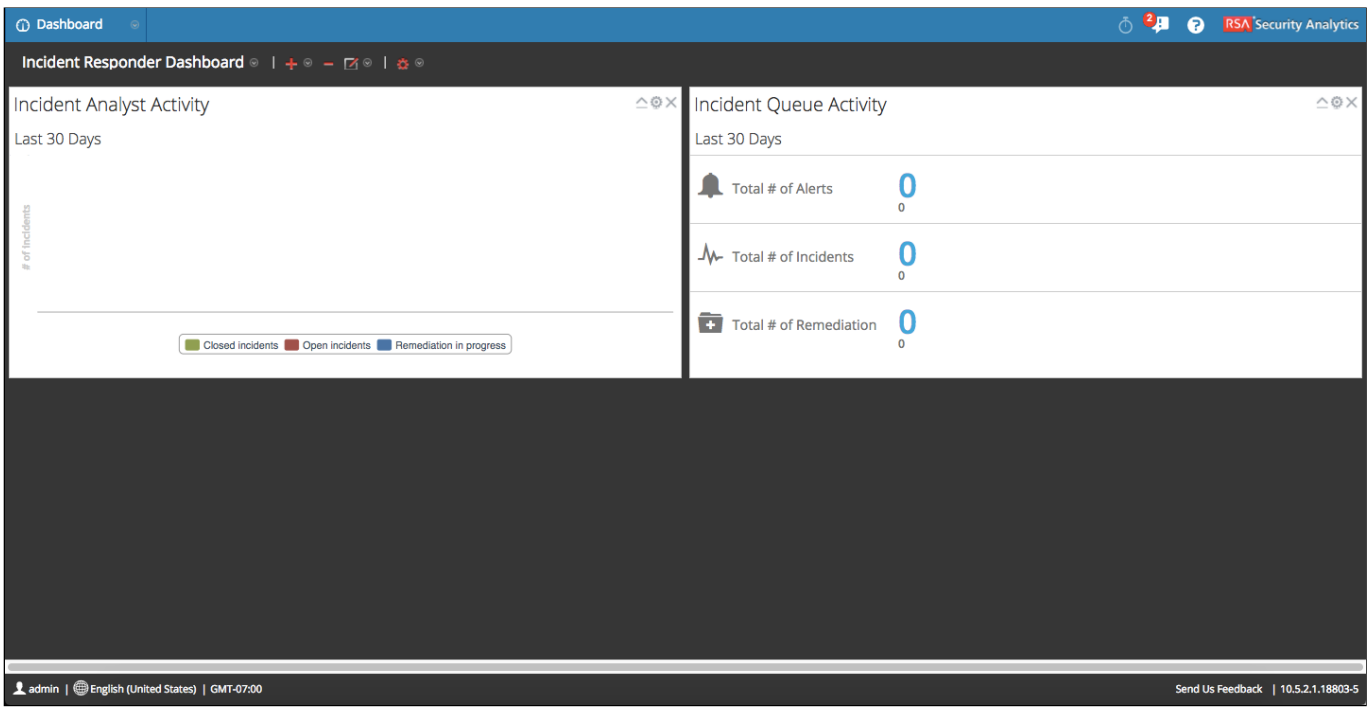
- **Reports Recent Run Report** - Displays a list of recently run RE reports

Top Tip: Most RE dashlets can be maximised with the button. Some can be refreshed with the button. Hovering over a chart will display contextual information about the place in the chart which you've hovered over. Some dashlet results can also be exported to PDF with the button. From some dashlets, one can also Investigate alerts from here by clicking on the chart and selecting **Investigate**.



Incident Responder Persona Dashlets

- **Incidents Analysts Activity** - Shows how many open incidents there are in IM over a selectable timeframe, how many have been closed, and how many are being remediated.
- **Incidents Queue Activity** - Displays how many alerts, incidents, and remediations there are in IM over a selectable timeframe.

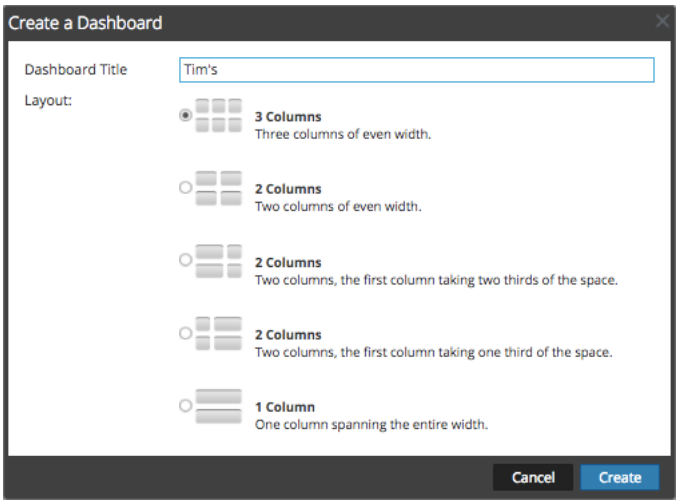


Malware Analyst Persona Dashlets - we shan't cover these, really, as we don't have MA in our lab.

- **Malware with High Confidence IOCs and High Scores** - Shows suspected malware detected with high confidence
- **Malware Scan Jobs List** - Shows Malware Analysis scan jobs
- **Malware Top Listing of Highly Suspicious Malware**
- **Malware Top Listing of Possible Zero Day Malware**

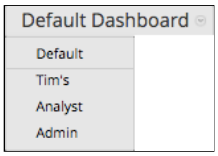
Creating a Dashboard

To add a new dashboard, click on the + dropdown, then choose **Create New Dashboard**. Note that the word "Dashboard" is automatically appended to the name you choose.



Selecting an Active Dashboard

The active dashboard name is the leftmost element displayed in the dashboard toolbar. It's also the selector for choosing the active dashboards.



Adding a Dashlet to a Dashboard

Click the add dropdown, and choose **Add Dashlet**. The Add a Dashlet dialog appears. Choose the dashlet type from the dropdown. Some dashlets have configuration options, and if so, they'll be displayed automatically.

Editing a Dashlet's Settings

Click the button in the upper righthand corner of a configurable dashlet.

Minimise a Dashlet

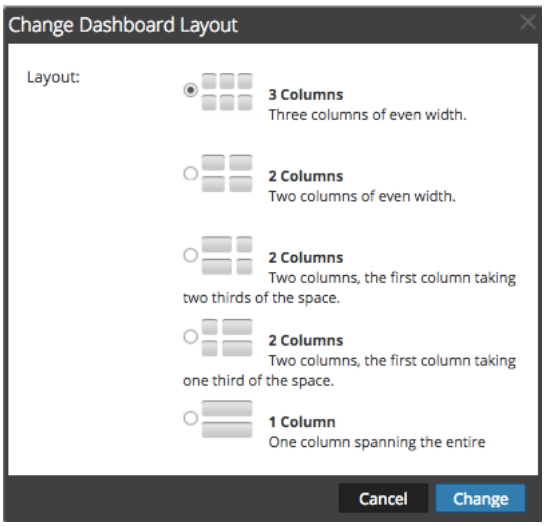
Click the button in the top righthand corner of a dashlet.

Removing a Dashlet from a Dashboard

Hit the grey at the upper righthand corner of a dashlet to remove the dashlet.

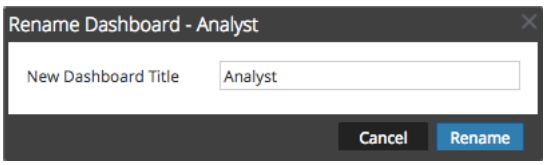
Changing a Dashboard Column Layout

Click the dropdown, then **Change Dashboard Layout**.



Renaming a Dashboard

Click the dropdown, then **Rename Dashboard**.



Sharing Dashboards With Other Users

Use the **Export Dashboard** function on the tools dropdown to export your dashboard. Share the exported file with your other SA users, and have them use the **Import Dashboard** function of the tools dropdown.

28. Licensing

As of version 10.5, SA licensing has become easier and has more options than it previously did.

Option 1: Trial Licenses

SA 10.5 comes with trial licenses for all components out-of-the-box. They will work for 90 days without annoyance. If you do not license your SA components within 90 days, on the 91st day, a large red banner will display across the top of the UI informing you that you are out of compliance with your license agreement. However, nothing will stop working! SA will continue working in perpetuity even without a valid license. This is a significant change from previous releases, where components would cease to function when the license expired, wreaking havoc in customers' environments.

Obtaining working SA licenses from RSA can be a challenge which has been known to, on occasion, drag on for months. Version 10.5 removed all license enforcement functionality from SA in order to prevent customers from going down due to a licensing snafu.

Option 2: Metered Licensing

Also new in SA 10.5 is metered licensing. In this mode, licenses do not have to be purchased for each individual service, but rather a license is purchased for aggregate consumption per-day. If you purchase a 1TB license, you can capture 1TB per day on any number of Decoders you wish to deploy in your environment. It doesn't matter whether it's 1 Decoder or 20 Decoders, so long as you keep your captured data below 1TB, you are fine.

As part of a metered license, a customer must agree to send an SA-generated report back to RSA to allow for billing based on consumption. This can be done directly from SA over the internet, or for airgapped deployments, the file can be submitted offline.

- Packet licensing can be purchased in increments of 1TB/day.
- SIEM licensing applies to Logs, and can be purchased in 50GB/day increments.
- Malware Analysis licensing is also offered in 1TB increments per-day.

Option 3: Service-based Perpetual Licensing

Licenses can also be purchased for individual components (Decoder, Log Decoder, Concentrator, MA, ESA, etc) to be perpetually valid. Perpetual licenses are not subjected to capture limits.

Option 4: Service-based Subscription Licensing

Similar to perpetual licensing, component licenses can be purchased for limited periods of time, at which time said license expires.

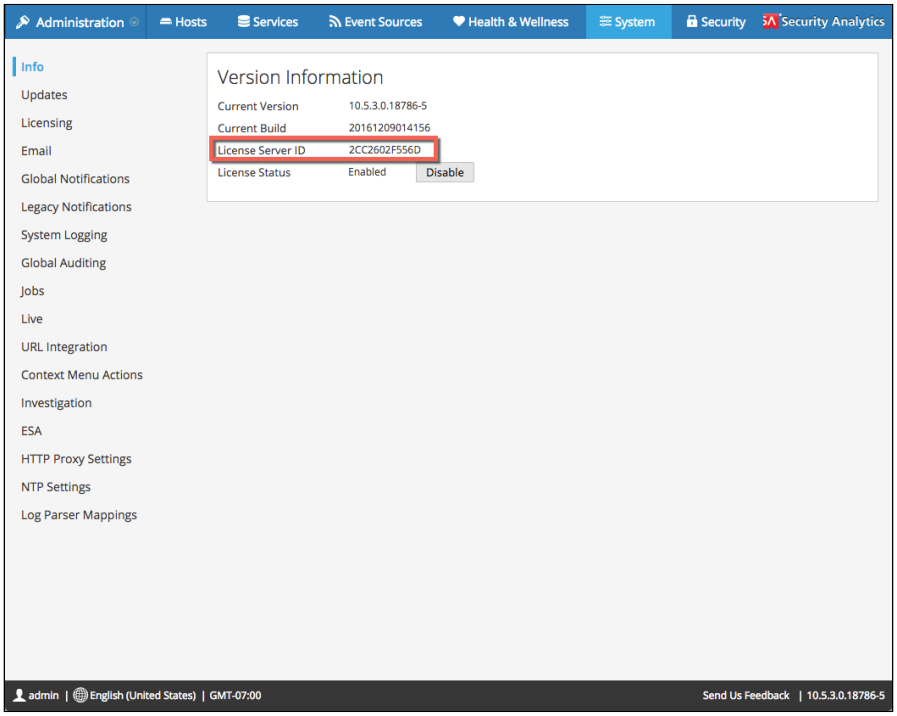
Option 5: A Mixture

Metered, perpetual, and subscription licenses can be mixed to suit a customer's needs.

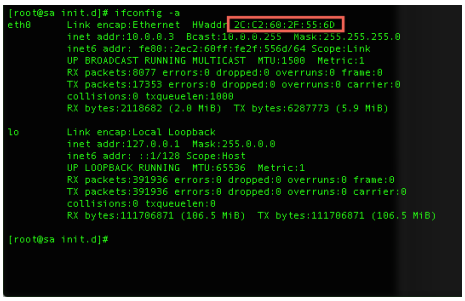
License Server ID

The **License Server ID** is what identifies your SA environment to RSA's licensing servers. As mentioned previously, it is tied to a MAC address of one of your system's NIC's. On a lab system, it will take the MAC of the first NIC on the system. In our lab, this is eth0.

On a physical RSA appliance, the License Server ID will take the MAC address of interface **em3**. On a physical RSA appliance, you will also see a bridge interface labeled **eth9**, bridged to em3. It will share em3's MAC address.



Lab License Server ID



Matches eth0 MAC Address on Lab System

Installing an SA License

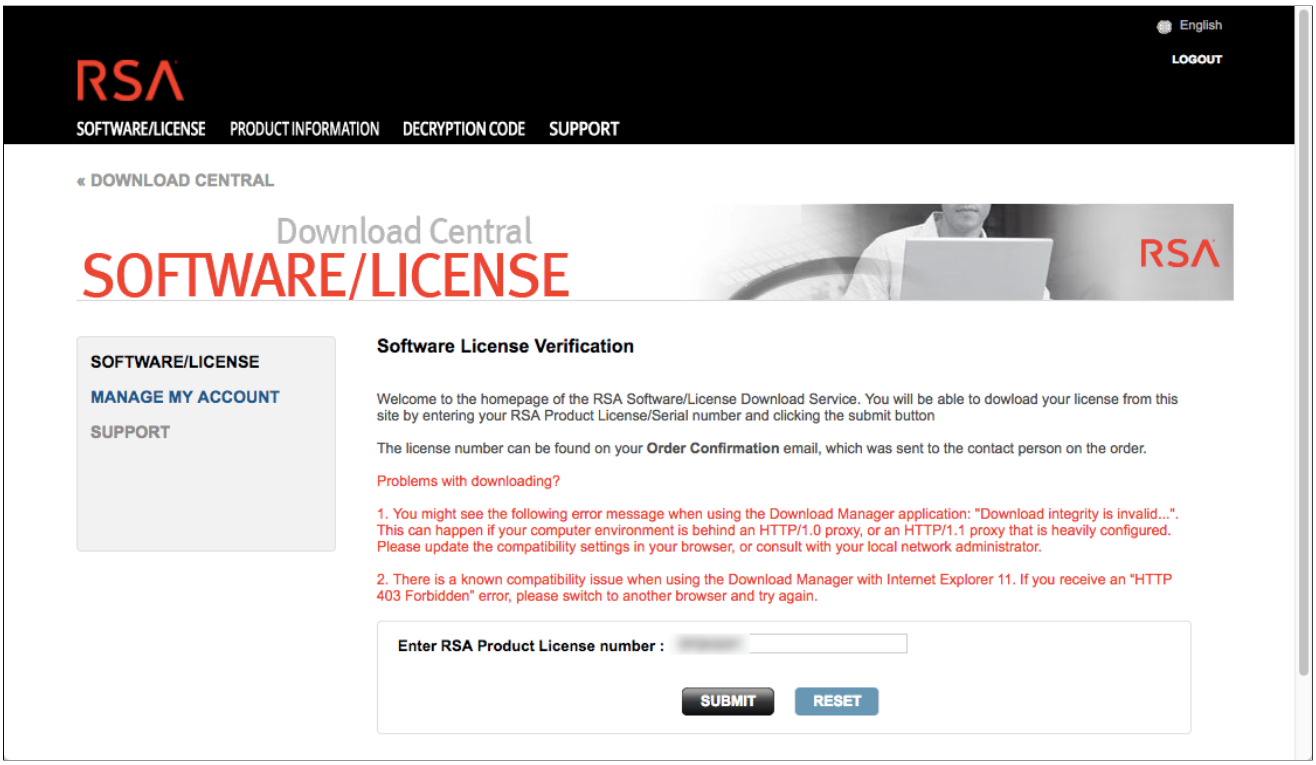
SA licenses are generated by RSA employees who work on a team called the License Seed Response team. They are responsible for generating licenses based on what was ordered by the customer. These licenses are created within a back end business system, from which Security Analytics obtains its licenses.

Within SA, all licensing of SA services is handled from the UI server itself. On this server, there is a glue service called **fne-server**, which provides the back end for the licensing, and it also communicates with the RSA licensing business systems via the Internet. There are two ways to install licenses: **online**, or **offline.** Offline mode is for airgapped environments or environments which require the use of a web proxy.

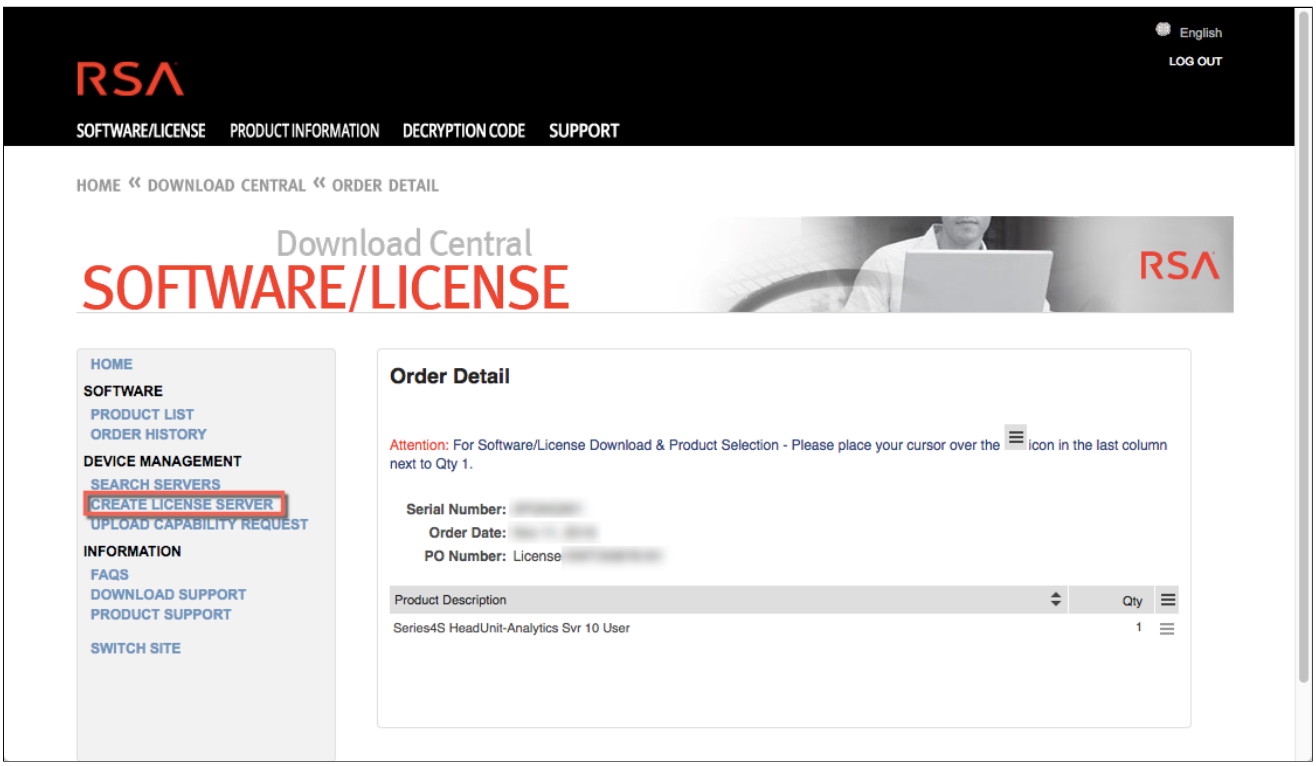
Mapping Licenses

Before licenses can be downloaded and applied to SA, the available licenses in RSA's back-end systems must be applied to a specific SA server, which is identified by its **License Server ID**, which SA bases on the MAC address of an installed NIC. These steps must be followed before SA can be licensed.

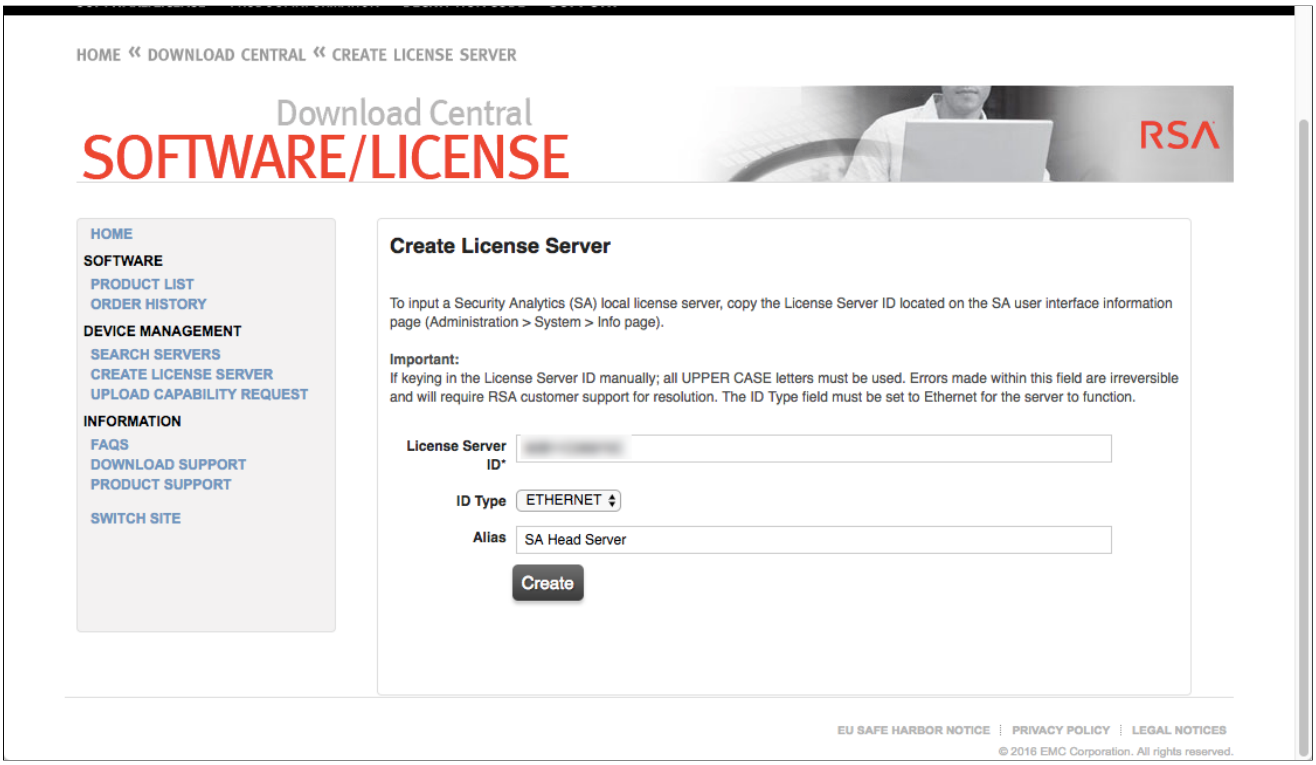
1. In a web browser, navigate to https://download.rsasecurity.com and enter the credentials that were provided you by RSA.
2. Enter the product license number or serial number provided to you by RSA. In the case of a physical SA appliance, this should be the serial number of the system.



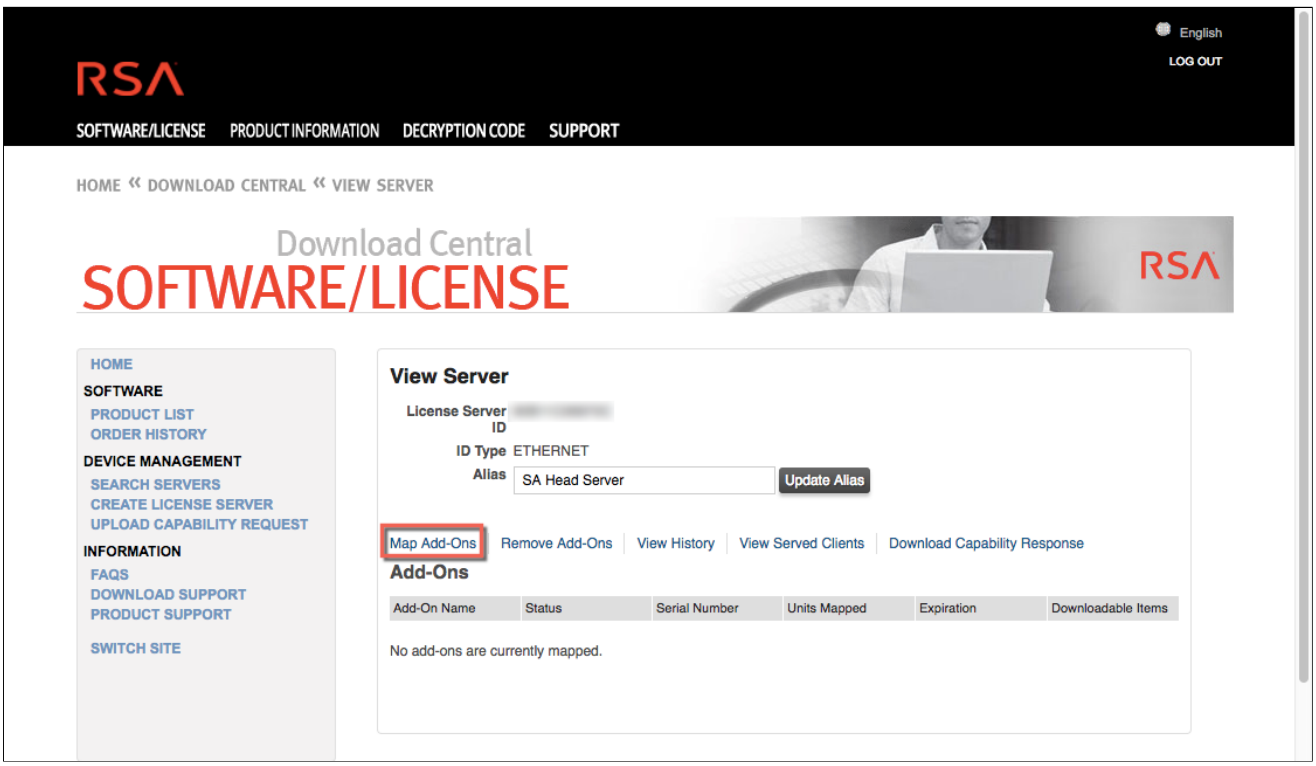
3. Click the **Create License Server** link.



4. Enter the license server ID in the first field. You obtain this from the **Administration->System->Info** page of the UI. Leave the ID type as Ethernet.

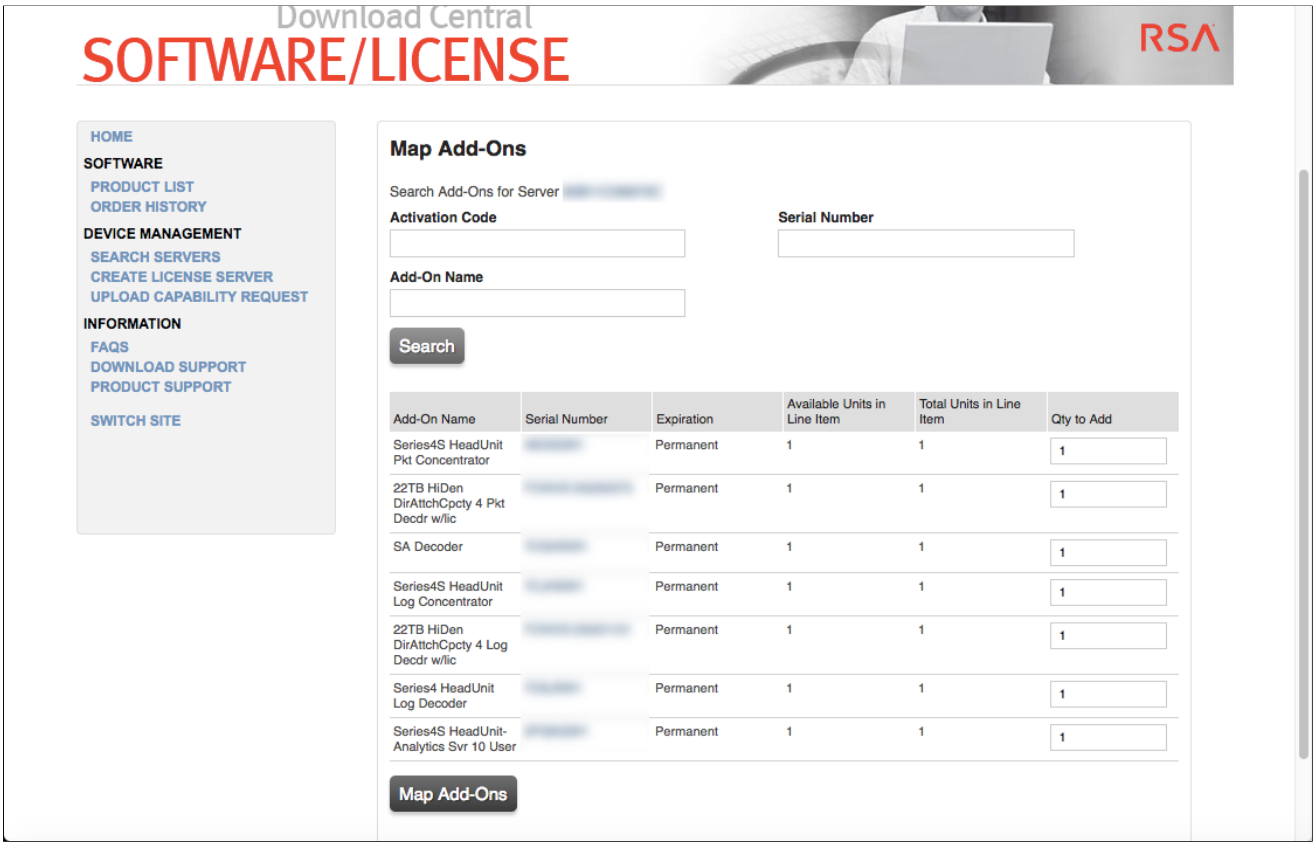


5. In the alias field, enter a friendly name for your SA server, like 'My SA Server' or 'Prod SA', or even its hostname.

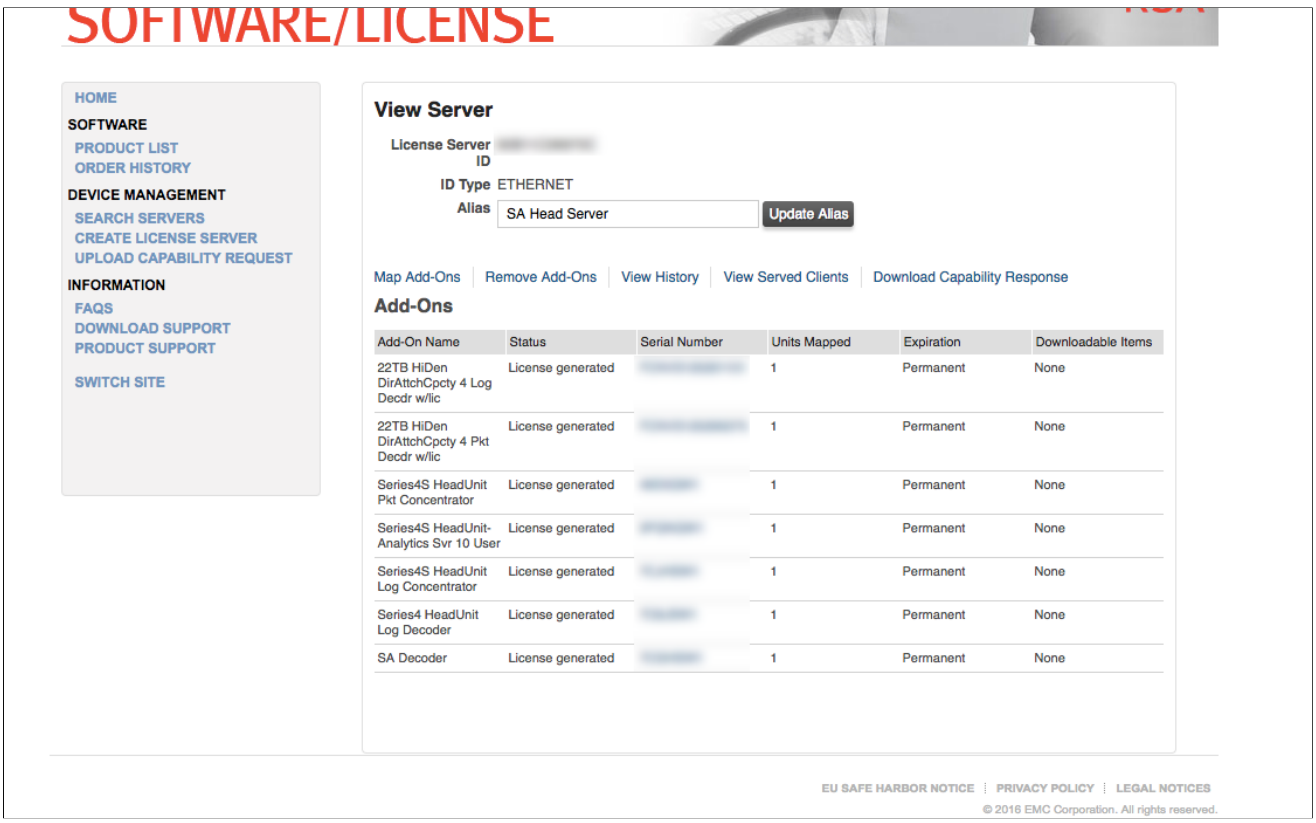


6. From the **View Server** page, click **Map Add-Ons**. If there are licenses available, you will see licenses which can be added to your license server, and how many of each type that are available. In the text box to the right of each line item, type in the number of licenses to map.

   Top Tip: What is shown on this page can be slightly nebulous. Some items may be straight forward, such as a Decoder or Broker, or ESA. You may also see licenses for storage and users. In service-based licensing, these are superfluous, as the amount of licensed storage is unlimited. You shouldn't have to map these. In metered licensing, it is unknown whether these must be mapped. This will be updated as more info becomes available.

7. 

8. Sync your licenses to your SA server using the below steps for either online mode or offline mode. After this, you should see your licenses as being mapped. Online mode is used by SA hosts which have Internet connectivity. Offline mode is for SA hosts in an airgap environment.



Syncing Licenses With Online Licensing

Online licensing requires a direct connection to the Internet from the UI server to host **rsasecurity.subscribenet.com** over port TCP 80. No other SA host will need to communicate with this site.

1. Follow all steps above to map your licenses to your SA server ID.
2. Navigate to **Administration->System->Licensing.**
3. In the **Overview** tab, click the **Licensing Options** dropdown, then **Refresh Licenses**. This launches an **fneserver** connection back to RSA to retrieve any licenses which you mapped above. Your licenses should now appear in either the **Service Based Licenses** pane or the **Metered Licenses** pane. You may have to refresh the page a few times in order for the synced licenses to show up.

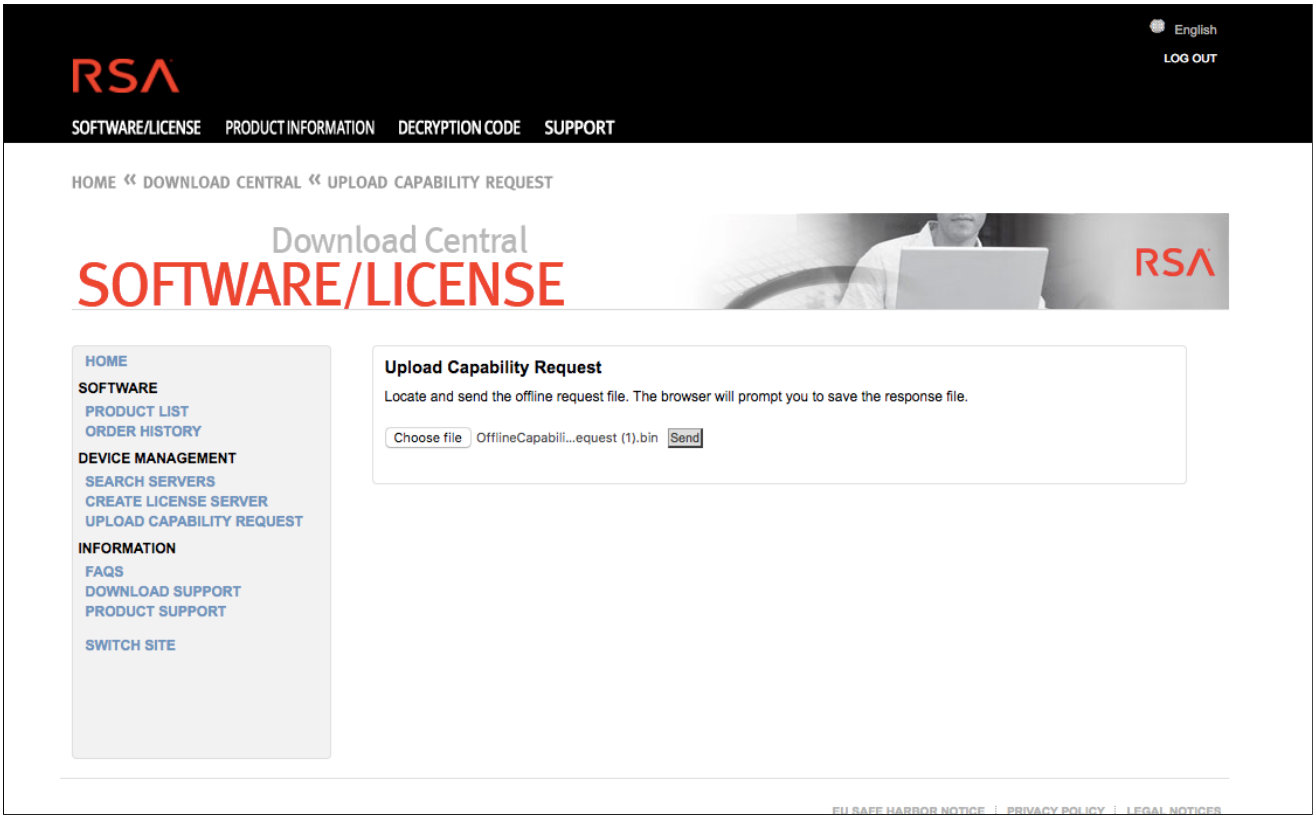
The Web Proxy Exception

If your environment requires the use of a web proxy, **offline licensing mode must be used**. **fneserver** does not support web proxies in RSA's implementation.

Syncing Licenses With Offline Licensing

Offline licensing is just a manual way of obtaining a license file. It's procedure goes like this:

1. Follow all steps above to map your licenses to your SA server ID.
2. Navigate to **Administration->System->Licensing.** Click the **Settings** tab.
3. Click the **Download Request** button.
4. On the RSA Downloads site, navigate to your License server ID page, and click **UPLOAD CAPABILITY REQUEST**.



5. Choose the file you downloaded in step 2, and click **Send**. A file should now be downloaded by your web browser. This is the capability response file.
6. Back in the UI, on the **Administration->System->Licensing->Settings** page, click the **Upload Response** button and select the capability response file downloaded in step 5. Click OK or Open.
7. You should now see your licenses in the **Overview** tab of **Administration->System->Licensing**.

<add SS's>

Which SA Services Require Licenses?

Only some SA services require licenses:

- Packet Decoder
- Log Decoder
- Concentrator
- Broker
- Archiver
- Full Malware Analysis
- ESA

Services that do not require licenses are:

- Log Collector - you can have as many of these as you'd like
- Reporting Engine
- Incident Management
- Co-Located Malware Analysis

- Warehouse Connector
- IMDB Extractor

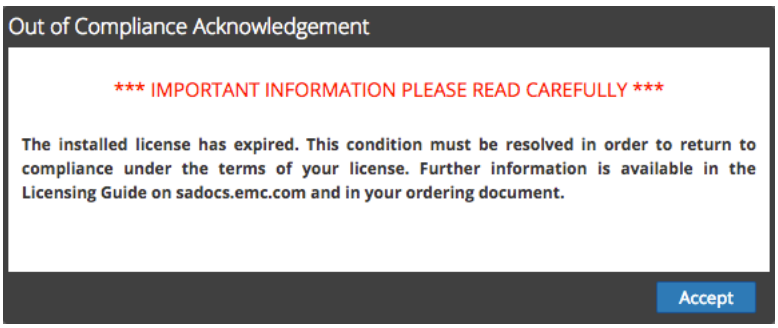
Applying Licenses to Services

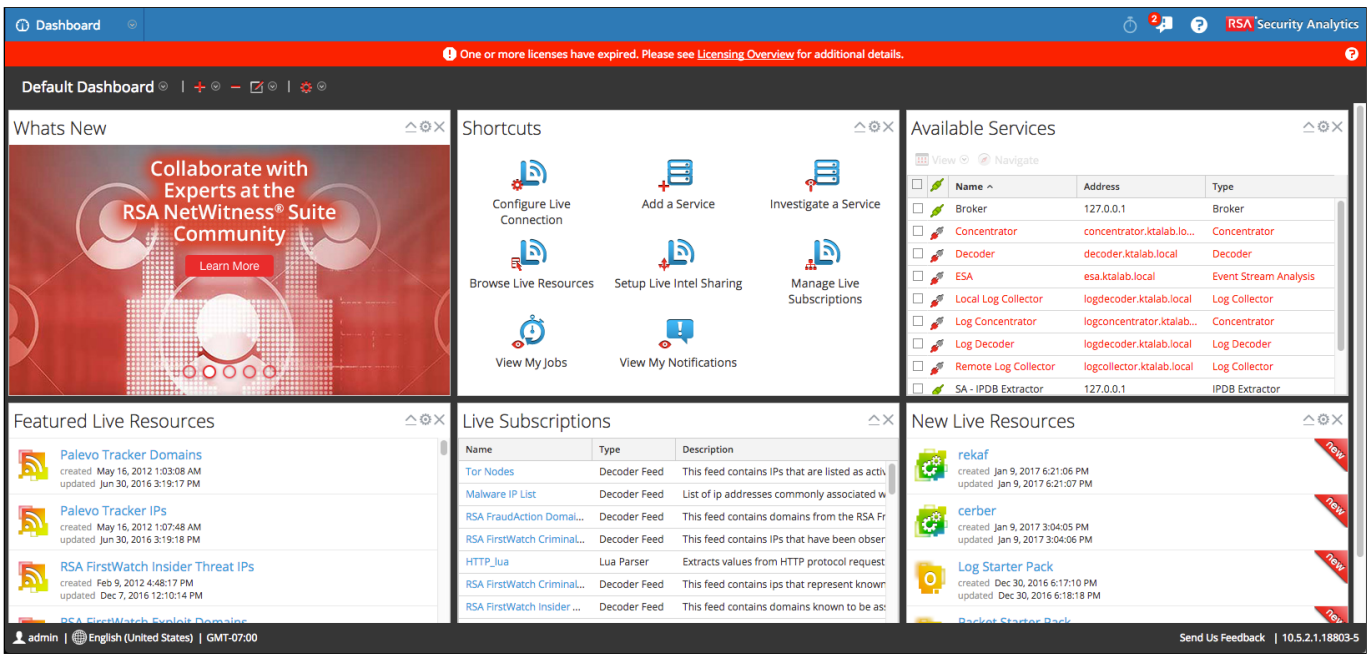
SA should automatically apply the appropriate licenses to its connected services once they've been added to the server. It may take a few minutes for all the licenses to get applied. There is, however, a manual way of applying licenses to and removing them from your SA services.

1. Navigate to **Administration->Services.**
2. Check the box of a service you wish to apply a license to.
3. Click the Licenses drop-down, then **Entitle Service**.

What Happens if a License Expires or if I Become Non-Compliant?

Practically nothing. Version 10.5 of SA removed all license enforcement code, and replaced it with nagware. Now, you will only see a pop-up warning you of the problem, and a red out-of-compliance banner will be displayed across the top of the UI. If this happens, you should work with RSA or your RSA channel partner to resolve the situation as soon as possible.





Troubleshooting Licensing

When the licenses you were expecting fail to appear, it can be frustrating. The best method of troubleshooting is by tailing the fneserver logfiles in the **/var/log/fne** directory of the SA server. Its three logfiles are called **fne-access.log**, **fne-error.log**, and **fneserver.log**. **fneserver.log** often is empty.

```
Info   : [25/Nov/2016:19:39:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:19:49:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:19:59:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:20:09:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:20:19:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:20:29:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:20:39:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:20:49:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:20:59:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:21:09:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:21:19:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:21:29:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:21:39:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:21:49:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:21:59:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:22:09:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:22:19:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:22:29:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:22:39:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:22:49:09 +0000] Server refresh completed.
Info   : [25/Nov/2016:22:59:09 +0000] Server refresh completed.
Info   : [05/Dec/2016:18:53:45 +0000] FNE Server starting.
Warn   : [05/Dec/2016:18:53:45 +0000] Log rotation policy of "" not recognized, defaulting to "size".
Warn   : [05/Dec/2016:18:53:45 +0000] Log rotation policy size syntax error in "" - using default of "10m".
Warn   : [05/Dec/2016:18:53:45 +0000] Log rotation policy of "" not recognized, defaulting to "size".
Warn   : [05/Dec/2016:18:53:45 +0000] Log rotation policy size syntax error in "" - using default of "10m".
Info   : [05/Dec/2016:18:53:46 +0000] Server (version 4.6.0) startup completed.
Info   : [05/Dec/2016:18:53:46 +0000] Server borrow interval set to 31536000.
Info   : [05/Dec/2016:18:53:46 +0000] Server renew interval set to 86400.
Info   : [05/Dec/2016:18:53:46 +0000] Server response lifetime set to 86400.
Info   : [05/Dec/2016:18:53:47 +0000] Server maintenance interval set to 259200.
Info   : [05/Dec/2016:18:53:47 +0000] Server expiration granularity set to "day".
Info   : [05/Dec/2016:18:53:47 +0000] Virtual machine detection disabled.
Info   : [05/Dec/2016:18:53:47 +0000] Server support for virtual clients is enabled.
Info   : [05/Dec/2016:18:53:47 +0000] FNO poll for updated licenses is enabled.
Info   : [05/Dec/2016:18:53:47 +0000] Online sync to FNO is enabled.
Info   : [05/Dec/2016:18:53:47 +0000] Offline sync to FNO is not enabled.
Info   : [05/Dec/2016:18:53:47 +0000] Inclusion of historical data into sync to FNO is enabled.
Info   : [05/Dec/2016:18:53:47 +0000] Sync from FNO is not enabled.
Info   : [05/Dec/2016:18:53:47 +0000] Sync to backup FNE server is not enabled.
Info   : [05/Dec/2016:18:53:47 +0000] Push architecture is not enabled.
Info   : [05/Dec/2016:18:53:47 +0000] Info message processing is not enabled.
Info   : [05/Dec/2016:18:53:47 +0000] Virtual host is allowed.
Info   : [05/Dec/2016:18:53:47 +0000] Virtual clients are allowed.
Info   : [05/Dec/2016:18:53:47 +0000] FNE Server startup complete.
Info   : [05/Dec/2016:18:53:47 +0000] Generating capability request (using server hostid "ETHERNET=2CC2602F556D").
Info   : [05/Dec/2016:18:53:47 +0000] Online sync to FNO is not needed at this time.
Info   : [05/Dec/2016:18:53:57 +0000] Successfully received response from FNO.
Info   : [05/Dec/2016:18:53:57 +0000] Server refresh completed.
Info   : [05/Dec/2016:19:03:57 +0000] Server refresh completed.
Info   : [05/Dec/2016:19:13:57 +0000] Server refresh completed.
Info   : [05/Dec/2016:19:23:57 +0000] Server refresh completed.
Info   : [05/Dec/2016:19:33:57 +0000] Server refresh completed.
Info   : [05/Dec/2016:19:43:57 +0000] Server refresh completed.
Waiting for data... (interrupt to abort)
```

29. SA Language Packs

For version 10.5, SA has been localised into several languages: Japanese, German, and Spanish. Support to these languages is added to the UI through the use of SA Language Packs. These language packs are an RPM package which is downloaded from SCOL and installed on the SA server.

Scope

- The scope of SA localisation is limited to only messages that are displayed by the UI from the UI's own code.
- **There is no Language Pack available for SA Core or other Java-based services.** UI messages which are displayed as a result of API calls to Core or Java services will still be displayed in English.
- Some messages will still be seen in English, but this is improving with subsequent SA releases.
- The list of supported languages is expanding with time.
- A language pack is not released for every SA release
- For those SA releases which do have a language pack, there is a delay between the availability of the SA release and its associated language pack.
- Usually the language pack will only target one particular service pack of SA, either the .1 or .2 point releases (i.e. 10.5.2, 10.6.1, etc). This allows each major/minor release time to mature somewhat before the language pack comes out, so that customers aren't stuck with outdated language packs.
- SA can still be upgraded if it has a language pack installed. Language Packs offer some forward-compatibility through the use of English fallback. Any UI messages in a newer SA release which were present at the time of the language pack's release will still be localised. Any UI messages which are new in the SA code update will be displayed in English.
- UI language is a per-user option with no global setting at present
- System log messages will still be displayed in English
- Localised SA release documentation is generally released simultaneously with or closely following its corresponding English documentation, with the exception of SA Online Help at http://sadocs.emc.com. Localised Online Help is usually released simultaneously with the Language Pack.
- If you find areas of the UI which you think should be localised, please feel free to open a suspected bug case with RSA Support.
- Language packs unfortunately do not yet include the ability to choose different international formats, such as date and time format and decimal separator.

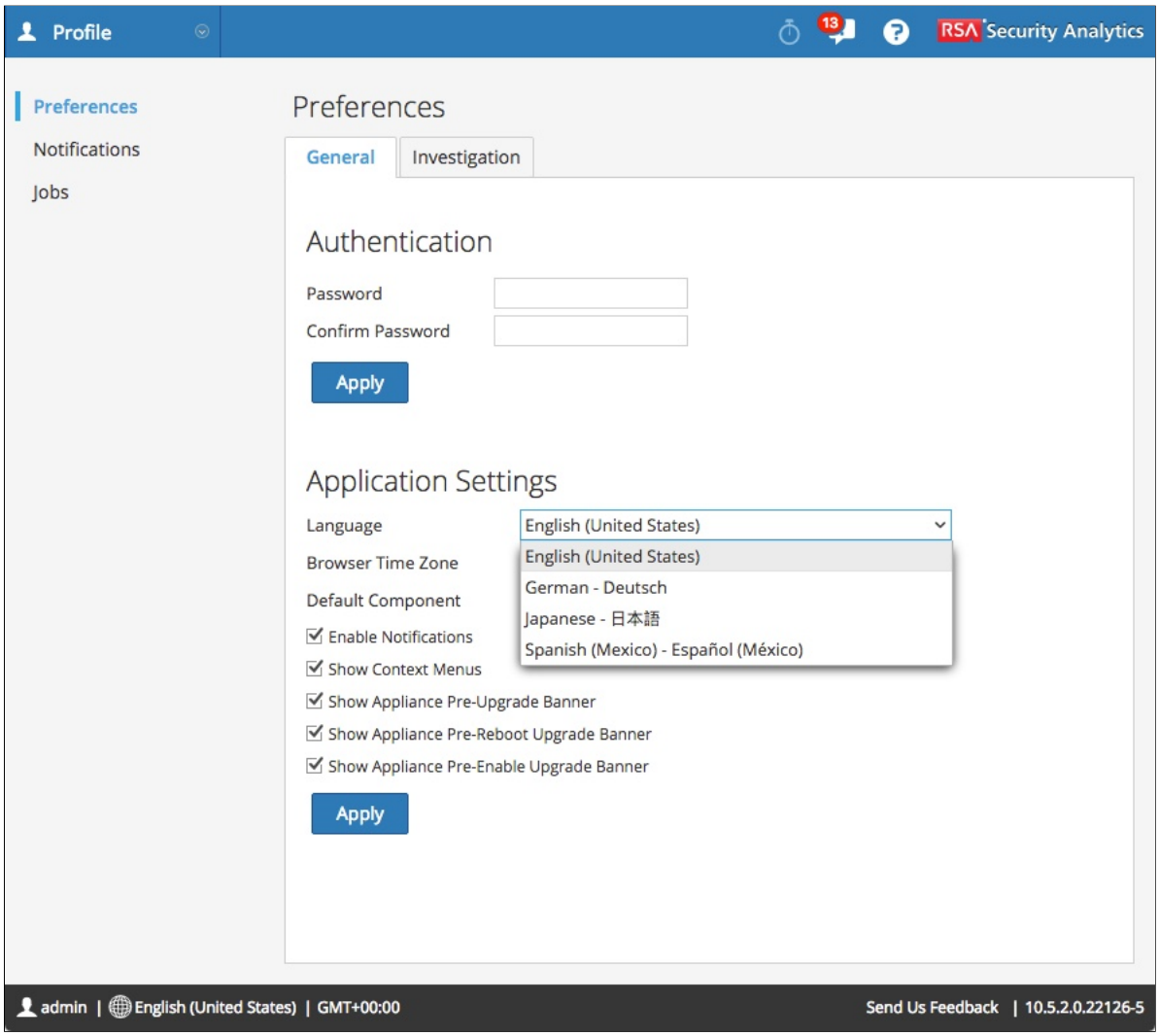
Installation

1. Download the Language Pack rpm file from the RSA Community site at https://community.rsa.com
2. Using SCP or SFTP, copy the language pack RPM file to your SA server.
3. Execute the following commands:

```
rpm -ivh <filename>
restart jettysrv
```

Selecting a Language

From the SA menu, select Profile. Then scroll down to the language setting, select your language, and hit apply.



Et voila! Your SA UI is now displayed in your chosen language.

Administration
ホスト
サービス
イベント ソース
ヘルスモニタ
システム
セキュリティ
情報
更新
ライセンス
メール
グローバル通知
レガシー通知
システム ログ
グローバル監査
ジョブ
Live
URL統合
コンテキスト メニュー
Investigation
ESA
HTTPプロキシ設定
NTP Settings
Log Parser Mappings
バージョン情報
現在のバージョン 10.5.2.0.22126-5
現在のビルド 20160222081211
ライセンス サーバID 2CC2602F556D
ライセンス ステータス 有効
無効化
admin | 日本語 | GMT+00:00
フィードバックの送信 | 10.5.2.0.22126-5

30. Querying With the NetWitness API

For some personas of SA users, rather than using SA's built-in interface, it is desirable to use custom-built tools to query and extract data from Security Analytics. In this lesson, you shall learn how to use key SDK calls for querying data through SA's REST API. API stands for Application Programming Interface, which is a standard term which describes low-level methods for interacting with an application outside of the normal user interfaces of the application.

NetWitness (SA Core) APIs

Earlier in the course, you learned that SA Core possesses two APIs, or protocols for interacting with the API tree:

- **C** - This is the original NetWitness API, which is used by NwConsole, and is also used by the SA UI to communicate with SA Core services. Making use of this API requires download of the NetWitness SDK from RSA, and also some knowledge of C programming. To obtain the SDK RPM file, extract the file nwsdk-<version>.el6.x86_64.rpm from your yum repository or the SA Upgrade .zip package. You can then install it on a CentOS system to get the files. You can also click here to download a version of the 10.5.3.0 RPM which we have converted into a .tgz file.

- **REST** - The RESTful API is an HTTP/HTTPS API for issuing API calls using a web client of some type, be it through a web browser, scripting language like Perl or Python, or with command line utilities like 'curl' or 'wget'.

These are commonly referred to as the NetWitness APIs, but they apply only to SA Core services. These are distinct from any APIs which SA's Java or glue services expose. What you can accomplish with both API types is nearly identical - both APIs use the same tree layout and API calls. The RESTful API is considerably easier to use and demonstrate, though, so it is with this which we shall work in our lesson.

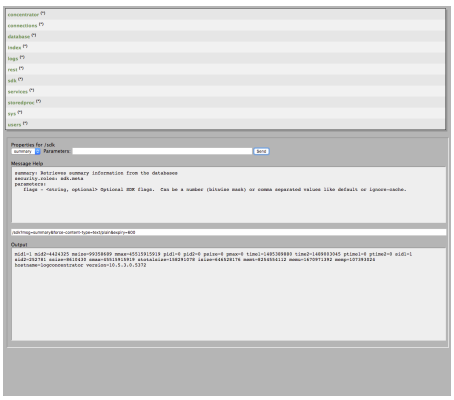
Accessing the REST API

Every SA Core service runs its own RESTful web server for service API calls. It can listen using either HTTP or HTTPS, but not both. HTTP is the default, but HTTPS can be enabled by setting the value of **/sdk/config/ssl** to **on**. Please bear in mind that any time one wishes to query data from SA Core's index, the REST call must be issued against a Core service which contains a full index (either real or virtual), such as a Broker, Concentrator, or Archiver. To get the REST API port number for each Core service, simply add 100 to its unencrypted port number:

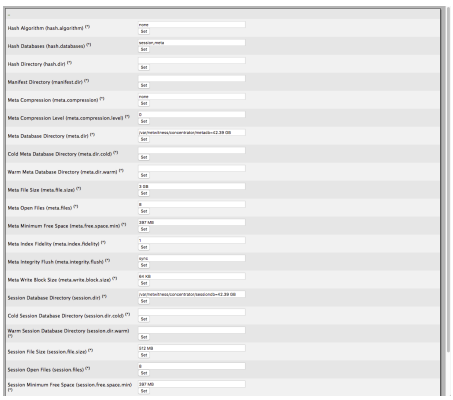
Service	REST API Port
Broker	50103
Concentrator	50105
Decoder	50104
Log Decoder	50102
Log Collector	50101
Archiver	50108

The REST API GUI

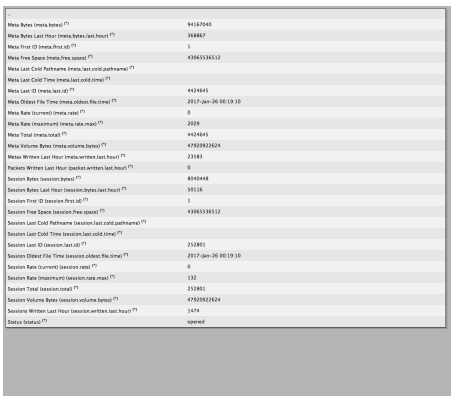
In addition to the basic REST interface, the REST port of SA Core also has a full HTTP user interface for interacting with the REST API. This interface is nearly identical to the **Explorer** view for Core services in the SA UI. In here, we can navigate through the tree, get and set configuration values, and also issue API calls. Unlike the SA UI's Explorer view, the RESTful UI also presents us with a very useful URI string which can be pasted into curl commands, which makes this interface invaluable for crafting RESTful API calls for use in our custom tools.



REST GUI - SDK Interaction



Setting Config With RESTful GUI



Viewing Stats With RESTful GUI

Issuing REST API Calls Using 'curl'

There are any number of methods for using the NetWitness RESTful API, such as with Perl or Python, or wget. We shall use '**curl**', which is a fairly standard UNIX/Linux tool for performing HTTP calls. **Curl** can be used to download files and web pages, but it can also issue REST API calls. It is also quite easy to use, and can easily be called from shell scripts or other scripts.

Since we are accessing a resource which requires authentication (i.e. the RESTful API), we must authenticate to SA using curl. Here, we are authenticating against the local SA Core user database, not the SA user database. This local account must have appropriate permissions to perform the API calls which you are using.

The easiest way to authenticate is to specify "**user:password@**" in the URL, just after http:// and before your hostname. This will leave your credentials visible, however, to anyone who can view the system's running processes:

```
$ curl "http://admin:netwitness@sa:50103/sdk?msg=summary"

<?xml version="1.0" encoding="utf-8"?><response flags="1073872897">
 <string>mid1=1 mid2=442211481 msize=8946135040 mmax=45515915919 pid1=0 pid2=0 psize=0 pmax=0 time1=1484858460 time2=1487369455
ptime1=0 ptime2=0 sid1=1 sid2=11185003 ssize=375623680 smax=45515915919 stotalsize=122017133005 isize=148404528 memt=16725958656
memu=16328232960 memp=16617472 hostname=sa.ktalab.local version=10.5.3.0.5372</string>
</response>
```

OR you can use the "**-u <user>**" command-line parameter, which will prompt you for a password. This is more secure, but you must specify the password every time you execute the command.

```
$ curl -u admin "http://sa:50103/sdk?msg=summary&force-content-type=application/json&expiry=30"

Enter host password for user 'admin':
{
    "flags" : 1073872897,
    "string" : "mid1=1 mid2=442211481 msize=8946135040 mmax=45515915919 pid1=0 pid2=0 psize=0 pmax=0 time1=1484858460
time2=1487369455 ptime1=0 ptime2=0 sid1=1 sid2=11185003 ssize=375623680 smax=45515915919 stotalsize=122017133005 isize=148404528
memt=16725958656 memu=16327479296 memp=16617472 hostname=sa.ktalab.local version=10.5.3.0.5372"
}
```

When we use curl in our lesson, we shall use the former "**user:pasword@**" method.

Breaking the URL Down

Let's now break down the URL of a simple REST API call against a Broker using one of the URL's we demonstrated above:

```
http://admin:netwitness@sa:50103/sdk?msg=summary&force-content-type=application/json
```

http://	Protocol specifier (http or https)
admin:	username to authenticate with
netwitness@	password to authenticate with
sa	host to connect to
:50103	Port specifier. In this case, a Broker REST port
/sdk	The API Tree path to the command we wish to execute.
?msg	'?' indicates that what follows is a parameter. In this case, the parameter name is 'msg', the value of which instructs the API which command to execute
=summary	The value of the 'msg' parameter. In this case, "msg=summary" means that the REST API should execute the 'summary' command.
&force-content-type	The '&' indicates the presence of an additional parameter, in this case the parameter 'force-content-type'
=application/json	The value of the 'force-content-type' parameter. In this case, the value is 'application/json', which is an optional formatter parameter (described below)

Special HTTP Parameters

When using curl or other utilities for interacting with the RESTful API, there are a few parameters which can help you:

- **force-content-type** - This controls how the output of command will be formatted, by specifying a supported MIME type. Supported types are:
    - **text/xml** - This is the default, if **force-content-type** is not specified.

```
$ curl "http://admin:netwitness@sa:50103/sdk?msg=summary&force-content-type=text/xml"

<?xml version="1.0" encoding="utf-8"?><response flags="1073872897">
 <string>mid1=1 mid2=442211481 msize=8946135040 mmax=45515915919 pid1=0 pid2=0 psize=0 pmax=0 time1=1484858460 time2=1487369455
ptime1=0 ptime2=0 sid1=1 sid2=11185003 ssize=375623680 smax=45515915919 stotalsize=122017133005 isize=148404528 memt=16725958656
memu=16370290688 memp=16891904 hostname=sa.ktalab.local version=10.5.3.0.5372</string>
</response>
```

- **application/json** - Format output using JSON format. This can be very handy in conjunction with Python.

```
$ curl "http://admin:netwitness@sa:50103/sdk?msg=summary&force-content-type=application/json"

{
    "flags" : 1073872897,
    "string" : "mid1=1 mid2=442211481 msize=8946135040 mmax=45515915919 pid1=0 pid2=0 psize=0 pmax=0 time1=1484858460
time2=1487369455 ptime1=0 ptime2=0 sid1=1 sid2=11185003 ssize=375623680 smax=45515915919 stotalsize=122017133005 isize=148404528
memt=16725958656 memu=16358260736 memp=16867328 hostname=sa.ktalab.local version=10.5.3.0.5372"
}
```

- **text/plain** - Use plain text formatting.

```
$ curl "http://admin:netwitness@sa:50103/sdk?msg=summary&force-content-type=text/plain"

mid1=1 mid2=442211481 msize=8946135040 mmax=45515915919 pid1=0 pid2=0 psize=0 pmax=0 time1=1484858460 time2=1487369455 ptime1=0
ptime2=0 sid1=1 sid2=11185003 ssize=375623680 smax=45515915919 stotalsize=122017133005 isize=148404528 memt=16725958656
memu=16357597184 memp=16908288 hostname=sa.ktalab.local version=10.5.3.0.5372
```

- **expiry** - A REST API-specific parameter which controls how long the command will be permitted to run before timing out and returning with a timeout error. The default is 30 seconds, with the exception of sdk-query, which is permitted to run indefinitely.

Behind the Scenes of SA Investigation

If we launch into a root drill in **Investigation->Navigate** in SA and watch the Core service logs, we can quickly learn what exactly SA is doing behind the scenes to pull the data that's displayed.

Here's some key lines from the Broker logs (full logfile here).

```
Feb 20 16:49:04 sa NwBroker[2584]: [SDK-Info] [audit] User admin (session 1229, 127.0.0.1:47486) has requested the SDK summary info:
flags=0
Feb 20 16:49:05 sa NwBroker[2584]: [SDK-Language] [audit] User admin (session 1229, 127.0.0.1:47486) has requested the SDK language:
id1=0 id2=0 size=1000 flags=1
Feb 20 16:49:05 sa NwBroker[2584]: [SDK-Language] [audit] User admin (session 1229, 127.0.0.1:47486) has issued language (channel
1239) (thread 27145): id1=0 id2=0 size=1000 flags=1
Feb 20 16:49:05 sa NwBroker[2584]: [SDK-Language] [audit] User admin (session 1229, 127.0.0.1:47486) has finished language (channel
1239, queued 00:00:00, execute 00:00:00, concentrator.ktalab.local:50005=00:00:00): id1=0 id2=0 size=1000 flags=1
Feb 20 16:49:05 sa NwBroker[2584]: [SDK-Info] [audit] User admin (session 1250, 127.0.0.1:46430) has requested the SDK summary info:
flags=1
Feb 20 16:49:05 sa NwBroker[2584]: [SDK-Session] [audit] User admin (session 1260, 127.0.0.1:46442) has requested the SDK session
info: id1=11185004 id2=11185003
Feb 20 16:49:08 sa NwBroker[2584]: [SDK-Values] [audit] User admin (session 1229, 127.0.0.1:47486) has issued values (channel 1273)
(thread 5461): fieldName=eth.src id1=1 id2=442211481 threshold=100000 size=20 flags=sessions,sort-total,order-descending,ignore-
cache where="time=\"2017-02-13 16:50:00\"-\"2017-02-20 16:49:59\""
Feb 20 16:49:09 sa NwBroker[2584]: [SDK-Aliases] [audit] User admin (session 1229, 127.0.0.1:47486) has requested the SDK aliases:
key=eth.type
Feb 20 16:49:10 sa NwBroker[2584]: [SDK-Values] [audit] User admin (session 1229, 127.0.0.1:47486) has issued values (channel 1297)
(thread 5461): fieldName=eth.type id1=1 id2=442211481 threshold=100000 size=20 flags=sessions,sort-total,order-descending,ignore-
cache where="time=\"2017-02-13 16:50:00\"-\"2017-02-20 16:49:59\""
```

From these logs, we can extrapolate that there are four API calls going on here:

- SDK-Summary
- SDK-Language
- SDK-Session
- SDK-Values
- SDK-Aliases

The 'flags' Parameter

Before we delve into the calls any further, it must be said that many API calls either permit or require several parameters to be included when making the call. One of these parameters is '**flags**'. The flags parameter impacts how the API call behaves, and it may impact the results. The valid values for **flags** depends on the API call being used. They are not well-documented, but if we consult the C header files included in the sdk RPM file, we can see which values are valid.

Let's take the flags for the **sdk-values** call, which we can see options in NwSDK.h under "**enum NwValuesFlags**":

NW_VALUES_FLAG_TOTAL_SESSIONS = 0x0001,
NW_VALUES_FLAG_TOTAL_SIZE = 0x0002,
NW_VALUES_FLAG_TOTAL_PACKETS = 0x0004,
NW_VALUES_FLAG_TOTAL_AGGREGATE = 0x0010, /**< Use the aggregate function parameter to produce totals */
NW_VALUES_FLAG_SORT_TOTAL = 0x0100,
NW_VALUES_FLAG_SORT_VALUE = 0x0200,
NW_VALUES_FLAG_ORDER_ASCENDING = 0x0400,
NW_VALUES_FLAG_ORDER_DESCENDING = 0x0800,
NW_VALUES_FLAG_IGNORE_CACHE = 0x1000, /**< Ignore any results in query cache and requery, affects only current query */
NW_VALUES_FLAG_CLEAR_CACHE = 0x2000, /**< Clear the cache, forcing it to rebuild with latest values, affects all users */
NW_VALUES_FLAG_QUERY_LOG = 0x4000, /**< Return a query log for debugging */
NW_VALUES_FLAG_DATABASE_SCAN = 0x8000, /**< Skip the index and scan the database, very slow */
NW_VALUES_FLAG_ESTIMATES = 0x10000

In the above, we can see that there are options for how the sessions are totaled, how they are sorted, which order they are sorted in, and miscellaneous parameters concerning caching, logging, and where we scan for the values. By adding the desired values together, we arrive at a value we can then pass in with the **flags** parameter.

We have included as reference all the valid flags for each SDK call in this lesson.

Making the Right Call

API Call (on /sdk node)	Purpose
summary	Tells the caller what the valid range of times, session ID's, and meta ID's are. This is often the starting point of querying NWDB data
language	Tells the caller what meta keys are available for querying, and their attributes
session	If you have one or a range of sessions ID's, it will tell you what their associated meta ID's are.
values	Returns meta values for a single selected meta key. Can be combined with where clause to narrow results.
query	Returns values in one or more selected meta keys from sessions matching on a SQL-like query statement, containing SELECT and WHERE clauses. Results can be aggregated as well. This is used by the Reporting Engine's NWDB rules.
aliases	Returns value aliases for a selected meta key. I.e. service=http is really service=80.
timeline	Returns discrete time slices with a count of the number of sessions or packets or the size within the selected time window.
deviceId	Identifies upstream service name for a given session and its associated upstream session ID.
msearch	Searches raw packet/log data and/or meta data for a value and returns a their session ID's and optionally text before and after the match, to add context. It also supports regex.

content	Returns either:
	• extracted files contained within sessions
	• raw pcap file from a session
	• raw log file from a session
	• HTML-rendered reconstruction pages of either:
	  ◦ best reconstruction
	  ◦ meta
	  ◦ HTML
	  ◦ mail
	  ◦ hex
	  ◦ packets
	  ◦ text

packets	Returns a file containing raw pcap or raw log data from a range of session ID's, packet ID's, a where clause, or a time range. It also can return logs in custom-rendered formats: CSV, XML, and JSON.

SDK-Summary

The **sdk-summary** call is often the starting point in obtaining values from SA Core. Its job is to inform the caller (us) about what ranges of meta, sessions, packets, and time are available for querying with other API calls. Given the following output from an sdk-summary call which we've re-used from a previous example, let's break down the output into its constituent parts:

Valid sdk-summary Parameters

flags (mandatory)	Bitwise Mask: NW_INFO_FLAG_DEFAULT = 0, /**< The default behavior of NwInfo. */ NW_INFO_FLAG_IGNORE_CACHE_WINDOW = 1 /**< Always get true session range (ignore cache window). */

Query: /sdk?msg=summary&force-content-type=text/plain

```
$ curl "http://admin:netwitness@concentrator:50105/sdk?msg=summary&force-content-type=text/plain"

mid1=1 mid2=439622619 msize=8946156568 mmax=45515915919 pid1=0 pid2=0 psize=0 pmax=0 time1=1484858460 time2=1487629454 ptime1=0
ptime2=0 sid1=1 sid2=11046398 ssize=375623924 smax=45515915919 stotalsize=122017133761 isize=429920096 memt=8254554112
memu=1575022592 memp=154034176 hostname=concentrator version=10.5.3.0.5372
```

SDK-summary Results

mid1	The first (lowest) meta DB ID which is available from the service
mid2	The last (highest) meta DB ID which is available
msize	The size of the meta DB, in bytes
mmax	The maximum possible size of the meta DB, in bytes
pid1	The first (lowest) packet DB ID which is available from the service
pid2	The last (highest) packet DB ID which is available from the service
psize	The size of the packet DB, in bytes
pmax	The maximum possible size of the packet DB, in bytes
time1	The time of the earliest session which has an entry in the index, in **epoch** time. In this case, 1484858460 converts to 19 Jan 2017 20:41:00 GMT
time2	The time of the latest session which has an entry in the index, in **epoch** time. In this case, 1487629454 converts to 20 Feb 2017 22:24:14 GMT
ptime1	The time of the oldest packet in the packet DB, in **epoch** time. As our query was against a concentrator and there is no packet DB, this is 0
ptime2	The time of the newest packet in the packet DB, in **epoch** time. As our query was against a concentrator and there is no packet DB, this is 0
sid1	The first (lowest) session DB ID which is available
sid2	The last (highest) session DB ID which is available
ssize	The size of the session DB, in bytes
smax	The maximum possible size of the session DB, in bytes
stotalsize	The size of the index summary database
isize	Memory used
memt	Memory total
memu	System memory
memp	Process memory
hostname	The host's hostname
version	The service version

Top Tip: **Epoch time** is a UNIX-y way of representing time. It is the number of seconds since 00:00:00 UTC on 1 January, 1970, not counting leap seconds. For a quick conversion from epoch to Gregorian calendar time, http://www.epochconverter.com is handy.

Using the above information, we can construct a fairly good picture as to what data is available for querying. We know the start and end time of the data, both in the index and in the packet db. This lets us know what the largest time range is that we can focus our queries on. We also know the first and last ID's in the meta, session, and packet databases, which we can use in subsequent API calls.

SDK-Language

The **sdk-language** call tells us what meta keys are available to us, given a few parameters.

Valid sdk-language Parameters

id1 (optional)	The first language ID
id2 (optional)	The last language ID
time1 (optional)	The starting time in **epoch** format, or as a string in the format "YYYY-MMM-DD HH:MM:S"
time2 (optional)	The ending time in **epoch** format, or as a string in the format "YYYY-MMM-DD HH:MM:S"
options (optional)	Unused
size (mandatory)	The number of results to return. Must be between 1 and 1677721
flags (optional)	NW_LANGUAGE_FLAG_DEFAULT = 0, /**< No value counting is done. Time1 and time2 parameters are ignored. */ NW_LANGUAGE_FLAG_NO_COUNT = 1, /**< Same as default. */ NW_LANGUAGE_FLAG_QUICK_COUNT = 2, /**< Returns the count of values from the most recent non-empty time slice only. Slices of index between time1 and time2 are considered. */ NW_LANGUAGE_FLAG_FULL_COUNT = 4 /**< Scan the time slices between time1 and time2 and return the unique value counts between those times in the results. */

Query: /sdk?msg=language&size=10&flags=4&id1=1&force-content-type=text/plain

```
$ curl "http://admin:netwitness@concentrator:50105/sdk?msg=language&size=10&flags=4&id1=1&force-content-type=text/plain"

[id1=11  id2=147
id1=1  id2=11046398  count=2032  format=32  value=Time  type=time  flags=2147484691  group=1
id1=1  id2=11046398  count=0  format=8  value=Remote Session ID  type=rid  flags=2147483906  group=2
id1=0  id2=0  count=0  format=65  value=Concentrator Source  type=cid  flags=2147484675  group=3
id1=1  id2=11046398  count=1  format=65  value=Decoder Source  type=did  flags=2147484691  group=4
id1=0  id2=0  count=0  format=65  value=Alerts  type=alert  flags=2147484163  group=5
id1=0  id2=0  count=64  format=65  value=Risk: Informational  type=risk.info  flags=2147484163  group=6
id1=0  id2=0  count=18  format=65  value=Risk: Suspicious  type=risk.suspicious  flags=2147484163  group=7
id1=0  id2=0  count=2  format=65  value=Risk: Warning  type=risk.warning  flags=2147484163  group=8
id1=0  id2=0  count=0  format=65  value=Feed Name  type=feed.name  flags=2147484418  group=9
id1=0  id2=0  count=0  format=65  value=Feed Category  type=feed.category  flags=2147484418  group=10
]
```

SDK-Language Results

Parsing the output of above, we can see which keys are available to us in subsequent queries.

id1	The first session ID that the key is available in. This will be listed as zero unless flags=2 or flags=4 (see **flags** parameter, in "Valid sdk-language Parameters" section)
id2	The last session ID that the key is available in. This will be listed as zero unless flags=2 or flags=4 (see **flags** parameter, in "Valid sdk-language Parameters" section)
count	The number of unique values stored in the index for the key
format	Refers to the format of the meta key, as defined by **enum NwVariantFormat** in **NwSdkTypes.h**: 1 = Int8 2 = UInt8 3 = Int16 4 = UInt16 5 = Int32 6 = UInt32 7 = Int64 8 = UInt64 9 = UInt128 10 = Float32 11 = Float64 32 = TimeT 64 = Binary 65 = Text 128 = nwIPv4 129 = nwIPv6 130 = nwMAC
value	Friendly name of meta key
type	Real name of meta key
flags	This is a bitwise mask, in integer format. This **flags** result has nothing to do with the flags parameter we passed into the sdk-language call. If we convert the value of **flags** to hex, we can then decode it using the below flags.

Defined by **enum NwLanguageResultFlags** in NwSDK.h:

NW_LANGUAGE_KEY_INDEX_MASK = 0x000F, /**< Mask for the index level. */

NW_LANGUAGE_KEY_INDEX_FILTER = 0, /**< Indicates the key is filtered, so data related to the key should be ignored. */
NW_LANGUAGE_KEY_INDEX_NONE = 1, /**< Indicates the data is kept but not indexed */
NW_LANGUAGE_KEY_INDEX_KEYS = 2, /**< Indicates the data is indexed only at the key level. */
NW_LANGUAGE_KEY_INDEX_VALUES = 3, /**< Indicates the data is indexed at key and value levels. */

****

NW_LANGUAGE_KEY_SPECIAL_MASK = 0x00F0, /**< Special flags about each language key */

NW_LANGUAGE_KEY_SPECIAL_SINGLETON = 0x0010, /**< Token appears only once per session */
NW_LANGUAGE_KEY_SPECIAL_VALUE_ALIASES = 0x0040, /**< Indicates this key has value aliases */
NW_LANGUAGE_KEY_SPECIAL_TRANSFORM = 0x0080, /**< Indicates this key has a transform */

****

NW_LANGUAGE_KEY_ACTION_MASK = 0x0F00, /**< Mask indicating default behavior in Investigator */

NW_LANGUAGE_KEY_ACTION_HIDDEN = 0x0100, /**< Indicates the default is to hide the key, user configurable from index-*.xml file. */
NW_LANGUAGE_KEY_ACTION_OPEN = 0x0200, /**< Indicates the default is to open the key, user configurable from index-*.xml file. */
NW_LANGUAGE_KEY_ACTION_CLOSE = 0x0300, /**< Indicates the default is to close the key, user configurable from index-*.xml file. */
NW_LANGUAGE_KEY_ACTION_AUTO = 0x0400, /**< Indicates the default is to auto open the key, user configurable from index-*.xml file. */

****

NW_LANGUAGE_KEY_PROTECTED_VALUE = 0x10000, /**< Values of this type should be treated as protected. */
NW_LANGUAGE_KEY_PROTECTED_TOKEN = 0x20000, /**< Indicates that the key is intended to contain tokenized values. */

****

NW_LANGUAGE_KEY_TRANSIENT = 0x800000, /**< Indicates that values for this key are not persisted and only exist during parsing on a decoder. */

****

NW_LANGUAGE_KEY_RESULT = 0x80000000 /**< Indicates this result is a response to a language call */

Example: given the following result: "id1=0 id2=0 count=64 format=65 value=Risk: Informational type=risk.info flags=2147484163 group=6"

A '**flags**' value of **2147484163** translates to **0x80000203** in hex. We can do this with a programming calculator by typing in the number in base 10, and then changing the display to base 16. You can also use this simple online converter at http://www.includehelp.com/programmer-calculator.aspx

	Using bitwise AND logic (you can learn about this here and here), we can unmask the values in each digit position, and what the parameters of the 'risk.info' key are: • **0x80000000** = This is the result of a language call (duh! we know this already) • **0x200** = The default setting of the key is to open the key • **0x3** = The key is indexed at the value level
group	Shows the key's position in the list of language keys

SDK-Session

SDK-Session returns the meta DB ID range for a given session DB ID range. This is useful when you have a range of session ID's, and you want to extract all of the meta from those sessions. This tells you which meta ID's to pull.

Valid SDK-session Parameters

id1 (mandatory)	The beginning session ID range
id2 (mandatory)	The end of the session ID range

Query: /sdk?msg=session&force-content-type=text/plain&expiry=600&id1=11185003&id2=11185004

```
$ curl "http://admin:netwitness@concentrator:50105/sdk?msg=session&force-content-
type=text/plain&expiry=600&id1=11185003&id2=11185004"

id1: 11185003
id2: 11185004
field1: 445439059
field2: 445439122
```

SDK-session Results

id1	The first session ID returned
id2	The last session ID returned
field1	The beginning meta ID for the starting session ID, id1
field2	The last meta ID for the ending session ID, id2

SDK-Values Call

The **SDK-Values** call returns a sorted list of meta values for a given session ID range.

Valid SDK-values Parameters

id1 (optional)	The starting meta id
id2 (optional)	The ending meta id
size (mandatory)	The number of unique meta values to return

flags (optional)	Defined by **enum NwValuesFlags** in **NwSDK.h**: NW_VALUES_FLAG_TOTAL_SESSIONS = 0x0001, NW_VALUES_FLAG_TOTAL_SIZE = 0x0002, NW_VALUES_FLAG_TOTAL_PACKETS = 0x0004, NW_VALUES_FLAG_TOTAL_AGGREGATE = 0x0010, /**< Use the aggregate function parameter to produce totals */ NW_VALUES_FLAG_SORT_TOTAL = 0x0100, NW_VALUES_FLAG_SORT_VALUE = 0x0200, NW_VALUES_FLAG_ORDER_ASCENDING = 0x0400, NW_VALUES_FLAG_ORDER_DESCENDING = 0x0800, NW_VALUES_FLAG_IGNORE_CACHE = 0x1000, /**< Ignore any results in query cache and requery, affects only current query */ NW_VALUES_FLAG_CLEAR_CACHE = 0x2000, /**< Clear the cache, forcing it to rebuild with latest values, affects all users */ NW_VALUES_FLAG_QUERY_LOG = 0x4000, /**< Return a query log for debugging */ NW_VALUES_FLAG_DATABASE_SCAN = 0x8000, /**< Skip the index and scan the database, very slow */ NW_VALUES_FLAG_ESTIMATES = 0x10000
threshold (optional)	Session threshold. Stops searching for new values after this number of sessions have been searched
fieldname (mandatory)	Which meta key to return meta values for
where (optional)	Query data to narrow the search results. This includes query language (from Investigation or Reporting), and also the timeframe (see below example).
aggregateFunction (optional)	If the **aggregate** flag is set, valid values are **count** and **sum** (defaults to count). This returns
aggregateFieldName (optional)	If using **aggregateFunction=count**, which key to aggregate on, or add the values together of. This should be a meta key of an integer type, usually either **size** or **packets.**
min (optional)	The lower limit of values to return. Only values greater than this value will be returned.
max (optional)	The upper limit of values to return. Only values less than this value will be returned.

Query:

/sdk?msg=values&force-content-type=text/plain&expiry=600&fieldName=eth.src&id1=1&id2=442211481&threshold=100000&size=20&flags=flags=sessions,sort-total,order-descending,ignore-cache&where=time="2017-02-13 16:50:00"-"2017-02-20 16:49:59"

```
$ curl "http://admin:netwitness@concentrator:50105/sdk?msg=values&force-content-type=text/plain&expiry=600&fieldName=eth.src&id1=1&id2=442211481&threshold=100000&size=20&flags=flags%3Dsessions%2Csort-total%2Corder-descending%2Cignore-cache&where=time%3D%222017-02-13%2016%3A50%3A00%22-%222017-02-20%2016%3A49%3A59%22"

[id1=1  id2=442211481
id1=10532311  id2=10970663  count=829108  format=130  value=EC:F4:BB:4F:B0:96  type=eth.src  flags=1  group=100000
id1=10500748  id2=10853163  count=118064  format=130  value=EC:F4:BB:4F:B2:45  type=eth.src  flags=1  group=100000
id1=10500661  id2=11046084  count=43042  format=130  value=EC:F4:BB:4F:C0:D7  type=eth.src  flags=0  group=0
id1=10508822  id2=11045891  count=30555  format=130  value=00:60:6E:00:9D:F9  type=eth.src  flags=0  group=0
id1=10500662  id2=11045914  count=22416  format=130  value=78:54:2E:9F:10:28  type=eth.src  flags=0  group=0
id1=10500852  id2=11045447  count=231  format=130  value=00:60:6E:42:B6:35  type=eth.src  flags=0  group=0
id1=10532959  id2=11046194  count=85  format=130  value=00:0C:29:34:F8:B1  type=eth.src  flags=0  group=0
id1=10504949  id2=11007145  count=54  format=130  value=EC:F4:BB:4F:C0:7E  type=eth.src  flags=0  group=0
id1=10500774  id2=11037051  count=50  format=130  value=10:BF:48:BA:EF:B8  type=eth.src  flags=0  group=0
id1=10533403  id2=11046193  count=25  format=130  value=00:0C:29:C9:B5:93  type=eth.src  flags=0  group=0
id1=10594377  id2=11002932  count=21  format=130  value=00:0C:29:3C:F8:47  type=eth.src  flags=0  group=0
id1=10506457  id2=11036088  count=13  format=130  value=00:0C:29:59:05:93  type=eth.src  flags=0  group=0
id1=10532781  id2=11037063  count=12  format=130  value=00:0C:29:31:9D:73  type=eth.src  flags=0  group=0
id1=10500663  id2=11046395  count=9  format=130  value=2C:C2:60:22:D0:E6  type=eth.src  flags=0  group=0
id1=10532949  id2=11037053  count=8  format=130  value=00:0C:29:6D:28:E9  type=eth.src  flags=0  group=0
id1=11001598  id2=11001603  count=6  format=130  value=EC:F4:BB:51:89:EC  type=eth.src  flags=0  group=0
id1=10500777  id2=11011103  count=5  format=130  value=60:2A:D0:95:4B:1C  type=eth.src  flags=0  group=0
id1=10532947  id2=11037050  count=4  format=130  value=3C:97:0E:11:A9:AA  type=eth.src  flags=0  group=0
id1=10532298  id2=11036489  count=4  format=130  value=00:50:56:B8:B3:C1  type=eth.src  flags=0  group=0
id1=10532951  id2=11037062  count=4  format=130  value=00:0C:29:C9:5F:2A  type=eth.src  flags=0  group=0
]
```

SDK-values Results

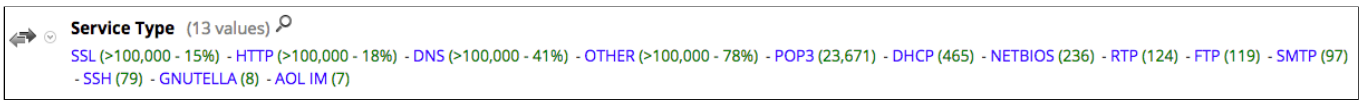
These are defined by struct **NwField** in **NwSdkTypes.h**

id1	The first meta ID where this value was seen
id2	The last meta ID where this value was seen
count	The number of results. This is by default the number of sessions containing this value, unless modified by the **aggregateFunction** and **aggregateFieldName** parameters.

format	Refers to the format of the meta key, as defined by enum **NwVariantFormat** in **NwSdkTypes.h:** 1 = Int8 2 = UInt8 3 = Int16 4 = UInt16 5 = Int32 6 = UInt32 7 = Int64 8 = UInt64 9 = UInt128 10 = Float32 11 = Float64 32 = TimeT 64 = Binary 65 = Text 128 = nwIPv4 129 = nwIPv6 130 = nwMAC
value	The meta value
type	The meta key name
flags	This is a bitwise mask, in integer format. This flags result has nothing to do with the flags parameter we passed into the sdk-values call. Defined by **enum NwFieldFlags** in NwSDK.h: NW_FIELD_FLAG_ESTIMATED_COUNT = 0x0001, /**< Threshold was reached or count was estimated */ NW_FIELD_FLAG_INVALID_UTF8 = 0x8000 /**< The original Text value was not valid UTF-8 and was modified to meet transfer encoding requirements. This is a transient flag and no changes were applied to the source data. */
group	From **NwSdkTypes.h**: The group number specifies a session id for the meta record and is used to group related pieces of meta together. All meta of the same group id are based from the same session.

SDK-Aliases

Returns any value aliases defined for a meta key, so that the alias can be displayed when rendering values for the user, as opposed to the original value. For instance, in Investigation, under the **Service Type** key, text values are displayed instead of the numeric value stored in the index (i.e **HTTP** instead of **80**). This corresponds to aliases defined in the **index-<*service*>.xml** and **index-<*service*>-custom.xml** files.



These Values Are Aliases For Real Values

Valid SDK-aliases Parameters

key (mandatory)	The meta key to retrieve aliases for
value (optional)	A specific value to return an alias for

Query: /sdk?msg=aliases&force-content-type=text/plain&expiry=600&key=service

```
$ curl "http://admin:netwitness@concentrator:50105/sdk?msg=aliases&force-content-type=text/plain&expiry=600&key=service"

0: alias=OTHER format=$alias
1000000: alias=KERNEL format=$alias
1000001: alias=USER format=$alias
1000003: alias=SYSTEM format=$alias
1000004: alias=AUTH format=$alias
1000005: alias=LOGGER format=$alias
1000006: alias=LPD format=$alias
1000008: alias=UUCP format=$alias
1000009: alias=SCHEDULE format=$alias
1000010: alias=SECURITY format=$alias
1000013: alias=AUDIT format=$alias
1000014: alias=ALERT format=$alias
1000015: alias=CLOCK format=$alias
1024: alias=EXCHANGE format=$alias
1080: alias=SOCKS format=$alias
110: alias=POP3 format=$alias
111: alias=SUNRPC format=$alias
1122: alias="MSN IM" format=$alias
119: alias=NNTP format=$alias
123: alias=NTP format=$alias
1344: alias=ICAP format=$alias
135: alias=RPC format=$alias
1352: alias=NOTES format=$alias
137: alias=NETBIOS format=$alias
139: alias=SMB format=$alias
143: alias=IMAP format=$alias
1433: alias=TDS format=$alias
1521: alias=TNS format=$alias
1533: alias=SAMETIME format=$alias
161: alias=SNMP format=$alias
1719: alias=H.323 format=$alias
1720: alias=RTP format=$alias
179: alias=BGP format=$alias
20: alias=FTPD format=$alias
2000: alias=SKINNY format=$alias
20000: alias=DNP3 format=$alias
2040: alias=SOULSEEK format=$alias
2049: alias=NFS format=$alias
21: alias=FTP format=$alias
22: alias=SSH format=$alias
23: alias=TELNET format=$alias
25: alias=SMTP format=$alias
3270: alias=TN3270 format=$alias
3389: alias=RDP format=$alias
3700: alias=DB2 format=$alias
443: alias=SSL format=$alias
502: alias=MODBUS format=$alias
5050: alias="YAHOO IM" format=$alias
5060: alias=SIP format=$alias
5190: alias="AOL IM" format=$alias
520: alias=RIP format=$alias
5222: alias="Google Talk" format=$alias
53: alias=DNS format=$alias
5900: alias=VNC format=$alias
6346: alias=GNUTELLA format=$alias
6667: alias=IRC format=$alias
67: alias=DHCP format=$alias
6801: alias=Net2Phone format=$alias
6881: alias=BITTORRENT format=$alias
69: alias=TFTP format=$alias
80: alias=HTTP format=$alias
8000: alias=QQ format=$alias
8002: alias=YCHAT format=$alias
8019: alias=WEBMAIL format=$alias
8082: alias=FIX format=$alias
```

SDK-aliases Results

Original Value	The meta value the alias applies to
alias=	The value after the equal sign is the aliased value to display instead of the original value
format=	This should always be $alias

Returning Meta With SDK-Query

The **SDK-query** call is useful for finding specific meta values that match a query statement. This is most commonly used by the Reporting Engine. It uses a SQL-like statement for its query parameter: "**SELECT * WHERE condition1 && condition2**". Constructing these statements can best be done from the Reporting Engine's rule builder. Watch the core service's logs as you execute a query, and paste the resulting query into the **query** parameter.

Valid SDK-query Parameters

id1	The starting meta ID to query. To make the query run from the most recent to oldest meta, give id1 a value greater than id2.
id2	The ending meta id to query.
size	The max number of values to return. If set to zero, returns all results.
query	SQL-like statement containing at least a SELECT clause, and optionally WHERE and THEN clauses.
flags	Defined by **enum NwQueryFlags** in **NwSDK.h:** NW_QUERY_FLAG_DEFAULT = 0x0000, /**< Default standard behavior */ NW_QUERY_FLAG_QUERY_LOG = 0x4000 /**< Return a query log for debugging */
threshold	Limits the number of sessions that the query will search to this value.

Query: /sdk?msg=query&force-content-type=text/plain&expiry=600&id1=1&id2=100000&threshold=10&query=select sessionid,alias.host WHERE service=443 && alias.host exists

```
$ curl "http://admin:netwitness@concentrator:50105/sdk?msg=query&force-content-
type=text/plain&expiry=600&id1=1&id2=100000&threshold=10&query=select%20sessionid%2Calias.host%20WHERE%20service%3D443%20%26%26%20al

[id1=16348  id2=16369
id1=16348  id2=16348  count=0  format=8  value=232  type=sessionid  flags=0  group=232
id1=16369  id2=16369  count=0  format=65  value=safebrowsing.google.com  type=alias.host  flags=0  group=232
]
[id1=16484  id2=16505
id1=16484  id2=16484  count=0  format=8  value=236  type=sessionid  flags=0  group=236
id1=16505  id2=16505  count=0  format=65  value=safebrowsing-cache.google.com  type=alias.host  flags=0  group=236
]
[id1=24992  id2=25021
id1=24992  id2=24992  count=0  format=8  value=493  type=sessionid  flags=0  group=493
id1=25013  id2=25013  count=0  format=65  value=safebrowsing.google.com  type=alias.host  flags=0  group=493
id1=25021  id2=25021  count=0  format=65  value=*.google.com  type=alias.host  flags=0  group=493
]
[id1=25135  id2=25164
id1=25135  id2=25135  count=0  format=8  value=497  type=sessionid  flags=0  group=497
id1=25156  id2=25156  count=0  format=65  value=safebrowsing-cache.google.com  type=alias.host  flags=0  group=497
id1=25164  id2=25164  count=0  format=65  value=*.google.com  type=alias.host  flags=0  group=497
]
[id1=100001  id2=100000
]
```

Top Tip: Notice how the results are grouped with square brackets. This is because the **size** parameter is set to 0, the default. Each group represents meta from an individual session. If you specify a non-zero value for the **size** parameter, only a single group will be returned by the query. I know of no reason why this is the case, but it is the case.

Query, with the size parameter specified: /sdk?msg=query&force-content-type=text/plain&expiry=600&id1=1&id2=1000000&threshold=100&size=20&query=SELECT sessionid,alias.host WHERE alias.host exists

```
$ curl "http://admin:netwitness@concentrator:50105/sdk?msg=query&force-content-
type=text/plain&expiry=600&id1=1&id2=1000000&threshold=100&size=20&query=SELECT%20sessionid%2Calias.host%20WHERE%20alias.host%20exist

[id1=298  id2=1000000
id1=32  id2=32  count=0  format=8  value=2  type=sessionid  flags=0  group=2
id1=50  id2=50  count=0  format=65  value=password-ned.hipchat.com  type=alias.host  flags=0  group=2
id1=55  id2=55  count=0  format=65  value=hipchat.com  type=alias.host  flags=0  group=2
id1=61  id2=61  count=0  format=65  value=hipchat.com  type=alias.host  flags=0  group=2
id1=66  id2=66  count=0  format=8  value=3  type=sessionid  flags=0  group=3
id1=84  id2=84  count=0  format=65  value=password-ned.hipchat.com  type=alias.host  flags=0  group=3
id1=93  id2=93  count=0  format=65  value=hipchat.com  type=alias.host  flags=0  group=3
id1=131  id2=131  count=0  format=8  value=5  type=sessionid  flags=0  group=5
id1=149  id2=149  count=0  format=65  value=password-ned.hipchat.com  type=alias.host  flags=0  group=5
id1=154  id2=154  count=0  format=65  value=hipchat.com  type=alias.host  flags=0  group=5
id1=160  id2=160  count=0  format=65  value=hipchat.com  type=alias.host  flags=0  group=5
id1=165  id2=165  count=0  format=8  value=6  type=sessionid  flags=0  group=6
id1=183  id2=183  count=0  format=65  value=password-ned.hipchat.com  type=alias.host  flags=0  group=6
id1=192  id2=192  count=0  format=65  value=hipchat.com  type=alias.host  flags=0  group=6
id1=245  id2=245  count=0  format=8  value=9  type=sessionid  flags=0  group=9
id1=263  id2=263  count=0  format=65  value=password-ned.hipchat.com  type=alias.host  flags=0  group=9
id1=268  id2=268  count=0  format=65  value=hipchat.com  type=alias.host  flags=0  group=9
id1=274  id2=274  count=0  format=65  value=hipchat.com  type=alias.host  flags=0  group=9
id1=279  id2=279  count=0  format=8  value=10  type=sessionid  flags=0  group=10
id1=297  id2=297  count=0  format=65  value=password-ned.hipchat.com  type=alias.host  flags=0  group=10
]
```

A single group is returned.

SDK-query Results

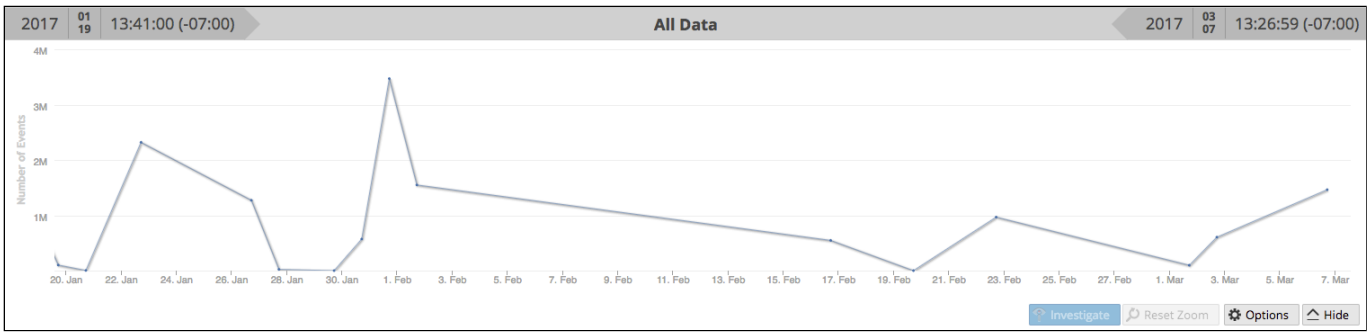
These are defined by struct **NwField** in **NwSdkTypes.h.**

id1	The starting meta ID for the value.
id2	The ending meta ID for the value. Since we are dealing with individual meta values, this should be the same value as id1.
count	Should always be 0
format	Refers to the format of the meta key, as defined by enum **NwVariantFormat** in **NwSdkTypes.h:** 1 = Int8 2 = UInt8 3 = Int16 4 = UInt16 5 = Int32 6 = UInt32 7 = Int64 8 = UInt64 9 = UInt128 10 = Float32 11 = Float64 32 = TimeT 64 = Binary 65 = Text 128 = nwIPv4 129 = nwIPv6 130 = nwMAC
value	The actual meta value returned by the query
type	The selected meta key

flags	Defined by **enum NwQueryResultFlags** in **NwSDK.h:** NW_QUERY_FLAG_AGGREGATE_SUM = 0x0001, /**< This result field is a sum() aggregate function */ NW_QUERY_FLAG_AGGREGATE_COUNT = 0x0002, /**< This result field is a count() aggregate function */ NW_QUERY_FLAG_AGGREGATE_MIN = 0x0003, /**< This result field is a min() aggregate function */ NW_QUERY_FLAG_AGGREGATE_MAX = 0x0004, /**< This result field is a max() aggregate function */ NW_QUERY_FLAG_AGGREGATE_AVG = 0x0005, /**< This result field is a avg() aggregate function */ NW_QUERY_FLAG_AGGREGATE_DISTINCT = 0x0006, /**< This result field is a distinct() aggregate function */ NW_QUERY_FLAG_AGGREGATE_FIRST = 0x0007, /**< This result field is a first() aggregate function */ NW_QUERY_FLAG_AGGREGATE_LAST = 0x0008, /**< This result field is a last() aggregate function */ NW_QUERY_FLAG_AGGREGATE_LEN = 0x0009, /**< This result field is a len() aggregate function */ NW_QUERY_FLAG_AGGREGATE_COUNT_DISTINCT = 0x000A, /**< This result field is a count_distinct() aggregate function */ NW_QUERY_FLAG_AGGREGATE_NOTHING = 0x000B, /**< This result filed is a nothing() aggregate function (will never appear in a result set) */ NW_QUERY_FLAG_AGGREGATE_MASK = 0x000F, /**< This mask indicates the result is a query aggregate function defined above */ NW_QUERY_INVALID_UTF8 = 0x8000 /**< The original Text value was not valid UTF-8 and was modified to meet transfer encoding requirements. This is a transient flag and no changes were applied to the source data. */
group	Corresponds to the first selected value returned by the query. In the case of our example, it's the session ID, since we selected that first in the query.

See Traffic Spikes With SDK-Timeline

The SDK-timeline call returns discrete time slices with a count of the number of sessions or packets or the size within that window. Coupled with graphing software, this can generate results like the timeline graph in Investigation.



Valid SDK-timeline Parameters

time1 (optional)	The starting time in epoch format or a time string in YYYY-MMM-DD HH:MM:SS format.
time2 (optional)	The ending time in epoch format or a time string in YYYY-MMM-DD HH:MM:SS format.
timezone (integer, optional)	The timezone to render the data in, hours from UTC. Valid values are **-13** to **13**. Defaults to UTC.
size (mandatory)	The number of time slices to return. Valid values are from 1 to 1677721.
flags	Defined by **enum NwValuesFlags** in **NwSDK.h**: NW_VALUES_FLAG_TOTAL_SESSIONS = 0x0001, NW_VALUES_FLAG_TOTAL_SIZE = 0x0002, NW_VALUES_FLAG_TOTAL_PACKETS = 0x0004, NW_VALUES_FLAG_TOTAL_AGGREGATE = 0x0010, /**< Use the aggregate function parameter to produce totals */ NW_VALUES_FLAG_SORT_TOTAL = 0x0100, NW_VALUES_FLAG_SORT_VALUE = 0x0200, NW_VALUES_FLAG_ORDER_ASCENDING = 0x0400, NW_VALUES_FLAG_ORDER_DESCENDING = 0x0800, NW_VALUES_FLAG_IGNORE_CACHE = 0x1000, /**< Ignore any results in query cache and requery, affects only current query */ NW_VALUES_FLAG_CLEAR_CACHE = 0x2000, /**< Clear the cache, forcing it to rebuild with latest values, affects all users */ NW_VALUES_FLAG_QUERY_LOG = 0x4000, /**< Return a query log for debugging */ NW_VALUES_FLAG_DATABASE_SCAN = 0x8000, /**< Skip the index and scan the database, very slow */ NW_VALUES_FLAG_ESTIMATES = 0x10000
where (optional)	Optional WHERE clause to filter the data.

bookendHours	The number of hours outside the given time1, time2 which will become part of the cache. Default is 24 hours.

Query: /sdk?msg=timeline&force-content-type=text/plain&expiry=600&size=24&time1=2017-Jan-19 00:00:00&time2=2017-Jan-20 00:00:00"

```
$ curl "http://admin:netwitness@concentrator:50105/sdk?msg=timeline&force-content-type=text/plain&expiry=600&size=24&time1=2017-Jan-
19%2000%3A00%3A00&time2=2017-Jan-20%2000%3A00%3A00"

[id1=1484858460 id2=1484859840
id1=1 id2=1085 count=1085 format=32 value=2017-Jan-19 20:41:00 type=minute flags=0 group=0
id1=1086 id2=9310 count=8225 format=32 value=2017-Jan-19 20:42:00 type=minute flags=0 group=0
id1=9311 id2=11226 count=1916 format=32 value=2017-Jan-19 20:43:00 type=minute flags=0 group=0
id1=11227 id2=16621 count=5395 format=32 value=2017-Jan-19 20:44:00 type=minute flags=0 group=0
id1=16622 id2=20335 count=3714 format=32 value=2017-Jan-19 20:45:00 type=minute flags=0 group=0
id1=20336 id2=24590 count=4255 format=32 value=2017-Jan-19 20:46:00 type=minute flags=0 group=0
id1=24591 id2=34004 count=9414 format=32 value=2017-Jan-19 20:47:00 type=minute flags=0 group=0
id1=34005 id2=40479 count=6475 format=32 value=2017-Jan-19 20:48:00 type=minute flags=0 group=0
id1=40480 id2=41069 count=590 format=32 value=2017-Jan-19 20:49:00 type=minute flags=0 group=0
id1=41070 id2=41972 count=903 format=32 value=2017-Jan-19 20:50:00 type=minute flags=0 group=0
id1=41973 id2=54441 count=12469 format=32 value=2017-Jan-19 20:51:00 type=minute flags=0 group=0
id1=54442 id2=64752 count=10311 format=32 value=2017-Jan-19 20:52:00 type=minute flags=0 group=0
id1=64753 id2=73726 count=8974 format=32 value=2017-Jan-19 20:53:00 type=minute flags=0 group=0
id1=73727 id2=81687 count=7961 format=32 value=2017-Jan-19 20:54:00 type=minute flags=0 group=0
id1=81688 id2=82633 count=946 format=32 value=2017-Jan-19 20:55:00 type=minute flags=0 group=0
id1=82634 id2=114125 count=31492 format=32 value=2017-Jan-19 20:56:00 type=minute flags=0 group=0
id1=114126 id2=126776 count=12651 format=32 value=2017-Jan-19 20:57:00 type=minute flags=0 group=0
id1=126777 id2=140188 count=13412 format=32 value=2017-Jan-19 20:58:00 type=minute flags=0 group=0
id1=140189 id2=151735 count=11547 format=32 value=2017-Jan-19 20:59:00 type=minute flags=0 group=0
id1=151736 id2=164389 count=12654 format=32 value=2017-Jan-19 21:00:00 type=minute flags=0 group=0
id1=164390 id2=177116 count=12727 format=32 value=2017-Jan-19 21:01:00 type=minute flags=0 group=0
id1=177117 id2=189534 count=12418 format=32 value=2017-Jan-19 21:02:00 type=minute flags=0 group=0
id1=189535 id2=189673 count=139 format=32 value=2017-Jan-19 21:03:00 type=minute flags=0 group=0
id1=189674 id2=189886 count=213 format=32 value=2017-Jan-19 21:04:00 type=minute flags=0 group=0
]
```

SDK-timeline Results

id1	The first session ID of the time window. If greater than time2, the range for the entire index is returned.
id2	The last session ID of the time window.
count	The number of sessions in the time window.
format	Should always be 32, which is TimeT.
value	The starting/ending time of the timerange
type	minute, hour, or day. You unfortunately can't choose this manually. It's picked automatically by the timeline function based on the length of time chosen.
flags	This is a bitwise mask, in integer format. This flags result has nothing to do with the flags parameter we passed into the sdk-values call. Defined by **enum NwFieldFlags** in NwSDK.h: NW_FIELD_FLAG_ESTIMATED_COUNT = 0x0001, /**< Threshold was reached or count was estimated */ NW_FIELD_FLAG_INVALID_UTF8 = 0x8000 /**< The original Text value was not valid UTF-8 and was modified to meet transfer encoding requirements. This is a transient flag and no changes were applied to the source data. */
group	Not meaningful in this context.

Find Upstream Device and Session ID with SDK-deviceId

If you're looking at a multi-source aggregating device like a Broker, you may need to know where a particular session came from, and what its session ID is on the upstream service. **SDK-deviceId will take care of this for you.**

Valid SDK-deviceId Parameters

session	The session ID from which to pull the upstream device ID and session ID information.

Query: /sdk?msg=deviceId&force-content-type=text/plain&expiry=600&session=11258939"

```
$ curl "http://admin:netwitness@sa:50103/sdk?msg=deviceId&force-content-type=text/plain&expiry=600&session=11258939"

device: concentrator.ktalab.local:50005
session: 11088872
```

SDK-deviceId Results

device	The name and port of the upstream device.
session	The upstream session ID for the session ID entered.

Search Content With SDK-msearch

It's possible to search for strings contained within both session data and meta data with the **SDK-msearch** call. It may be very slow to return results, depending on the size of the databases being searched.

Valid SDK-msearch Parameters

sessions (optional)	A range of session to search
packets (optional)	A range of packet ID's
search (mandatory)	String to search for (can be regex if **regex** flag is set. Each word specified is logically AND'ed together. Both words must appear in the session, but not necessarily together For example: If I specify search="Tim Underhay", both "Tim" and "Underhay" must both be found in the session. The text "Tim's last name is Underhay" would match this. You can exclude words using the minus (-) operator. I.e. "star -wars". Results would be returned if "star" is found and "wars" isn't.
where (optional)	Optional WHERE query string to filter the result set before searching
limit (optional)	The maximum number of sessions to search
flags	A comma separated list defining search behaviour: **regex** - Regular expression search **sp** - Search Packets **sm** - Search meta **ci** - Case insensitive **pre** - Pretext. Returns the characters before the search hit, to provide context to the data found. **post** - Posttext. Returns the characters after the search hit, to provide context to the data found. **ds** - Decode sessions (i.e. convert/decode email attachments before searching them) **precache** - Undocumented flag to speed up searching packet DB

Query: /sdk?msg=msearch&force-content-type=text/plain&expiry=600&search=pwned&where=time="2017-02-22 16:48:00"-"2017-03-01 16:47:59" && service=25&flags=sm,ci

```
$ curl "http://admin:netwitness@concentrator:50105/sdk?msg=msearch&force-content-
type=text/plain&expiry=600&search=pwned&where=time%3D%222017-02-22%2016%3A48%3A00%22-%222017-03-
01%2016%3A47%3A59%22%20%26%26%20service%3D25&flags=sm%2Cci"

[id1=11087826 id2=11087826
id1=11087826 id2=11087826 count=0 format=65 value=pwned type=hit flags=0 group=11087826
]
[id1=11088872 id2=11088872
id1=11088872 id2=11088872 count=0 format=65 value=pwned type=hit flags=0 group=11088872
]
[id1=11104094 id2=11104094
id1=11104094 id2=11104094 count=0 format=65 value=pwned type=hit flags=0 group=11104094
]
[id1=11104103 id2=11104103
id1=11104103 id2=11104103 count=0 format=65 value=pwned type=hit flags=0 group=11104103
]
[id1=11104408 id2=11104408
id1=11104408 id2=11104408 count=0 format=65 value=pwned type=hit flags=0 group=11104408
]
[id1=11104471 id2=11104471
id1=11104471 id2=11104471 count=0 format=65 value=pwned type=hit flags=0 group=11104471
]
[id1=11104512 id2=11104512
id1=11104512 id2=11104512 count=0 format=65 value=pwned type=hit flags=0 group=11104512
]
[id1=11120568 id2=11120568
id1=11120568 id2=11120568 count=0 format=65 value=pwned type=hit flags=0 group=11120568
]
[id1=11120605 id2=11120605
id1=11120605 id2=11120605 count=0 format=65 value=pwned type=hit flags=0 group=11120605
]
[id1=11131563 id2=11131563
id1=11131563 id2=11131563 count=0 format=65 value=pwned type=hit flags=0 group=11131563
]
[id1=11132216 id2=11132216
id1=11132216 id2=11132216 count=0 format=65 value=pwned type=hit flags=0 group=11132216
]
```

SDK-msearch Results

id1	Low session ID.
id2	High session ID. Should be same as id1.
count	Appears to always be zero
format	Refers to the format of the meta key, as defined by enum **NwVariantFormat** in **NwSdkTypes.h**: 1 = Int8 2 = UInt8 3 = Int16 4 = UInt16 5 = Int32 6 = UInt32 7 = Int64 8 = UInt64 9 = UInt128 10 = Float32 11 = Float64 32 = TimeT 64 = Binary 65 = Text 128 = nwIPv4 129 = nwIPv6 130 = nwMAC
value	What was matched
type	Should always be 'hit'
flags	The return flags for msearch are not documented.
group	Grouping identifier. Should be the same as the session ID

Event Reconstruction - SDK-Content Call

The **sdk-content** call is used by Event Reconstruction in Investigation to display the packets for a session. **This is the main call you can use to extract files and pcaps from an individual session.** Using this call with the REST API means the valid parameters are different than the valid parameters for the C API. It is invoked using "**/sdk/content**" rather than "/sdk?message=sdk". You CANNOT use the REST GUI to construct your URL as it will not display valid parameters and will display an invalid URI.

This call returns files rather than textual data. The http **content-type** returned depends on what is selected for the **render** parameter.

You can see valid parameters at **http://*<host>*:*<restport>*/sdk/content**. Below, we document valid REST call parameters,

REST Content Help

session	The session id that will be returned in the response.
render	The render type (defined in NwSDK.h) for content or one of the following render types: text, hex, packets, web, mail, raw, rtp, voip, meta, im, files or pcap
maxSize	The maximum size of the session returned (bytes), zero is unlimited. Default is zero.
base64	If 1, encode the multipart/mixed response as base64. Otherwise binary is the default.
flags	Optional flags for content rendering, see the SDK header file NwContent flags parameter for more info.
includeFileTypes	Semicolon separated list of file extensions that will be extracted (e.g. ".ext1;.ext2").
excludeFileTypes	Semicolon separated list of file extensions that will be excluded (e.g. ".ext1;.ext2").
renameFileTypes	Used to rename certain files from one or more extensions to another. For example: renameFileTypes=.download|.octet-stream|.program|.exe;.jpeg|.jpg For the above example, all files ending in .download, .stream or .program will be renamed to .exe All files ending in .jpeg will be renamed to .jpg

Command Examples:

Content Type	Query	Command	Mime Type
Web Auto Reconstruction (presented as HTML page)	/sdk/content?session=12019743&render=0	curl "http://admin:netwitness@concentrator:50105/sdk/content?session=12019743&render=0" > 0.html	text/html
Meta Details (presented as HTML page)	/sdk/content?session=12019743&render=1	curl "http://admin:netwitness@concentrator:50105/sdk/content?session=12019743&render=1" > 1.html	text/html
Mail Reconstruction (presented as HTML page)	/sdk/content?session=11251719&render=5"	curl "http://admin:netwitness@sa:50103/sdk/content?session=11251719&render=5" > 5-mail.html	text/html
PCAP of captured session	/sdk/content?session=12019743&render=100	curl "http://admin:netwitness@concentrator:50105/sdk/content?session=12019743&render=100" > 100.pcap	application/cap
RAW Session Payload	/sdk/content?session=12019743&render=102	curl "http://admin:netwitness@concentrator:50105/sdk/content?session=12019743&render=102" > 102.out	application/x-netwitness-raw
Limited Session Meta XML	/sdk/content?session=12019743&render=103	curl "http://admin:netwitness@concentrator:50105/sdk/content?session=12019743&render=103" > 103.xml	text/xml
Full Session Meta CSV	/sdk/content?session=12019743&render=104	curl "http://admin:netwitness@concentrator:50105/sdk/content?session=12019743&render=104" > 104.csv	text/csv
Full Session Meta Tab-Separated	/sdk/content?session=12019743&render=105	curl "http://admin:netwitness@concentrator:50105/sdk/content?session=12019743&render=105" > 105.txt	text/plain
NetWitness .NWD file (for thick Investigator client)	/sdk/content?session=12019743&render=106	curl "http://admin:netwitness@concentrator:50105/sdk/content?session=12019743&render=106" > 106.nwd	application/x-netwitness-nwd
Logfile	/sdk/content?session=3478&render=108	curl "http://admin:netwitness@logconcentrator:50105/sdk/content?session=3478&render=108" > 108.log	text/plain
Google Protobuf of NWD File (session, meta, packets)	/sdk/content?session=12019743&render=109	curl "http://admin:netwitness@concentrator:50105/sdk/content?session=12019743&render=109" > 109.protobuf	application/x-google-protobuf

Valid SDK-content Parameters

session (mandatory)	The single session ID to retrieve the packets from
render (integer, optional)	Defined by **enum NwContentTypes** in **NwSDK.h**: Values 0 through 9 return HTML web pages of the desired view which can be displayed in a web browser, with a content-type of text/html. Values greater than 100 and higher return files. NW_CONTENT_TYPE_AUTO = 0, /**< Auto Select HTML Content View (text/html) */ NW_CONTENT_TYPE_DETAILS = 1, /**< HTML Meta Details View (text/html) */ NW_CONTENT_TYPE_TEXT = 2, /**< HTML Text View (text/html) */ NW_CONTENT_TYPE_HEX = 3, /**< HTML Hex View (text/html)*/ NW_CONTENT_TYPE_PACKETS = 4, /**< HTML Packet View (text/html)*/ NW_CONTENT_TYPE_MAIL = 5, /**< HTML Mail View (text/html)*/ NW_CONTENT_TYPE_WEB = 6, /**< HTML Web Page View (text/html)*/ NW_CONTENT_TYPE_VOIP = 7, /**< HTML VOIP View (text/html)*/ NW_CONTENT_TYPE_IM = 8, /**< HTML IM View (text/html)*/ NW_CONTENT_TYPE_FILES = 9, /**< HTML Listing of all files found in session (text/html)*/ NW_CONTENT_TYPE_PCAP = 100, /**< Pcap Packet File (application/cap)*/ NW_CONTENT_TYPE_RAW = 102, /**< Raw Content (application/x-netwitness-raw)*/. This is raw session payload, and does not separate source and destination sides of the session, which can be modified with **flags** parameter NW_CONTENT_TYPE_XML = 103, /**< Meta XML File (text/xml)*/ This returns metadata from the session, but only a subset. NW_CONTENT_TYPE_CSV = 104, /**< Meta Comma Separated File (text/csv) */ Returns a CSV of the session's meta. This is a fuller meta set displayed in the Meta view of Session Reconstruction, but still omits certain fields like **did, cid,** and **rid**. If you want these as well, use **sdk-query** instead. NW_CONTENT_TYPE_TXT = 105, /**< Meta Tab Separated File (text/plain)*/ Same as CSV, but tab-separated. NW_CONTENT_TYPE_NWD = 106, /**< Netwitness Data File (application/x-netwitness-nwd)*/ Returns a NWD file which can be imported directly containing session, meta, and packet data into Thick Investigator client without needing to re-parse. NW_CONTENT_TYPE_FILE_EXTRACTOR = 107, /**< Extract files from common protocols */ **This returns a multipart MIME response of type 'multipart/mixed', one part for every file extracted from the session. You must use a client that can handle each response correctly (not curl)!** NW_CONTENT_TYPE_LOGS = 108, /**< Log extract (captured logs, LF delimited) (text/plain)*/ NW_CONTENT_TYPE_PROTOBUF = 109 /**< Content of the NWD as a Google Protocol Buffer object (application/x-google-protobuf)*/
maxSize (optional)	The maximum size of of the session returned. Zero means unlimited, and is the default.
base64 (integer, optional)	If set to 1, encode the response as base64. Binary is the default.
flags (integer, optional)	Defined by **enum NwContentFlags** in **NwSDK.h:** NW_CONTENT_FLAG_STREAM1 = 0x00001, /**< Return only request stream content. */ NW_CONTENT_FLAG_STREAM2 = 0x00002, /**< Return only response stream content. */ NW_CONTENT_FLAG_SINGLE_COLUMN = 0x00004, /**< Format generated web page as a single column with requests and responses interleaved. */ NW_CONTENT_FLAG_PACKET_PAYLOAD = 0x00008, /**< Include only session payload. */ NW_CONTENT_FLAG_DECODEAS_EBCDIC = 0x00010, /**< Convert session payload from EBCDIC to ASCII. */ NW_CONTENT_FLAG_DO_NOT_EMBED = 0x00020, /**< Do not embed application/ audio/video traffic into the generated web page. */ NW_CONTENT_FLAG_UNCOMPRESS_TEXT = 0x00040, /**< Unzip web content in text view. */ NW_CONTENT_FLAG_DECODE_SSL = 0x00080, /**< Attempt to decrypt SSL sessions if the encryption key is provided. */ NW_CONTENT_FLAG_STRIP_STYLE_TAGS = 0x00100, /**< Removes all <style> tags from the original html document. */ NW_CONTENT_FLAG_IGNORE_CACHE = 0x01000, /**< Ignore any content in cache and requery, affects only current request. */ NW_CONTENT_FLAG_NO_EMBEDDED_EXE = 0x02000, /**< Do not look for or extract hidden/embedded PE files when performing file extraction. */ /* New flags only used when returning google protocol buffers */ NW_CONTENT_FLAG_INCLUDE_DUPS = 0x04000, /**< Include packets otherwise removed by assembly */ NW_CONTENT_FLAG_INCLUDE_HEADERS = 0x08000, /**< Include packet header meta information */ NW_CONTENT_FLAG_CAPTURE_ORDER = 0x10000 /**< Do not assemble packets, return in capture order*/
includedFileTypes (optional)	Semicolon-separated list of file extensions that will be extracted (e.g. ".ext1;.ext2").
excludeFileTypess (optional)	Semicolon-separated list of file extensions that will be excluded (e.g. ".ext1;.ext2").

renameFileTypes (optional)	Semicolon-separated list which changes file extension of extracted files to be a new extension. The extensions are linked together by the pipe (|) character, and the last extension listed in a set of pipes is the destination file extension. **Example:** renameFileTypes=.jpeg|.jpg - Renames files with .jpeg file extension to have the .jpg extension renameFileTypes=.tar.gz|.tgz - Renames files with .tar.gz file extension to have the .tgz extension renameFileTypes=.a|.b|.c - Renames files with .a and .b extensions to have the .c extension.

SDK-content Results

Because the sdk-**content** call returns files, we've attached some of the output files here.

Best HTML Reconstruction as Web Page

Mail Reconstruction as Web Page

Meta Details Web Page

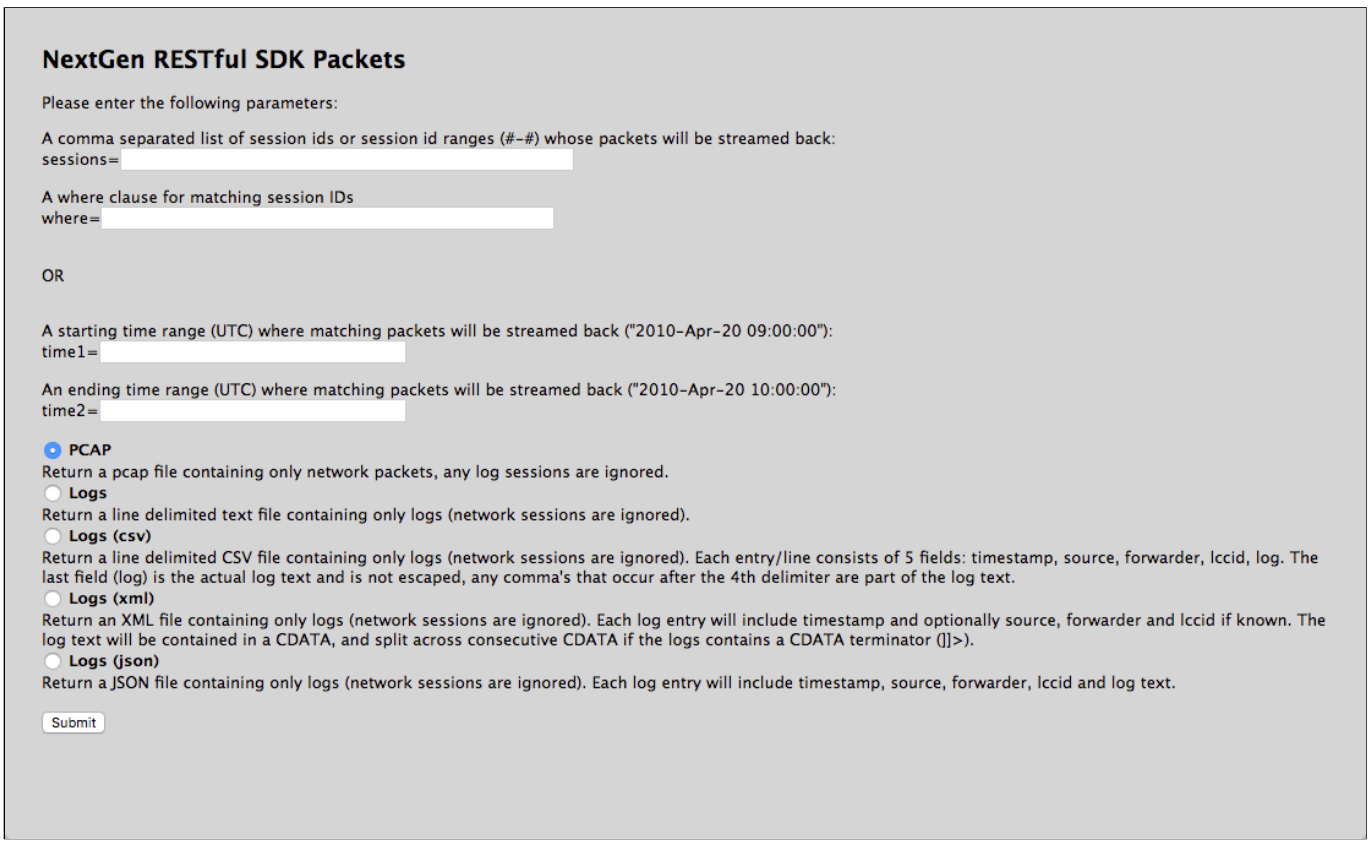
Meta CSV

Meta Tab Separated

Log File

Dump Packets to Disk With SDK-packets

The **SDK-packets** call returns raw pcap or log data. It can also format log data as CSV, XML, or JSON. Similar to the SDK-content call, the SDK-packets call is invoked using "**/sdk/packets**" rather than "/sdk?msg=packets". For this reason, you cannot use this directly from the Explorer page of the RESTful GUI. But if you navigate to http://<yourservice>:<restport>/sdk/packets, then you will be presented a custom page from which you can execute the command.



Valid SDK-packets Parameters

op (mandtory)	One of: **start**, **cancel**, or **processed.**

sessions (optional)	A range of session ID's to return
packets (optional)	A range of packet ID's to return
where (optional)	A where clause to narrow the result set.
time1 (optional)	A UTC starting time range that will be streamed back, in YYYY-MMM-DD HH:MM:SS format.
time2 (optional)	A UTC ending time range that will be streamed back, in YYYY-MMM-DD HH:MM:SS format.
flags (optional)	Defined by enum **NwPacketsFlags** in **NwSDK.h**: NW_PACKETS_FLAG_DEFAULT = 0x0000, /**< Default fastest option */ NW_PACKETS_FLAG_SESSION_ID = 0x0001, /**< Request the session id with each packet. */
devices (optional)	A comma-separated list of upstream service names (as defined in ***/<serviceType>/devices***) to query. If not specified, all upstream services are queried.
render	One of: **pcap** - Returns raw pcap data with content-type application/octet-stream **logs** - Returns raw log data with content-type text/plain **text/csv** - Returns log data formatted as CSV with content-type text/plain **text/xml** - Returns log data formatted as XML with no content-type set **application/json** - Returns log data formatted as JSON with content-type application/json

Command Examples:

Content-Type	Query	Command	Mime-T
RAW PCAP	/sdk/packets?op=start&where=service=25&render=pcap	curl "http://admin:netwitness@concentrator:50105/sdk/packets?op=start&where=service%3D25&render=pcap" > pcap.pcap	applicat stream
RAW logs	/sdk/packets?op=start&where=device.type=rsaflow&render=logs	curl "http://admin:netwitness@logconcentrator:50105/sdk/packets?op=start&where=device.type%3Drsaflow&render=logs" > log.log	text/plai
Logs formatted as CSV	/sdk/packets?op=start&where=device.type=rsaflow&render=text/csv	curl "http://admin:netwitness@logconcentrator:50105/sdk/packets?op=start&where=device.type%3Drsaflow&render=text%2Fcsv"	no type
Logs formatted as XML	/sdk/packets?op=start&where=device.type=rsaflow&render=text/xml	curl "http://admin:netwitness@logconcentrator:50105/sdk/packets?op=start&where=device.type%3Drsaflow&render=text%2Fxml" > log.xml	text/xml
Logs formatted as JSON	/sdk/packets?op=start&where=device.type=rsaflow&render=application/json	curl "http://admin:netwitness@logconcentrator:50105/sdk/packets?op=start&where=device.type%3Drsaflow&render=application%2Fjson" > log.json	applicat

SDK-packets Results

Because the sdk-packets call returns files, we've attached output files here.

Raw Log Data

Log Data Formatted as CSV

Log Data Formatted as XML

Log Data Formatted as JSON

Now that you know what these calls are, the next step is putting this knowledge to work in your own custom scripts and tools!

31. Working With NwConsole

NwConsole is a power user tool which allows one to interact with the NetWitness Explorer/API tree. It is a command shell for SA, allowing for both interactive and non-interactive commands. It allows one to:

- View the contents of the tree or one of its nodes
- Set a value for a configuration node
- Execute API calls
- Script actions on or against the tree
- Execute special built-in commands which are not available through the regular API.

NwConsole used to be available for Mac but unfortunately RSA has stopped publishing this version. There is an RPM available for CentOS, though, and it is installed by default on all SA appliances.

Top Tip: The legacy NetWitness 9.8.5.20 version for Macintosh will work with SA 10.5, though some of the root commands may not be available. You can download it from this link. **CAUTION**: **DO NOT UNDER ANY CIRCUMSTANCES** perform any database checks or repairs on a version 10.x database using a 9.8 version if NwConsole. You will almost certainly lose data!

```
[[root@concentrator ~]# NwConsole
RSA Security Analytics Console 10.5.3.0.5372
Copyright 2001-2016, RSA Security Inc.  All Rights Reserved.

[> help
Built-in Commands:
            ?: display this message
         help: display this message
        index: explore index data (advanced)
          sdk: execute SDK commands
      reindex: reindex a collection
      dbcheck: database integrity check
          log: perform operations on a log database
      memtest:
         hash: view or create file hashes
    diskspeed: test disk speed
   blockspeed: Tests various write block sizes to determine best setting
    metaspeed: Tests read performance over an existing meta db
         feed: work with feed files
        pause: wait for user input when running a script file
   createflex: create a flex parser
  compileflex: compile all flex parsers in a directory
encryptparser: encrypt all parsers in a directory
       import: import packets
      upgrade: send an upgrade package to a server
     makepcap: convert packet database files to pcap files
         quit: exit this program
         exit: exit this program
          bye: exit this program
      license: Output details of all license files in a directory
     nwdstrip: Convert full NWD file into just session and meta file
      logfake: Create a fake log pcap file
       fmanip: Manipulate a file with XOR and check for embedded PEs
       vslice: Validate index slices
       makec3: Generate C3 Test Data
          lua: Execute a lua script
     avro2nwd: Convert AVRO files to NWD files
     avrodump: Display schema and contents of AVRO file (for debugging)
     logParse: Parse line delimited logs on stdin and post results to stdout
 httpAggStats: Http Aggregation Status
     netbytes: Display statistics on network interface utilization
        login: connect to a remote collection
[> login localhost:50005 admin netwitness
Successfully logged in as session 598
[[localhost:50005] /> /sdk/stats ls
    123:0x0000000000200200 /sdk/stats/cache.window.time.begin (Cache Window Time Begin) = None
    124:0x0000000000200200 /sdk/stats/cache.window.time.end (Cache Window Time End) = None
    122:0x0000000000100100 /sdk/stats/queries
    126:0x2000000000200200 /sdk/stats/queries.active (Queries Active) = 0
    125:0x2000000000200200 /sdk/stats/queries.pending (Queries Pending) = 0
    127:0x2000000000200200 /sdk/stats/values.calls (Values Calls) = 0
    128:0x2000000000200200 /sdk/stats/values.calls.cached (Values Calls Cached) = 0

[localhost:50005] />
```

Getting Started

The first thing you'll want to do with NwConsole is launch it, and then log in to an SA Core service. Just type **NwConsole** from the command line to launch it.

```
[root@concentrator ~]# NwConsole
RSA Security Analytics Console 10.5.3.0.5372
Copyright 2001-2016, RSA Security Inc. All Rights Reserved.

> login localhost:50005 admin netwitness
Successfully logged in as session 533
```

You login with the **login** command, using the syntax: "**login host:port[:ssl] <user> [password]**"

Useful Commands

Command	Description
login host:port[:ssl] <user> [password]	Logs into an SA Core service. Host and port are mandatory. :ssl is not. If password is not specified, the user will be prompted for it. When scripting, login is the first command which should be passed to NwConsole. The ports are: Broker: **50003** (non-ssl) / **56003** (ssl) Concentrator: **50005** (non-ssl) / **56005** (ssl) Packet Decoder: **50004** (non-ssl) / **56004** (ssl) Log Decoder: **50002** (non-ssl) / **56002** (ssl) Log Collector: **50001** (non-ssl) / **56001** (ssl) Archiver: **50008** (non-ssl) / **56008** (ssl) Workbench: **50007** (non-ssl) / **56007** (ssl) Warehouse Connector: **50020** (non-ssl) / **56020** (ssl)
help	Shows help for NwConsole. Its output changes depending on whether you are logged into a Core service or not.

dbcheck [-header] [-autofix] [-chatty] [-dump] <*filename or pathname, wildcards accepted*>	Checks an NWDB file for problems, and using the **-autofix** parameter, it can repair some of the problems. **Be sure you stop the Core service before using this!**
	
	Top Tip: Do **NOT** use this on index files (files in **/var/netwitness/<*service*>/index**). Use only on MetaDB, SessionDB, and PacketDB files.
	
	Example:
	
	`> dbcheck -autofix /var/netwitness/concentrator/metadb/meta-000000001.nwmdb`
	`----------`
	`PERFORMING TEST: DB INTEGRITY`
	`----------`
	
	`Verifying file /var/netwitness/concentrator/metadb/meta-000000001.nwmdb`
	`The index flags are 1, which translates to INDEX_32, fidelity is 1`
	`The index file /var/netwitness/concentrator/metadb/meta-000000001.nwmdbindex does not contain a user defined header.`
	`The index contains offsets for 48884329 objects.`
	`The file "/var/netwitness/concentrator/metadb/meta-000000001.nwmdb" has a magic number of 1884076757 (0x704CBAD5).`
	`Version is 10.`
	`The file header size is 20. File creation date from header is 2017-Jan-19 21:49:18 UTC.`
	`The file flags value is 1052706: NEW_FILE_GROUP, OBJECT_SIZE_16_BITS.`
	`Second file flags is 1.`
	`This is a VARIABLE object size database file.`
	`The serializer header length is 19468.`
	`The user defined header length is 8.`
	`Header Dump:`
	`00000000 01 00 00 00 00 00 00 00 [........]`
	`Header value as a unsigned 64 bit int: 1`
	
	`0% Object 0 has a size of 14 at offset 19516.`
	`1% Object 504244 has a size of 10 at offset 7925438.`
	`2% Object 977814 has a size of 10 at offset 15850857.`
	`3% Object 1464232 has a size of 14 at offset 23776289.`
	`4% Object 1967108 has a size of 7 at offset 31701714.`
	`5% Object 2463187 has a size of 8 at offset 39627151.`
	`6% Object 2885308 has a size of 10 at offset 47552576.`
	`7% Object 3372953 has a size of 14 at offset 55478006.`
	`...`
	`99% Object 48389441 has a size of 16 at offset 784617360.`
	`The object store contains 48884329 objects.`
	`100% File scan complete. Total file size is 755.82 MB`
	`File is valid`

vslice dir=<index *directory*>	Checks the consistency of index all slices. Index slices are stored in **/var/netwitness/<service>/index,** where N is an integer.
	
	Example:
	
	`> vslice dir=/var/netwitness/concentrator/index`
	`Examining slices for reference.id`
	`Examining slices for threat.category`
	` validating slice 15`
	` validation OK`
	` 7 values`
	` data leakage to vulnerability`
	` validating slice 14`
	` validation OK`
	` 4 values`
	` informational to suspicious`

import [-noSource] [collection=<collection>] [-create] [node=<node>] *filename.ext*[;*filename.ext*;*.*ext*]	Imports packets from pcap files into a Decoder or Archiver. **collection=** is for Archiver.

makepcap {source=<*pathname*>} [dest=<*pathname*>] [filenum=#[-#]] [packetid=#[-#]] [time1=<time>] [time2=<time>] [ip=<IP Address>] [single={1|0}] [gzip={1|0}]	Exports packets to a pcap file.
	
	`Usage: makepcap {source=<pathname>} [dest=<pathname>] [filenum=#[-#]] [packetid=#[-#]] [time1=<time>] [time2=<time>] [ip=<IP Address>] [single={1|0}] [gzip={1|0}]`
	`  source - Required, the directory where the packet db files reside`
	`  dest - The destination directory for pcap files, uses source dir by default`
	`  filenum - The file number to operate on or range of file numbers, use 999999999 for no limit on upper range`
	`  packetid - Only extract the packet IDs within the specified range`
	`  time1 - Start time (UTC) for all packets to be extracted from packet db (e.g., time1="2009-Dec-17 08:00:00")`
	`  time2 - Stop time (UTC) for all packets to be extracted from packet db (e.g., time2="2009-Dec-17 08:05:00")`
	`  ip - Limit to a single IP address, will match both source and destination`
	`  single - Create a single output file if 1, 0 is default`
	`  gzip - Compress output files if 1, 0 is default`
	`  fileType - pcap, pcapng or log, default is pcap`
	`  delimiter - Log output line ending, default is \n (e.g., for windows you can use \r\n)`
	`  delete - If true, deletes the db file after processing, default is false`
	
	Example:
	
	`> makepcap source=/var/netwitness/decoder/packetdb dest=/root ip=10.0.0.0/8 single=1 fileType=pcap`
	`40 files found`
	`packet database file: packet-000000027.nwpdb, starting packet ID is 166847558`
	`  - converting to file: packet-000000027.pcap`
	`  - creating new file`
	`  - wrote 432744 packets (99%) timestamp: 01/02/17 01:57:14`
	`  - finished writing 432831 packets`
	`  - all packets written in time sequence`
	
	`packet database file: packet-000000028.nwpdb, starting packet ID is 175981447`
	`  - wrote 432991 packets (99%) timestamp: 01/02/17 01:58:42`
	`  - finished writing 432994 packets`
	`  - all packets written in time sequence`
	
	`packet database file: packet-000000029.nwpdb, starting packet ID is 176066909`
	`...`

Scripting With NwConsole

NwConsole can be used as a substitute for the REST API, but we believe the REST API is a more elegant solution, as one can use whatever technology one would like with it, and you are not bound to the CentOS operating system. Having said that, it is a perfectly valid tool for scripting by using NwConsole in non-interactive mode. If you have a look at the **nwtech.sh** script distributed by RSA which, ahem, someone writing this course also wrote, it is chock full of calls to NwConsole.

To execute a non-interactive API call directly from the command line, execute NwConsole with the **-c "someCommand"** option. It is possible to execute multiple commands consecutively by use one **-c** option per command. If you're performing API calls on a Core service, the first command should be **-c "login host:port user password".**

Top Tip: When using the **-c** option, the command that follows must be enclosed in either double quotes, single quotes, or must make use of escape characters, i.e. the backslash.

Valid: **NwConsole -c "login localhost:50004 admin netwitness"**

Valid: **NwConsole -c 'login localhost:50004 admin netwitness'**

Valid: **NwConsole -c login\ localhost:50004\ admin\ netwitness**

Invalid: **NwConsole -c login localhost:50004 admin netwitness**

NwConsole 'Folders'

Because the NetWitness API is a tree, the tree can be traversed in NwConsole similar to a filesystem. Use the syntax "**cd *<folder>***". The prompt will always show the current path.

Examples:

```
[localhost:50005] /> cd /sdk
[localhost:50005] /sdk

[localhost:50005] /sdk> cd /concentrator/config
[localhost:50005] /concentrator/config

[localhost:50005] /concentrator/config>
```

Executing an API Call

To execute an API call, use the syntax: "***/<folder> msgName parameter1=value1 parameter2=value2***". If you have **cd**'d into a directory, you can use a period (**.**) for the folder name.

Examples:

```
[localhost:50005] /> /sdk summary
mid1=1 mid2=565471624 msize=11511767089 mmax=45515915919 pid1=0 pid2=0 psize=0 pmax=0 time1=1484858460 time2=1489168598 ptime1=0
ptime2=0 sid1=1 sid2=14180077 ssize=482209934 smax=45515915919 stotalsize=157319696781 isize=354064688 memt=8254554112
memu=1583185920 memp=41574400 hostname=concentrator version=10.5.3.0.5372
```

Getting Help on API Calls

To learn what messages are available for a particular API node, use the syntax: "**/<folder> help**"

Example:

```
[localhost:50005] /> /sdk help
[description: A container node for other node types
security.roles: everyone,sdk.content,sdk.content & sdk.meta,sdk.content & sdk.packets,sdk.manage,sdk.meta
message.list: The list of supported messages for this node
ls: [depth:<uint32>] [options:<string>] [exclude:<string>]
mon: [depth:<uint32>] [options:<uint32>]
info:
help: [msg:<string>] [op:<string>] [format:<string>]
count:
query: [id1:<uint64>] [id2:<uint64>] [size:<uint32>] [query:<string>] [flags:<string>] [threshold:<uint64>]
search: session:<uint64> size:<uint32> search:<string>
cancel: handle:<uint32>
values: [id1:<uint64>] [id2:<uint64>] size:<uint32> [flags:<string>] [threshold:<uint64>] fieldName:<string> [where:<string>]
[aggregateFunction:<string>] [aggregateFieldName:<string>] [min:<string>] [max:<string>]
xforms: [key:<string>]
msearch: [sessions:<string>] [packets:<string>] search:<string> [where:<string>] [limit:<uint64>] [flags:<string>]
stopMon:
session: id1:<uint64> id2:<uint64>
aliases: key:<string> [value:<string>]
packets: op:<string> [sessions:<string>] [packets:<string>] [where:<string>] [time1:<date-time>] [time2:<date-time>] [flags:
<uint32>] [devices:<string>]
content: session:<uint64> [packet:<uint64>] [maxSize:<size>] [failOnCacheMiss:<bool>] [renderType:<uint32>] [renderFlags:<uint32>]
[renderOptions:<string>]
summary: [flags:<string>]
deviceId: session:<uint64>
language: [id1:<uint64>] [id2:<uint64>] [time1:<date-time>] [time2:<date-time>] [options:<string>] [flags:<string>] size:<uint32>
delCache:
precache: [sessions:<string>] [packets:<string>] [waitOnComplete:<bool>]
timeline: [time1:<date-time>] [time2:<date-time>] [timezone:<int32>] size:<uint32> [flags:<string>] [where:<string>] [bookendHours:
<uint32>]
validate: op:<string> [validateLanguage:<bool>] [fieldName:<string>] [where:<string>] [query:<string>]
reconfig: [update:<bool>] [memory:<size>]
]
```

To find out what the parameters of an API call accepts, use the syntax: "***/<folder>* help message=*<apiCall>***" You can also substitute '**m=**' for '**message=**'.

Examples:

```
[localhost:50005] />  /sdk help message=query

[query: Performs a query against the meta database
security.roles: sdk.meta
parameters:
 id1 - <uint64, optional> The starting meta id (to run the query from most recent to oldest meta, make id1 larger than id2)
 id2 - <uint64, optional> The ending meta id
 size - <uint32, optional> The max number of entries to return, or just stream back all results if zero
 query - <string, optional> The query string to use
 flags - <string, optional> The flags to use for query. Can be a number (bitwise mask) or comma separated values like query-log.
 threshold - <uint64, optional> Query optimization to stop processing results after the threshold is reached (useful with select
aggregate functions). Zero means no threshold (the default).

]
```

Server Commands

Top Tip: If logged into a Core service, executing the '**help**' command without any parameters will tell you what **server commands** are available. **Server commands** are distinct from API calls in that they are more like generic methods or functions that don't operate on specific nodes of the API tree.

```
[localhost:50005] />  help
Server Commands: login, tlogin, vlogin, logout, cd, ls, dir, get, info, set, send,
 ping, mon, delmons, file, upload, import, upgrade, timeout
 packets, dlogs, addLic, record, replay, netspeed, compress
```

Interesting server commands are:

<table>
<tr><th>Command</th><th>Description</th></tr>
<tr><td>login</td><td>Logs in to an SA Core service using password authentication.


```
> login
login <server:port[:ssl]> <username> <password>
```


Example:


```
> login localhost:56005:ssl admin netwitness
Successfully logged in as session 1534
```

</td></tr>
</table>

tlogin

The **tlogin** server command performs a trusted connection login using an SSL certificate. This is extremely useful if you don't want to type in credentials.

Top Tip: Using tlogin, one should be able to connect to any Core service in the environment from the puppet master server, as its public key gets pushed to every Core service.

- You must use the HTTPS port for this connection.
- You must use a certificate public/private key pair which is trusted by the Core service. These certificates are generally installed in the Core service by puppet.
- The **cert=** parameter is for the PEM certificate containing the public key. **This must be a full PEM certificate**, not just a file containing a public key. (i.e. in **/var/lib/puppet/ssl/certs**)
- The **key=** parameter is for PEM certificate containing the private key. This need not be specified if the private key is in the same file as the **cert=** parameter, but this is very rare.
- You must specify a username to be used for the connection. This need not be defined on the core service.
- You must also specify a role name that the connection will use, such as '**Administrators**' or '**Operators**'. This role must be defined already on the core service.

```
tlogin {server=ip or hostname} {port=port #} {username=user account} {group=g1,g2} {cert=cert path} [key=key path]
Usage: Open a trusted SSL connection to an existing service
 server - The server name or ip address to connect to
 port - The port to connect to
 username - The user account to use. NOTE: This account does not need to exist on the service.
 group - The administrative group or groups (csv) this user belongs to. NOTE: These must exist on the service
 cert - The trusted certificate pathname to use for the SSL connection
 key - The trusted key pathname to use for the SSL connection (this is optional if the key is in the cert file)
```

The trusted public certificates are installed in **/etc/netwitness/ng/<*service*>/trustpeers**.

```
[root@concentrator ~]# find /etc/netwitness/ng/concentrator
/etc/netwitness/ng/concentrator
/etc/netwitness/ng/concentrator/trustpeers
/etc/netwitness/ng/concentrator/trustpeers/e8066ed0.0
/etc/netwitness/ng/concentrator/trustpeers/5f77e40f.0
/etc/netwitness/ng/concentrator/storedproc
/etc/netwitness/ng/concentrator/storedproc/groupby.luasp
/etc/netwitness/ng/concentrator/truststore
/etc/netwitness/ng/concentrator/truststore/3dc25656.0
```

The **truststore** subdirectory contains the puppet CA's public certificate, which is used to validate the trusted certificates which the user connects with.

If you are using NwConsole to connect to a localhost Core service which has been provisioned (enabled) by the SA head server, you can use the localhost's puppet keypair, which can be found in **/var/lib/puppet/ssl** under the **private_keys** dir and the **certs** subdirectories. You should also be able to do this from the SA head server to any Core service which is managed by it.

```
[root@concentrator ~]# find /var/lib/puppet/ssl/
/var/lib/puppet/ssl/
/var/lib/puppet/ssl/crl.pem
/var/lib/puppet/ssl/private_keys
/var/lib/puppet/ssl/private_keys/4fe924bb-8ed9-4f65-a731-8767f0a37992.pem
/var/lib/puppet/ssl/private
/var/lib/puppet/ssl/public_keys
/var/lib/puppet/ssl/public_keys/4fe924bb-8ed9-4f65-a731-8767f0a37992.pem
/var/lib/puppet/ssl/certs
/var/lib/puppet/ssl/certs/ca.pem
/var/lib/puppet/ssl/certs/4fe924bb-8ed9-4f65-a731-8767f0a37992.pem
/var/lib/puppet/ssl/certs/sa.pem
/var/lib/puppet/ssl/certificate_requests
/var/lib/puppet/ssl/certificate_requests/4fe924bb-8ed9-4f65-a731-8767f0a37992.pem
```

Example connecting from localhost (using certificate names from above):

```
> tlogin server=localhost port=56005 username=admin group=Administrators cert=/var/lib/puppet/ssl/certs/4fe924bb-8ed9-
4f65-a731-8767f0a37992.pem key=/var/lib/puppet/ssl/private_keys/4fe924bb-8ed9-4f65-a731-8767f0a37992.pem
Successfully logged in as session 1490
```

logout

Logs out of a Core service.

cd	Like in UNIX-like OS'es, **cd** changes directory. Use a forward slash as the path delimiter.

ls	Like in UNIX-like OS'es, **ls** lists the contents of an API node. Note that there is an **ls** server command and also an **ls** API call. Using **ls** without specifying a path will use the server command.
	
	Use the **depth=<*n*>** parameter to include subdirectories in the listing, where ***n*** is an integer indicating how many subdirectories levels to display.
	
	`[localhost:50005] /database/config> ls`
	
	`hash.algorithm (Hash Algorithm) = none`
	`hash.databases (Hash Databases) = session,meta`
	`hash.dir (Hash Directory)`
	`manifest.dir (Manifest Directory)`
	`meta.compression (Meta Compression) = none`
	`meta.compression.level (Meta Compression Level) = 0`
	`meta.dir (Meta Database Directory) = /var/netwitness/concentrator/metadb=42.39 GB`
	`meta.dir.cold (Cold Meta Database Directory)`
	`meta.dir.warm (Warm Meta Database Directory)`
	`meta.file.size (Meta File Size) = 3 GB`
	`meta.files (Meta Open Files) = 8`
	`meta.free.space.min (Meta Minimum Free Space) = 397 MB`
	`meta.index.fidelity (Meta Index Fidelity) = 1`
	`meta.integrity.flush (Meta Integrity Flush) = sync`
	`meta.write.block.size (Meta Write Block Size) = 64 KB`
	`session.dir (Session Database Directory) = /var/netwitness/concentrator/sessiondb=42.39 GB`
	`session.dir.cold (Cold Session Database Directory)`
	`session.dir.warm (Warm Session Database Directory)`
	`session.file.size (Session File Size) = 512 MB`
	`session.files (Session Open Files) = 8`
	`session.free.space.min (Session Minimum Free Space) = 397 MB`
	`session.integrity.flush (Session Integrity Flush) = sync`
	`session.write.block.size (Session Write Block Size) = 32 KB`

dir

The **dir** command produces almost the same output as **ls**, except that it displays the full path of each item, and another identifier which isn't really useful.

Using **dir** is the same as issuing the **ls** API call on a specific folder. I.e. "**dir /database/config"** produces the same output as "**/database/config ls"**

depth=<n> is also valid for the **dir** command.

```
[localhost:50005] /database/config> dir /database/config

 256:0x0000000000400300 /database/config/hash.algorithm (Hash Algorithm) = none
 257:0x0000000000400300 /database/config/hash.databases (Hash Databases) = session,meta
 255:0x0000000000400300 /database/config/hash.dir (Hash Directory)
 221:0x0000000000400300 /database/config/manifest.dir (Manifest Directory)
 241:0x0000000000400300 /database/config/meta.compression (Meta Compression) = none
 242:0x0000000000400300 /database/config/meta.compression.level (Meta Compression Level) = 0
 228:0x0000000000400300 /database/config/meta.dir (Meta Database Directory) = /var/netwitness/concentrator/metadb=42.39
GB
 230:0x0000000000400300 /database/config/meta.dir.cold (Cold Meta Database Directory)
 229:0x0000000000400300 /database/config/meta.dir.warm (Warm Meta Database Directory)
 253:0x0000000000400300 /database/config/meta.file.size (Meta File Size) = 3 GB
 250:0x0000000000400300 /database/config/meta.files (Meta Open Files) = 8
 235:0x0000000000400300 /database/config/meta.free.space.min (Meta Minimum Free Space) = 397 MB
 243:0x0000000000400300 /database/config/meta.index.fidelity (Meta Index Fidelity) = 1
 245:0x0000000000400300 /database/config/meta.integrity.flush (Meta Integrity Flush) = sync
 247:0x0000000000400300 /database/config/meta.write.block.size (Meta Write Block Size) = 64 KB
 231:0x0000000000400300 /database/config/session.dir (Session Database Directory) =
/var/netwitness/concentrator/sessiondb=42.39 GB
 233:0x0000000000400300 /database/config/session.dir.cold (Cold Session Database Directory)
 232:0x0000000000400300 /database/config/session.dir.warm (Warm Session Database Directory)
 254:0x0000000000400300 /database/config/session.file.size (Session File Size) = 512 MB
 251:0x0000000000400300 /database/config/session.files (Session Open Files) = 8
 236:0x0000000000400300 /database/config/session.free.space.min (Session Minimum Free Space) = 397 MB
 244:0x0000000000400300 /database/config/session.integrity.flush (Session Integrity Flush) = sync
 248:0x0000000000400300 /database/config/session.write.block.size (Session Write Block Size) = 32 KB
```

```
[localhost:50005] /database/config> /sdk ls depth=10

 121:0x0000000000100100 /sdk/config
 129:0x0000000000400300 /sdk/config/cache.dir (Cache Directory) = /var/netwitness/concentrator/cache
 130:0x0000000000400300 /sdk/config/cache.size (Cache Size) = 4 GB
 134:0x0000000000400300 /sdk/config/cache.window.minutes (Cache Window Minutes) = 15
 132:0x0000000000400300 /sdk/config/max.concurrent.queries (Max Concurrent Queries) = 3
 133:0x0000000000400300 /sdk/config/max.pending.queries (Max Pending Queries) = 24
 139:0x0000000000400300 /sdk/config/max.query.groups (Maximum numbers of groups collected by GROUP BY) = 1000000
 136:0x0000000000400300 /sdk/config/max.where.clause.cache (Max Where Clause Cache) = 1 GB
 140:0x0000000000400300 /sdk/config/max.where.clause.sessions (Maximum number of sessions retrieved by a query or
msearch) = 10000000
 135:0x0000000000400300 /sdk/config/packet.read.throttle (Packet Read Throttle) = 0
 138:0x0000000000400300 /sdk/config/parallel.query (Parallelize all query operations) = 0
 137:0x0000000000400300 /sdk/config/parallel.values (Parallelize all values operations) = 2
 131:0x0000000000400300 /sdk/config/system.roles (System Roles) = 0
 120:0x0000000000100100 /sdk/stats
 123:0x0000000000200200 /sdk/stats/cache.window.time.begin (Cache Window Time Begin) = 2017-Mar-10 21:02:11
 124:0x0000000000200200 /sdk/stats/cache.window.time.end (Cache Window Time End) = 2017-Mar-10 21:17:11
 122:0x0000000000100100 /sdk/stats/queries
 126:0x2000000000200200 /sdk/stats/queries.active (Queries Active) = 0
 125:0x2000000000200200 /sdk/stats/queries.pending (Queries Pending) = 0
 127:0x2000000000200200 /sdk/stats/values.calls (Values Calls) = 0
 128:0x2000000000200200 /sdk/stats/values.calls.cached (Values Calls Cached) = 0
```

get

The **get** server command returns the value of a specific node key. It works for both configuration and statistic nodes. I.e. if you want the value of /sdk/config/cache.window.minutes, issue the command "**get /sdk/config/cache.window.minutes**"

[localhost:50005] /sdk/config> get /sdk/config/cache.window.minutes
[15]

set

The **set** server command allows one to set a new value for a configuration node. Note that for **set**, the parameter must be in the current path which you are cd'd into.

```
[localhost:50005] /sdk/config> set cache.window.minutes 20
[Success]

[localhost:50005] /sdk/config> get cache.window.minutes
[20]
```

ping	Tests the connection for the connected Core service and returns the time which it took to receive a response.
	
	`[localhost:50005] /> ping`
	`Ping received response in 2 ms`

mon	The **mon** server command monitors a given folder node or key node for changes in values, and records them to the screen. If a folder is specified, then all immediate child values are monitored. If a specific value node is specified, then it will only monitor that node.
	
	Example:
	
	`[localhost:50005] /sdk/stats> mon queries.active`
	`mon(126) completed using channel 1152`
	`[localhost:50005] /sdk/stats> Monitor: /sdk/stats/queries.active = 0`
	
	We then issue an Reporting Engine query, and NwConsole instantly spits out:
	
	`Monitor: /sdk/stats/queries.active = 1`
	`Monitor: /sdk/stats/queries.active = 0`

delmons	The **delmons** command removes all **mon** command monitors.
	
	`[localhost:50005] /sdk/stats> delmons`
	`1 monitors removed`

packets	The **packets** server command is about the same as the "**/sdk packets"** command, except that it writes the files directly to the local disk. The **pathname** parameter must be used, which specifies which directory to write the output files to on disk.
	
	`[localhost:50005] /sdk/stats> packets`
	`Usage: packets {sessions=<#,#-#,#> | time1=<"YYYY-MM-DD HH:MM:SS"> time2=<"YYYY-MM-DD HH:MM:SS">} {pathname=<pathname for pcap/logs>} [append={0|1}]`
	`NOTE: The current node must be /sdk ("cd /sdk"), use extension ".pcap.gz" to compress output`
	
	`Parameters:`
	`sessions - Comma separated list of session IDs or session ID ranges whose packets will be downloaded (e.g., "1,3-10,98-120,1001"`
	`packets - Comma separated list of packet IDs or packet ID ranges whose packets will be downloaded (e.g., "1,3-10,98-120,1001"`
	`where - Where clause to determine which sessions to download (must be submitted to a service with an index)`
	`time1 - Start time (UTC) for all packets to be extracted from packet db (e.g., time1="2009-Dec-17 08:00:00")`
	`time2 - Stop time (UTC) for all packets to be extracted from packet db (e.g., time2="2009-Dec-17 08:05:00")`
	`pathname - The pcap or log file that the packets will be written to`
	`append - If 1, will append to an existing file, otherwise it overwrites, default is overwrite`

dlogs

The **dlogs** server command downloads logs from the Core service logs database. Use the parameters laid out by "**/logs help m=download**".

```
localhost:50005] /sdk/stats> send /logs help msg=download
[download: Downloads log entries
security.roles: logs.manage
parameters:
 id1 - <uint64, optional> The starting log number to download
 id2 - <uint64, optional> The ending log number to download
 time1 - <date-time, optional> The starting log date to download, can be posix time (seconds since 1970) or DD-MM-YY
HH:MM:SS
 time2 - <date-time, optional> The ending log date to download, can be posix time (seconds since 1970) or DD-MM-YY
HH:MM:SS
 op - <string, {enum-one:start|next|cancel}> The operation to perform (start, next, cancel)
 logTypes - <string, optional> The type(s) of log messages to retrieve (debug,info,audit,warning,failure)
 match - <string, optional> Case insensitive string to match in each log message
 regex - <string, optional> Regular expression to match in each log message
 timeFormat - <string, optional, {enum-one:posix|simple}> The time format used in each log message, default is posix
time (seconds since 1970)
 batchSize - <uint32, optional, {range:2000 to 10000}> The number of logs to retrieve per batch, default is 2000

]
```

```
[localhost:50005] /sdk/stats> /logs help m=download
[download: Downloads log entries
security.roles: logs.manage
parameters:
 id1 - <uint64, optional> The starting log number to download
 id2 - <uint64, optional> The ending log number to download
 time1 - <date-time, optional> The starting log date to download, can be posix time (seconds since 1970) or DD-MM-YY
HH:MM:SS
 time2 - <date-time, optional> The ending log date to download, can be posix time (seconds since 1970) or DD-MM-YY
HH:MM:SS
 op - <string, {enum-one:start|next|cancel}> The operation to perform (start, next, cancel)
 logTypes - <string, optional> The type(s) of log messages to retrieve (debug,info,audit,warning,failure)
 match - <string, optional> Case insensitive string to match in each log message
 regex - <string, optional> Regular expression to match in each log message
 timeFormat - <string, optional, {enum-one:posix|simple}> The time format used in each log message, default is posix
time (seconds since 1970)
 batchSize - <uint32, optional, {range:2000 to 10000}> The number of logs to retrieve per batch, default is 2000

]
```

record

The **record** server command records the commands you subsequently enter into a file, which can be replayed in NwConsole at a later time using the **replay** command.

```
[localhost:50005] /sdk/stats> record
Usage: record {pathname=<pathname>} [append={0|1}] [stop=true]
 pathname - the output path where the messages will be recorded
 append - If True, will append to existing file, zero overwrites (default)
 stop - If True, will stop recording to the file and close it
```

replay

The **replay** server command replays the commands from the output file from the **replay** command in NwConsole.

```
[localhost:50005] /sdk/stats> replay
Usage: replay {pathname=<pathname>} [timeout=secs] [exit=<exit code>] [verbose=<true/false>]
 pathname - the input pathname of a previously recorded session that will be replayed
 timeout - The timeout in seconds to wait for a response (default is 600 seconds)
 exit - When results do not match, exit the process with this error code
 verbose - Output results even when they match
 skip - Skip the first N tests total - Only run 'total' tests then exit, default is all[localhost:50005] /sdk/stats>
```

netspeed

The **netspeed** server command performs a speed test between the host which NwConsole is currently running on and the remote Core service that it's logged into.

Top Tip: Don't forget to use quotes for the value of the transfer= parameter, as it is a two word parameter.

```
[localhost:50005] /sdk/stats> netspeed
Usage: netspeed transfer=<amount>
Examples: netspeed transfer=4GB
 netspeed transfer="512 MB"
```

```
[localhost:50005] /sdk/stats> netspeed transfer="100 MB"
Requesting 1600 messages of size 64 KB
Response from compress command: Compression changed from 0 to 0, compression level is 0
Received 13.06 MB at a transfer rate of 13.04 MB/sec or 109 Mbps
Received 88.23 MB at a transfer rate of 44.02 MB/sec or 369 Mbps
Received 100.05 MB at a transfer rate of 46.4 MB/sec or 389 Mbps
```

Root Commands

Root commands are NwConsole operations which do not require a logged-in connection to a Core service. It can check databases, validate index slices, dump logs, test speed, and many other things. Executing the '**help'** command without being logged in to a Core service will produce this output:

```
> help
Built-in Commands:
            ?: display this message
         help: display this message
        index: explore index data (advanced)
          sdk: execute SDK commands
      reindex: reindex a collection
      dbcheck: database integrity check
          log: perform operations on a log database
      memtest:
         hash: view or create file hashes
    diskspeed: test disk speed
   blockspeed: Tests various write block sizes to determine best setting
    metaspeed: Tests read performance over an existing meta db
         feed: work with feed files
        pause: wait for user input when running a script file
   createflex: create a flex parser
  compileflex: compile all flex parsers in a directory
encryptparser: encrypt all parsers in a directory
       import: import packets
      upgrade: send an upgrade package to a server
     makepcap: convert packet database files to pcap files
         quit: exit this program
         exit: exit this program
          bye: exit this program
      license: Output details of all license files in a directory
     nwdstrip: Convert full NWD file into just session and meta file
      logfake: Create a fake log pcap file
       fmanip: Manipulate a file with XOR and check for embedded PEs
       vslice: Validate index slices
       makec3: Generate C3 Test Data
          lua: Execute a lua script
     avro2nwd: Convert AVRO files to NWD files
     avrodump: Display schema and contents of AVRO file (for debugging)
     logParse: Parse line delimited logs on stdin and post results to stdout
 httpAggStats: Http Aggregation Status
     netbytes: Display statistics on network interface utilization
        login: connect to a remote collection
```

Issue one of the commands without parameters to learn what the usage is.

```
> dbcheck
Usage: dbcheck [-header] [-autofix] [-chatty] [-dump] <filename or pathname, wildcards accepted>
```

32. Puppet

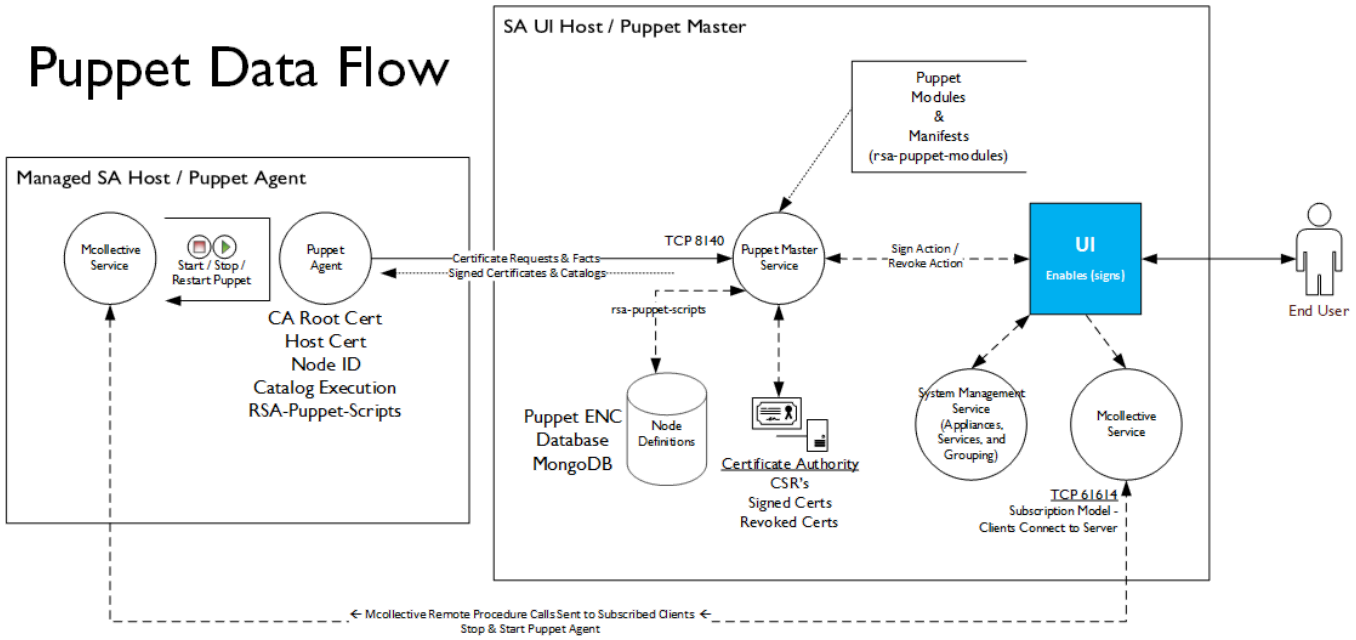
One of the least understood parts of SA is Puppet. It is an open source third-party technology, written in Ruby, with its own unique paradigm, and how it fits into the rest of the SA suite can be confusing. It's actually not too bad once you get it. Hopefully we can help to dispel some of the knowledge gaps here, and make it a bit more accessible.

DANGER! ATTENTION! ACHTUNG! ¡PELIGRO!

All commands contained in this lesson may be considered dangerous, and if used improperly, could render an SA system unusable. **USE AT YOUR OWN RISK!!!**

How Does it Work?

- Puppet uses a declarative language to control the configuration of Unix-like and Windows systems.
- Manifest files, or recipes, describe what state end systems should be in. These are stored on the puppet master server.
- A **puppet agent** runs on all SA hosts, which is joined to the puppet master.
- Puppet uses PKI, or Public Key Infrastructure, to secure itself.
- The puppet master functions as a certificate authority for all SA nodes.
- When a new puppet agent is first started, it generates a Node ID for itself, which is used by the puppet master to identify that node. This is in GUID format. E.g. **f11f6740-7ad4-45d3-baa0-10516dea1b16.**
- The puppet agent also generates a PKI keypair and CSR, or Certificate Signing Request. The CSR and public key are then submitted by the puppet agent to the puppet master on its first attempt to run. The puppet agent then waits for that CSR to be signed before it can take any action. The subject of this CSR is the puppet node ID.
- For a new SA host to be provisioned, the puppet master certificate authority on the SA UI host must sign the new host's automatically-generated certificate signing request, or CSR. This is done by an Administrator from the **Administration->Hosts** page of the UI.
- Once a puppet agent is managed by its puppet master, the agent discovers **facts** about itself with a tool called **facter**, which then gets submitted to the puppet master.
- The puppet master then uses what is called an **ENC**, or **External Node Classifier**, to store what type of host the puppet agent is. I.e. a packet Decoder, Log Decoder, Concentrator, etc. The classification is then stored in a local MongoDB.
- Based on the facts the puppet agent submits and the manifest files stored on the puppet master, the puppet master then compiles a **catalog** for the managed host, which is transmitted back to the puppet agent to be run.
- The puppet agent on the managed host then executes the catalog, making any configuration changes that the catalog requires. Examples of changes that puppet can make are:
  - The installation of packages
  - The uninstallation of packages
  - Starting services
  - Stopping services
  - Restarting services
  - Modifying service configuration files
  - Execution of local executables to achieve specific ends, such as creating a java certificate store, or generating yum repository metadata.
- Each SA puppet agent contains a package of dependency scripts which are used by the local puppet agent. These scripts extend the functionality of the standard puppet agent and allow it to work with components that it wasn't originally designed to work with, like Security Analytics.
- Puppet can be implemented in a number of ways, but in the SA implementation, only a single puppet master is supported in an SA environment.



Puppet Packages

Versions are based on SA 10.5.3.0.

puppet-server-3.8.2-1.el6.noarch	The puppetmaster server, which runs only on the SA head server.
puppet-3.8.2-1.el6.noarch	The puppet agent, which runs on all SA hosts, including the SA server
rsa-puppet-modules-10.5.3.0.3107-5.el6.noarch	The puppet modules, or recipes stored on the puppet master.
rsa-puppet-scripts-10.5.3.0.3107-5.el6.noarch	The puppet scripts which run on each managed puppet node by the puppet agent.

Puppet File Locations

Location	Role	Description
/etc/puppet	Agent & Master	Puppet configuration files.
/etc/puppet/scripts	Agent & Master	Custom RSA scripts for puppet.
/etc/puppet/modules	Master	Directory containing puppet module definitions, or *recipes*.
/etc/puppet/modules/<*moduleName*>/manifests/init.pp	Master	init.pp manifest files define the configuration for a given module. They define configuration files, service states, firewall ports, etc. It's helpful to remember that this file defines a **state** that the managed host should match once its catalog has executed.
/etc/puppet/puppet.conf	Agent & Master	The main puppet configuration file. Contains the puppet node ID. Also contains directives for the puppet master.
/etc/puppet/csr_attributes.yaml	Agent	Defines CSR attributes needed by the puppet master to communicate with the managed puppet host, which are fqdn and ipaddress.
/var/lib/puppet	Agent & Master	Contains ruby dependencies used by the puppet agent and puppet master.
/var/lib/puppet/ssl	Agent & Master	Contains SSL certificates and keys used by the puppet agent.
/var/lib/puppet/ssl/ca	Master	Contains files for the puppet master certificate authority (CA), including CA public and private CA keyfiles, inventory, CSR's, signed certificates, revoked certificates, and the certificate revocation list.
/var/lib/puppet/node_id	Agent	Contains the puppet node ID.
/var/lib/puppet/classes.txt	Agent & Master	Contains a list of all puppet classes applied to a host.

The Puppet Master

The puppet master is a relatively lightweight service which runs on the SA host. It hosts a kind of web server, running on TCP port <port>, which puppet agents connect to to pull configurations. It holds modules, manifest files, compiles catalogs, and hosts a certificate authority, or CA. The CA is capable of signing and revoking certificates.

Top Tip: **DO NOT** sign or revoke a puppet CSR from the command line. Always use the GUI. If you don't follow this advice, the host will not get set up properly.

Starting and Stopping Puppet Master

To start the puppet master service: **service puppetmaster start**

To stop the puppet master service: **service puppetmaster stop**

The Puppet Agent

The puppet agent is the service running on each managed puppet host, which executes catalogs compiled by the puppet master for the host. It also generates certificates for itself and transmits them to the puppet master. It is written in ruby.

```
[root@logconcentrator ~]# ps -ef | grep puppet
root 1726 1 0 15:44 ? 00:00:06 /usr/bin/ruby /usr/bin/puppet agent --waitforcert=10
```

The puppet agent by default runs with the **--waitforcert=10** parameter. This tells the agent that if the puppet master hasn't signed the agent's CSR yet, to keep waiting for it to be signed, checking again every 10 seconds.

puppet.conf

The file **/etc/puppet/puppet.conf** is the agent's main configuration file. The case of SA's agent, it has a very simple configuration, essentially specifying only the certificate name, which is the node ID, and the server, which is the puppet master.

For Puppet Agent:

```
[main]
rundir = /var/run/puppet
certname = 4fe924bb-8ed9-4f65-a731-8767f0a37992
logdir = /var/log/puppet
ssldir = $vardir/ssl

[agent]
localconfig = $vardir/localconfig
classfile = $vardir/classes.txt
server = puppetmaster.local
```

For Puppet Master:

```
[main]
node_terminus = exec
pluginsync = true
dns_alt_names = puppetmaster.local
certname = d015576d-c69d-4f06-865c-3a58711a23e5
logdir = /var/log/puppet
trusted_node_data = true
ssldir = $vardir/ssl
rundir = /var/run/puppet
external_nodes = /etc/puppet/scripts/mongo_enc.py

[agent]
localconfig = $vardir/localconfig
classfile = $vardir/classes.txt
server = puppetmaster.local
```

In SA, the puppet server is always called **puppetmaster.local**, which is is automatically added to the **/etc/hosts** file as part of the initial system configuration. Do you remember when an SA host asks you for IP details and the IP address of the SA UI server, when logging into a newly-built SA hosts for the first time? That SA server IP is added to **/etc/hosts** for host **puppetmaster.local**.

```
# Created by NetWitness Installer on Tue May 10 03:36:09 UTC 2016
127.0.0.1 concentrator localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 concentrator localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.0.14 concentrator.ktalab.local concentrator
10.0.0.3 puppetmaster.local
```

csr_attributes.yaml

The file **/etc/puppet/csr_attributes.yaml** is also a key configuration file for the puppet agent. It comes into play when the agent starts up for the first time. Its purpose is to guide the creation of the certificate signing request which gets submitted to the puppet master. The two pertinent and variable attributes are fqdn and ipaddress. If you ever change the hostname or IP address of an SA system, it is necessary to Remove and Repurpose the host in **Administration->Hosts** in the UI, then reset puppet certs, and then update the hostname and/or IP address in this file before starting the agent again. The procedure for this can be found later on this lesson.

```
custom_attributes:
 1.2.840.113549.1.9.7: fqdn=concentrator.ktalab.local,ipaddress=10.0.0.14,type=base
```

Puppet Logs

When running as a service, the puppet agent logs to **/var/log/messages**.

If the certificate hasn't yet been signed, you will see these messages in the logs:

```
May 10 21:45:27 NWAPPLIANCE16378 python: Starting puppet agent: #033[60G[#033[0;32m OK #033[0;39m]#015
May 10 21:45:40 NWAPPLIANCE16378 puppet-agent[1687]: Did not receive certificate
May 10 21:45:50 NWAPPLIANCE16378 puppet-agent[1687]: Did not receive certificate
May 10 21:46:01 NWAPPLIANCE16378 puppet-agent[1687]: Did not receive certificate
May 10 21:46:11 NWAPPLIANCE16378 puppet-agent[1687]: Did not receive certificate
May 10 21:46:21 NWAPPLIANCE16378 puppet-agent[1687]: Did not receive certificate
May 10 21:46:33 NWAPPLIANCE16378 puppet-agent[1687]: Did not receive certificate
```

Once an agent's certificate has been signed, puppet will log messages similar to these:

```
May 10 21:46:44 NWAPPLIANCE16378 puppet-agent[1687]: Starting Puppet client version 3.6.2
May 10 21:46:45 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/facts.d]/mode) mode changed '0755' to '0775'
May 10 21:46:46 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/facts.d/applianceVersion.sh]/ensure) defined content as
'{md5}bd28d8db285f1eba04a07cba339afddd'
May 10 21:46:46 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/facts.d/rabbitmq_disk_limit.sh]/ensure) defined content
as '{md5}024146c308f810f4ea7801a8167fb4d1'
May 10 21:46:46 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/facts.d/rabbitmq_plugins_dir.sh]/ensure) defined content
as '{md5}51a1cd5e71760131d27c3e6cccf0d697'
May 10 21:46:46 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/facts.d/nw_admin_path.sh]/ensure) defined content as
'{md5}a25c56de3e8587371aba1c79b3568169'
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet]/ensure) created
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/provider]/ensure) created
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/provider/java_ks]/ensure) created
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/provider/file_line]/ensure) created
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/util]/ensure) created
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/util/ipcidr.rb]/ensure) defined content as
'{md5}e1160dfd6e73fc5ef2bb8abc291f6fd5'
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/provider/firewall.rb]/ensure) defined content
as '{md5}32d2f5e5dcc082986b82ef26a119038b'
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/provider/firewallchain]/ensure) created
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/provider/firewall]/ensure) created
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/provider/firewall/iptables.rb]/ensure)
defined content as '{md5}470fdc741c7d60843573b23a2c9fc321'
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/provider/java_ks/keytool.rb]/ensure) defined
content as '{md5}6ac5a79da78281e667515e383151584f'
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/parser]/ensure) created
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/parser/functions]/ensure) created
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/parser/functions/ensure_packages.rb]/ensure)
defined content as '{md5}fbed5c0c9bf82b7746e01f15f89d184f'
May 10 21:46:50 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/parser/functions/validate_hash.rb]/ensure)
defined content as '{md5}e9cfaca68751524efe16ecf2f958a9a0'
May 10 21:46:51 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/parser/functions/count.rb]/ensure) defined
content as '{md5}9eb74eccd93e2b3c87fd5ea14e329eba'
May 10 21:46:51 NWAPPLIANCE16378 puppet-agent[1754]: (/File[/var/lib/puppet/lib/puppet/parser/functions/validate_bool.rb]/ensure)
defined content as '{md5}a712634a000024398b3c6cd4ecc46463'
```

A Refreshing Agent

The puppet agent will re-execute every 30 minutes to ensure the managed host always has a fresh configuration. In some rare circumstances, such as when activating audit logging, SA will make certain types of configuration changes which requires the puppet agent to run again to activate the changes. Sometimes the UI will use **mcollective** to trigger another puppet run (see below) by sending the agent process the appropriate signal, but sometimes it won't. If you're impatient, you can always restart the puppet agent from the command line to trigger a new run much sooner with "**service puppet restart**".

This 30 minute cycling of the agent also comes into play during SA upgrades. Sometimes an upgrade requires multiple agent runs in order for all new configuration changes to take effect, and because there are **20?** minutes between runs, an upgrade can be mistaken to have failed if everything doesn't come up quickly after a reboot, when really it just needs some more time for another catalog run to complete. If, after a host has been upgraded and

rebooted, Health & Wellness tells you that there are service problems after the host comes back up, then I generally watch the agent logs to confirm whether the agent is still executing. If it isn't, restarting puppet one more time will resolve these problems in a good number of cases.

Top Tip: Linux signal USR1 will cause the puppet agent to reload and start a new catalog run. Signal TERM will stop the agent.

```
[root@concentrator ~]# grep puppet-agent /var/log/messages | grep -i caught
May 10 03:59:34 concentrator puppet-agent[1682]: Caught USR1; calling reload
May 10 05:01:36 concentrator puppet-agent[1682]: Caught TERM; calling stop

[root@concentrator ~]# tail -f /var/log/messages | grep puppet-agent &

[root@concentrator ~]# kill -USR1 `pgrep puppet`
[root@concentrator ~]# Mar 9 16:32:27 concentrator puppet-agent[1723]: Caught USR1; storing reload
Mar 9 16:32:28 concentrator puppet-agent[1723]: Processing reload
Mar 9 16:32:46 concentrator puppet-agent[6527]: (/Stage[main]/Ssh/Exec[fix-ssh]/returns) executed successfully
Mar 9 16:33:11 concentrator puppet-agent[6527]: (/Stage[main]/Yumconfig/Exec[disable-Centos-Repos]/returns) executed successfully
Mar 9 16:33:18 concentrator puppet-agent[6527]: Finished catalog run in 33.66 seconds

[root@concentrator ~]# kill -TERM `pgrep puppet`
[root@concentrator ~]# Mar 9 16:33:55 concentrator puppet-agent[1723]: Caught TERM; storing stop
Mar 9 16:33:59 concentrator puppet-agent[1723]: Processing stop
```

Starting and Stopping Puppet Agent

To start the puppet agent service: **service puppet start**

```
[root@concentrator ~]# service puppet start
Starting puppet agent: [ OK ]
```

To stop the puppet agent service: **service puppet stop**

```
[root@concentrator ~]# service puppet stop
Stopping puppet agent: [ OK ]
```

To check the puppet agent status: **service puppet status**

```
[root@concentrator ~]# service puppet status
puppet (pid 10452) is running...
```

Watching the Puppet Agent Logs

If you grow tired of having to watch /var/log/messages for puppet messages, especially when they're mixed in with other application logs, you can **tail** the logfile and pipe the output to **grep** using the string "**puppet-agent"**.

```
[root@concentrator ~]# tail -f /var/log/messages | grep puppet-agent
```

Running the Puppet Agent from the Command Line

If you'd rather not watch logs at all, the puppet agent can be run directly from the command line with the command "**puppet agent -t**". The agent will then execute its catalog.

Top Tip: After doing this, don't forget to start the agent again with "**service puppet start**", or you won't get any new configurations!

```
[root@concentrator ~]# service puppet stop
Stopping puppet agent: [ OK ]

[root@concentrator ~]# puppet agent -t
Info: Retrieving pluginfacts
Info: Retrieving plugin
Info: Loading facts
Info: Caching catalog for 4fe924bb-8ed9-4f65-a731-8767f0a37992
Info: Applying configuration version '1489075329'
Notice: /Stage[main]/Ssh/Exec[fix-ssh]/returns: executed successfully
Notice: /Stage[main]/Yumconfig/Exec[disable-Centos-Repos]/returns: executed successfully
Notice: Finished catalog run in 36.44 seconds

[root@concentrator ~]# service puppet start
Starting puppet agent: [ OK ]
```

Top Tip: You can press CTRL-C to stop the puppet agent if it's running from the command line, though it can take many seconds for it to actually stop.

Viewing Puppet Node Classification

To view what type of node puppet thinks a host is, use the **/etc/puppet/scripts/mongo_enc.py** script. Pass in the node ID of the host to be identified as a parameter.

```
[root@sa scripts]# ./mongo_enc.py f11f6740-7ad4-45d3-baa0-10516dea1b16
_class: com.rsa.smc.sa.admin.management.domain.bean.Nodes
classes:
 appliance: ''
 base: ''
 concentrator: ''
```

Every SA host should have the 'base' class defined, and every SA Core host should also have the 'appliance' class defined. The 'appliance' class refers to the the legacy appliance service.

Note that if the mongo_enc script isn't able to find the node ID, it will not output an error, but rather list no '_class' entry and will only show 'base' as a class.

```
[root@sa ~]# /etc/puppet/scripts/mongo_enc.py badnodeid

classes:
 base: ''
```

Top Tip: Do not try to modify the ENC Mongo database to change a node's definition. Because SA also stores its own definitions about hosts and their installed services in the SMS service, it will not pick up the changes you make to the ENC, and you'll end up with a bad configuration. The only legitimate reason a node classification might need to be changed is because a critical service was stopped at the time the puppet agent first ran on the host. Puppet classifies nodes based on what services are **running** at the time of the first agent run, not what services are installed. This is most common with the Warehouse Connector service, because it gets installed by default on Decoder appliances, but it is by default not started (most customers don't use this service), therefore puppet does not classify the node as a Warehouse Connector unless the administrator starts the WC service prior to enabling the host in SA. Should this ever happen to you:

1. **Remove and Repurpose** the host in the UI in **Administration->Hosts**.
2. Reset the puppet configuration of the host (see instructions later in this lesson).
3. Start the service which wasn't previously part of the node classification (i.e. the Warehouse Connector). E.g. "**start nwwarehouseconnector**"
4. Start the puppet agent on the host with "**puppet agent start**".
5. Enable the host in **Administration->Hosts** in the UI.

Viewing Puppet SSL Certificates

On the puppet master, one can view the certificates which have been signed. Each signed certificate is located in **/var/lib/puppet/ssl/ca/signed**, and is identified by its node ID:

```
[root@sa ca]# ls -l /var/lib/puppet/ssl/ca/signed
total 28
-rw-r--r--. 1 puppet puppet 2009 Feb 18 17:53 4fe924bb-8ed9-4f65-a731-8767f0a37992.pem
-rw-r--r--. 1 puppet puppet 2009 Jan 19 19:40 a666cbc4-baa8-4733-82d8-4664985d2c10.pem
-rw-r--r--. 1 puppet puppet 2009 Jan 19 19:41 bd3ffdfa-f606-4881-ae8a-97a727465601.pem
-rw-r--r--. 1 puppet puppet 2102 Jan 18 23:18 d015576d-c69d-4f06-865c-3a58711a23e5.pem
-rw-r--r--. 1 puppet puppet 2009 Jan 19 19:41 d7eca16a-a11d-4bc4-a0df-42b0ac9d2cbd.pem
-rw-r--r--. 1 puppet puppet 2009 Jan 19 19:44 e80a3c23-c9ff-4c0d-acc0-4d40336c7f56.pem
-rw-r--r--. 1 puppet puppet 2009 Jan 19 19:44 f11f6740-7ad4-45d3-baa0-10516dea1b16.pem
```

To view the decoded contents of the certificate, use "**openssl x509 -in <filename.pem> -text**". Notice the Subject: line. Its value will be "**CN= *<puppet node ID>***".

```
[root@sa signed]# openssl x509 -in f11f6740-7ad4-45d3-baa0-10516dea1b16.pem -text

Certificate:
 Data:
 Version: 3 (0x2)
 Serial Number: 8 (0x8)
 Signature Algorithm: sha256WithRSAEncryption
 Issuer: CN=Puppet CA: d015576d-c69d-4f06-865c-3a58711a23e5
 Validity
 Not Before: Jan 18 19:44:16 2017 GMT
 Not After : Jan 18 19:44:16 2022 GMT
 Subject: CN=f11f6740-7ad4-45d3-baa0-10516dea1b16
 Subject Public Key Info:
 Public Key Algorithm: rsaEncryption
 Public-Key: (4096 bit)
 Modulus:
 00:ae:9a:9a:13:c5:b4:b9:73:c0:ad:57:3d:97:a1:
 79:0e:f4:d2:51:85:33:6b:0b:2c:b3:f8:78:38:61:
 6e:9d:46:3b:b1:13:44:87:3a:36:89:0a:ef:62:26:
 42:33:05:fb:8f:57:6e:b5:4e:df:26:58:1f:9f:cf:
 c0:3c:41:20:95:e5:f1:87:15:54:85:01:06:7d:2c:
 61:b9:cc:ad:92:bb:98:ee:c6:25:ca:c6:23:19:99:
 7f:1a:f8:74:d5:55:7b:48:8b:fe:e9:18:d3:9b:1c:
 75:56:ae:9e:34:6a:a3:c2:02:08:0b:c4:d8:29:46:
 a1:e2:0f:1a:db:42:aa:c5:3e:10:b7:4c:d0:06:dd:
 8b:0a:48:b7:ed:19:0e:2f:c2:51:6e:65:15:50:b9:
 b2:97:e2:e0:bd:8e:0e:53:86:86:ae:29:f9:ca:16:
 42:c3:2b:99:ec:eb:d3:2c:a6:3a:f1:d6:d0:1f:7d:
 b0:53:da:db:9a:7f:a9:55:1b:93:67:c3:95:5e:88:
 a7:fc:44:7e:44:56:99:48:11:5e:01:1f:37:6b:c8:
 15:be:3b:ca:b5:a6:17:23:34:62:17:4f:c2:e3:60:
 9e:e7:42:39:b8:79:24:6c:31:24:69:a6:8d:e8:85:
 f1:f9:55:13:ec:5d:4b:80:b1:96:a6:b4:74:5b:3f:
 ce:33:17:21:ad:39:3d:60:de:40:1a:31:6d:6a:77:
 43:d3:b0:27:94:21:ac:85:bb:8f:03:69:87:85:39:
 93:c4:c8:e2:61:52:28:06:8b:2b:3b:9e:05:6c:0e:
 0a:b8:8c:99:42:5d:f7:da:59:e1:70:71:2f:17:d4:
 39:96:24:6c:60:bb:7c:0d:55:df:b6:17:3a:a9:b5:
 ad:d6:8e:64:09:7f:7b:c0:ce:f6:06:c5:2e:fd:c8:
 51:c3:0a:1c:66:b8:9a:82:fd:cb:87:84:aa:19:eb:
 c4:46:a5:59:39:e0:3c:c3:be:ce:06:10:06:62:18:
 bb:36:41:2e:16:04:a4:53:cf:10:71:24:cc:88:4d:
 a4:ba:cc:a2:bd:18:2d:1f:b8:f2:fd:5f:95:91:c3:
 86:57:0f:26:ba:0b:af:2b:7d:0e:b0:85:43:ad:66:
 01:48:0b:37:3a:bd:3a:e8:a6:7b:54:ad:f6:30:16:
 99:54:79:c9:97:74:2a:da:54:96:f8:12:44:61:c0:
 c2:9e:bd:67:d4:b1:8e:08:f5:07:5d:70:d7:5f:ac:
 86:73:66:6d:24:39:d2:34:e8:5c:83:75:14:a2:0c:
 9d:84:10:a0:69:64:06:a5:d4:dd:8c:54:1f:86:55:
 8c:7b:f2:12:89:de:db:52:6a:e7:86:47:78:62:80:
 bb:47:95
 Exponent: 65537 (0x10001)
 X509v3 extensions:
 X509v3 Subject Key Identifier:
 82:27:EB:E0:C2:50:24:D7:20:0A:02:16:7A:5F:22:D6:B1:6B:DF:41
 X509v3 Extended Key Usage: critical
 TLS Web Server Authentication, TLS Web Client Authentication
 Netscape Comment:
 Puppet Ruby/OpenSSL Internal Certificate
 X509v3 Authority Key Identifier:
 keyid:66:EE:3C:8E:9A:19:AC:5C:1C:8E:D8:D4:81:F4:E8:32:5A:11:99:A1
X509v3 Basic Constraints: critical
 CA:FALSE
 X509v3 Key Usage: critical
 Digital Signature, Key Encipherment
 Signature Algorithm: sha256WithRSAEncryption
 7d:e8:74:2c:f9:13:60:52:c1:92:c3:42:7a:97:3b:9e:df:d5:
 03:59:3b:cd:9c:be:18:e5:50:b7:ef:10:de:e7:2e:17:5b:4c:
 bf:37:69:74:05:84:d5:73:2e:89:e3:b7:92:cf:9e:f7:80:81:
 7c:64:f3:9d:83:1c:81:51:2b:67:c6:4d:2f:2d:94:91:0b:66:
 3b:82:fd:f6:c5:e4:c3:96:bd:7f:74:2b:62:5d:0a:59:ce:a2:
 97:12:30:e7:b0:53:75:21:fd:84:f7:83:74:78:ac:52:c6:1f:
 4f:7a:78:c6:50:d9:08:16:0b:65:3e:91:7f:c5:7c:37:c9:53:
 a7:e6:7a:04:51:9d:43:34:5d:53:d6:b9:ff:8b:7d:a2:bc:5f:
 50:fe:a5:bc:31:c3:13:bd:8a:1b:1b:6d:16:fc:24:51:fa:ca:
 8a:84:0c:d8:8a:dc:e3:4f:b8:bd:e7:aa:e0:68:c1:6d:01:af:
 c3:bb:e3:94:92:36:fe:54:87:75:0f:48:ec:71:e7:1c:58:04:
 2a:da:98:ef:95:97:19:5b:5b:61:4a:0a:3f:67:81:c5:4b:c6:
 14:3d:4d:07:2e:c2:29:74:08:43:84:2e:ac:eb:4f:91:1b:62:
 7c:b6:9a:38:b5:eb:94:6d:34:2a:d7:ee:a3:d0:28:68:c4:ed:
 5e:e4:ed:98:02:8b:6a:2e:e6:d4:6c:72:d2:eb:78:f9:d1:10:
```

```
 c1:dd:a6:2d:a3:e0:bb:85:4e:25:0c:fb:99:88:ae:37:e0:9e:
 8a:aa:eb:ed:63:57:35:9e:23:c4:b8:27:54:cb:08:68:16:33:
 a4:63:28:b6:63:90:2a:ca:ef:2b:97:80:7e:47:6a:01:8b:f0:
 52:30:df:f2:6b:8d:6a:b7:6a:a1:0f:9d:07:2b:dc:38:f1:d7:
 3b:f2:84:b2:dd:4d:e6:cb:f4:12:86:57:c7:a2:0b:55:d3:11:
 14:2c:4b:b8:ef:ea:17:ff:62:4e:af:2a:ed:f7:56:9f:92:8d:
 63:d3:0e:02:26:1b:78:d3:d8:c6:ab:d5:5b:0c:03:18:e7:1c:
 04:85:e7:8e:0a:50:8c:cb:73:05:6d:02:87:00:6c:90:68:e6:
 76:07:46:70:b3:38:a8:19:72:6f:c1:d8:4c:82:ef:76:d3:0c:
 03:a7:00:3e:9c:8b:70:0f:30:7c:9b:aa:90:e8:00:4c:17:1b:
 89:c4:13:22:6a:b6:d5:24:3d:fd:a8:1d:2a:99:4e:8c:18:e0:
 bd:ca:2e:e5:fb:ec:80:7d:30:c5:bb:78:9b:da:6b:7e:77:16:
 fa:f6:52:87:2d:15:24:f5:95:e4:1f:bd:ac:78:08:e8:99:c8:
 10:ef:e4:43:38:66:8a:60
-----BEGIN CERTIFICATE-----
MIIFnzCCA4egAwIBAgIBCDANBgkqhkiG9w0BAQsFADA6MTgwNgYDVQQDDC9QdXBw
ZXQgQ0E6IGQwMTU1NzZkLWM2OWQtNGYwNi04NjVjLTNhNTg3MTFhMjNlNTAeFw0x
NzAxMTgxOTQ0MTZaFw0yMjAxMTgxOTQ0MTZaMC8xLTArBgNVBAMMJGYxMWY2NzQw
LTdhZDQtNDVkMy1iYWEwLTEwNTE2ZGVhMWIxNjCCAiIwDQYJKoZIhvcNAQEBBQAD
ggIPADCCAgoCggIBAK6amhPFtLlzwK1XPZeheQ700lGFM2sLLLP4eDhhbp1GO7ET
RIc6NokK72ImQjMF+49XbrVO3yZYH5/PwDxBIJXl8YcVVIUBBn0sYbnMrZK7mO7G
JcrGIxmZfxr4dNVVe0iL/ukY05scdVaunjRqo8ICCAvE2ClGoeIPGttCqsU+ELdM
0AbdiwpIt+0ZDi/CUW5lFVC5spfi4L2ODlOGhq4p+coWQsMrmezr0yymOvHW0B99
sFPa25p/qVUbk2fDlV6Ip/xEfkRWmUgRXgEfN2vIFb47yrWmFyM0YhdPwuNgnudC
Obh5JGwxJGmmjeiF8flVE+xdS4Cxlqa0dFs/zjMXIa05PWDeQBoxbWp3Q9OwJ5Qh
rIW7jwNph4U5k8TI4mFSKAaLKzueBWwOCriMmUJd99pZ4XBxLxfUOZYkbGC7fA1V
37YXOqm1rdaOZAl/e8DO9gbFLv3IUcMKHGa4moL9y4eEqhnrxEalWTngPMO+zgYQ
BmIYuzZBLhYEpFPPEHEkzIhNpLrMor0YLR+48v1flZHDhlcPJroLryt9DrCFQ61m
AUgLNzq9Ouime1St9jAWmVR5yZd0KtpUlvgSRGHAwp69Z9Sxjgj1B11w11+shnNm
bSQ50jToXIN1FKIMnYQQoGlkBqXU3YxUH4ZVjHvyEone21Jq54ZHeGKAu0eVAgMB
AAGjgbowgbcwHQYDVR0OBBYEFIIn6+DCUCTXIAoCFnpfItaxa99BMCAGA1UdJQEB
/wQWMBQGCCsGAQUFBwMBBggrBgEFBQcDAjA1BglghkgBhvhCAQ0EKFB1cHBldCBS
dWJ5L09wZW5TU0wgSW50ZXJuYWwgQ2VydGlmaWNhdGUwHwYDVR0jBBgwFoAUZu48
jpoZrFwcjtjUgfToMloRmaEwDAYDVR0TAQH/BAIwADAOBgNVHQ8BAf8EBAMCBaAw
DQYJKoZIhvcNAQELBQADggIBAH3odCz5E2BSwZLDQnqXO57f1QNZO82cvhjlULfv
EN7nLhdbTL83aXQFhNVzLonjt5LPnveAgXxk852DHIFRK2fGTS8tlJELZjuC/fbF
5MOWvX90K2JdClnOopcSMOewU3Uh/YT3g3R4rFLGH096eMZQ2QgWC2U+kX/FfDfJ
U6fmegRRnUM0XVPWuf+LfaK8X1D+pbwxwxO9ihsbbRb8JFH6yoqEDNiK3ONPuL3n
quBowW0Br8O745SSNv5Uh3UPSOxx5xxYBCramO+VlxlbW2FKCj9ngcVLxhQ9TQcu
wil0CEOELqzrT5EbYny2mji165RtNCrX7qPQKGjE7V7k7ZgCi2ou5tRsctLrePnR
EMHdpi2j4LuFTiUM+5mIrjfgnoqq6+1jVzWeI8S4J1TLCGgWM6RjKLZjkCrK7yuX
gH5HagGL8FIw3/JrjWq3aqEPnQcr3Djx1zvyhLLdTebL9BKGV8eiC1XTERQsS7jv
6hf/Yk6vKu33Vp+SjWPTDgImG3jT2Mar1VsMAxjnHASF544KUIzLcwVtAocAbJBo
5nYHRnCzOKgZcm/B2EyC73bTDAOnAD6ci3APMHybqpDoAEwXG4nEEyJqttUkPf2o
HSqZTowY4L3KLuX77IB9MMW7eJvaa353Fvr2UoctFST1leQfvax4COiZyBDv5EM4
Zopg
-----END CERTIFICATE-----
```


Puppet Command Cheat Sheet

Command	Role	Description
puppet agent -t --waitforcert 10	Agent	Runs puppet agent on its first run and checks for a signed certificate every 10 seconds.
puppet agent -t	Agent	Normal puppet agent run. Exits if certificate has not been signed by puppet master.
puppet agent -t --debug	Agent	Puppet run with full debug output.
puppet cert list	Master	Lists outstanding certificate requests (CSR's).

puppet cert list --all	Master	Lists signed certificates in addition to CSR's.
		
		`[root@sa puppet]# puppet cert list --all`
		`+ "4fe924bb-8ed9-4f65-a731-8767f0a37992" (SHA256)`
		`3F:C1:24:9E:DF:E7:58:BD:C4:ED:64:A4:78:7C:2C:06:92:BC:38:5A:45:16:EC:90:69:AD:C3:03:4F`
		`+ "a666cbc4-baa8-4733-82d8-4664985d2c10" (SHA256)`
		`A4:B5:B0:C0:AB:7C:65:A6:50:4F:FC:30:19:10:0B:34:78:CD:75:25:E8:83:19:F2:41:B3:17:F0:4F`
		`+ "bd3ffdfa-f606-4881-ae8a-97a727465601" (SHA256)`
		`4F:3F:68:46:95:BC:DB:96:E0:67:EE:43:1B:4D:BC:3D:7F:54:4A:03:13:08:76:D4:33:48:50:E8:93`
		`+ "d015576d-c69d-4f06-865c-3a58711a23e5" (SHA256)`
		`47:5B:F1:74:03:7D:CE:06:2C:AA:77:AF:41:CF:09:97:43:85:0E:F6:9B:A2:FF:D4:A2:FC:A9:A2:39`
		`(alt names: "DNS:d015576d-c69d-4f06-865c-3a58711a23e5", "DNS:puppetmaster.local")`
		`+ "d7eca16a-a11d-4bc4-a0df-42b0ac9d2cbd" (SHA256)`
		`F7:9A:EE:D9:15:93:46:A4:43:1B:A3:5D:45:32:66:02:A5:7D:83:CA:C9:47:79:02:5F:CC:70:83:44`
		`+ "e80a3c23-c9ff-4c0d-acc0-4d40336c7f56" (SHA256)`
		`38:16:86:10:6D:4A:EF:FB:0A:BF:30:7B:55:2E:8F:EE:61:B2:1B:AF:98:76:2E:98:97:18:84:A2:7D`
		`+ "f11f6740-7ad4-45d3-baa0-10516dea1b16" (SHA256)`
		`09:65:77:AC:CF:EA:10:2B:E3:30:03:C8:3E:70:0F:75:41:F8:28:BA:21:F7:5D:B0:FD:C5:A8:FB:B4`

puppet cert clean <*nodeID*>	Master	Revokes a host's certificate and removes certificate files related to it.

puppet ca list --all	Master	Supposed to list certificates in the CA.
		
		Note that puppet will always respond with the following error. This error is itself erroneous, and is c a bug in puppet. "**puppet cert list --all**" should work instead. We've included this command in ca is ever fixed.
		
		`[root@sa puppet]# puppet ca list --all`
		`Error: The certificate retrieved from the master does not match the agent's private ke`
		`Certificate fingerprint:`
		`3F:C1:24:9E:DF:E7:58:BD:C4:ED:64:A4:78:7C:2C:06:92:BC:38:5A:45:16:EC:90:69:AD:C3:03:4F`
		`To fix this, remove the certificate from both the master and the agent and then start`
		`run, which will automatically regenerate a certficate.`
		`On the master:`
		` puppet cert clean d015576d-c69d-4f06-865c-3a58711a23e5`
		`On the agent:`
		` 1a. On most platforms: find /var/lib/puppet/ssl -name d015576d-c69d-4f06-865c-3a58711`
		`-delete`
		` 1b. On Windows: del "/var/lib/puppet/ssl/d015576d-c69d-4f06-865c-3a58711a23e5.pem" /f`
		` 2. puppet agent -t`
		
		`Error: Try 'puppet help ca list' for usage`

puppet ca list --pending	Master	CA command to list pending CSR's.
puppet ca list --no-pending	Master	The --no-pending switch can be appended to the "puppet ca list" CA command to exclude pending from the listing.
puppet ca list --signed	Master	Supposed to list signed certificates in the CA. Errors out with the same error as "puppet ca list --al Included in case this gets fixed in a future release.
puppet ca list --no-signed	Master	The --no-signed switch can be appended to the "puppet ca list" CA command to exclude signed ce from the listing.
puppet ca destroy <node*ID*>	Master	Deletes a pending certificate request.
/etc/puppet/scripts/delNode.py <nodeID>	Master	Removes a host from the External Node Classifier, or ENC.
/etc/puppet/scripts/mongo_enc.py <nodeID>	Master	Obtains the classes defined in the ENC for a given puppet managed host.

Generating a New Node ID

If, and only if, a puppet node is not being managed by a puppet master, and should you have a need to change the node ID of a puppet node, here is the procedure. This could be the case for virtual machines which are being copied.

1. Run: **rm -f /var/lib/puppet/node_id**
2. Run: **/etc/puppet/scripts/nodeId.py**
3. Run: **NODEID=`cat /var/lib/puppet/node_id`; sed -i'' -r "s/certname\s?=\s?.*$/certname = $NODEID/g" /etc/puppet/puppet.conf**

Resetting Puppet Configuration

It is possible to reset puppet's configuration back to its initial state with this tool developed by RSA, and customised by Kensington Tech. Note that this also resets the RabbitMQ configuration, as well.

Puppet_agent_reset_certs_non_interact_V0.51-ktal.sh

The KTA version of scripts removes the **/var/lib/puppet/ssl** directory to avoid any certificate conflicts (this will automatically be re-created the next time the agent runs). It also does not re-start the puppet agent at the end of the script, in case the admin needs to make additional changes.

To run the script, SCP this file to the SA host in question, and execute the script with "**./Puppet_agent_reset_certs_non_interact_V0.51-ktal.sh**".

```
# ./Puppet_agent_reset_certs_non_interact_V0.51-ktal.sh

Stopping puppet agent: [ OK ]
Stopping node sa@localhost ...
Resetting node sa@localhost ...
Stopping rabbitmq-server: rabbitmq-server.
Stopping collectd: [ OK ]
Shutting down mcollective: [ OK ]
killing Erlang.............Killed
This is not a Packet Decoder
This is not a Broker
This is not a Log Decoder
This is not a WareHouse Connector
This is not a Log Decoder
This is not an Archiver
This is not an Archiver/Workbench
This is a core appliance Decoder, Concentrator or Archiver
```

Correcting Annoyances

Puppet sometimes does some annoying things, like re-starting a service you deliberately wanted to stop (such as Incident Management, if you don't use it), or removing the custom firewall port you opened in **iptables**.

CRITICAL Top Tip: You absolutely must remember to perform these customisations again after every upgrade of SA, as the manifest files will likely be overwritten during an upgrade, thus negating your changes.

Opening a New Firewall Port

If you'd like to permanently open a new firewall port, you'll need to add your port to a puppet manifest. If you merely modify the iptables configuration of a host, then puppet will swiftly undo your changes.

1. On the puppet master, edit the file **/etc/puppet/modules/base/manifests/init.pp**
2. Close to the top of the file, after the first include statement but before the first firewall statement, add the following text block, modifying port and proto (**tcp** or **udp**). Note that on the first line, the description must be in the format "*N sometext*", where N is a number. The rule will not apply if not in this format.

   ```
   firewall {'200 Some description of this rule':
    port => [12345],
    proto => tcp,
    action => accept
    }
   ```

3. Save the file and quit.
4. On the puppet master, issue the command "**server puppetmaster restart**".

The next time your puppet agents run, they will all open the new port in their iptables configuration.

If you don't want to open the port on all SA hosts, you cannot target specific nodes, but instead of modifying the **base** manifest file, you can instead modify the manifest for the specific class of device you wish to target. For instance, to open the port only on Concentrators, modify the file **/etc/puppet/modules/concentrator/manifests/init.pp** instead of **/etc/puppet/modules/base/manifests/init.pp.**

Disabling Incident Management

If you don't use SA's Incident Management feature, it can consume quite a bit of system resources on your SA host, which might be better used elsewhere. The problem is, if you stop the service and disable it from running, puppet will just start it right back up again. To stop this madness:

1. On the puppet master, edit the file **/etc/puppet/modules/incident-management/manifests/init.pp**
2. At the very bottom of the file, just above the last closing bracket, change the service block to read thus (so that enable is now **false** instead of **true**, and ensure is **stopped** rather than **running**):

```
service { 'rsa-im':
 enable => false,
 ensure => stopped,
 subscribe => [Class['rsa-java'], file['carlos.keystore']]
 }
```

3. Save the file and quit.
4. On the puppet master, issue the command "**server puppetmaster restart**".

During the next catalog run, the incident management service ('rsa-im') will stop on its own and not start back up again.

Interactions with MCollective

The **mcollective**, or Marionette Collective, service is a separate service which is also developed by Puppet. Its job is to orchestrate remote commands run by the SA server host against all managed puppet hosts. What commands get run isn't very important, but it's important to know that **mcollective** will sometimes trigger the stopping, starting, and restarting of the puppet agent at various times, especially during upgrades.

Occasionally during an upgrade, mcollective will stop the puppet agent but for an unknown reason, never re-start it. If you find that the puppet agent it isn't running when it should be (especially after a reboot), then you may need to start puppet manually.

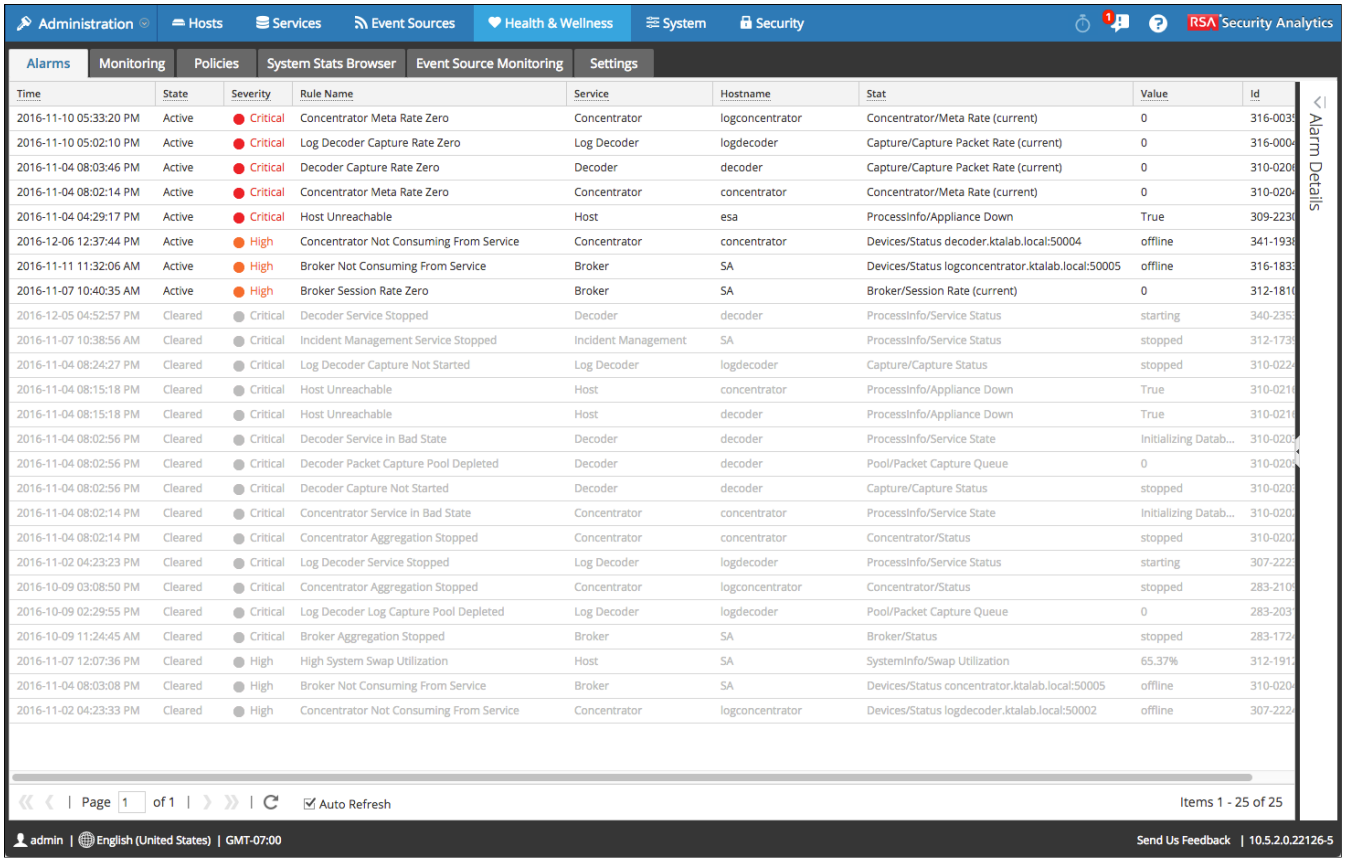
33. Health & Wellness

Prior to version 10.4 of SA, its Health & Wellness functionality was a bit of a hodge-podge, and difficult to use at best. In version 10.4 of Security Analytics, for the first time, RSA made the health of the SA platform a proper part of the product story by introducing an ad hoc UI for monitoring system functionality. It built on this further in version 10.5 by incorporating a mechanism for proactively alerting on health problems, based on policies. You'll see Health & Wellness referred to as H&W in this article.

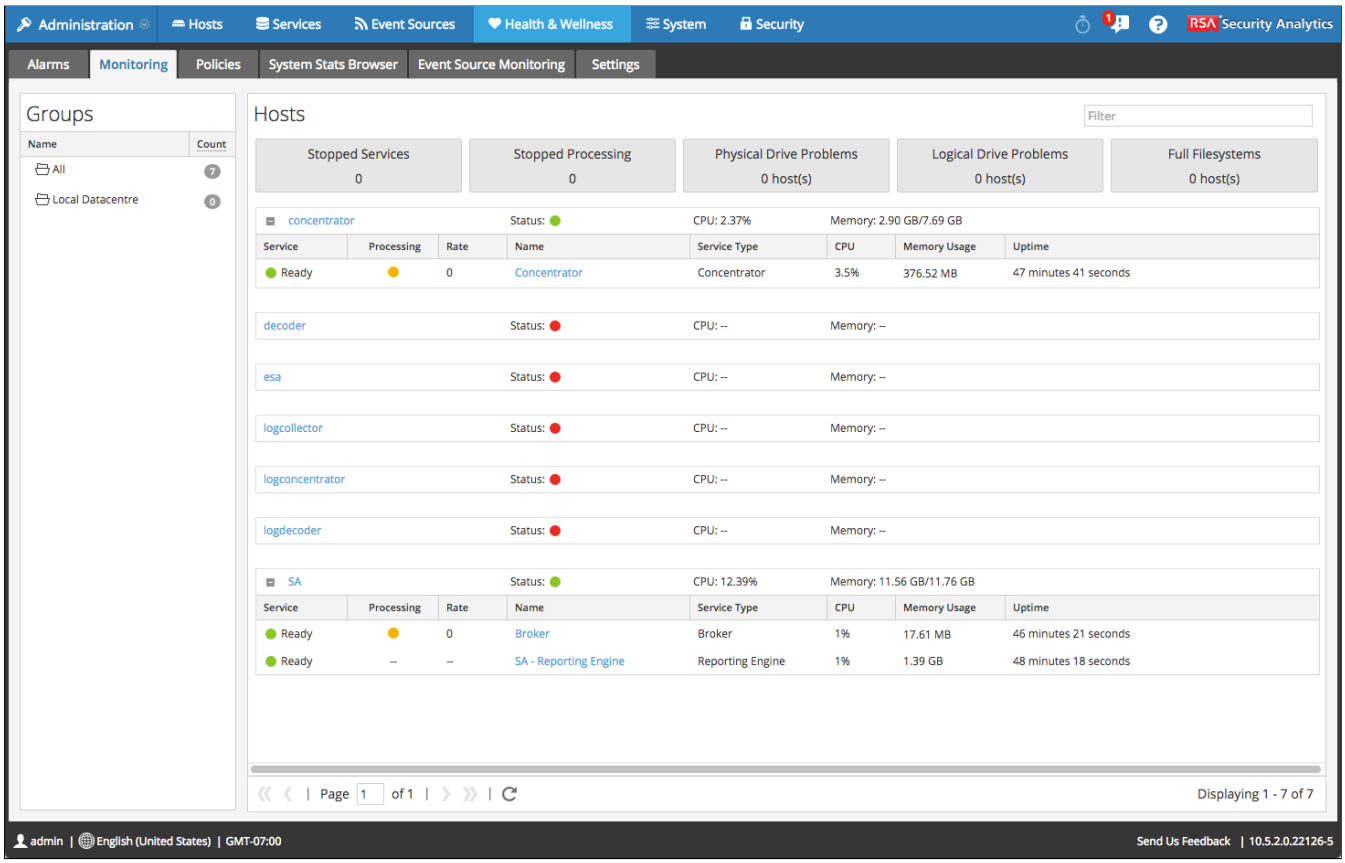
Intro

SA's Health & Wellness functionality is comprised of four main UI pages, all found under **Administration->Health & Wellness**:

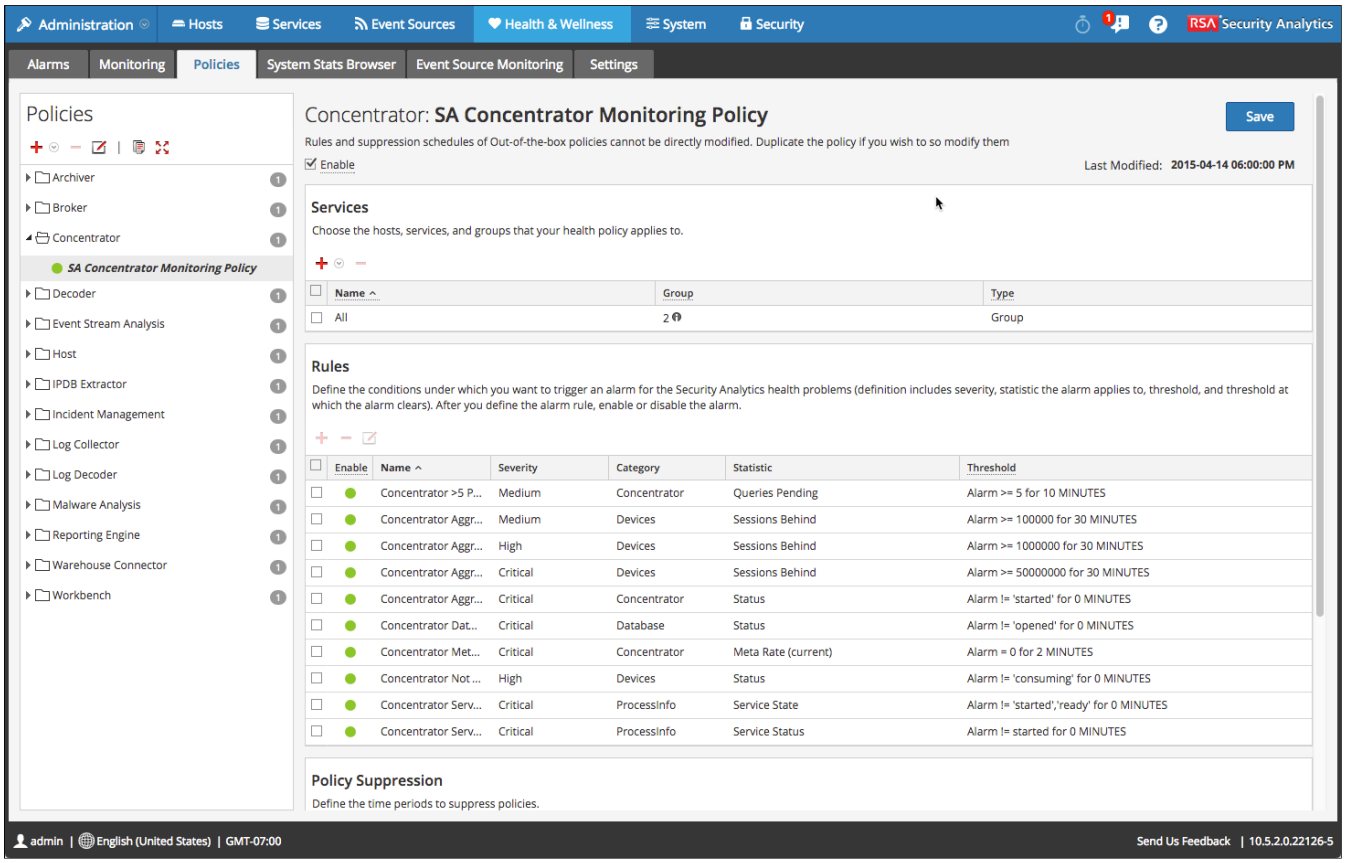
- **Alarms** - Shows active and inactive health alarms in a rolled-up view.



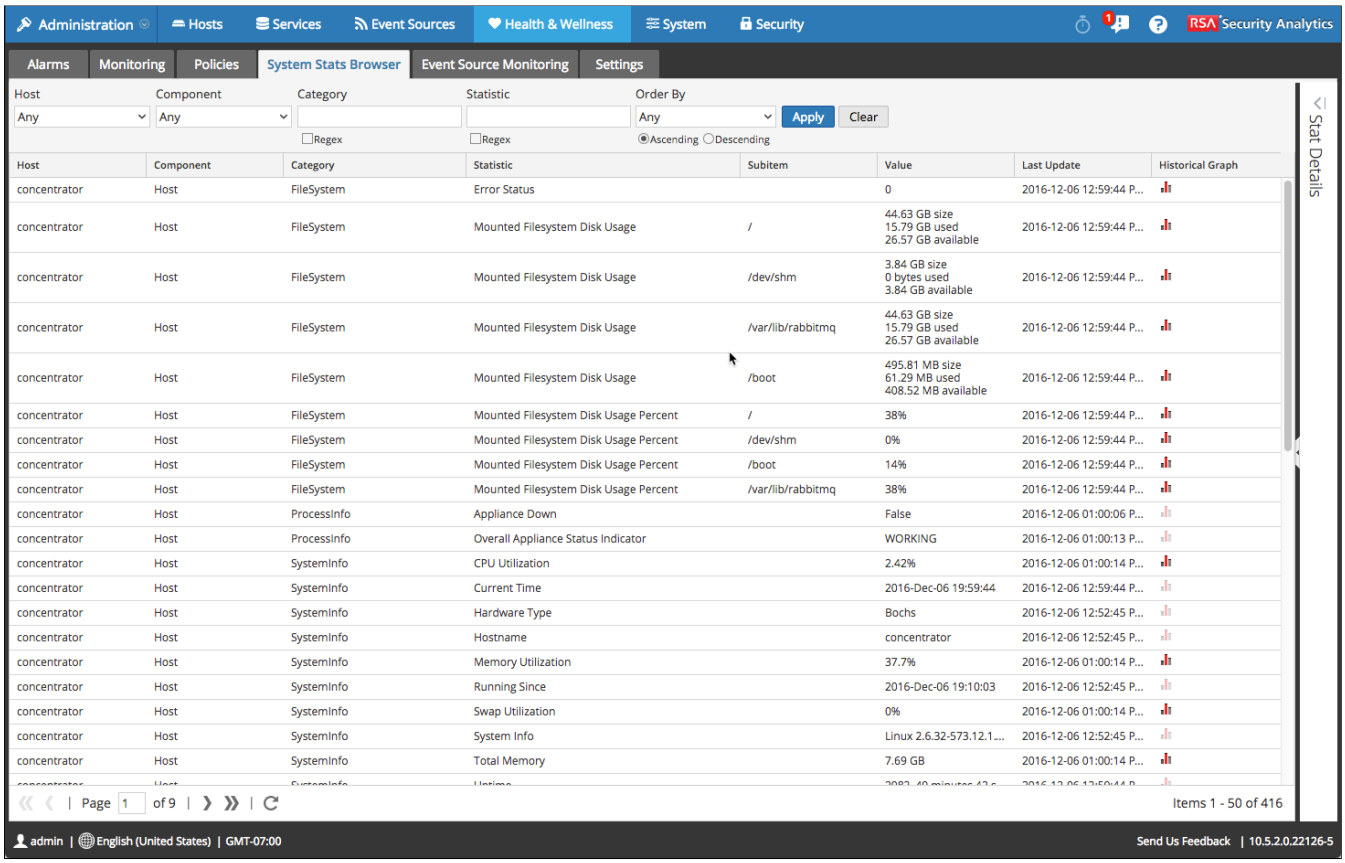
- **Monitoring** - This is the ad hoc view of H&W. Its functionality is somewhat limited and is not configurable. It can also sometimes be misleading, but it can sometimes still be useful.

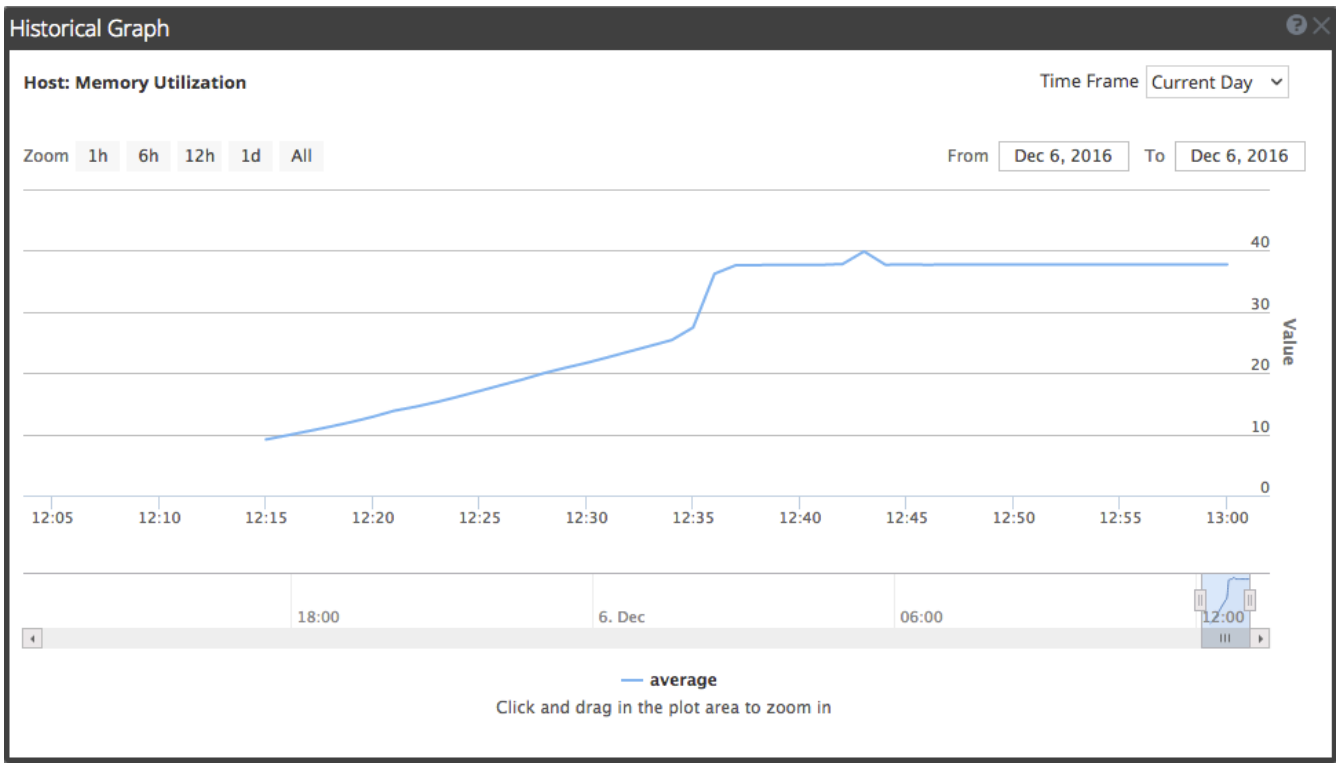


- **Policies** - These are the H&W policies which control when SA will generate a Health alarm, and under what conditions.



- **System Stats Browser** - A power user tool for viewing the raw health statistics and RRD graphs.

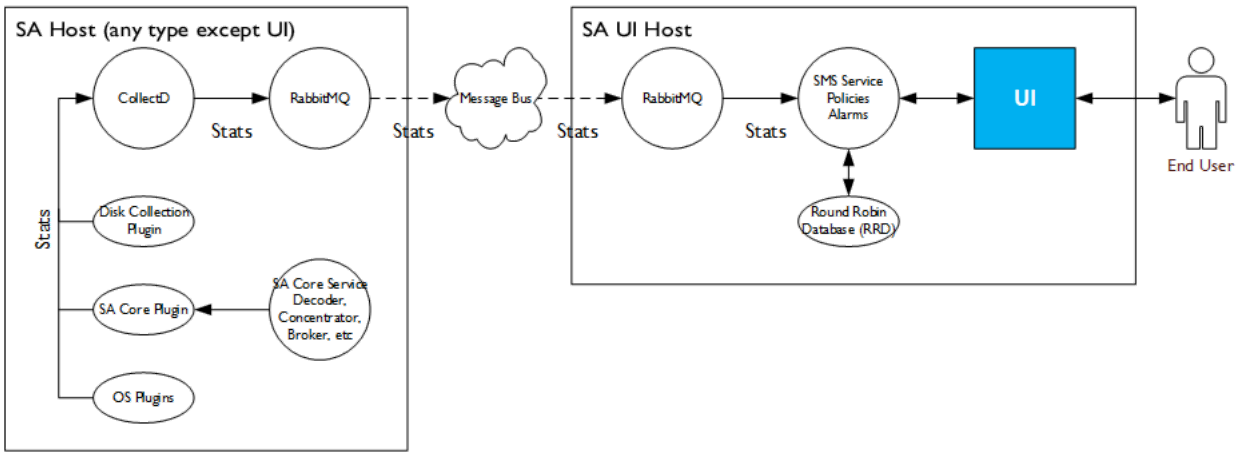




Under the Bonnet - Statistics, CollectD, RabbitMQ, and SMS

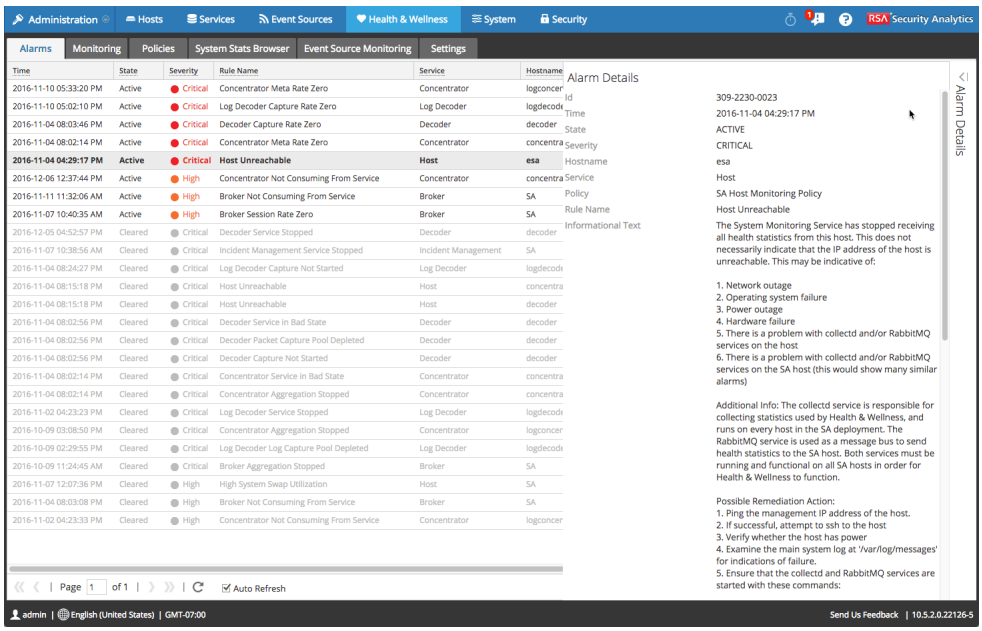
SA's H&W functionality is based on the collection and aggregation of health statistics from each SA host. The underlying glue service that enables the process of statistics collection is called **CollectD**, which is an open source product. It in turn feeds RabbitMQ, which transports the stats to the SA UI host. The RabbitMQ service running on the SA host then picks up the stats, which then feeds them to the SMS service, which in turn stores them in a Round-Robin-Database (RRD), and also can generate alarms from the statistics.

Health & Wellness Data Flow

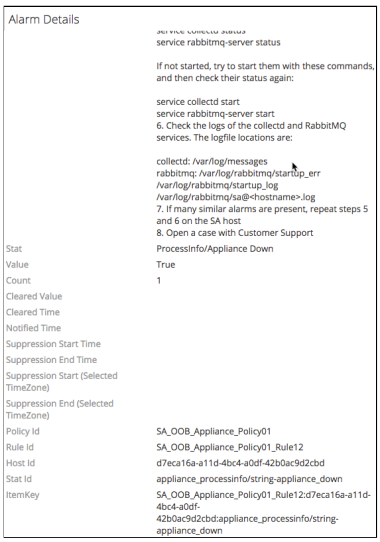


Alarms View

This is fairly self-explanatory. Active Health alarms are shown with a colourful severity, and inactive alarms (i.e. ones that have cleared) are greyed out. You can gain additional details on an alarm by clicking on the alarm andthen clicking on the Alarm Details drawer on the right-hand side of the page. This will give you some low-level info about the alarm, and some helpful text on resolving it.



Alarms View



Alarm Details Pane

Monitoring View

The H&W Monitoring view, or as I call it, the Ad Hoc view, was the first UI component to be introduced for Health & Wellness. It presents a rolled-up, 1,000 foot view of the health of one's SA hosts and services. It relies on hard-coded logic to produce warning lights of green, amber, and red, to indicate whether there is a problem or not. There is a light for the host state, for the service state, and for the 'processing' state. There is also a bar across the top of the page which some key statistics, such as how many services are stopped, how many have stopped processing, how many have drive problems, and full filesystems.

The problem with the coloured lights is that they cannot always be relied upon to be accurate. For instance, most deployments use Virtual Log Collectors to handle the heavy-lifting of log collection, with the Local Collector only serving as a conduit to the Log Decoder. Because of this, the processing light of a Local Log Collector will almost always show as red, unless a log collection process is started.

logdecoder Status:

Service	Processing	Rate	Name
Ready	Collection stopped	--	Local Log Collector

To see why a light is amber or red, simply hover over it. An amber light usually means that the service is in a good state, but is not seeing data.

logconcentrator Status: CPU: 3.13% Memory: 1.22 GB/7.81 GB

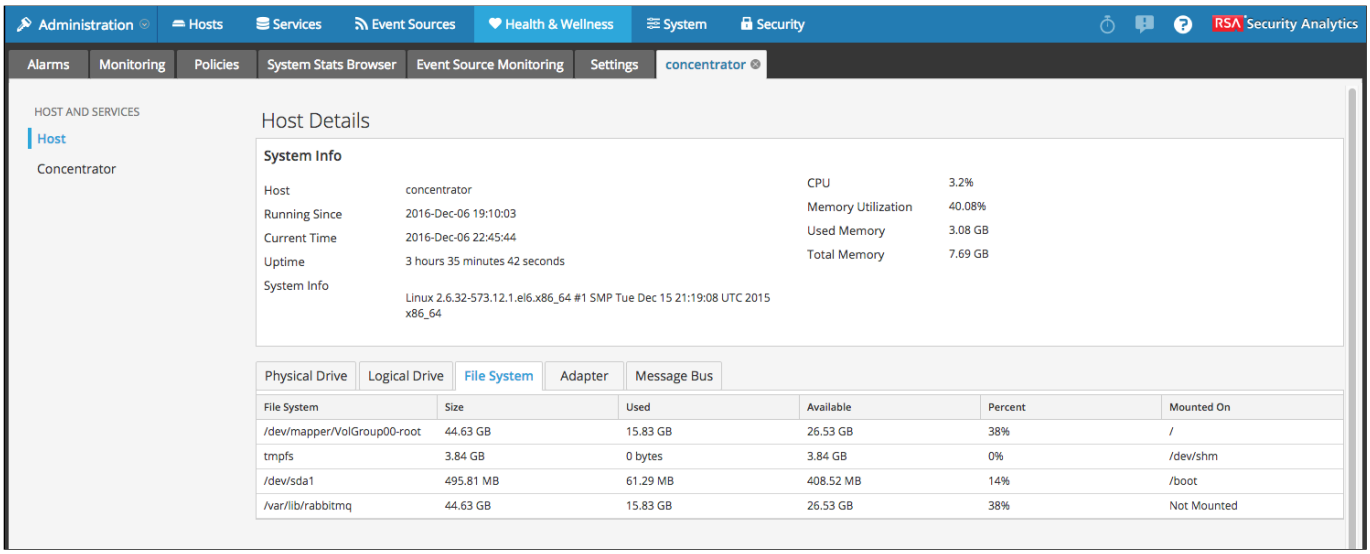
Service	Processing	Rate	Name	Service Type	CPU	Memory Usage	Uptime
Ready	Aggregation Started Rate: 0	0	Log Concentrator	Concentrator	3.7%	111.88 MB	1 hour 53 minutes 53 seconds

Also, a Broker only aggregates occasionally, so its processing light will usually show as amber.

Host Details Tab

Within the Monitoring view, one can also drill into the hosts and individual services. By clicking on a host or service, you will be taken to a new view, in a new tab which will appear at the top of the page. The left-hand pane contains links to either the host, which is the operating system and hardware, and also any SA services which are on the host. Click on these links to see stats related to them.

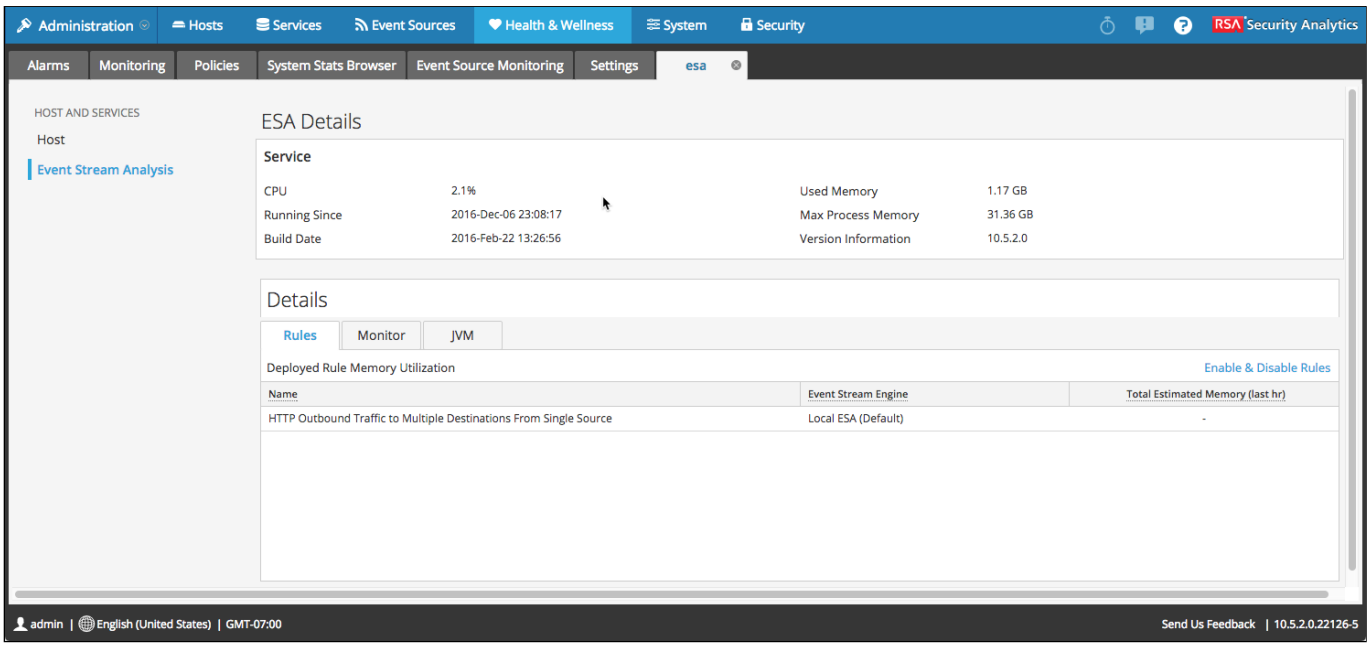
Within the Host tab, you can see tabs related to RAID physical and logical drives, filesystems, and the message bus. The Physical Drive, Logical Drive, and Adapter tabs will not populate unless the system is equipped with a hardware RAID controller and RAID drives, so in a virtual lab such as ours, nothing will be populated in those tabs.

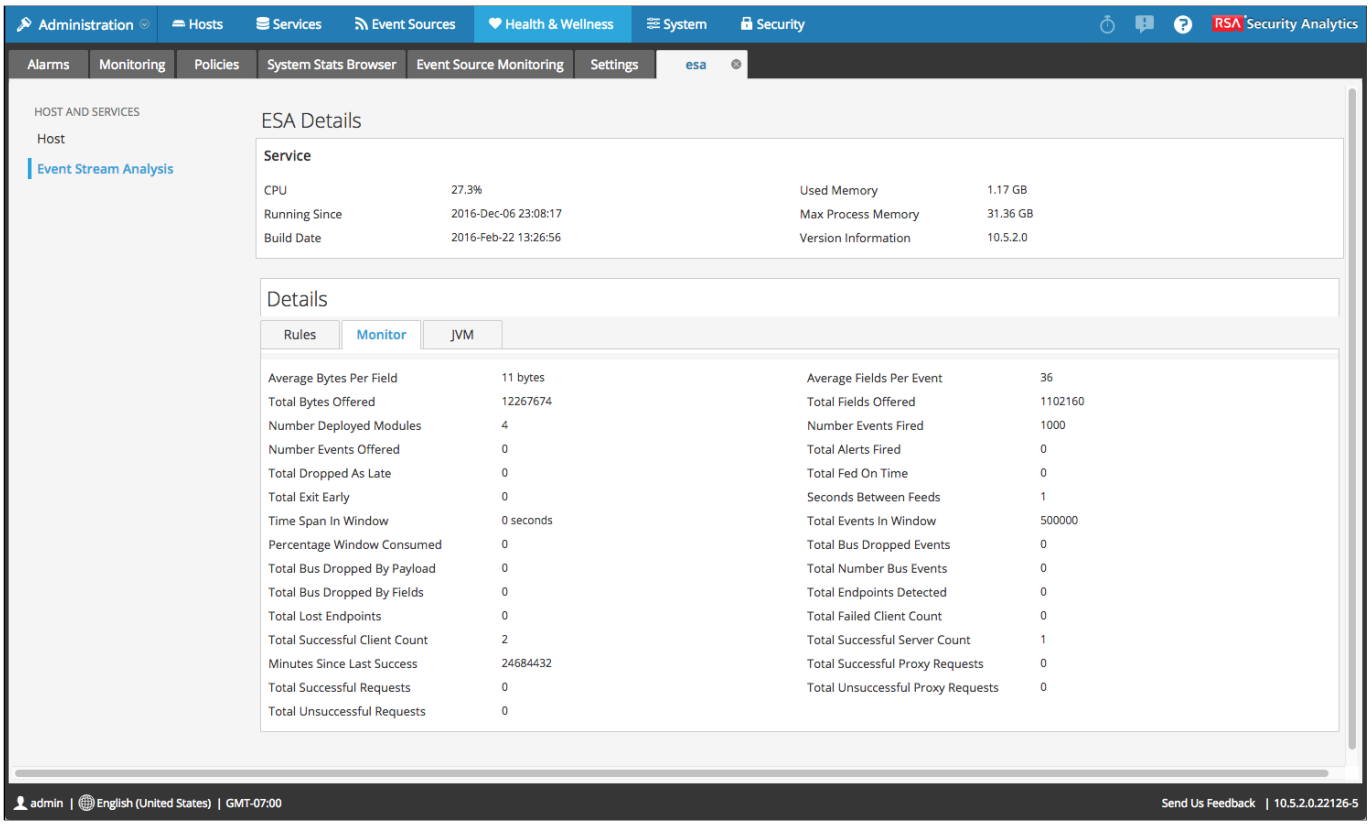


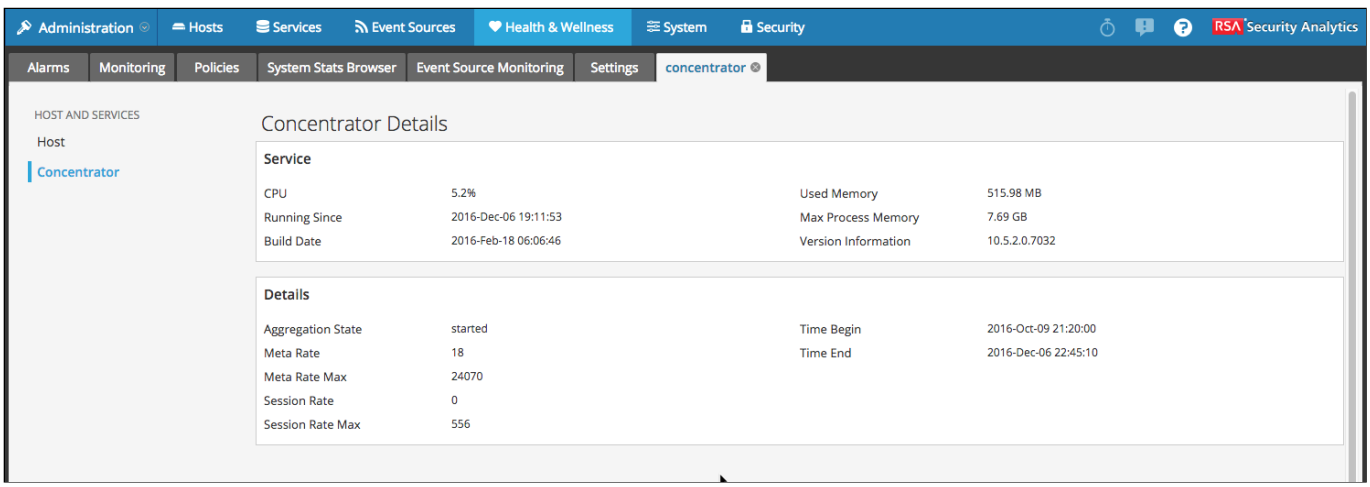
Service Details Tab

The service details tab shows stats related to individual services. These may not be very enlightening, with the possible exception of the ESA service, which may show some useful memory statistics.

Top Tip: H&W will not display memory stats for individual ESA rules unless memory stats collection is turned on, which it isn't by default. This isn't something you want to leave turned on all the time, as it can adversely affect performance.



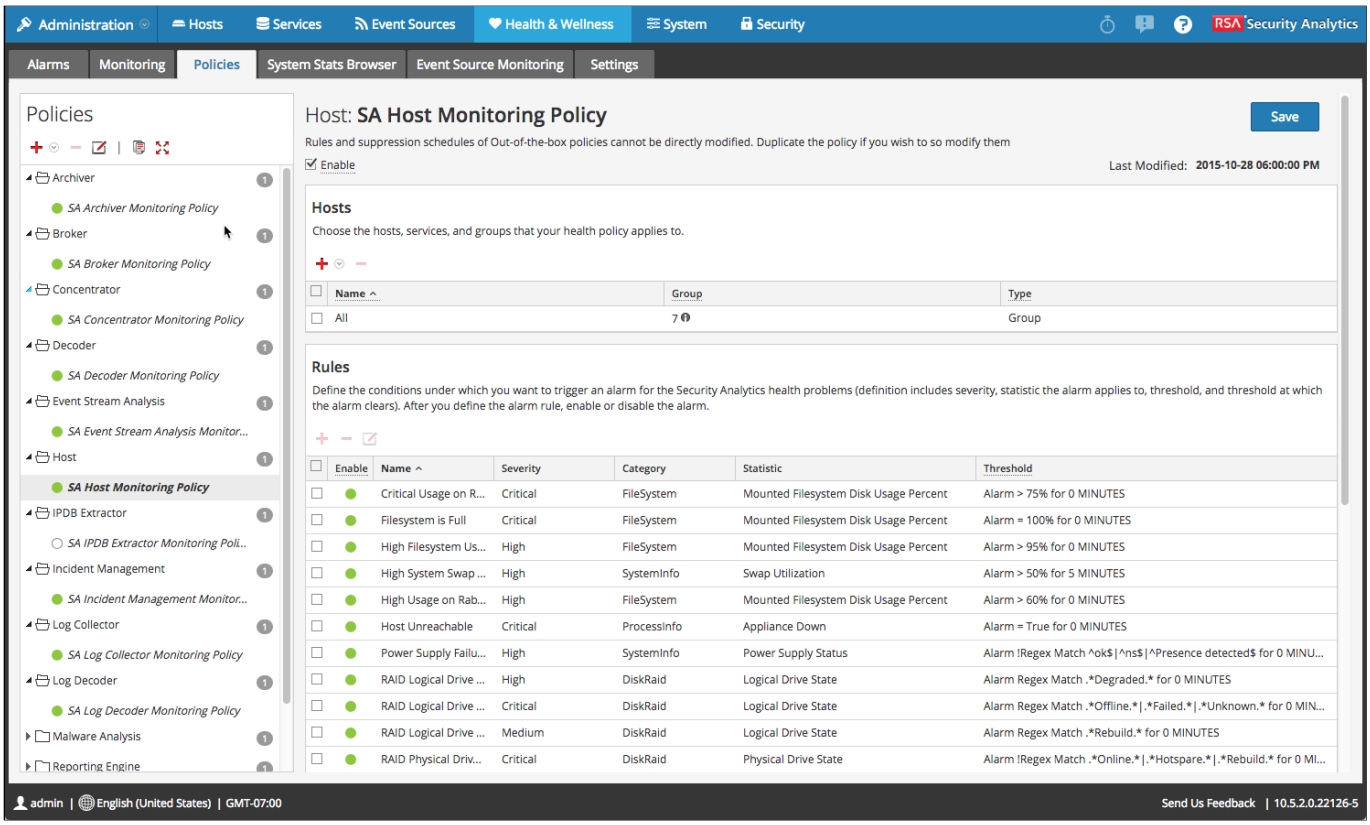




Policies Tab

The Policies tab of Health & Wellness gives Administrators control over when health alarms are created and when they clear, through the use of Health Policies. A Health Policy can be for either a Host or for a Service. A quick look here shows that SA is boxed with many out-of-the-box Health Policies. These policies are a good starting point for any customer.

A policy can apply to only a single service type or to hosts.



Out-of-the-Box Health Policies

RSA has taken a stab at guessing what the correct settings are for a broad range of customers, but you may find that sometimes the default policy settings don't work for your environment. It's okay, that can be fixed.

Top Tip: You know that a Health policy is an OOTB policy if its name is in *Italics*.

Another Top Tip: The rules and suppressions of OOTB policies cannot be edited directly. One must duplicate the policy and modify the duplicate. The reason for this is to allow RSA to update OOTB policies in new releases without overwriting your customisations. Don't forget to disable the OOTB policy, as you don't want duplicate alarms!

Inside a Health Policy

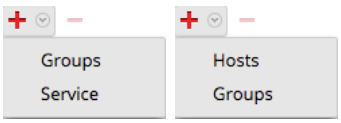
A health policy is divided into four sections:

- **Hosts** or **Services** - This shows which hosts (for a host policy) or services (for a service policy) that the policy will apply to.
- **Rules** - These are the meat of the policy, which define conditions under which alarms are created and cleared, and an alarm's severity.
- **Policy Suppression** - This allows the Administrator to tell H&W NOT to alarm during certain periods, for instance during known maintenance windows, or during periods of low traffic, such as on weekends.
- **Notifications** - Here is where one can configure recipients to receive health alarms. These can be of types email, syslog, or SNMP trap.

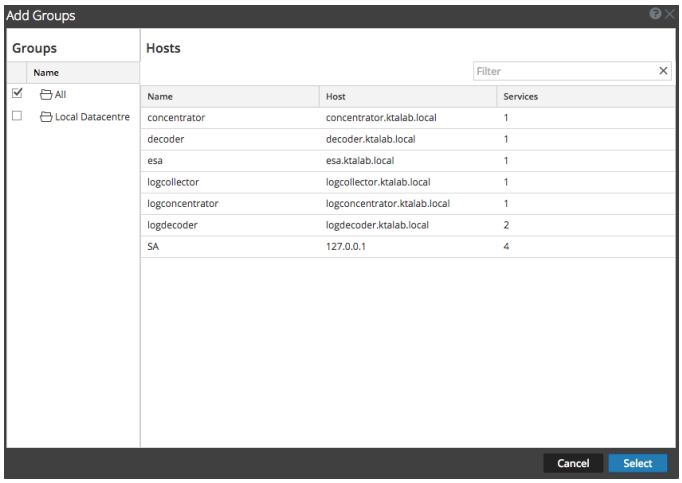
Applying a Policy to Hosts or Services

For a policy to have any effect, you must first apply it to hosts or services. You can choose individual hosts/services, or a group of hosts/services, as defined in **Administration->Hosts** and **Administration->Services**. Choose the All group to have it apply to all hosts/services of that type.

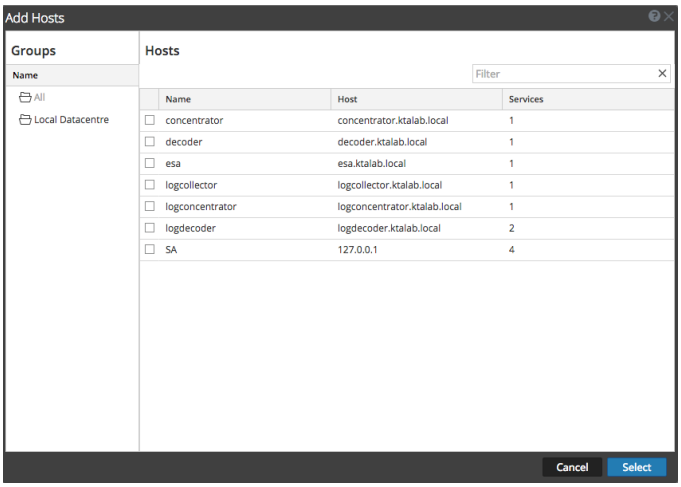
1. Simply click the + drop-down, then choose either Groups or Service/Host.



2. You will now see a dialog which will allow you to choose a group, or individual hosts or services, case depending. Tick the box for the groups or individual objects you would like the policy to apply to, and click **Select**.



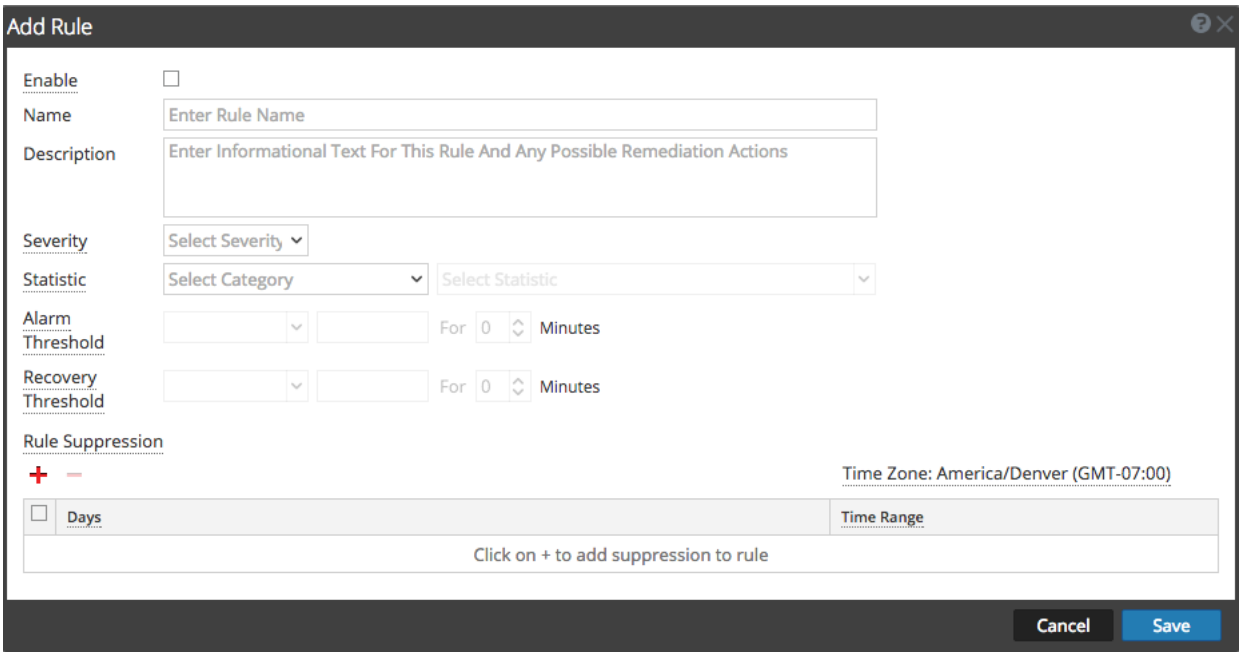
Adding a Group



Adding Individual Hosts

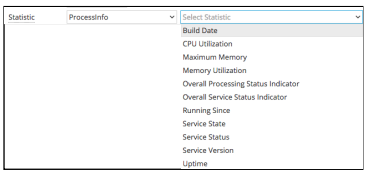
Creating or Editing a Rule

Once you've chosen your hosts or services, you are ready to create or modify a rule. Click the + button to create one.

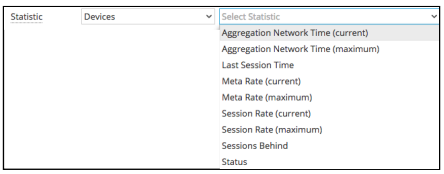


These are the things one can define:

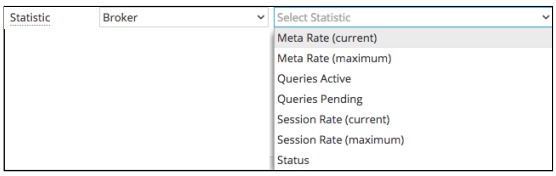
- **Enable** - Self-explanatory
- **Name** - The rule name, which will appear in any alerts generated by the rule.
- **Severity** - Either Critical, High, Medium, or Low
- **Statistic** - Select a statistic you wish to monitor here. Statistics are divided in to categories, such as **Broker**, (applies to the Broker service itself) **Devices** (stats for upstream services which the Broker consumes from), or **ProcessInfo** (things like max memory, cpu %, etc). First select a statistic category, and then an individual statistic.



ProcessInfo Category Stats



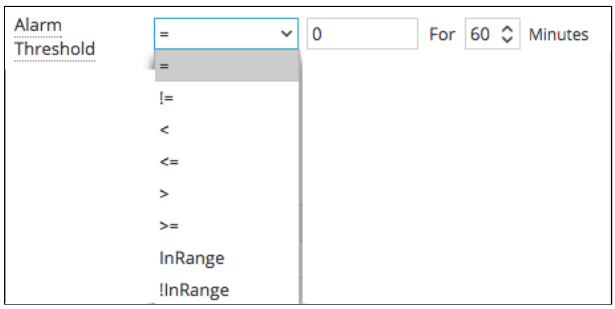
Devices Category Stats



Broker Category Stats

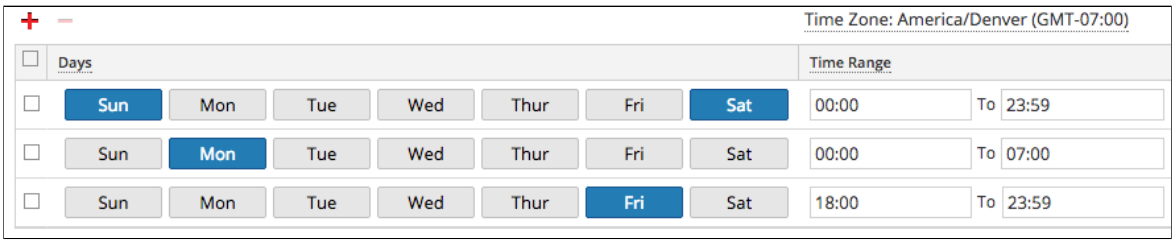
Top Tip: The System Stats Browser can really help you here to find the correct statistic (covered below).

- **Alarm Threshold** - Specifies a conditional, a value that must be matched, and the length of time that the condition must be true in order for the rule to trigger an alarm.



- **Recovery Threshold** - Specifies a conditional, a value that must be matched, and the length of time that the condition must be true in order for a triggered alarm to clear.

- **Rule Suppression** - One or more time ranges during which the rule will not apply. You simply choose the days of the week, and the time of day, in 24 hour format (HH:MM). This uses the timezone setting which is configured for the Policy Suppression section of the policy.

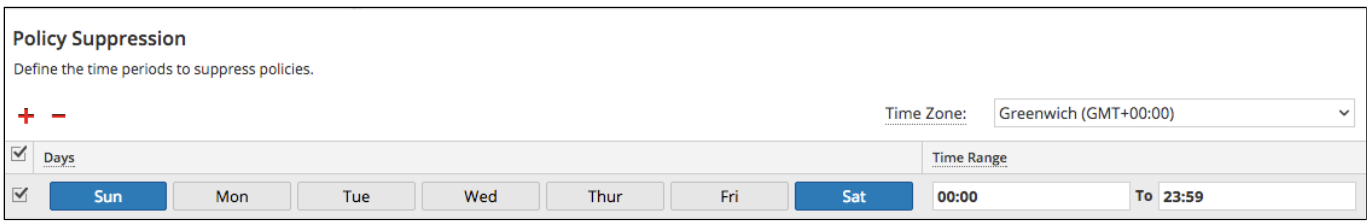


A Typical After-Hours Suppression

Top Tip: Every alarm must have both an alarm threshold and a recovery threshold. One cannot have an alarm trigger without the possibility of clearing it.

Adding a Policy Suppression

One can suppress all alarms for an entire policy on a schedule, in a similar way to suppressing an individual rule. This is done from the **Policy Suppression** section (surprise!). The only difference between this and Rule Suppression is that Policy Suppression includes a timezone setting. This should be set because all date and time measurements must be relative to a specific place. For instance, if we were to suppress a policy's alarms on Monday between midnight and 8am, is that midnight in Sydney or midnight in London? Thus, the timezone selector.



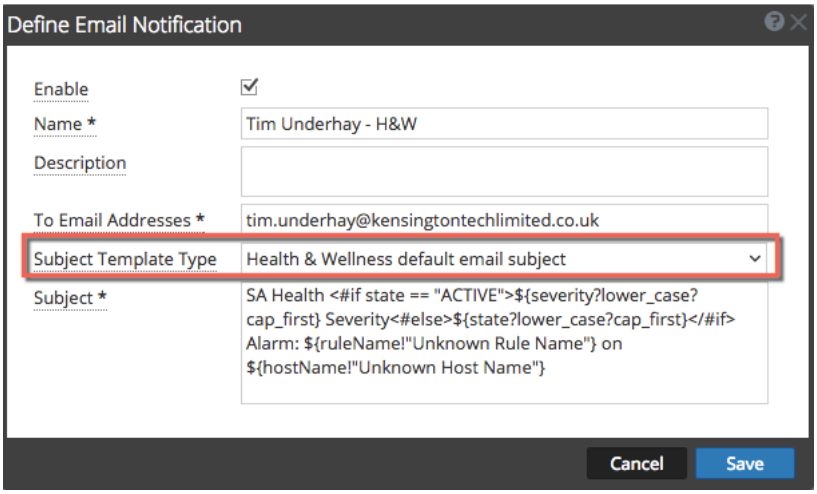
Alarm Notifications

Policies can be configured to send a notification to an email address, SNMP server, or syslog server, which for quite obvious reasons can be quite beneficial to an organisation. To do this:

1. Simply use the + drop-down to add a row to the **Notifications** section of a Health Policy, then choosing the notification type (Email, SNMP, or Syslog).

2. Choose a recipient (this must be defined in the **Global Notifications Output** tab See the lesson on Notifications for guidance on configuring Global Notifications).

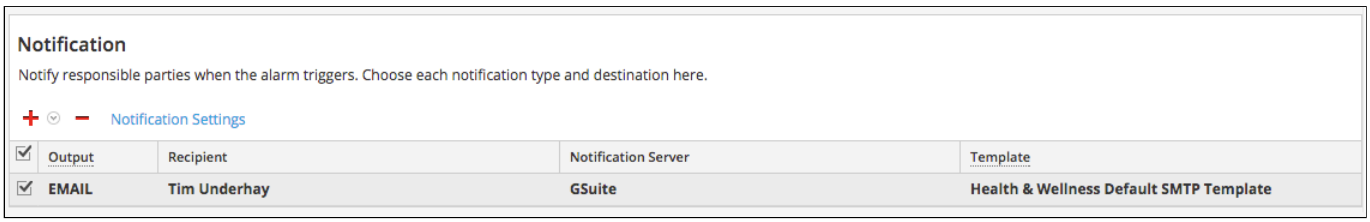
Top Tip: In order for an email alert subject line to contain relevant information (like the hostname and alert name), you will need to choose the **Subject Template Type** for Health & Wellness in your email output object in **Administration->System->Global Notifications->Output**:



3. Choose a notification server (also defined in **Global Notifications**, under **Servers**).

4. Choose an appropriate template for your notification type (also defined in **Global Notifications**, under **Templates**). There are three OOTB templates: one for SMTP, one for Syslog, and one for SNMP Traps.

Top Tip: SA's Health & Wellness does not support SNMP polling, only traps.



noreply@securityanalytics.com
SA Health Critical Severity Alarm: Log Decoder Capture Not Started on logdecoder
To: Underhay Tim

Inbox - Kensington Tech 13:58

RSA Security Analytics
Health Alarm Notification

Log Decoder Capture Not Started on logdecoder

State
Active

Severity
Critical

Host
logdecoder

Service
Log Decoder

AlarmId
342-2058-0011

Policy
SA Log Decoder Monitoring Policy

Rule
Log Decoder Capture Not Started

Statistic
Capture/Capture Status

Value
stopped

Time
December 7, 2016 8:58:24 PM UTC

Additional Information
Capture is not started on this Log Decoder service, therefore no log packet data is being captured.

Possible Remediation Action:
From Security Analytics, attempt to start capture on the service at Administration -> Services -> <this service> -> Actions -> System.

If capture will not start, please consult the service logfile on this Log Decoder host at /var/log/messages for indicators of the problem.

If you cannot resolve the issue, please open a case with Customer Support.

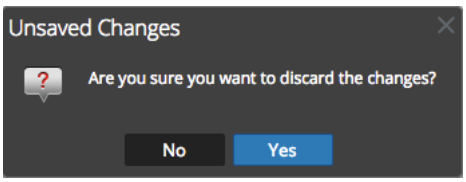
Enabling or Disabling a Policy

To enable or disable a policy, tick or clear the Enable box in the upper-left hand side of the policy.

Top Tip: When disabling a policy, all alarms associated with it will automatically be cleared.

Saving a Policy

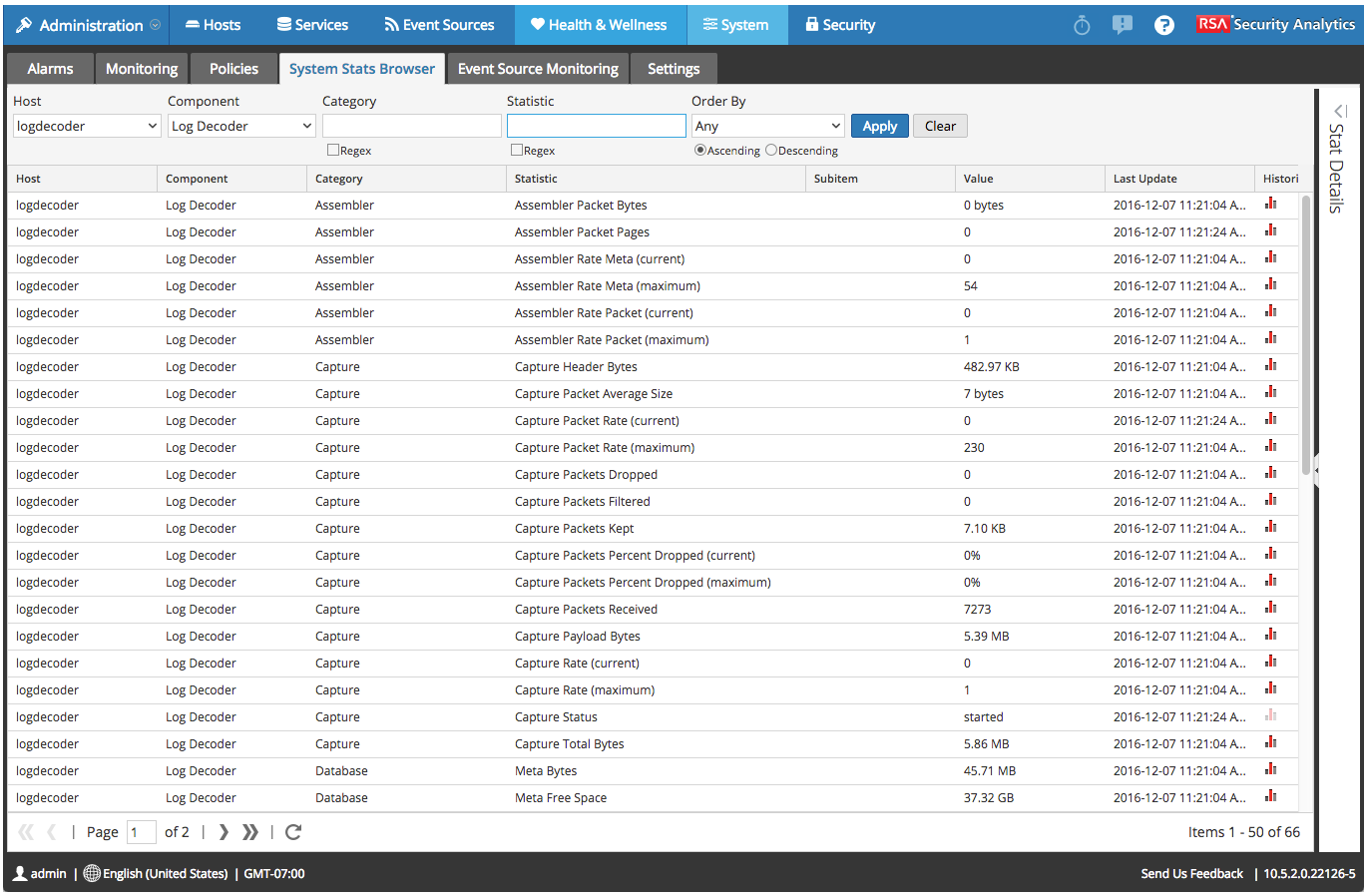
Be sure to hit the Save button in the upper-right hand corner of the policy! If you forget to do this and try navigate away, you'll get this pop-up:



The System Stats Browser

As mentioned earlier, the System Stats Browser is a tool for power users to dig into raw H&W statistics. From here you can filter on

- **Host** - Self-explanatory
- **Component** - e.g. Host, Log Collector, MessageBus, System Monitor, Concentrator, Decoder, Log Decoder, Broker, Archiver
- **Category** - The category of statistic available under the selected component. E.g. if you've selected Log Decoder as the component, you can choose Assembler, Capture, Database, Pool, and ProcessInfo. You know which stat categories are available by selecting a component, then leave category blank, and then hit **Apply**. The results are paginated, so you can click through for more pages of results.
- **Statistic** - The individual statistic available under the selected category. If you don't know, just leave it blank.



Host	Component	Category	Statistic	Subitem	Value	Last Update	Histori
logdecoder	Log Decoder	Assembler	Assembler Packet Bytes		0 bytes	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Assembler	Assembler Packet Pages		0	2016-12-07 11:21:24 A...	
logdecoder	Log Decoder	Assembler	Assembler Rate Meta (current)		0	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Assembler	Assembler Rate Meta (maximum)		54	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Assembler	Assembler Rate Packet (current)		0	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Assembler	Assembler Rate Packet (maximum)		1	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Header Bytes		482.97 KB	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Packet Average Size		7 bytes	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Packet Rate (current)		0	2016-12-07 11:21:24 A...	
logdecoder	Log Decoder	Capture	Capture Packet Rate (maximum)		230	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Packets Dropped		0	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Packets Filtered		0	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Packets Kept		7.10 KB	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Packets Percent Dropped (current)		0%	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Packets Percent Dropped (maximum)		0%	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Packets Received		7273	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Payload Bytes		5.39 MB	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Rate (current)		0	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Rate (maximum)		1	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Capture	Capture Status		started	2016-12-07 11:21:24 A...	
logdecoder	Log Decoder	Capture	Capture Total Bytes		5.86 MB	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Database	Meta Bytes		45.71 MB	2016-12-07 11:21:04 A...	
logdecoder	Log Decoder	Database	Meta Free Space		37.32 GB	2016-12-07 11:21:04 A...	

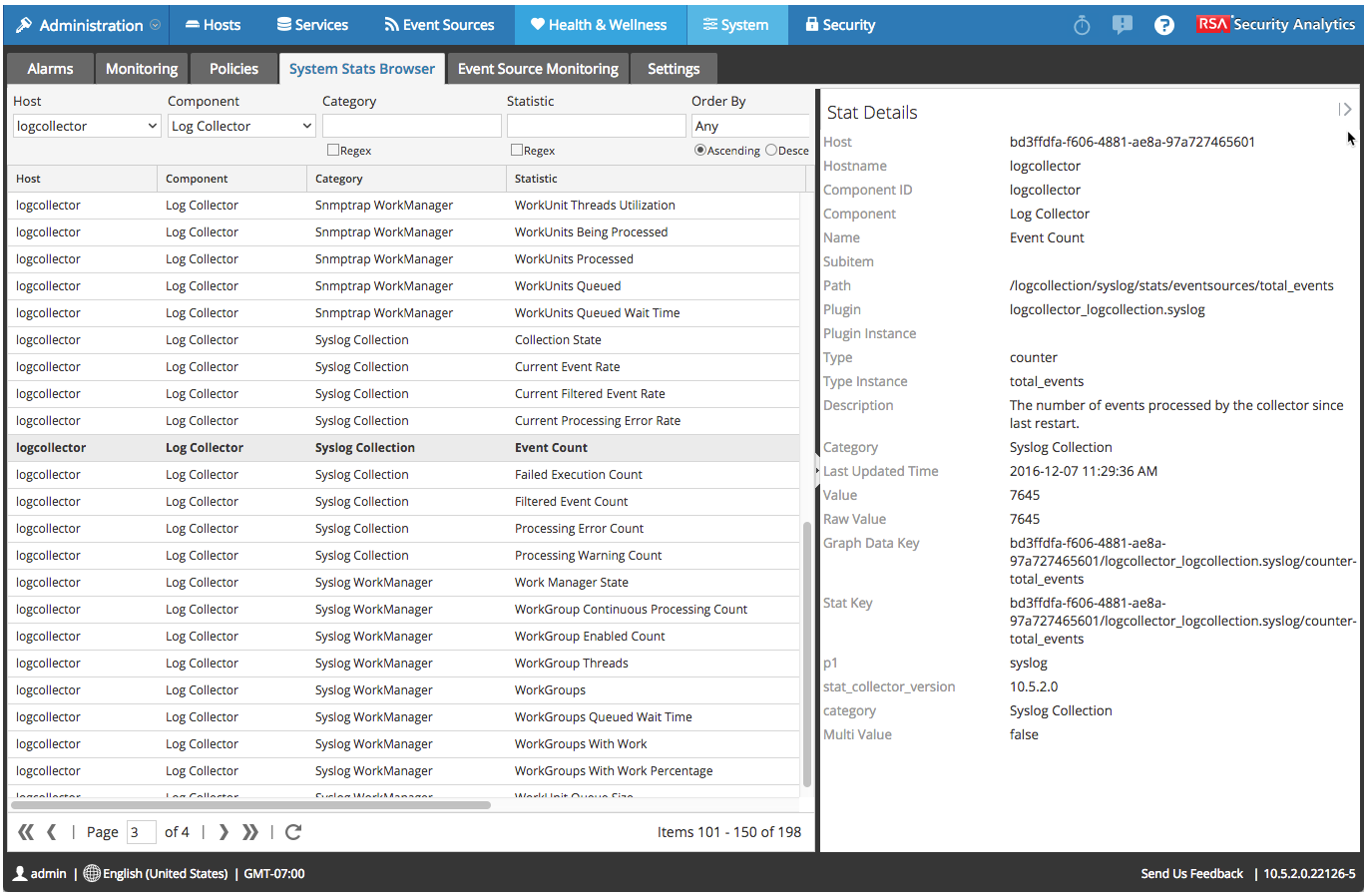
Subitems

You may see more than one of a certain type of statistic for a given service. This happens when there is more than one value for a given statistic. Each individual value will get its own row, and is identified by the SubItem column. E.g. for filesystem statistics, there is more than one filesystem present on a given host, so every filesystem mount point is its own SubItem.

logdecoder	Host	FileSystem	Mounted Filesystem Disk Usage Percent	/	13%	2016-12-07 11:24:06 A...
logdecoder	Host	FileSystem	Mounted Filesystem Disk Usage Percent	/dev/shm	0%	2016-12-07 11:24:06 A...
logdecoder	Host	FileSystem	Mounted Filesystem Disk Usage Percent	/boot	13%	2016-12-07 11:24:06 A...
logdecoder	Host	FileSystem	Mounted Filesystem Disk Usage Percent	/var/lib/rabbitmq	13%	2016-12-07 11:24:06 A...

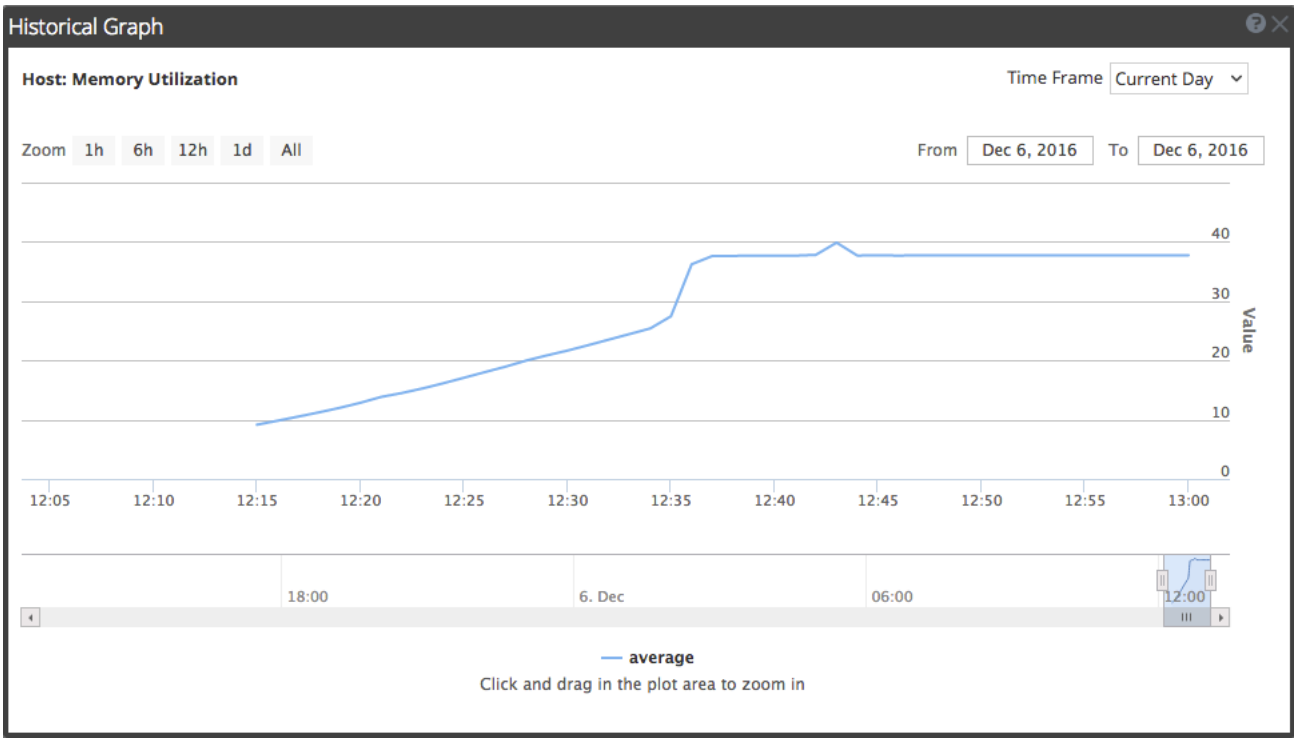
More Stat Details

By expanding the **Stat Details** drawer on the right-hand side of the System Stats Browser, you can pull up the raw details of the particular stat.



RRD Graphs

Notice that for some statistics, there is a button in the Historical column. This is available only for numerical statistic types. Clicking this will display an RRD Graph, which can then be further filtered for a specific timeframe through use of the Zoom buttons, or by click-dragging on the chart itself, or by dragging the handles on the timeline at the bottom of the dialog.



The Event Source Monitoring and Settings Tabs

You may have noticed two tabs under H&W labelled **Event Source Monitoring** and **Settings**. These pages are deprecated and will be removed in a future release of SA. They were for legacy ESM and H&W capabilities that have since been replaced by newer feature sets. If you have these set up in your environment, you should migrate away from them ASAP.

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Alarms | Monitoring | Policies | System Stats Browser | Event Source Monitoring | Settings

This page is deprecated and will be removed in future release.
To monitor Event Sources, see Event Source Management. To monitor Archiver and Warehouse Connector, see Health Policy.

Event Source | Archiver | Warehouse Connector

Event Source Monitoring

These setting dictate when to generate notifications for event sources that the system is no longer receiving logs from and where to send those notifications.

Configure email or distribution list.
Configure Syslog and SNMP Trap servers.

	Regex	Source Type	Source Host	Time Threshold	
				Hours	Minutes

☐ Email ☐ Syslog Notification ☐ SNMP Trap Notification

Decommission

admin | English (United States) | GMT-07:00 Send Us Feedback | 10.5.2.0.22126-5

34. Event Source Monitoring

With those pesky SA for Logs deployments, it can be critical for an organisation to know when a device which sends logs to their monitoring system, stops logging. When a log event source either stops logging, logs too little, or logs too much - security personnel need to know about it. Enter Event Source Monitoring, or ESM for short.

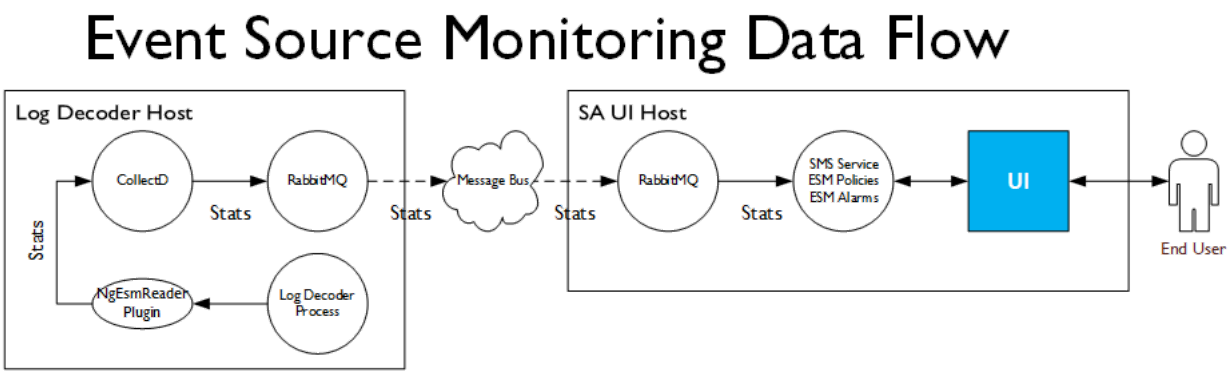
What is an Event Source?

Simply put, an Event Source is any device that sends a log to SA. It is identified by its IP address and its Event Source Type, which typically corresponds to the name of the log parser which parsed the log message.

Note: Please forgive the examples given in this page, as nearly all of the logs in our lab environment are being injected via a script from a single source IP, and all event sources shown in the example have the same IP address. In the real world, this should not happen.

ESM Architecture

ESM and Health & Wellness share a lot of infrastructure. One might say that ESM is actually a part of Health & Wellness. Like Health & Wellness, ESM also relies on statistics collected by CollectD from the Log Decoder process, through a plugin called **NgEsmReader**. ESM takes a policy-based approach to alerting personnel of when there are anomalies in logging volume from a particular event source or group of sources.



Event Source Groups

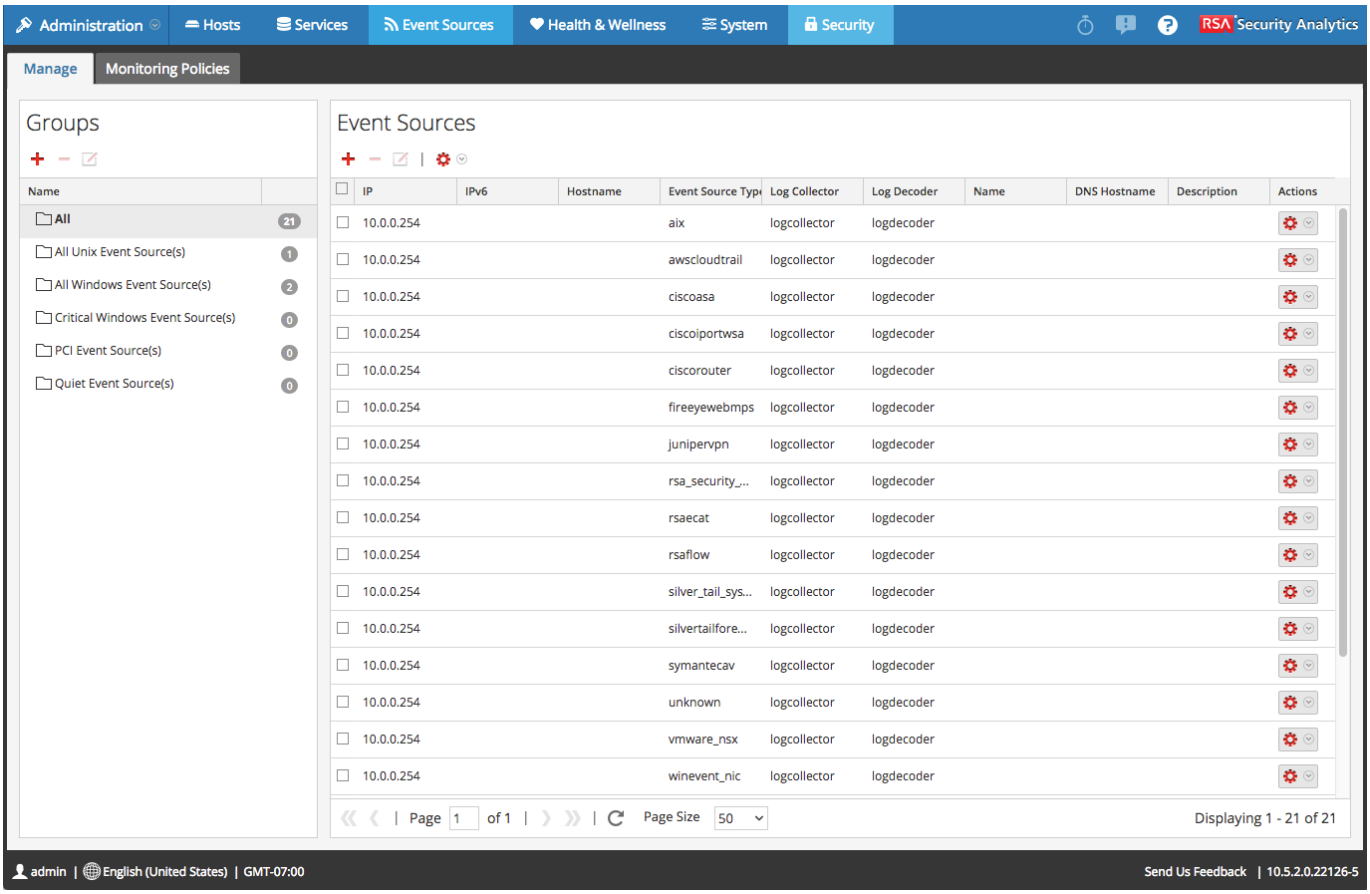
The first thing one most take note of with ESM is **event source groups**. An **event source group** is a collection of log event sources, defined by one or more characteristics, such as event source type, OS type, asset criticality (PCI environment, HR system, etc), or when an event source is a known low-logger. An event source can be added to an Event Group in two ways:

1. **Dynamically**: by setting one or more conditions in the group's definition, event sources are automatically added to the ESM group as soon as they send a log.

2. **Manually** - Event sources can be added one at a time by administrators.

3. **Bulk Import** - Multiple hosts can be imported at once via Bulk import from a CSV file. This CSV can be created manually or exported from an external asset management system, or CMDB.

Let's navigate to the **Administration -> Event Sources -> Manage** tab of the UI. We can see several out of the box event groups - All, Unix, Windows, Critical Windows, PCI, and Quiet Event Sources.



Creating an Event Source Group

Notice the different event source types above. Let's create an event source group for Cisco ASA's.

1. Under Groups, click the + button. The Create an Event Group dialog is displayed.

2. Enter "**Cisco ASA Event Sources**" for the Group Name.

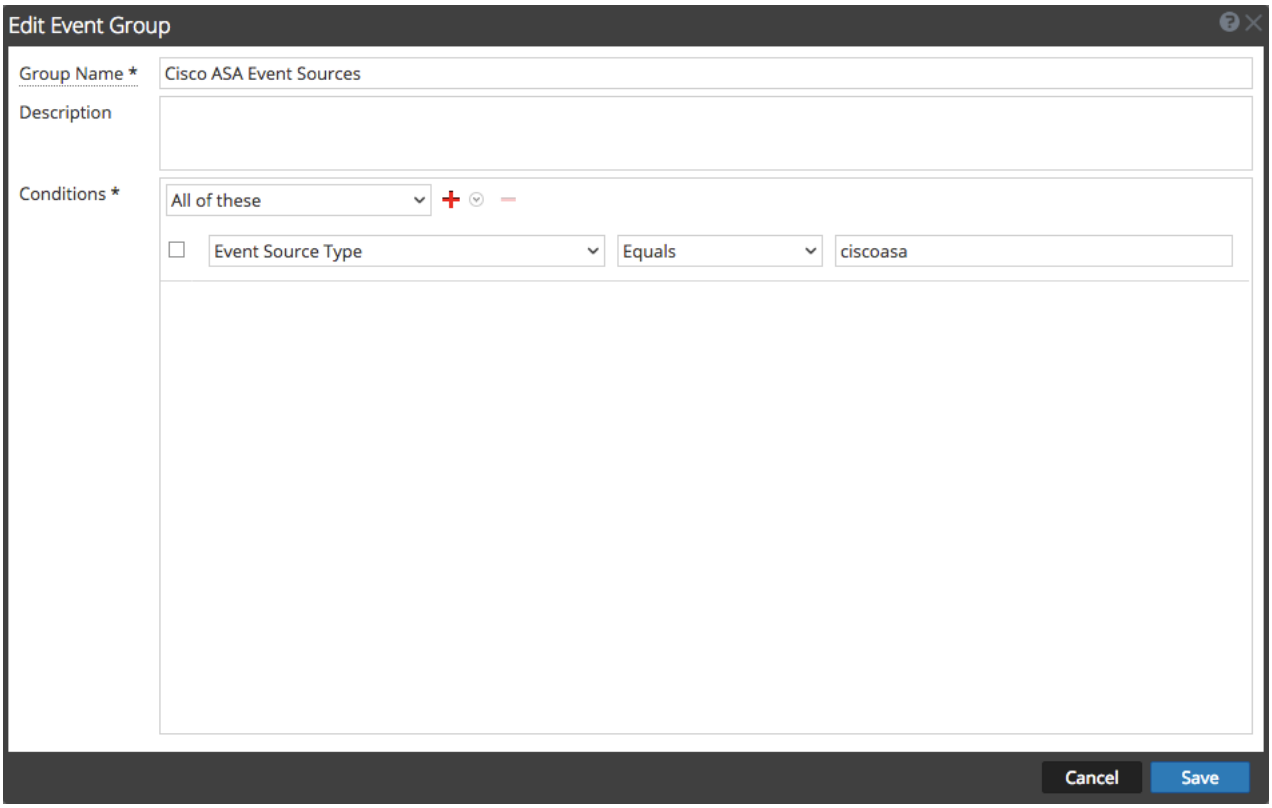
2. Under Conditions, leave **All of these** selected. Click the + drop-down, and then **Add Condition**.

3. In the Select Attribute box, select **Event Source Type.**

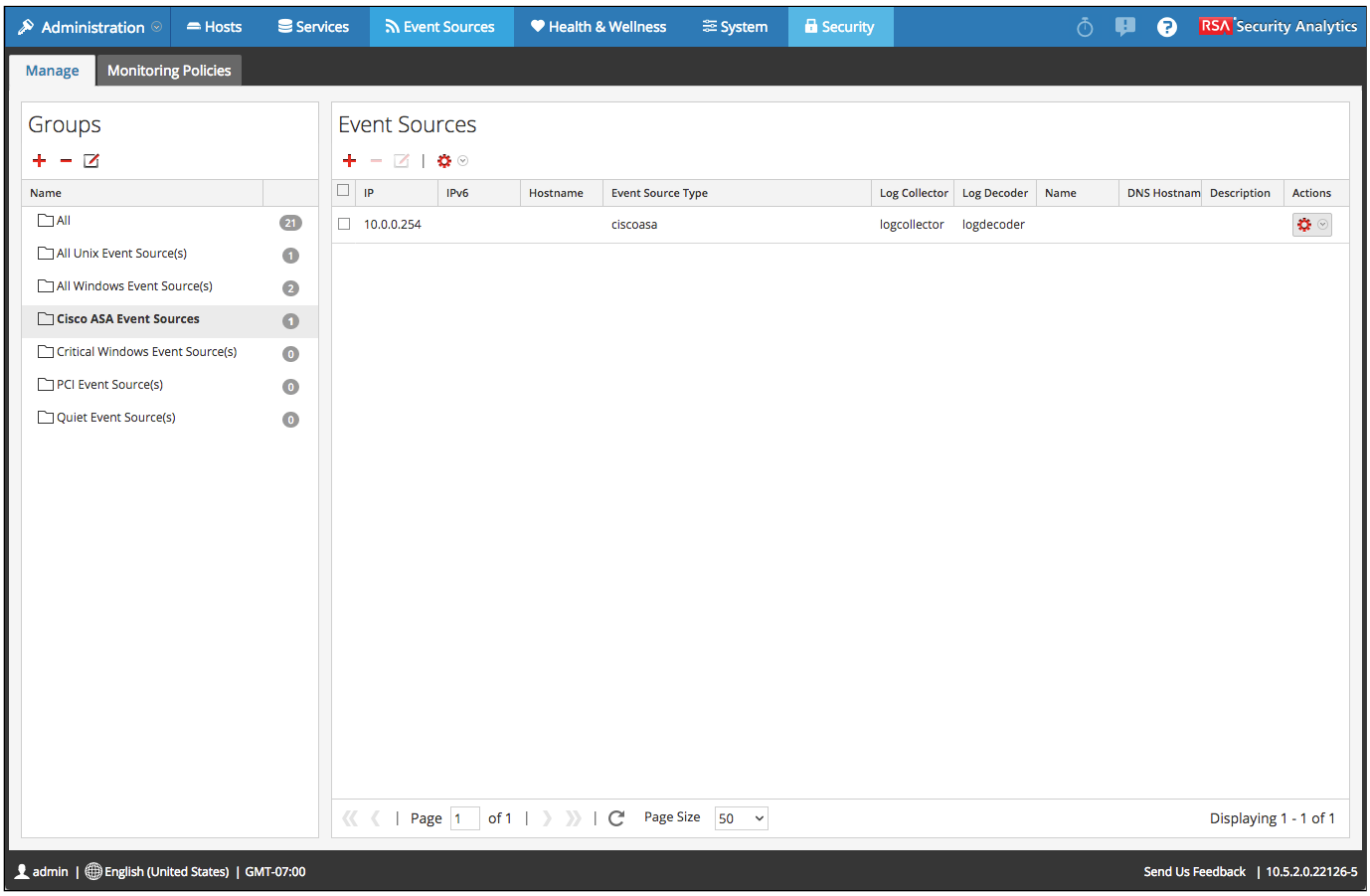
4. For **Select Operator**, choose **Equals**. Available operators are Equals, Not Equals, In (meaning a list of values), Not in, Like (contains), Not like, Greater than, and Less than.

5. For **Enter Value**, type "**ciscoasa".**

6. Click **Save.**

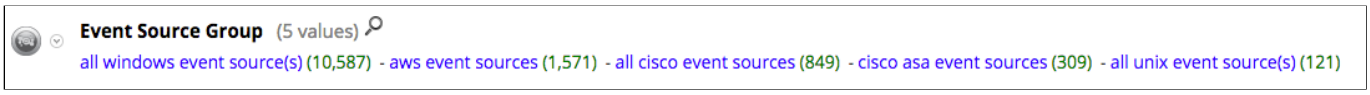


You will now see an event source group called **Cisco ASA Event Sources,** containing a single event source.



Re-Injecting Event Source Groups

Once an event source is part of a group, something very cool happens. ESM automatically creates a feed containing the IP's of all event sources and its associated groups, and then injects this feed back into all Log Decoders. You can then use the Event Source Group meta key, or **device.group** key, to query and drill in on for members of specific event source groups.

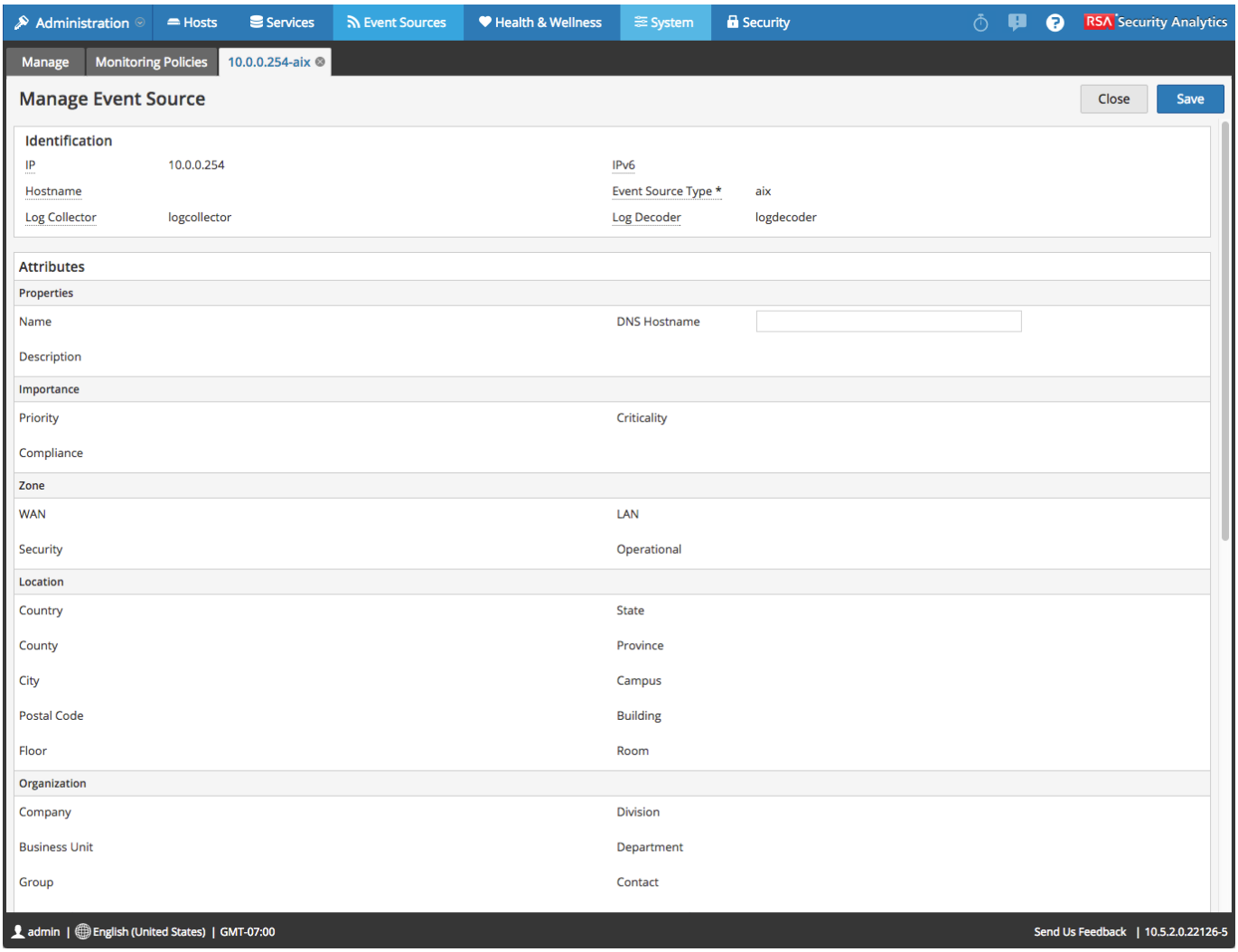


Lab Exercise:

1. Create an event source group for all event sources whose logs the Log Decoder has not been able to identify. This type of group can be very useful for finding log parsing problems.
2. Create an event source group for Cisco devices of any type.

Event Source Definition

By clicking on an individual event source inside of ESM, you access the event source's properties page. Some of the information on this page will be auto-populated, while others must be user-defined. Some, like IP, hostname, and event source type, can't be modified. Don't worry, it isn't necessary to input all this information by hand for every event source. Click **Save** when finished.

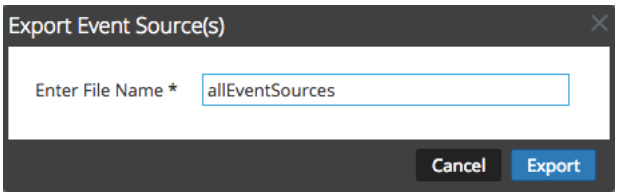


Bulk Importing Event Sources

Rather than adding all this info by hand, which let's face it, would be awful, we can import it all through a CSV file. You're on the hook to create this file, but if you have a CMDB, this may not be so hard. To see the CSV file format, export your event sources:

Exporting Your Existing Event Sources

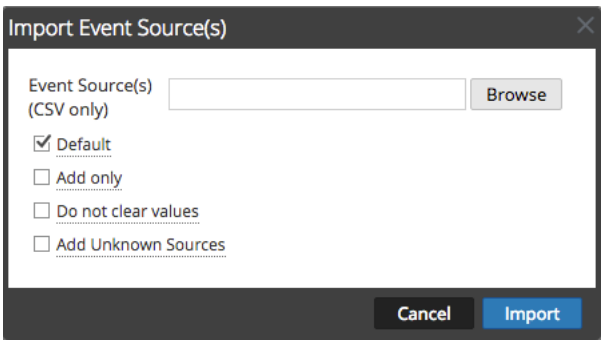
1. From within ESM, select a group
2. From the drop-down, select **Export Group (.csv)**
3. Finally, enter a filename and click **Export**. A CSV file will automatically download to your browser. Download me



To Import Your CSV:

1. From within ESM, select a group.

2. From the drop-down, select **Import.**



3. Choose your CSV file.

4. Either leave Default selected, or tick the other boxes as you see fit. Default means that it will overwrite an existing value in SA, it will clear the value in SA if the value in the CSV is blank, and new columns in the CSV are ignored and not cleared.

5. Click **Import.**

ESM Monitoring Policies

A Monitoring Policy defines high and low thresholds which will trigger an ESM alarm for its associated ESM group. Every ESM group defined on the **Manage** tab has an associated ESM policy, which can optionally be enabled or disabled.

1. Click on an ESM group from the left-hand column. Every ESM Group has a policy associated with it.

2. Optionally define a low threshold - If any event source which belongs to the group sends fewer logs than this number of events within the defined timeframe, an ESM alarm will be triggered.

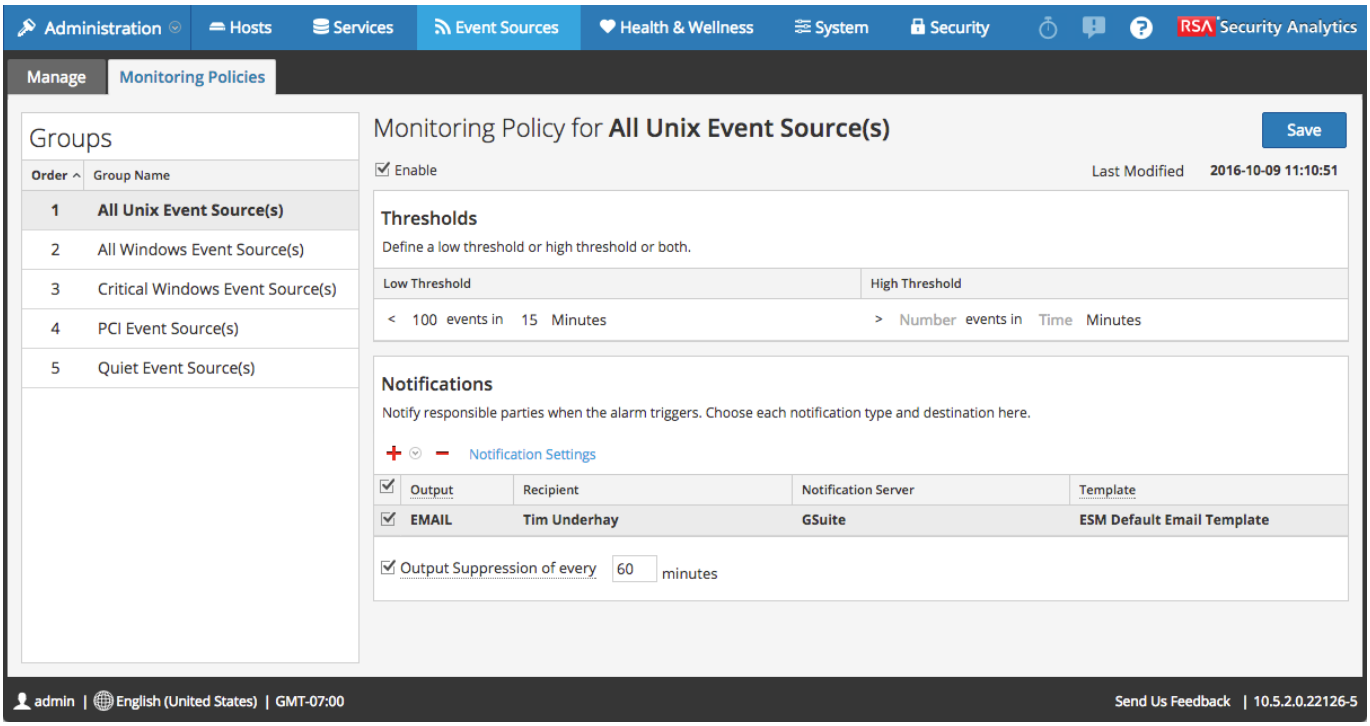
3. Optionally define a high threshold - If any event source which belongs to the group sends more logs than this number of events within the defined timeframe, an ESM alarm will be triggered.

4. Optionally configure notification recipients. As in H&W, Email, SNMP Trap, and Syslog notifications are available, and OOTB templates are available for alarms.

5. Choose output suppression interval. This means that notifications will not be re-sent during the configured timeframe. At the end of this window, another alarm notification will be sent.

6. Tick the **Enable** box

7. Click **Save**



Grouping Order

The order of the groups which are shown on the policies page matters when policies get applied to the event sources. If an event source belongs to more than one group, only the first policy that the event source belongs to will apply to it.

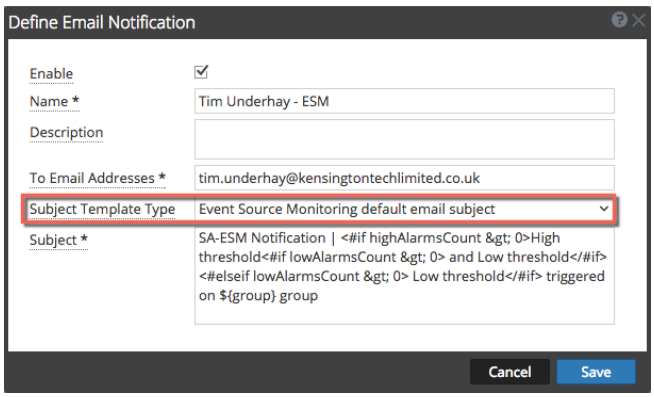
Groups can be re-ordered by dragging them up and down.

ESM Alarms

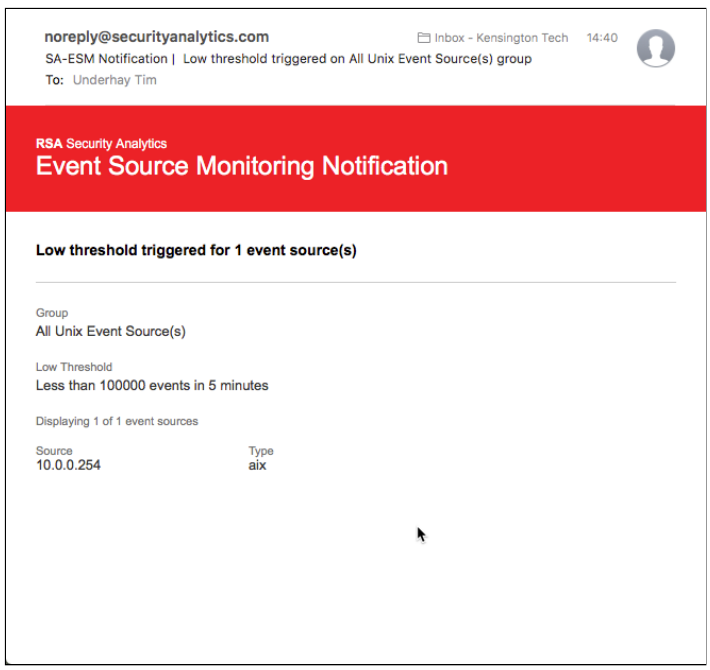
Whenever the thresholds defined in an ESM Monitoring Policy are breached, an alarm will be generated. This alarm will be sent to whichever notification recipients are defined in the policy (refer to the lesson on Global Notifications for more information).

ESM alarms will automatically clear once the logging level of the event source falls back within its policies thresholds.

Top Tip: In order for an email alert subject line to contain relevant information (like the hostname and group name), you will need to choose the **Subject Template Type** for Event Source Monitoring in your email output object in **Administration->System->Global Notifications->Output**:



Email Notification with Subject Template



ESM Notification Email Example

ESM Stats Browser

As in Health & Wellness, there is a stats browser for raw ESM statistics, available under **Administration->Health & Wellness->Event Source Monitoring**. It also can show RRD graphs of event source logging.

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Alarms | Monitoring | Policies | System Stats Browser | Event Source Monitoring | Settings

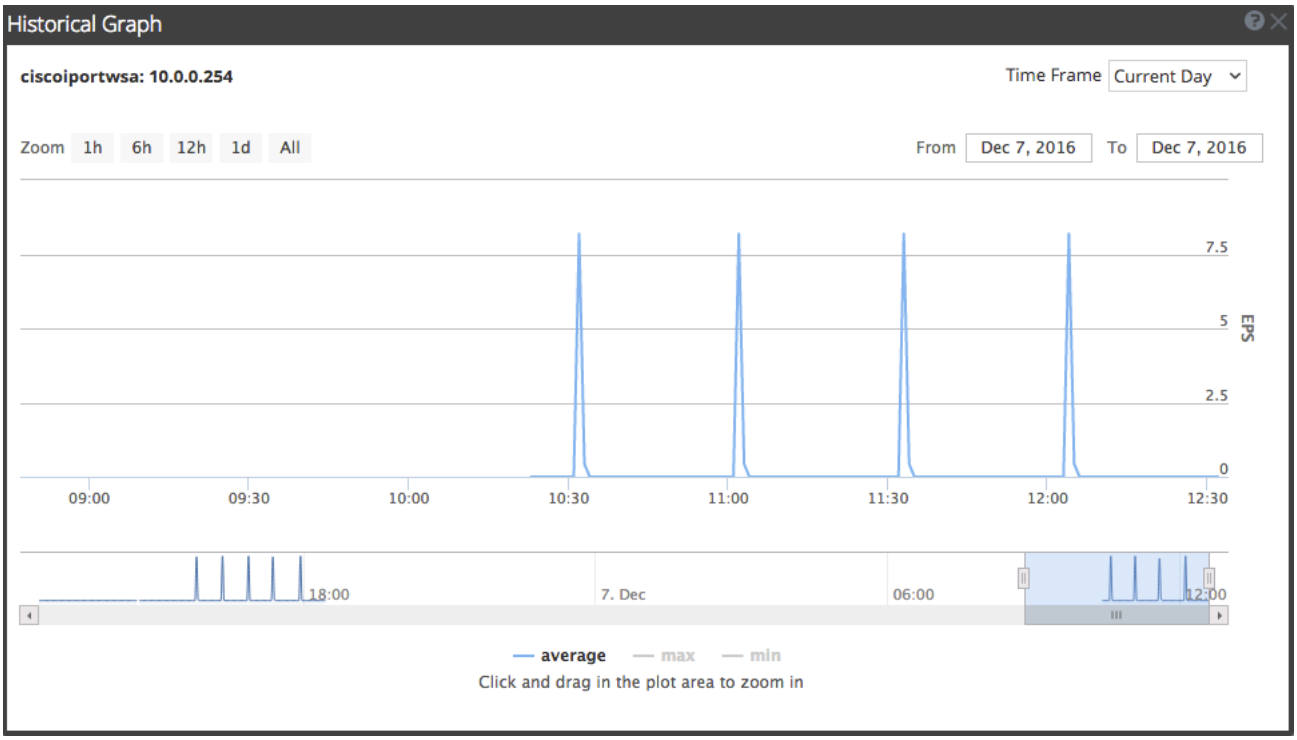
Event Source | Event Source Type: Any | Log Collector: Any | Log Decoder: Any | Time Frame: Any | Order By: Any | Apply | Clear | Export as CSV

Regex | Received / Not Received | Ascending / Descending

Event Source	Event Source Type	Log Collector	Log Decoder	Count	Idle Time	Last Collected Time	Historical Graph
10.0.0.254	aix	logcollector	logdecoder	1287	28 min	2016-12-07 12:01:42 P...	
10.0.0.254	awscloudtrail	logcollector	logdecoder	163384	28 min	2016-12-07 12:01:46 P...	
10.0.0.254	ciscoasa	logcollector	logdecoder	36153	28 min	2016-12-07 12:02:20 P...	
10.0.0.254	ciscoiportwsa	logcollector	logdecoder	59972	27 min	2016-12-07 12:02:48 P...	
10.0.0.3	ciscorouter	logcollector	logdecoder	854	2 hours, 5 min	2016-12-07 10:24:29 A...	
10.0.0.254	ciscorouter	logcollector	logdecoder	2668	28 min	2016-12-07 12:02:20 P...	
10.0.0.254	fireeyewebmps	logcollector	logdecoder	118	31 min	2016-12-07 11:58:33 A...	
10.0.0.254	junipervpn	logcollector	logdecoder	2340	28 min	2016-12-07 12:01:42 P...	
10.0.0.3	netscreenidp	logcollector	logdecoder	51	2 hours, 5 min	2016-12-07 10:24:29 A...	
10.0.0.254	rsa_security_analytics...	logcollector	logdecoder	20423	2 hours, 5 min	2016-12-07 10:24:29 A...	
10.0.0.254	rsaecat	logcollector	logdecoder	3627	28 min	2016-12-07 12:01:42 P...	
10.0.0.254	rsaflow	logcollector	logdecoder	234	28 min	2016-12-07 12:01:42 P...	
10.0.0.3	securityanalytics	logcollector	logdecoder	32178	29 min	2016-12-07 12:00:33 P...	
10.0.0.254	silver_tail_systems_for...	logcollector	logdecoder	117	28 min	2016-12-07 12:01:42 P...	
10.0.0.254	silvertailforensics	logcollector	logdecoder	117	28 min	2016-12-07 12:01:42 P...	
10.0.0.254	symantecav	logcollector	logdecoder	1755	28 min	2016-12-07 12:01:46 P...	
10.0.0.254	unknown	logcollector	logdecoder	540	2 hours, 5 min	2016-12-07 10:24:29 A...	
10.0.0.3	unknown	logcollector	logdecoder	13428	2 hours, 5 min	2016-12-07 10:24:29 A...	
10.0.0.254	vmware_nsx	logcollector	logdecoder	819	28 min	2016-12-07 12:01:42 P...	
10.0.0.254	winevent_nic	logcollector	logdecoder	69479	2 min	2016-12-07 12:27:48 P...	
10.0.0.254	winevent_snare	logcollector	logdecoder	43290	28 min	2016-12-07 12:02:20 P...	

Page 1 of 1 | Items 1 - 21 of 21

admin | English (United States) | GMT-07:00 | Send Us Feedback | 10.5.2.0.22126-5



Legacy ESM

The ESM functionality you have seen here is new to SA 10.5. There was previously a different ESM component to SA, with considerably less functionality. This legacy functionality is considered to be deprecated, and will be removed in a future release of SA. Should you wish to see it, it can be found under **Administration -> Health & Wellness -> Event Source Monitoring**.

Administration | Hosts | Services | Event Sources | Health & Wellness | System | Security | RSA Security Analytics

Alarms | Monitoring | Policies | System Stats Browser | Event Source Monitoring | Settings

This page is deprecated and will be removed in future release.
To monitor Event Sources, see Event Source Management. To monitor Archiver and Warehouse Connector, see Health Policy.

Event Source | Archiver | Warehouse Connector

Event Source Monitoring

These setting dictate when to generate notifications for event sources that the system is no longer receiving logs from and where to send those notifications.

Configure email or distribution list.
Configure Syslog and SNMP Trap servers.

	Regex	Source Type	Source Host	Time Threshold	
				Hours	Minutes

☐ Email ☐ Syslog Notification ☐ SNMP Trap Notification

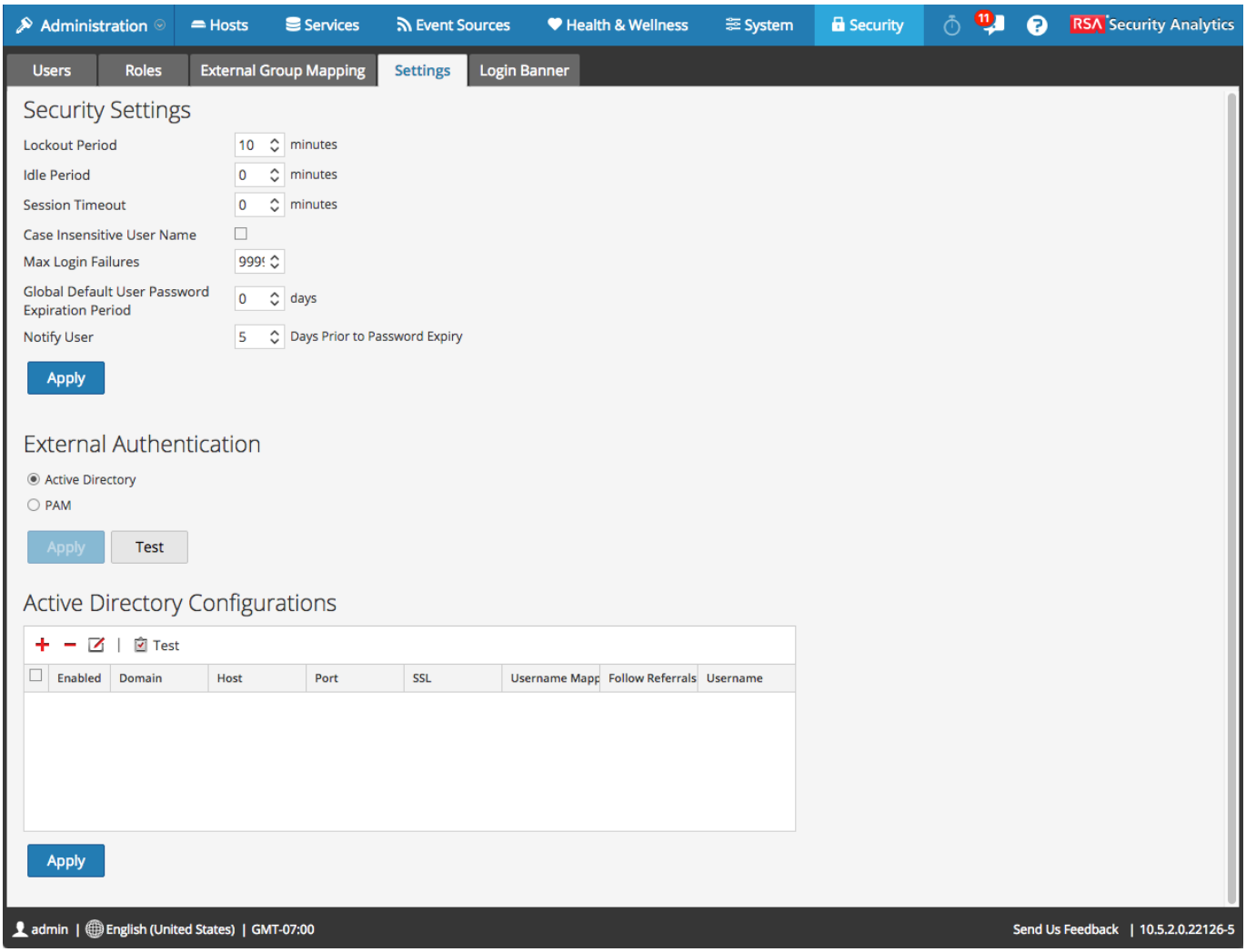
Decommission

admin | English (United States) | GMT-07:00 Send Us Feedback | 10.5.2.0.22126-5

35. Integrated Active Directory Authentication

One needn't bother with defining all SA users as SA system users. SA supports direct integration with Microsoft Active Directory (AD). To configure this, navigate to **Administration->Security->Settings->External Authentication** in the UI.

Under **External Authentication**, select **Active Directory**. This will use SA's internal AD authentication mechanism, which uses LDAP.



Prerequisites

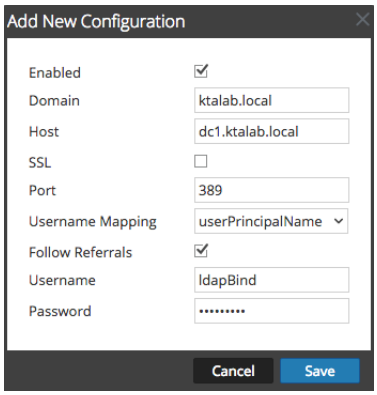
You need to know a few things before you can set AD authentication set up:

- A dedicated Active Directory account to use for LDAP binding.
- What AD groups do you want to allow to access SA, and which SA role(s) do you wish each AD group to map to? E.g. I want to allow my AD group 'socEngineers' to access SA, and I will map them to SA's **Operators** role. I may also have an AD group called 'socAnalysts' which in turn I want to have access to SA with **Analysts** role permissions.
- Your AD domain name
- The hostname or IP of a domain controller
- The LDAP port
  - The standard LDAP port is 389. This is standard LDAP and will authenticate only against the configured domain.
  - Port 636 is for LDAPS, or LDAP with SSL. In order to use SSL with LDAP on an AD server, the server must first have an SSL certificate installed, which does not happen by default. The procedure to do this is beyond the scope of this course.
  - Port 3268 is for the AD Global Catalog port. The Global Catalog is a way of querying an entire AD forest across multiple domains.
  - Port 3269 is the SSL-enabled Global catalog LDAP port.
- Whether to use the **userPrincipalName** (UPN) AD field for the username or the **sAMAccountName** field. This is usually set to UPN. If one doesn't work, then try the other.
- Whether to follow LDAP referrals. This is almost always 'yes'.

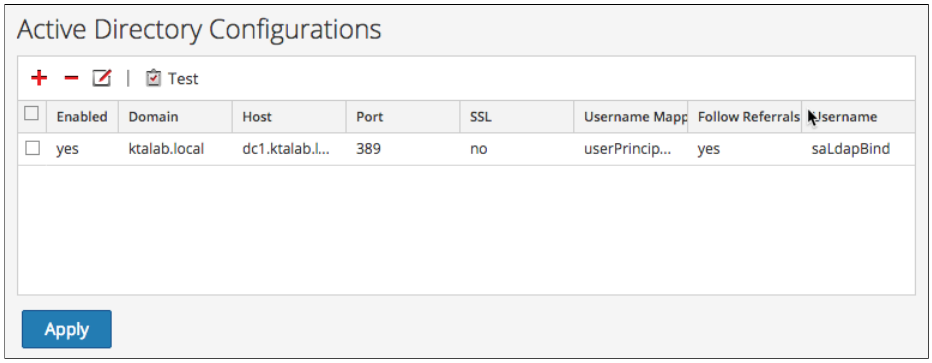
AD Settings

1. Under Active Directory Configurations, click the + button to add a configuration for a new AD Domain.

2. Fill in the settings based on what you learned when researching the prerequisites.



Adding a new AD Configuration



List of AD Configurations

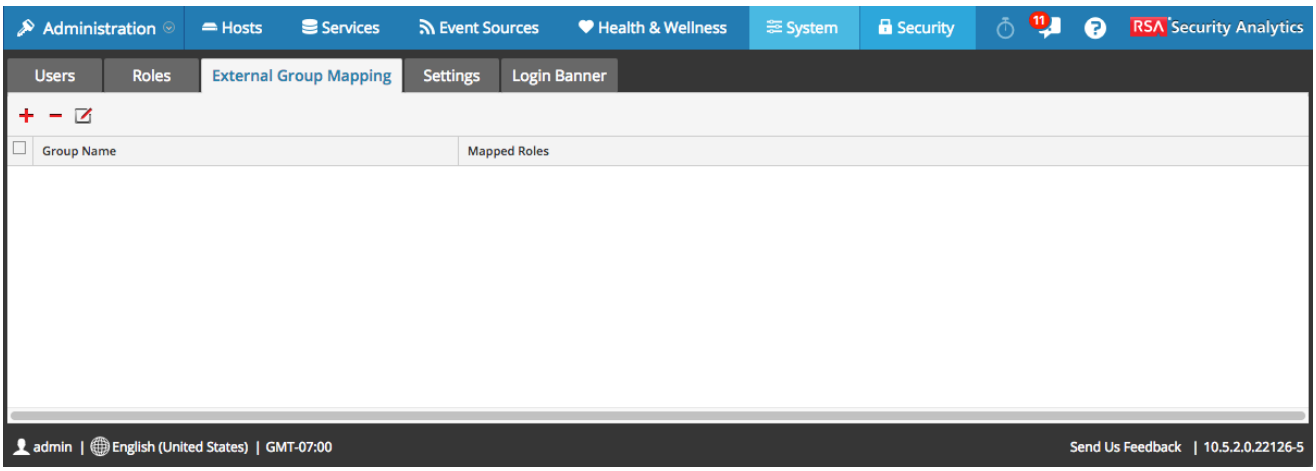
3. Click the Test button to confirm AD connectivity.

4. Click **Apply**. Don't forget this!

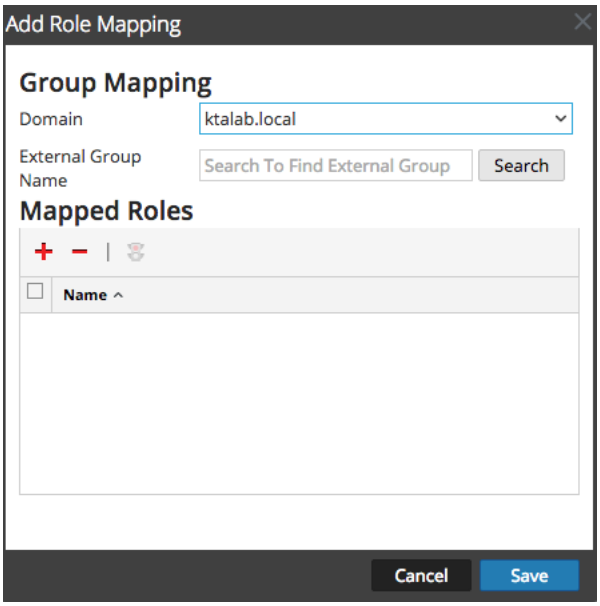
Mapping External Groups to SA Roles

In order for AD users to be permitted to login, one or more AD groups that they belong to must first be mapped to SA roles.

1. Navigate to **Administration->Security->External Group Mapping** in the UI.



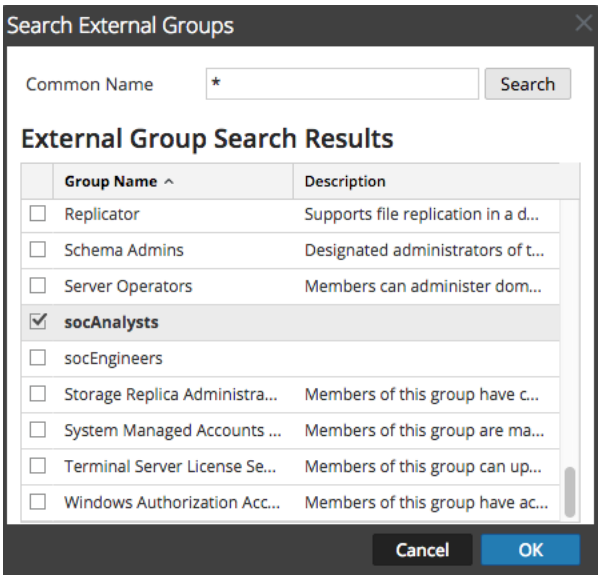
2. Click the + button. Select an AD Domain from the **Domain** dropdown. AD connectivity must be working before this will populate.



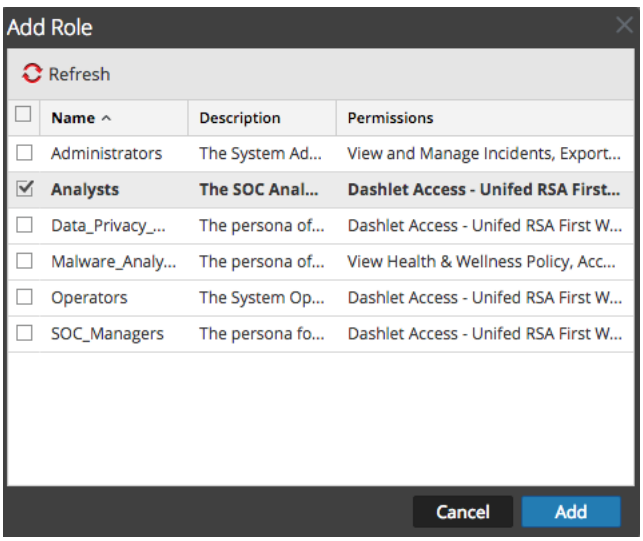
3. Click **Search** to select an AD group.

4. Type an asterisk in the **Common Name** text box and click **Search**. You may also type a filter string. All AD groups should now be displayed.

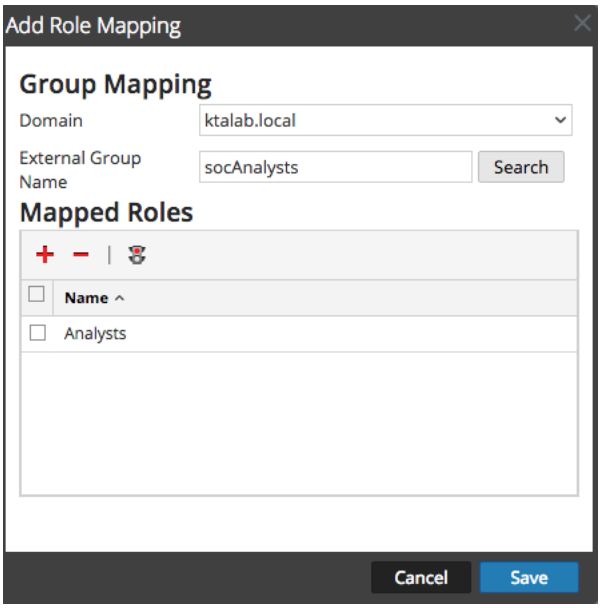
5. Select an AD Group and click **OK**.



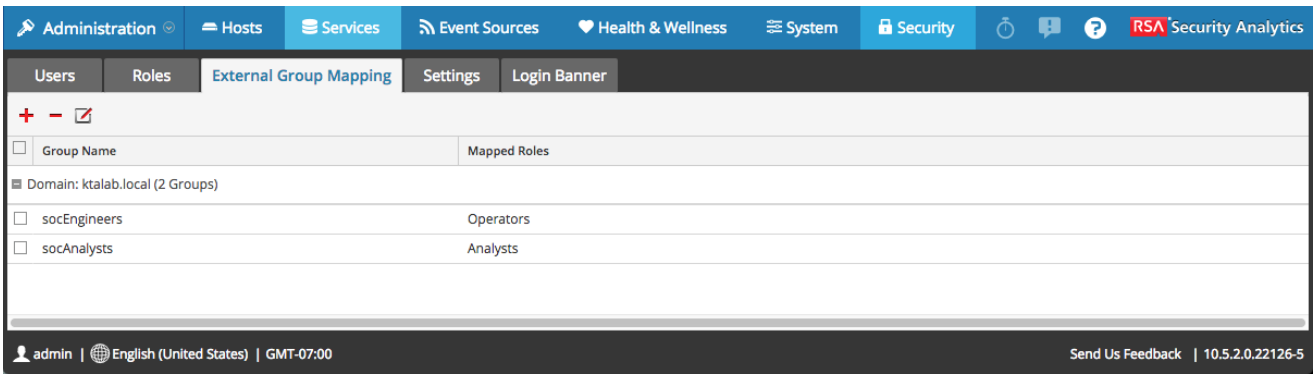
6. Click the + button to select an SA role or roles that you wish to map to the AD group and click **Add**.



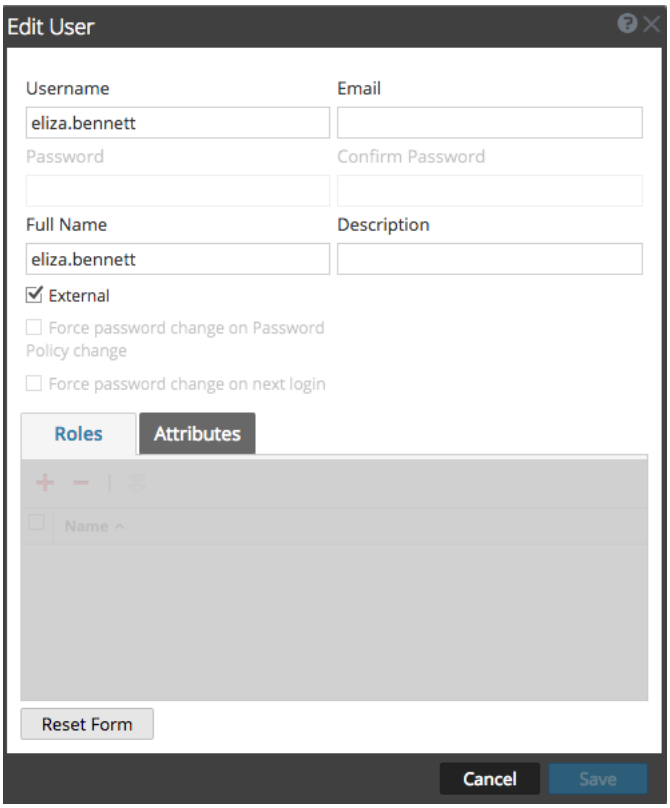
7. Click **Save.**

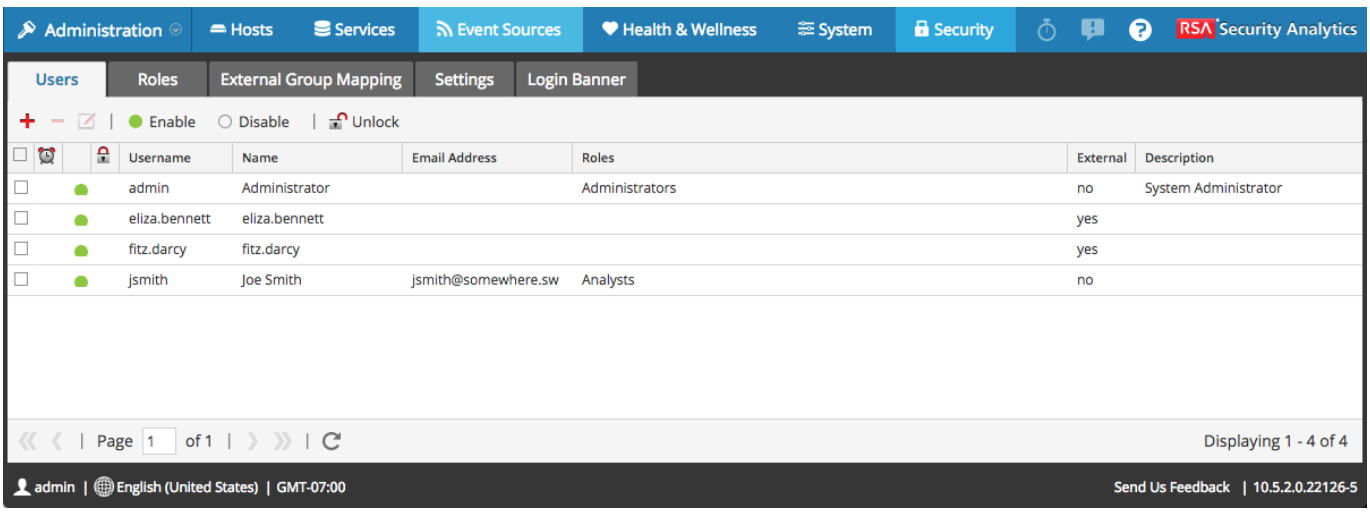


8. If all has gone well, then you should now be able to login to SA as an AD user who is a member of the AD group you just mapped.



Top Tip: Every AD or external user who logs into SA will automatically have a user account created for them under **Security->Users.** Their **external** flag will be set to **yes,** and the roles tab of their user profile will be greyed out, as the roles are mapped automatically for external users.





Lab Exercise

Now you give it a go. There is already a Windows domain controller in the lab.

Objective: Log in to SA using credentials for both of these users

Username	Password	AD Group Membership
fitz.darcy	NetWitness1!	socEngineers
eliza.bennett	NetWitness1!	socAnalysts

Base Parameters:

AD Domain Name	Domain Controller Hostname	LDAP Bind Account Name	LDAP Bind Account Password
ktalab.local	dc1.ktalab.local	ldapBind	ld@pb!nd1

Can I Require That AD Users First be Created in SA by an Administrator?

Unfortunately, this is not possible at present.

36. Audit Logging

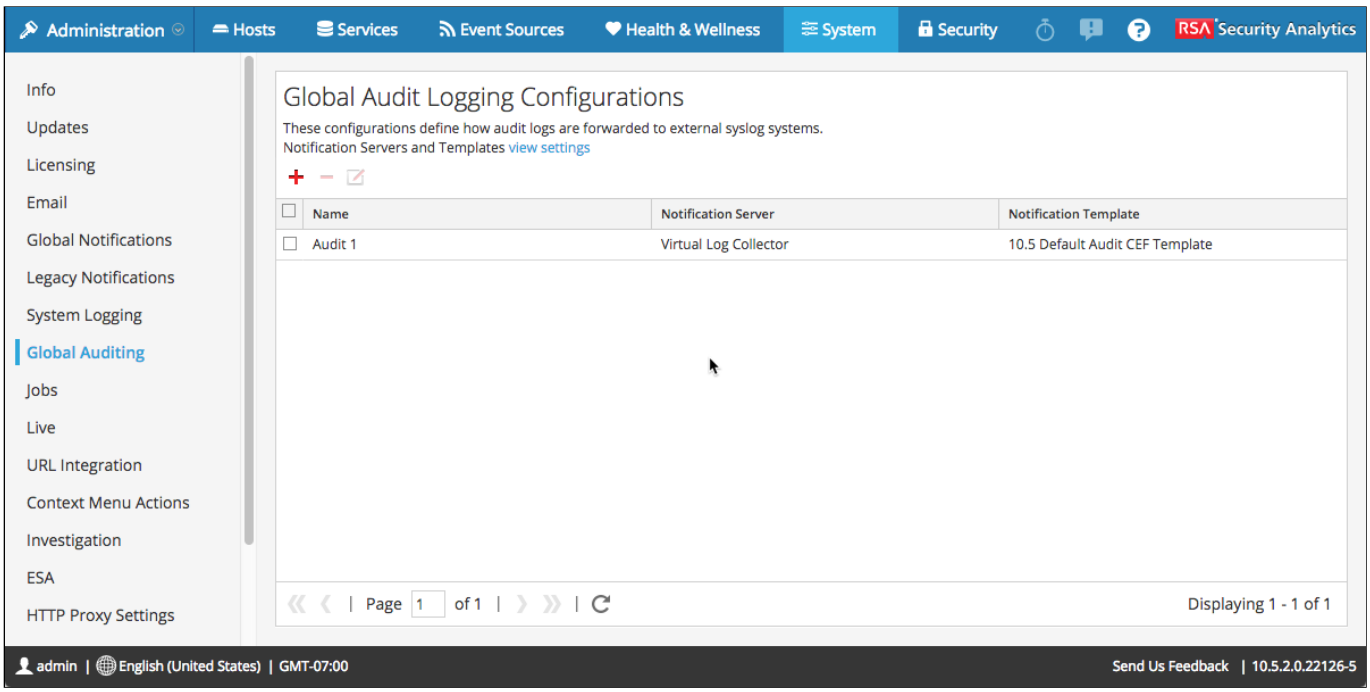
New with Security Analytics 10.5 is an audit logging feature, whereby SA services are now capable of generating audit logs of SA user actions in a consistent format, to track what SA users are doing inside the system. Many organisations need audit logging to fulfil compliance and/or regulatory requirements.

Scope

Every SA service supports audit logging, both Core and Java services.

Configuration

The configuration of audit logging is configured in the UI under **Administration -> System -> Global Auditing**.



What Gets Logged?

Any event that a user performs in SA. This is any queries performed by the user or any configuration changes made.

Where Does it Log To?

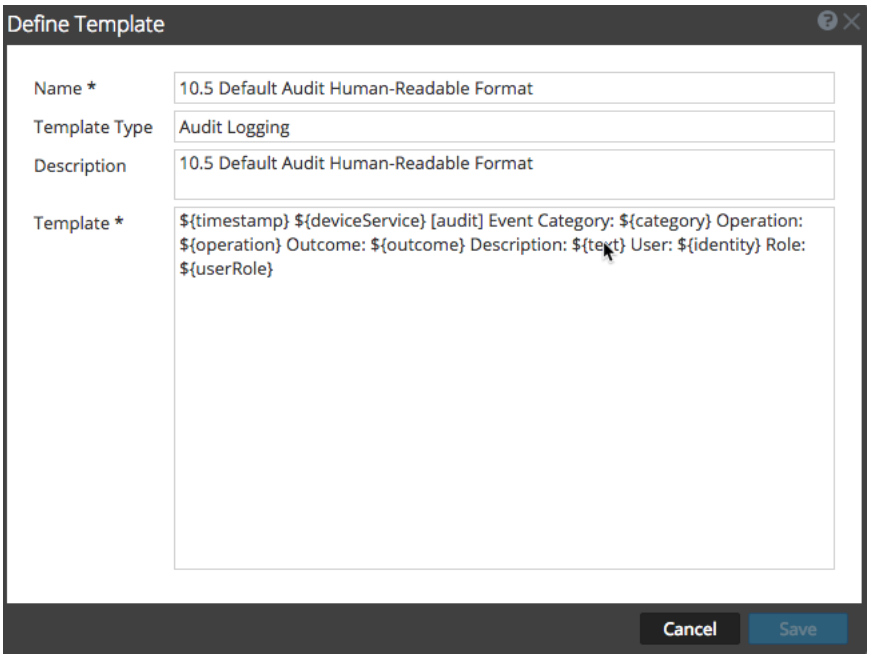
SA's audit logging feature sends logs via syslog to any syslog receiver. This can be a third-party syslog server product, or it can be an SA **Log Decoder** or **Remote Log Collector**, if you wish to ingest audit logs back into SA.

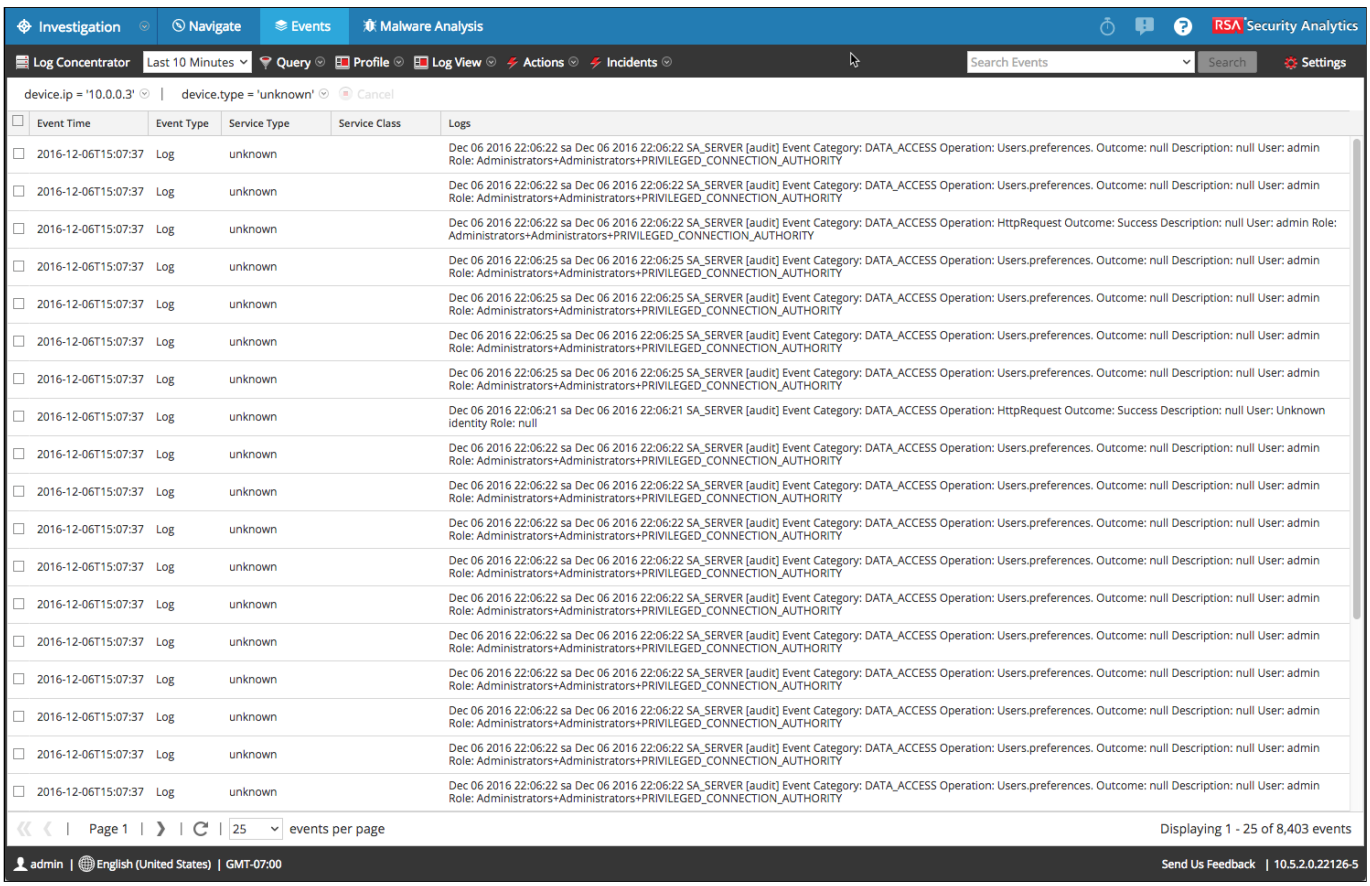
Please note that if you wish to consume SA audit logs with SA for Logs so as to be able to drill into and report in its meta, **you will need to own a Log Decoder**.

Audit Log Templates

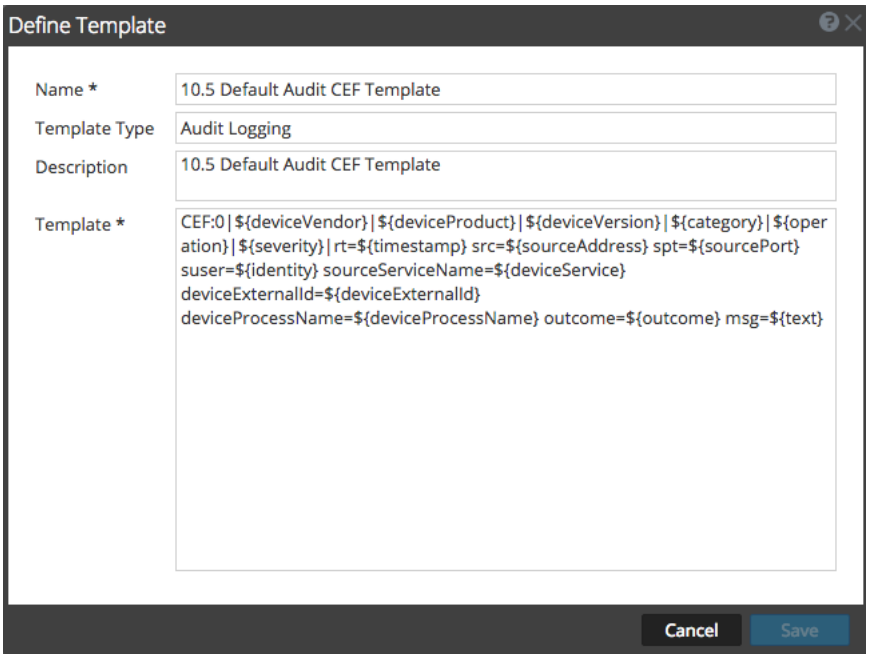
SA is packaged with two templates which can be used for audit logging.

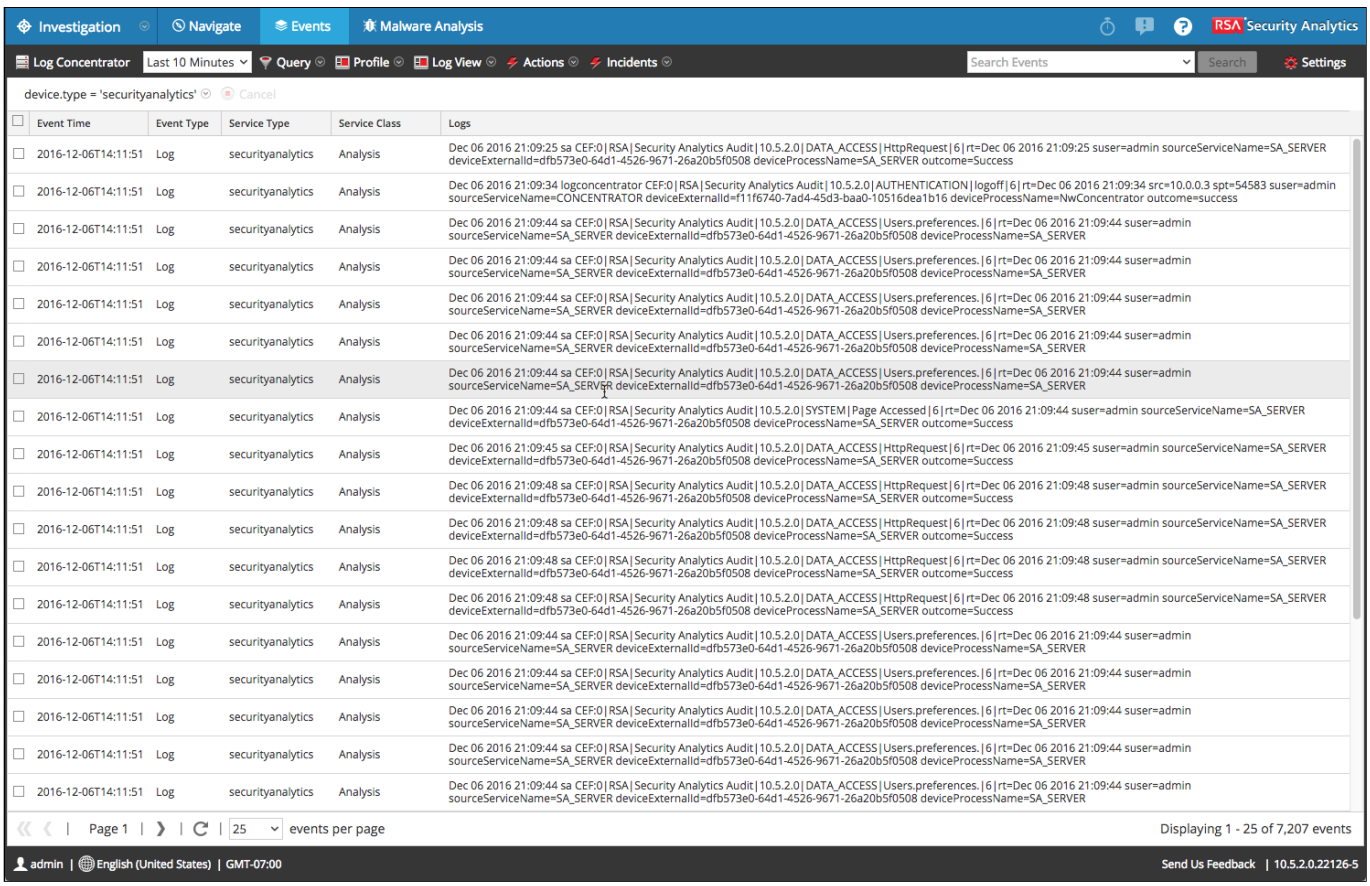
- **Human-Readable, or "10.5 Default Audit Human-Readable Format"**: This can be used to send syslogs to a syslog server where humans will be reading the logs directly.





- **CEF, or "10.5 Default Audit CEF Template"**: This uses the ArcSight Common Event Format, a de facto standard in logging formats. This format is easy to parse, but can be quite difficult for humans to read from a logfile. Use this if you are ingesting your audit logs with SA for Logs.





Enabling Audit Logging

Audit logging isn't turned on by default, so if you wish to use it, you must configure it. To enable audit logging:

1. Click the + button

2. Give a friendly name to the configuration

3. Choose a notification syslog service. See the lesson on Global Notifications for configuring one of these.

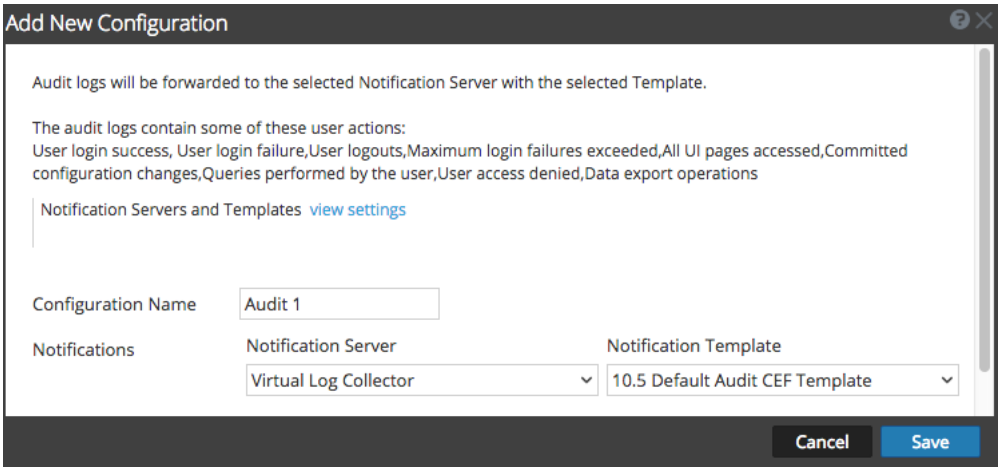
4. Choose a template notification template type. If you are using a SA for Logs to consume audit logs, you will want to choose "**10.5 Default Audit CEF Template**". If humans will be reading these logs, there is another template which produces logs which are much easier on the eyes called "**10.5 Default Audit Human-Readable Format**"

5. Click **Save**. Your audit log settings will not take effect until the local **puppet-agent** on the SA server has had a chance to re-run, which may take up to half an hour or so. If you're impatient, you can always run "**service puppet restart**" from the command line to force it to run sooner.

***puppet-agent logs after changing audit logging config*:**

```
Feb 10 21:04:29 sa puppet-agent[2795]: (/Stage[main]/Rsa-audit-server/File[/etc/logstash/conf.d/rsa-audit-server-
output.conf]/content) content changed '{md5}6712fdb8f122c9132c3d2192a6dbdb87' to '{md5}c277fb2d3773b46867d15777449ad2f6'
Feb 10 21:04:55 sa puppet-agent[2795]: (/Stage[main]/Rsa-audit-server/Service[logstash]) Triggered 'refresh' from 1 events
```

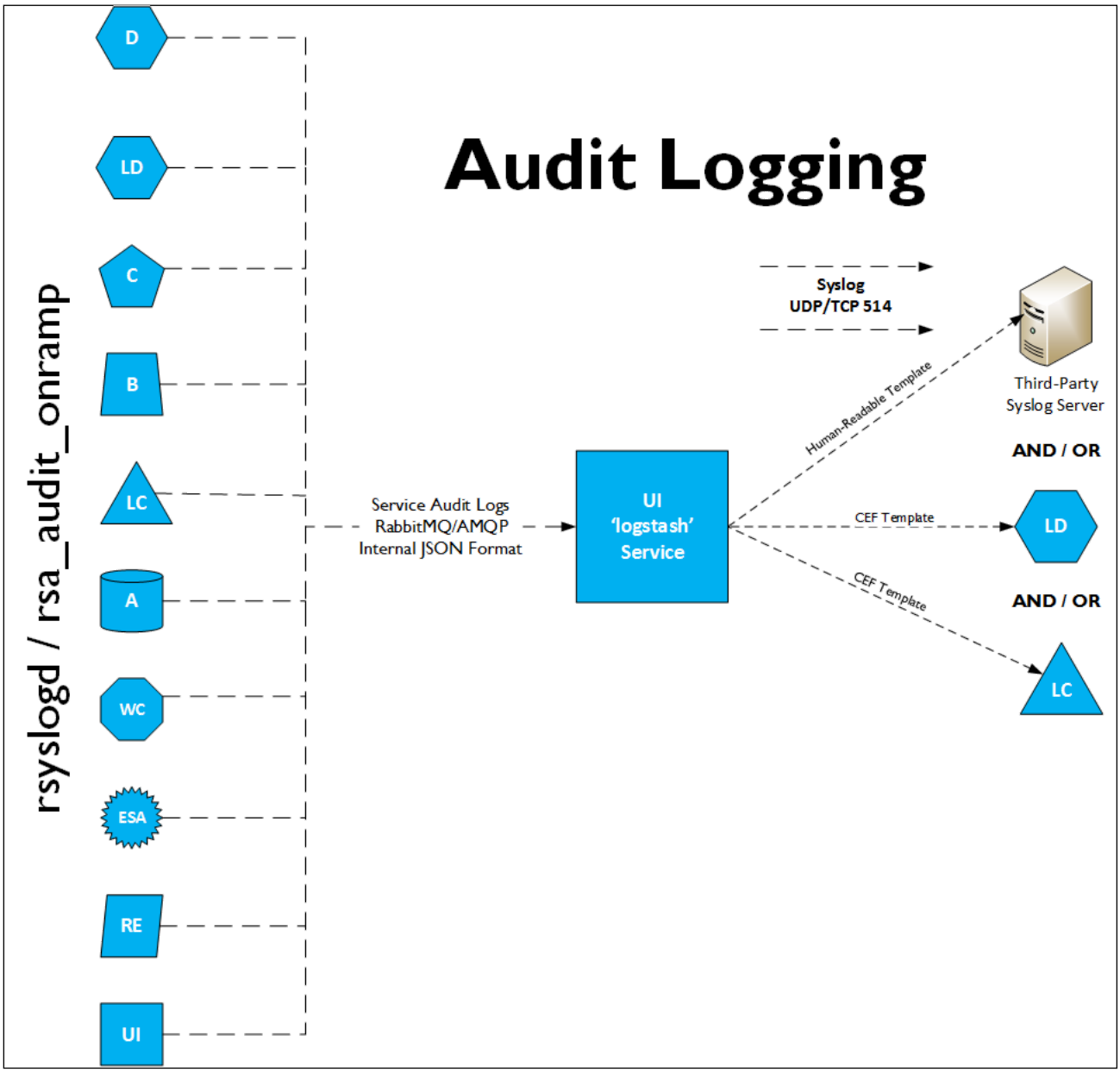
Top Tip: If you are consuming CEF-formatted audit logs with SA for Logs (i.e. a Log Decoder), you will first want to install and enable the **CEF** parser from RSA Live.



How Does it Work?

The process of generating audit logs is multi-step:

1. An SA user performs an action which triggers an audit log.
2. The SA service (Core or Java) generates an audit syslog message which is captured by rsyslogd running on each SA host.
3. The rsyslogd service runs a plugin called rsa_audit_onramp, which captures SA's audit syslog messages, and then transforms them into JSON format, and then forwards the audit message over the RabbitMQ Message Bus to the UI host.
4. The **logstash** service running on the SA UI host reads the audit JSON-formatted audit log off the message bus.
5. The **logstash** service transforms the JSON-formatted log into a syslog message using the template specified in **Administration->System->Global Auditing**.
6. The template-formatted audit syslog message is transmitted by the Logstash service to one or more syslog receivers specified in **Administration->System->Global Auditing**.

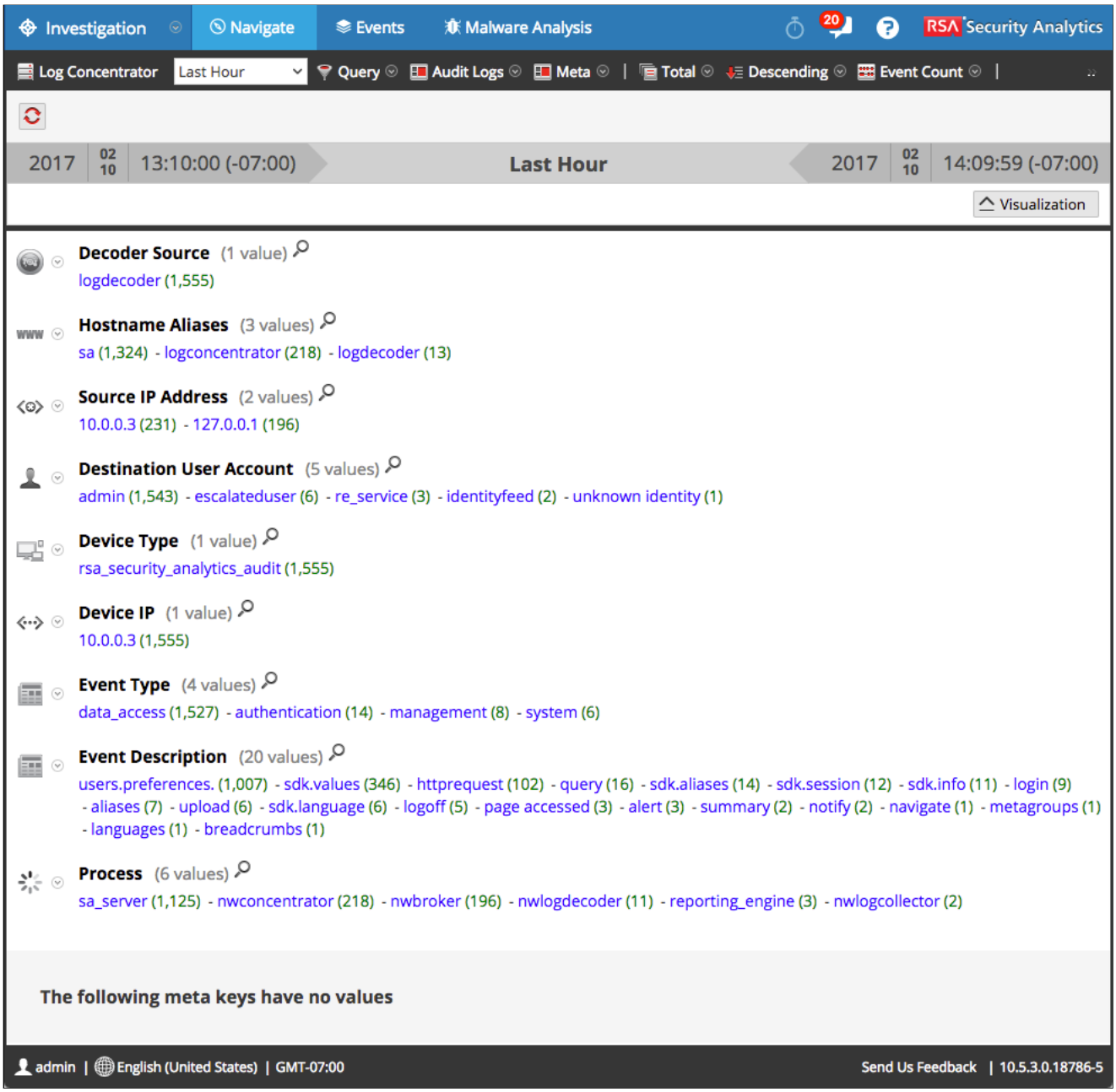


Taxonomy

The taxonomy of audit log field values is not yet completely consistent between every service (i.e. one service may say *logon* and another *login*), though I expect this will improve with future SA releases.

Device Type for Audit Logs

SA Audit logs are parsed as **device.type='rsa_security_analytics_audit'**. To see your audit logs, query for this in **Investigation** or **Reporting**.



Troubleshooting logstash

- The logstash service logs to **/var/log/logstash/logstash.log** .
- Its configuration files are located in **/etc/logstash/conf.d**. They are:
  - **rsa-audit-server.conf** - This tells the logstash service details about connecting to rabbitmq to pull the audit log data

```
#
# Copyright (c) RSA The Security Division of EMC
#
input{
    rabbitmq{
        host => "localhost"
        port => 5672
        vhost => "/rsa/system"
        exchange => "carlos.audit"
        key => ""
        queue => "ls-audit"
        auto_delete => true
    }
}
```

  - **rsa-audit-server-output.conf** - This tells the logstash service how to output audit log data: what methods (tcp or udp) to use, where to transmit to, and the template with which to format messages.

```
#
# Copyright (c) RSA The Security Division of EMC
#
output {
        rsatcp{
            host => "logdecoder"
            port => 514
            codec => sacodec {
                template =>
"CEF:0|${deviceVendor}|${deviceProduct}|${deviceVersion}|${category}|${operation}|${severity}|rt=${timestamp} src=${sourceAddress}
spt=${sourcePort} suser=${identity} sourceServiceName=${deviceService} deviceExternalId=${deviceExternalId}
deviceProcessName=${deviceProcessName} outcome=${outcome} msg=${text}"
                facility => "USER"
            }
    }
}
```

Starting and stopping logstash

Starting:

```
service logstash start
```

Stopping:

```
service logstash stop
```