1. Splunk 是机器数据的引擎。使用 Splunk 可收集、索引和利用所有应用程序、服务器和设备生成的快速移动型计算机数据。Splunk利用内在的机器学习方法，对收集到的数据自动进行整合，制表。

2. SplunkBase是Splunk的社区，里面有社区人士专门为某些特别的应用例如AWS, VMWARE等任务写的Add-on包。

3. Splunk主要有3个组件。Indexer, Search Head, Forwarder.

Indexer(索引器)： 用于将unstructed的数据变为structured. 建立好文件夹保存到Splunk数据库中。并且执行具体的搜索工作。

Search Head（搜索头）: 使得可以使用 Splunk搜索语法去搜索数据，并且整合返回搜索结果。提供各种的结果图标。

Forwarder(转发器): 是Splunk Enterprise的实例， 安装在服务器上，将搜集的数据发送到Indexer中。

Note: 这三个基本组建可以部署在单个机器上，也可以部署在分布式的结构上。

4. Splunk的三种角色(roles):

Admin: 管理员可以安装apps, 创建 knowledge objects.

Power: 可以创建knowledge objects， 分享knowledge objects, 以及完成实时搜索

User: 只能看自己或者被分享的knowledge objects.

5. Splunk自带两个Apps, 一个是Searching & Report。另外一个Home App这个App用于添加管理其它App.

6. Splunk 常用的启动命令: ./splunk start, 停止命令 ./splunk stop, 查看帮助指令信息 ./splunk help

设置Splunk在机器启动时启动， ./splunk enable boot-start

7. Splunk主要有三种手段去获取数据并且为数据建立索引(indexing) 。第一个种是上传文件，主要用于测试以及不变数据。 第二种是选择文件进行监视（monitoring），该文件会不断变化，用于检测数据变化。最后一种是forwarder, forwarder是生产环境比较常用的一种，从远程机器中不断将数据发送给Splunk 的indexer.

8. SourceType 即是表明数据的来源是什么。Splunk中有很多预定义的数据来源， 例如csv文件，json文件等等。当Splunk接收到数据时, Splunk会尽可能的自动识别数据源类型。然后运用这些预定义的sourcetype让Splunk根据预定义的格式去读取，一般在预览的结果上都可以很好的识别。 如果遇到一些特别的格式，或者Splunk无法自动识别，我们可以自定义一个新的sourcetype，自定义读取规则然后保存为一个新的sourcetype.

9. 当数据根据自定义的sourcetype中的规则，或者Splunk预先定义的规则进去Splunk后。Splunk会为这些数据进行索引(indexing). 推荐的做法是将数据分类的放入不同的数据index中，不要全部数据都放入同一个数据索引中。这样可以加快数据的搜索时间，还可以在以后根据role的不同，使得不同用户，根据权限，去访问不同的数据index.

10. 在使用Splunk对数据进行搜索，发送邮件通知，数据监控时，首先需要将数据通过上传，监控，或者Forwarder的方式引入到Splunk中，让Splunk为这些数据建立 起索引（indexing）先。

11.当我们在Splunk进行搜索的时候将进行搜索任务，搜索任务默认持续10分钟。10 分钟以后，Splunk需要重新进行查找以更新。

12. Splunk有三种搜索模式， Fast模式只返回搜索的字段。Verbose模式将返回尽可能多的返回字段。Smart模式将根据运行的搜索类型动态返回。

13.Splunk 搜索完毕以后，我们可以看到事件列表以及时间轴。通过时间轴，我们可以在时间轴上拖动从而对某个时段进行具体的搜索。在事件列表上我们可以选择关键字，去更新搜索的条件。我们还可以共享这个搜索结果，改变搜索结构的有效期，根据搜索的结果查看图标，保存为文件，或者结果面板等。

14. Splunk搜索语言 (Splunk Search Language) （简称SPL）:

主要由5个元素组成：

|  |  |
| --- | --- |
| \* 万能匹配符号 | AND 逻辑与（两个搜索条件什么都不写，默认为and） |
| OR 逻辑或 (小写的or 不行) | NOT逻辑否 (例如 not match(A,”B”) |
| \ 转义字符 | =，！=等于不等于（用于数值或者字符比较） |
| >,>=,<,<= （用于数值比较） |  |

**Search Term:** 搜索条件词，利于等，不等于…

**Commands:** 告诉Splunk我们要对搜索结果执行什么操作。例如计算统计数据，画图，制表

***fields***: 让搜索包含，或者去除某个字段。 (fields status clientip 只提取status clientip) (fields – status clientip 不提取status和clientip字段)， 限制提取字段(field extraction)可以改善性能，而去除提取的字段(field exclusion)对性能没有变化，只会改变显示的结果。

table: 让提取的字段以表格的形式展示 (table status clientip, 展示status和clientip字段组成的表格)

rename: 重命名字段 rename 提取字段 as 重命名字段

dedup： 对指定的提取字段去掉重复的结果, 为了性能的提供应该尽早在搜索中使用

sort: 根据提取的字段对搜索结果排序 sort +/- field\_name

top: 显示数据集中数量最高的几位。

rare: 与top相反，显示数据集中数量最低的几位

stats: 生成数据的统计数据命令。（我们可以在搜索中使用多个stats函数但是必须保证他们处于同一管道中，且只能有一个stats 命令， count, distinct\_count, sum, list, value为stats类型的函数）

Stats count： 统计符合条件的事件数量, count(X) X为field的名称

Stats distinct\_count: 统计符合条件的事件唯一计数

Stats sum: 统计符合条件的事件计数总和

Stats list ： 列出数据中某一字段的所有出现过的值

Stats value: 列出数据中某一字段的所有的不重复出现过的值

**Functions:** 告诉Splunk我们对搜索结果在执行操作的时候，需要执行的逻辑

**Arguments:** 函数的参数

**Clauses:** 说明系统结果进行如何分组或者定义

By: 将统计结果进行分组。

例如符合条件的数目有3000个。然后我们使用By语句，By product name, 那边符合条件的这3000个会根据product name 再具体划分。

例如一条Splunk搜索语句：

Sourcetype=acc\* status =200 | stats list(product\_name) as “Game Sold” | top “Games….

Sourcetype=acc\* status =200 为Search Term

| 为搜索管道，代表把结果交给下一个管道处理

Stats 为 command

List 为Function名 product\_name为参数， as为Clauses 定义为” Games Sold”

又例如chart values (RegressionUpgradeTo)

chart为command

values为Function名 RegressionUpgradeTo为参数

15. Splunk搜索结果的用户界面左边含有搜索的字段，一般包含有两种。Selected Fields以及Interesting Fields. Selected Field是比较重要的字段，其中也显示在事件的列表中。Interesting Field是在搜索结果中至少含有20%的覆盖率的，有可能有价值的字段。我们可以让Interesting Field中的字段加入Selected Fields中。

16. 我们在搜索框中，可以设置筛选条件，筛选事件结果。我们可以填入，事件那内容是否含有某个关键字，事件是什么sourcetype, 事件中是否有某一字段等。

17. Splunk字段类型表示中，#表示数字类型. a 表示字符类型。

18. Splunk搜索的结果默认会在10分钟后过期。但是Splunk搜索的历史纪录很久都不过期。

19. Splunk在搜索的时候，我们指定搜索条件，例如index, 时间范围，等会让搜索更快速更优质的搜索结果。

20. Splunk搜索指定最早时间为earlies=, 指定最晚时间latest=，值可以指定为-3h（三小时以前）也可以指定具体日期时间

22. 想看查看Splunk存有哪些index情况时，可以在 Web界面上点击Settings->Data->Indexes, 或者使用指令 | REST /services/data/indexes | table title, currentDBSizeMB 用于查看Splunk中现在的index情况。

23. Splunk让搜索的结果方便的形成Dashboard(仪表盘)或者报告(report)，并且分享给其他用户。报告可以选择定时生成，可以选择生成只含报表或者图标的报告。而Dashboard(仪表盘)上可以放置多个Panel, 并且放置时间选择器，在Panel上改变时间，整个仪表盘都会产生变化。

24. Splunk在搜索语句的时候，一般都不用加双引号。如果字符串中包含有空格，则必须有双引号。

25. Splunk 展示的时候，创建一个dashboard, 这个dashboard上可以放很多的 Panel, 然后每个Panel都能放置一个图表 (graph, chart) 或者报告(report)，每个图表或者报告先通过search命令生成。然后放在Panel上，然后再加入dashboard中。在dashboard中，可以拖动Panel去改变位置。

26. User, Power, Admin都可以创建Report.

27. Pivot 是数据透视图工具，可以让工具在不使用Splunk 搜索语言的知识背景下。通过GUI选择生成图标报告等。Pivot有两种，一种是管理员预先定义数据模型，然后用户使用。另一个是快速Pivot，管理员甚至都不需要预先定义数据模型，用户直接进行GUI层面的搜索(在保存时Pivot时，会自动创建数据模型) 。

28. Splunk有transforming和non-transforming命令区别。Transforming命令，例如stats, chart, timechart, table将会令数据变为图表，表格。而non-transforming命令只是筛选数据，不会将数据变为表格，图表形式。这种non-transforming命令Splunk会提供instant pivot按钮。而transforming的命令不会，因为这种命令已经生成了图表。

29. Splunk的lookup功能通过数值域的映射，为现存的数据中加入新的数据。例如当前数据中有HTTP状态码204， 但是我们希望在搜索出来的结果中显示204的文字说明。此时HTTP状态码的文字说明不在数据集中，我们通过一个外部的CSV文件来补充说明，这个外部补充的CSV文件就是Lookup文件。在这个CSV文件中含有两列数据，HTTP状态码和状态说明。当引入Splunk时候，HTTP状态码作为数值域的映射，数据集中的状态码与外部CSV文件中的状态码相同时，在搜索的结果中加入HTTP状态说明。

30. Splunk的Lookup分为2步，第一步是引入外部文件，第二步是建立Lookup。建立Lookup又可以让Splunk自动映射或者在Splunk Searching Language中指定。

例如手动映射： | lookup products\_lookup productid as product Id OUTPUT product\_description as Description

| lookup lookup表名 表名中的域与数据集中的域关联 OUTPUT lookup表中的product\_description别名输出Description

31. Splunk中 某个报告或者搜索的Run As（通常有Owner和User可以选）的意思是，当运行该搜索的时候，可以搜索的数据集是Owner可见的数据集或者是User可见的数据集。作用的避免生成的报告暴露权限外可以看到的数据。

32. Splunk中可以定时的运行搜索生成报告。也可以定时或者实时不断的搜索达到条件的时候，达到报警效果，进而执行动作（包括，发送邮件，执行脚本，点击链接等等）。我们还可以设置报警的频率，故障发生的频率等

33. Splunk默认以时间从晚到早返回搜索的结果。

34. SplunkBase的Slideshow App 简单易用，达到幻灯片定时切换展示图标的效果。Slideshow这个App无法产生dashboard或者report，这个App所展示的dashboard还有report是由其它App（例如Searching&Report）产生的，然后把权限改为AllApps, 共享给slideshow的。

SplunkBase的Website Monitoring用于监控网站的反应时间 (response time, response status等)

SplunkBase的Splunk Dashboard Examples包含Splunk生成图标的的教学以及编辑，提供方便使用的Visual Editor. 并且整个图标以XML的形式表示。

35. 在Web管理界面中可以加入或者删除修改的Index.

36. Splunk的license一开始是Trail License 过一段时间以后将变为Free License, Free License则失去所有用户权限控制等等。Trail License无法被再次激活，接下来只能一直使用free license或者买license.

Free License 无法达到的效果为：

--分布式搜索无法执行

--转发的数据只能转发到Splunk实例，无法转发给非Splunk实例

--实时发送警告等动作无法执行，但允许利用实时数据生成图表

--报告加速生成无法执行

--无法作为部署服务器

--无任何验证以及角色管理

----无登录认证，无需密码

----登陆进去都是admin，无法创建其他用户

----搜素在所有的index上，相当于只能指定index= “\*”

如果我们建立的splunk的独立的(standalone)的架构，那么license可以是本地的。如果是分布式的部署，那么我们要将其中一个实例配置为license master,并且配置证书池，让其他 splunk实例作为licence slave.

37. 我们创建报告，或者图标或者dashboard的时候，实在具体的APP环境下生成。所生成出来的结果默认情况下属于该App, 其他App并不能可见。例如在Searching&Reporting中 能搜索到的dashboard或者report，要不就是在自身Searching&Reporting中生成的dashboard或者report, 要不就是在其他App生成，但是其中设置了对所有App可见。

38. 用于发送数据的Splunk实体叫做Splunk Forwarder。 Splunk Forwarder可以远程发送数据到Indexer当中。并且通常在数据发生地，产生地安装Forwarder. Forwarder不会消耗太多的资源，所有在产生数据的机器中不会影响其运行。

用于接收数据的Splunk实体叫做Splunk Receiver. 接收方通常是Splunk Indexer将数据保存以及搜索。

39. 有三种forwarders:

1. 通用转发器 (universal forwarder): 只包含需要转发数据的组件。最轻量化。它是一套独立的软件，而不是Splunk Enterprise实例，所以不从Splunk Enterprise 上启动。并且配置过程和配置重型转发器或轻量转发器都不同。

2. 重型转发器 (heavy forward) : 是一个完整的Splunk企业版实例， 他自身可以index, search, 并且转发data. 只有少部分功能缺失。

3. 轻量转发器 (light forwarder) : 也是完整的Splunk企业版实例， 但是有许多功能被禁用，以减少资源需求。基本被通用转发器替代。

40. Splunk在使用Forwarder进行数据转发的时候，可以有多个转发器，逐次转发最终到达目的服务器。

41. 由于通用转发器不是Splunk Enterprise实例，所以无法在数据发送之前对数据进行分析，改变，或者根据数据的内容进行转发。但是重型转发器，轻量转发器可以。

42. 转发的数据分类：

--原始数据（Raw）: 转发器不对转发的数据进行任何处理，直接发送。

--未解析数据 (Unparsed Data): 转发器对数据添加某些头内容，例如来源，数据类型等。

--解析数据 (Parsed Data): 转发器对数据进行解析，或者其中的域，并且根据其中的域值进行路由等操作。

我们称为解析数据与解析数据为加工过数据(Cooked Data). 默认情况下转发器发送加工过数据（通用转发器发送未解析数据，重型解析器发送解析数据）

43. 部署forwarder时候，要检查Splunk Enterprise 实例（接受数据者）是否和转发器版本兼容。部署完以后，可以在receiver上求证是否存在发送来的数据，搜索语句host=<forwarded host name>

44. 在部署forwarder环境的时候，一般我们先配置receiver. 在receiver上启动接受数据选项，我们可以通过Splunk Web方式, CLI命令行的方式, 对inputs.conf的方式进行配置。（主要的达到的效果是让receiver监听某个端口）

45.通用转发器Universal forwarder不会自动启动，也需要通过. /splunk start启动。 每次改变转发器的配置时候，也需要重启转发器才能生效。

46. 配置通用转发器，由于通用转发器上没有Splunk完整的实例，所以我们无法通过WEB GUI界面配置。我们可以通过CLI，修改配置文件，通过部署服务器的方式修改。

47.通过转发器的配置文件(存在$SPLUNK\_HOME/etc/system/local/)：

--inputs.conf: 控制转发器如何搜集数据

--outputs.conf: 控制转发器如何发送数据

--server.conf: 控制转发器链接和效能

--deploymentclient.conf 控制转发器链接部署服务器配置

P. S ***outputs.conf这个配置可能在多个地方都存在，例如可以在apps目录下有，在$SPLUNK\_HOME/etc/system/local 下也有，转发器根据优先级将把他们所有的配置都整合在一起，然后作为此次启动的配置***。

P. S 如果我们修改outputs.conf一般都只会对$SPLUNK\_HOME/etc/system/local/进行修改。不会对APP中outputs.conf进行修改。

48. Splunk的应用Apps便放在$SPLUNK\_HOME/etc/apps下面。

49. 在转发器中有两种处理单元，tcpout和syslog. 通用转发器只支持tcpout.

通用转发器outputs.conf配置示例：

[tcpout] --- 代表使用tcpout处理器头，并且这一节是全局配置范围。有些属性只能在全局范围内配置，例如 defaultGroup

defaultGroup = default-autolb-group ---- 代表目标组名为default-autolb-group, 转发器将自动向这些组转发数据

[tcpout:default-autolb-group] ---- 配置目标组节，在这里定义一系列属于此组的接收器

server = 135.121.250.159:9997 ----地址为135.121.250.159 端口号为9997

[tcpout-server://135.121.250.159:9997] ----对具体主机额外配置，这里具体主机必须是某个目标组中的成员。并且此处对该主机的配置高于全局配置。

50. 我们还可以通过CLI对forwarder进行配置。通过CLI进行配置不需要直接编辑配置文件，优点是不会再配置文件中输入错误的字符，缺点是有一些配置参数无法从CLI进行配置需要直接对配置文件进行文本修改。通过CLI修改的配置splunk会自动写入文件中。

（API Documentation: http://docs.splunk.com/Documentation/Splunk/7.0.2/Data/MonitorfilesanddirectoriesusingtheCLI）

--配置转发器向receiver发送数据：

./splunk add forward-server <host name or ip address>:<listening port>

--配置转发器新增需要转发的数据：

./splunk add

--查看,添加，删除转发器将向哪些receiver发送数据：

./splunk list forward-server

./splunk add forward-server Ipaddress:port

./splunk remove forward-server Ipaddress:port

--查看转发器将向将发送什么数据：

./splunk list monitor

P.S 改变转发的目标服务器的时候，需要重启splunk服务

P.S 不仅可以monitor文件还可以monitor文件夹。

P.S 可以指定Monitor的文件发送到receiver的时候，是以怎样的sourcetype和存在哪个index中

51. Splunk Index以后的数据放在文件形式的DB中，路径在类似于$SPLUNK\_DB/galvatron\_data/db。 我们可以在Index页面找到

52.想到从Splunk删除时间的时候，先搜索然后使用管道| 导入到delete中。注意这样删除 只会让搜索不会找到删除的条目，但是磁盘空间并不会减少。 需要清空磁盘空间，只有等数据过期，或者删除整个index.

53. Splunk在Monitor文件的时候，Json格式的文件是比较好的选择，因为格式清晰，对Splunk来说非常易读。JSON中不要保留回车，最好直接一行。

54. Splunk使用transforming指令去生成图表

Chart 指令会将返回的结果成为一个表格，但是并不直接生成表格，我们进一步选择具体的图标类型后，从该表格中进一步 (Visualization)形成图表. 在该表格中将会存在的列为：求得stats function的列， by分组的列，以及在chart 以后通过eval 命令再加入的值

chart [<chart-options>] [agg=<stats-agg-term>]

( <stats-agg-term> | <sparkline-agg-term> | "("<eval-expression>")" )...

[ BY <row-split> <column-split> ] | [ OVER <row-split> ] [BY <column-split>] ]

--其中stats-agg-term可以包含stats-func，例如avg (), var (), sum ()等。

55. timechart指令是transforming指令，特别用于生成与时间有关的图表。X轴将是时间。

使用文档（<http://docs.splunk.com/Documentation/Splunk/7.0.2/SearchReference/Timechart#Stats_function_options>）

语法格式为：timechart [sep=<string>] [format=<string>] [partial=<bool>] [cont=<bool>] [limit=<int>] [agg=<stats-agg-term>] [<bin-options>... ] ( (<single-agg> [BY <split-by-clause>] ) | (<eval-expression>) BY <split-by-clause> )

--其中代表<single-agg> [BY <split-by-clause>] 或者<eval-expression>)BY <split-by-clause> 必须填写。在<single-agg> [BY <split-by-clause>]中single-agg又必须填写，在<eval-expression> BY <split-by-clause> 中<eval-expression>又必须填写。

--其中eval-expression会被展开成<math-exp> | <concat-exp> | <compare-exp> | <bool-exp> | <function-call>

--其中single-agg会被展开成count | <stats-func>(<field>)

--其中stats-func代表各种各样的statistic function例如avg(), count(),max()等，查看文档具体每个函数需要的参数

--timechart有默认的时间跨度，例如选择24小时内结果的时候，每个时间跨度采样点是30分钟。我们也可以在timechart命令中指定改写。Splunk的时间采样点总是以整体开始的，而不是以第一次有数据的时间点开始数时间间隔的。（例如5分钟时间间隔，那么是00，05，10这样）

56. 当时间传入Splunk Index中的域时，使用数字不包含单位。例如7s是无法生成图表Splunk将其看为字符串，而7是可以得。

57. 统计函数list(X)返回域名为X的100个值。Values(X)返回域名为X的所有值。

58. fields函数使得搜索结果只包含或者去除显示某些域。

Table函数也会让搜索结果只含某些域名。但是不同的是table会将结果以tabular的形式返回。而且 返回的结果只包含所指定的域，以\_开头的内在域都会被剔除。而fields只是不显示，但是内在域没有去除。内在域去不去除，关系到是否可以使用chart或者time chart去生成图表。Chart与time chart在生成图标的时候，需要内在域例如\_time.

59. HomeDashboard是一进入Splunk以后显示的dashboard。任何一个dashboard都可以设为HomeDashboard。

60. 在Searching&Report的Dashboard页面中。有三个选择，ALL|Yours|This APP。 这三个选择刚好对应权限的三种选择。ALL显示的是所有dashboard，包括其他APP中dashboard权限选择ALL APPS的图表。Yours下面显示的是现在登陆者（例如Admin），创建的所有dashboard。This APP代表的是在这个APP下面能看见的所有dashboard。

61. 当我们在创建一个dashboard，打算向里面加入一个新的panel，panel上放置一个新的图表的时候，一般先使用搜索去生成该图表，然后选择加入到该Dashboard中。如果是更改该Dashboard的标题名称，权限等，则直接Edit该图表。

62. Scheduled Report 不允许有Time picker出现，Scheduled Report出现的结果永远是最后一次运行的结果

63. Splunk 的Eval函数，Eval函数用于通过表达式计算出新的值，加入到表格中形成新的Field。

格式：eval <field>=<expression>["," <field>=<expression>]...

--在eval中指定添加的域名filed并且该域名是由后面expression逻辑产生的。如果该域名在表格中已经存在，则会覆盖该域。如果不存在则加入该域到表格中。

--eval与stats不同之处在于，eval会计算并且在表格中加入新的域。Stats只是计算数值而已，不会添加额外的域。

64. Splunk在搜索时候，在TimeRangerPicker中选择了时间以后，虽然在搜索框选中没有显示时间限制语句，但是搜索中是生效的。

# 65. 当传入JSON进去Splunk的时候，Splunk在默认情况下会自动提取key作为列名，将value作为值处理。如果key对应的是另外一个字典，那么Splunk会继续平铺该字典，将前面所有key以.连接，作为列名。将最后一层的value作为值。这种行为叫做（automatic key-value field extraction）

但是我们可以通过改变props.conf来改变这种行为（无法在Splunk Web上面配置）。我们可以设置Splunk只为某种sourcetype, host, index执行是否自动提取。但是注意，我们无法精确到具体key的行为。

（<https://docs.splunk.com/Documentation/Splunk/7.0.2/Knowledge/Automatickey-valuefieldextractionsatsearch-time>）

66. Splunk把每一行的数据成为event, 其中的列成为fields. 当Splunk读入数据或者index数据的时候，Splunk会做一个域提取（field extraction）工作.

Fields的类型：

|  |  |  |
| --- | --- | --- |
| Type of field | List of fields | Description |
| Internal fields（内部域） | \_raw, \_time, \_indextime, \_cd, \_bkt | 域名中以\_开头的为内部域。  \_raw代表了事件的原始数据  \_time代表了事件的发生时间  \_indextime代表了事件的index时间  \_cd代表了事件在index中的地址  \_bkt代表了事件所存放的bucketID |
| Default fields(默认域) | Host,index,linecount,punct, source,sourcetype,splunk\_server,timesetamp | 这些域会自动在index的时候加入到域当中。  Host: 表明事件产生的IP地址或者服务器域名  Index: 表明时间存在哪个index当中  line count: 表明在index之前，这个事件包含多少行  source：表明传入的文件名，数据流名等。  Sourcetype：表明使用什么数据类型  Splunk\_server: 表明事件存在那个Splunk Server上，在分布式Splunk部署中有用 |

67. Table指令不允许重命名域名，要重命名域名只能在Table之前或之后。

68. where命令，where的格式为where <eval-expression>，其eval-expression可以放置很多东西。可以是常规的字符串比较，可以是正则表达式等。搜索的结果只会显示让eval-expression为真的结果。需要注意的是在where中，如果不加单引号或者双引号，都会被看作是field name而不是字符串。

<http://docs.splunk.com/Documentation/Splunk/latest/SearchReference/Where?r=searchtip>

69. Splunk的match语句格式为match (SUBJECT, "REGEX”) ， 在match内容上，文档要求双引号，就必须双引号。单引号是无法生效的。

70. Splunk的Real-time 搜索主要用于防火墙或者病毒。Real-Time Search会在后台中建立一个永远进行的进程，不断的搜索数据，如果有险情则会报警。相比之下，一般的search则不会一直搜索，只搜索一次。

71. 当我们在dashboard上建立一个图表，图表后面其实是一个search. 这个search如果是实时的，那么他在后台一直运行，但是不是一有变化显示在dashboard panel上。他只是后台一直运行而已。如果想要dashboard panel上会隔一段时间变化，则需要设置auto refresh delay. （需要注意的是auto refresh delay是该网页或者该窗口一直打开的情况下，如果是我们手动再打开一个网页去观察，尽管没有设置auto refresh delay,但这相当于再一次请求网页，Splunk会运行一次sarch,我们还是可以看到新的数据，）

72. Splunk的图表split by我们可以从statistic tab中可以观察的到，它根据给定的域进行分割。对statistic tab 中的表格划分更多的行和列，从而进一步画图。所以分割前后，可以从statistic tab的表格中看得出。

73. chart允许我们split X行， split Y列。 而timechart其实是强制以时间split X行，可设置split Y列。

74. Splunk中的report是单个搜索形成的图表，表格，无法防止多个图表。Dashboard则是综合内容，可以放置多个图表，链接等内容。

75. Splunk 6.1，report可以embeded, embeded report的意思是将图表防止到非Splunk的环境中展示，例如生成iframe地址显示在其他外部的网页中。Embeded Report不具备Splunk Report中的所有特性，例如embeded report不支持实时搜索，不支持表格排序等。 但是dashboard暂时还无法embeded.

76. Splunk要求scheduled report不能带有time picker, 要求embeded report必须是scheduled的。当report scheduled to run的时候，可以发送邮件等动作去提醒用户。并且embeded report处将会更新。

77. Embeded Report在schedule to run以后，在运行的期间，embeded report将显示空白。所以要避免，频繁的运行耗时长的report， 这样会很长一段时间都是空白。hu

78. Splunk的forwarder 我们可以通过WEB GUI配置，可以通过CLI配置，可以通过文件直接编辑配置。但是最终这些都会写到Configuration File中。我们一般采用直接对Configuration 修改，因为这样的修改容易追踪。

不管在system还是在apps中：

Inputs.conf定义了要引入那些数据

Outputs.confi定义了forwarder将数据发送到哪里

Prof.conf定义了sourcetype的参数

79. 可以在Splunk中设置alert, alert是设置一个search，当search的条件满足时，将触发一个alert action. 该alert action可以是执行一段脚本或者发送邮件等等。

80. RPM-based Linux (RHEL, CentOS, Fedora, openSUSE). RPM本来的含义是Rehat Package Management, 后来发展成许多的linux都支持RPM来安装系统包。所以说RPM的linux时，现在RHEL, CentOS, Fedora, openSUSE都支持。

81.

在search bar中使用shift+enter进行换行。

82.splunk的rex命令，用于使用正则表达去提取原事件中的值，或者替换原事件中的值。

rex [field=<field>] ( <regex-expression> [max\_match=<int>] [offset\_field=<string>] ) | (mode=sed <sed-expression>)

83.

Splunk中的append命令，用于将分搜索的结果填充到一个搜索语句的结果上。从而可以达到在一个图中，有多条线条的效果，每个线条是一个搜索结果。

84.

在搜索的时候，如果不指定条件index是哪个index,那么Splunk默认会搜索所有默认index,例如main.

85. splunk的rest命令，用于在splunk中使用splunk的rest api节点然后返回数据作为搜索结果。

86.

Splunk的license如果在当天的0点到24点之间超过了可以index的数据量时，将会收到license violation warning. 这个此时数据还将被index，也可以继续搜索。但是如果在任意一个30天的周期内，收到5次或者更多时，将无法search,需要联系客服重新申请临时的license或者购买更大的license.

87.

Case函数，case (“X”, “Y”, …..), 执行该语句时，splunk从左到右进行审视每一个表达式X, 如果碰到第一个X为true,则返回对应Y值，如果一直无法匹配则返回Null.

88.

rex 命令，我们可以采用rex命令从event中提取字段作为额外的field, 也可以替换字段中的内容。

例如：

source="cisco\_esa.txt" | rex field=\_raw "From: <(?<from>.\*)> To: <(?<to>.\*)>"

这样将从中提出额外的field, from和to

89.

Transaction命令，我们使用transaction命令添加条件来来发现在原事件中哪一些应该如何被归类为一个个的transction（例如，maxspan每个事务最长时间跨度）。 同时，使用transaction以后还会被搜索提供两个额外的域，duration以及eventcount. Duration域将是事务的开始和结束时间， eventcount是该事务中所含的事件数。

Transaction语法：

transaction [<field-list>] [name=<transaction-name>] [<txn\_definition-options>...] [<memcontrol-options>...] [<rendering-options>...]

其中的field-list是指，将事件（每一个事件对应每一条记录）中含有这个域的相同的值作为同一个transaction.

90.

rex是Splunk的命令，用于利用正则表达从event中提取新的域。

match是Splunk的函数，用于判断某个域是否满足正则表达，输出true还是false.

where是Splunk的命令，用于和Splunk其他的函数合用，过滤出函数评定为true的值。

91.

Splunk搜索的时候，有时候可以看到在job在有红色地感叹号，代表这个搜索有错误。可以点击红色感叹号查看，错误可能包括但不限于，\*用于匹配标点，在搜索语句中检索不存在的域等原因。

92.

当Splunk去monitor某个数据源的时候：

如果该数据源是json格式，那么splunk会将其中地key都作为field,然后我们可以使用where match(field, 正则表达)来过滤符合要求地记录。

如果该数据源是文本格式，那么splunk会除了自动提取，host, sourcetype以外splunk不会自动提取其他地field.我们需要自己用rex来在文字中进行提取， 可能还会用 transaction等来将多条文字记录化为一组。 如果我们想过滤一些无用地行， 我们不能使用where match, 因为splunk没有提取到field. 我们可以直接 写”筛选内容来筛选符合内容的行。例如：source="job/main\_dev/\*/console" ( "Build duration in seconds" OR "Finished: SUCCESS" )， 将source中普通文字，这样的搜索会让搜索结果中只留下含有( "Build duration in seconds" 或者 "Finished: SUCCESS"地文本文字行。

93.

Stats Function分为四类

|  |  |
| --- | --- |
| Aggregate functions | avg() count() sum() range() min() max() … |
| Event order functions | earliest() first() last() latest() |
| Multivalue stats and chart function | list() values() |
| Time functions | per\_day() per\_hour() per\_minute() per\_second() |