实验 - 读取服务器日志

目标

第 1 部分：使用 Cat、More 和 Less 读取日志文件

第 2 部分：记录文件和系统日志

第 3 部分：日志文件和 Journalctl

1. 背景/场景

日志文件是用于排除故障和监控的重要工具。不同的应用生成不同的日志文件，每一个日志文件都包含自己的一组字段和信息。虽然日志文件之间的字段结构可能会有所更改，但用于读取它们的工具基本上是相同的。在本实验中，同学们将了解并练习使用读取日志文件的常用工具。

1. 所需资源

* CyberOps Workstation 虚拟机
* 互联网接入

1. 使用 Cat、More、Less 和 Tail 读取日志文件

日志文件是用于记录由应用、服务或操作系统本身触发的特定事件的文件。日志文件通常以纯文本形式存储，是排除故障不可或缺的资源。

* 1. 打开日志文件。

日志文件通常包含纯文本信息，可以通过能够处理文本的几乎任何程序（例如文本编辑器）进行查看。但是，由于便捷性、可用性和速度等原因，一些工具比其他工具更为常用。此部分重点介绍四个基于命令行的程序：**cat**、**more**、**less** 和 **tail**。

**Cat**，从“concatenate”（连接）一词派生，它是一款基于命令行的 UNIX 工具，用于读取并在屏幕上显示文件的内容。由于使用简单且可以打开一个文本文件，并在纯文本终端中显示，因此如今 **cat** 是使用较为广泛的程序。

* + 1. 启动 **CyberOps Worstation 虚拟机**并打开一个终端窗口。
    2. 在终端窗口中，发出以下命令以显示 **logstash-tutorial.log** 文件的内容，它位于 **/home/analyst/lab.support.files/** 文件夹中：

analyst@secOps ~$ **cat /home/analyst/lab.support.files/logstash-tutorial.log**

文件内容应在终端窗口中滚动，直到所有内容都显示出来。

将 **cat** 用于较大的文本文件存在哪些缺点？

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

用于可视化日志文件的另一个常用工具是 **more**。与 **cat** 类似，**more** 也是一款基于命令行的 UNIX 工具，它可以打开基于文本的文件并在屏幕上显示文件内容。**cat** 和 **more** 之间的主要区别在于，**more** 支持分页符，允许用户一次查看一个文件的内容。可以使用空格键来显示下一页。

* + 1. 在同一个终端窗口中，使用下面的命令再次显示 **logstash-tutorial.log** 文件的内容。此时，使用 **more**：

analyst@secOps ~$ **more /home/analyst/lab.support.files/logstash-tutorial.log**

文件内容应在终端窗口中滚动，直到显示一页。按空格键前进到下一页。按 Enter 键显示下一行文本。

使用 **more** 存在哪些缺点？

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Less** 工具在 **cat** 和 **more** 功能的基础上构建，允许按页显示文件的内容，同时也允许用户选择查看之前显示的页面。

* + 1. 在同一个终端窗口中, 使用 **less** 再次显示 **logstash-tutorial.log** 文件的内容：

analyst@secOps ~$ **less /home/analyst/lab.support.files/logstash-tutorial.log**

文件内容应在终端窗口中滚动，直到显示一页。按空格键前进到下一页。按 Enter 键显示下一行文本。使用向上和向下箭头键将文本文件进行前后滚动。

使用键盘上的“**q**”键退出 **less** 工具。

* + 1. **Tail** 命令显示文本文件的末尾。默认情况下，**tail** 显示文件中的最后十行。

使用 tail 显示 **/home/analyst/lab.support.files/logstash-tutorial.log** 文件的最后十行。

analyst@secOps ~$ **tail /home/analyst/lab.support.files/logstash-tutorial.log**

218.30.103.62 - - [04/Jan/2015:05:28:43 +0000] "GET /blog/geekery/xvfb-firefox.html HTTP/1.1" 200 10975 "-" "Sogou web spider/4.0(+http://www.sogou.com/docs/help/webmasters.htm#07)"

218.30.103.62 - - [04/Jan/2015:05:29:06 +0000] "GET /blog/geekery/puppet-facts-into-mcollective.html HTTP/1.1" 200 9872 "-" "Sogou web spider/4.0(+http://www.sogou.com/docs/help/webmasters.htm#07)"

198.46.149.143 - - [04/Jan/2015:05:29:13 +0000] "GET /blog/geekery/disabling-battery-in-ubuntu-vms.html?utm\_source=feedburner&utm\_medium=feed&utm\_campaign=Feed%3A+semicomplete%2Fmain+%28semicomplete.com+-+Jordan+Sissel%29 HTTP/1.1" 200 9316 "-" "Tiny Tiny RSS/1.11 (http://tt-rss.org/)"

198.46.149.143 - - [04/Jan/2015:05:29:13 +0000] "GET /blog/geekery/solving-good-or-bad-problems.html?utm\_source=feedburner&utm\_medium=feed&utm\_campaign=Feed%3A+semicomplete%2Fmain+%28semicomplete.com+-+Jordan+Sissel%29 HTTP/1.1" 200 10756 "-" "Tiny Tiny RSS/1.11 (http://tt-rss.org/)"

218.30.103.62 - - [04/Jan/2015:05:29:26 +0000] "GET /blog/geekery/jquery-interface-puffer.html%20target= HTTP/1.1" 200 202 "-" "Sogou web spider/4.0(+http://www.sogou.com/docs/help/webmasters.htm#07)"

218.30.103.62 - - [04/Jan/2015:05:29:48 +0000] "GET /blog/geekery/ec2-reserved-vs-ondemand.html HTTP/1.1" 200 11834 "-" "Sogou web spider/4.0(+http://www.sogou.com/docs/help/webmasters.htm#07)"

66.249.73.135 - - [04/Jan/2015:05:30:06 +0000] "GET /blog/web/firefox-scrolling-fix.html HTTP/1.1" 200 8956 "-" "Mozilla/5.0 (iPhone; CPU iPhone OS 6\_0 like Mac OS X) AppleWebKit/536.26 (KHTML, like Gecko) Version/6.0 Mobile/10A5376e Safari/8536.25 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"

86.1.76.62 - - [04/Jan/2015:05:30:37 +0000] "GET /projects/xdotool/ HTTP/1.1" 200 12292 "http://www.haskell.org/haskellwiki/Xmonad/Frequently\_asked\_questions" "Mozilla/5.0 (X11; Linux x86\_64; rv:24.0) Gecko/20140205 Firefox/24.0 Iceweasel/24.3.0"

86.1.76.62 - - [04/Jan/2015:05:30:37 +0000] "GET /reset.css HTTP/1.1" 200 1015 "http://www.semicomplete.com/projects/xdotool/" "Mozilla/5.0 (X11; Linux x86\_64; rv:24.0) Gecko/20140205 Firefox/24.0 Iceweasel/24.3.0"

86.1.76.62 - - [04/Jan/2015:05:30:37 +0000] "GET /style2.css HTTP/1.1" 200 4877 "http://www.semicomplete.com/projects/xdotool/" "Mozilla/5.0 (X11; Linux x86\_64; rv:24.0) Gecko/20140205 Firefox/24.0 Iceweasel/24.3.0"

* 1. 主动跟踪日志。

在某些情况下，向日志文件写入日志条目时，最好监控日志文件。对于这些情况而言，**tail -f** 命令非常有用。

* + 1. 使用 tail -f 主动监控 **/var/log/syslog** 文件的内容：

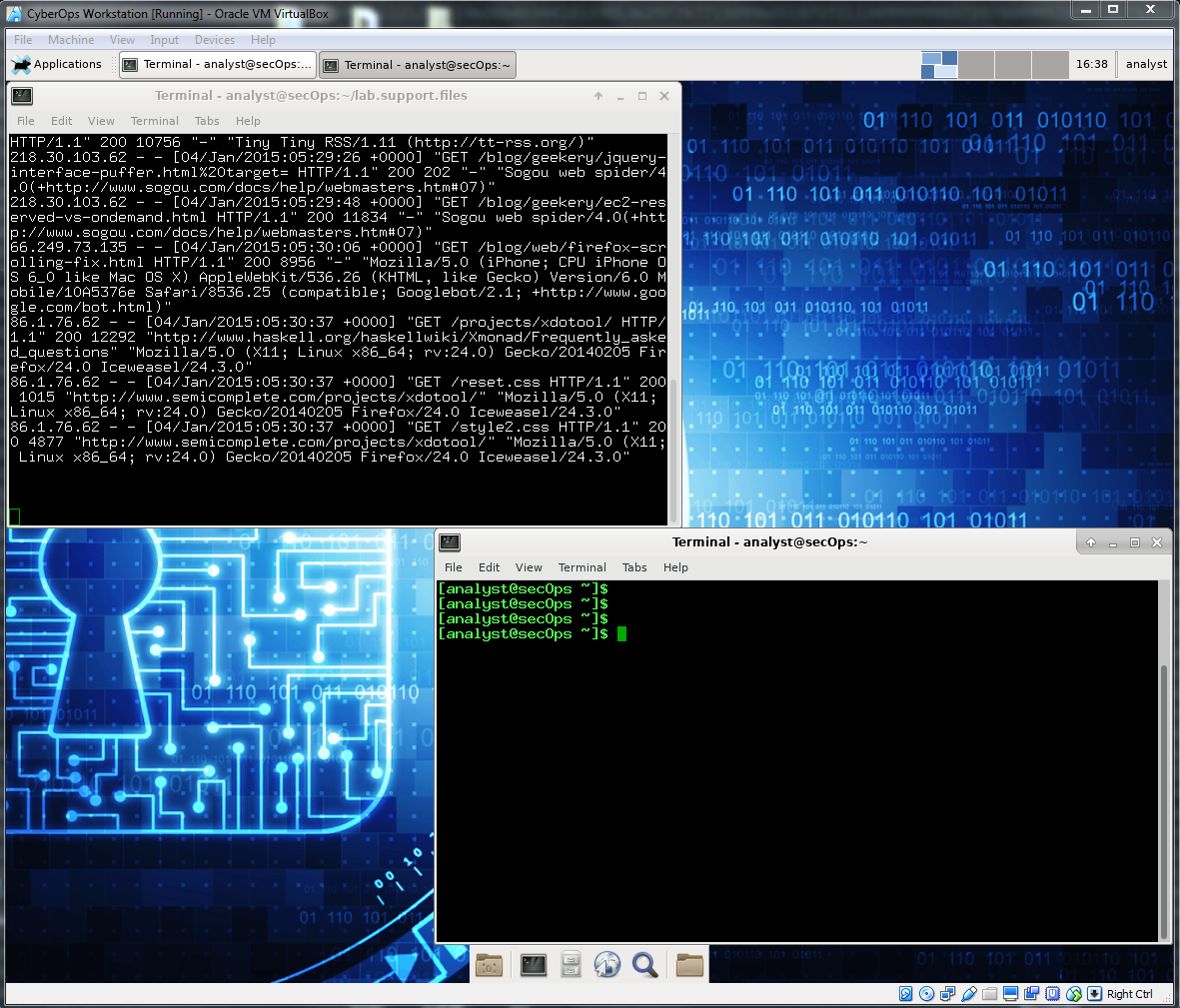
analyst@secOps ~$ **sudo tail –f /home/analyst/lab.support.files/logstash-tutorial.log**

**tail** 和 **tail -f** 的输出有什么区别？请说明原因。

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* + 1. 要在操作过程中观察 tail –f，请打开第二个终端窗口。安排同学们的显示器，以便同学们可以看到两个终端窗口。重新调整窗口大小，以便同学们可以同时看到它们，如下图所示：



顶部的终端窗口正在运行 **tail -f**，以监控 **/home/analyst/lab.support.files/logstash-tutorial.log** 文件。使用底部的终端窗口向所监控的文件添加信息。

为了使可视化更容易，请选择顶部的终端窗口（运行 **tail -f** 的窗口），并按 Enter 键几次。此操作将在文件的当前内容和新添加的信息之间添加几行。

* + 1. 选择底部的终端窗口并输入以下命令：

analyst@secOps ~$ **echo "this is a new entry to the monitored log file" >> lab.support.files/logstash-tutorial.log**

使用上面的命令可将“这是向所监控日志文件添加的新条目“消息附加到 **/home/analyst/lab.support.files/logstash-tutorial.log** 文件。由于 tail –f 此时正在监控该文件，因此会在文件中添加一行。顶部窗口应实时显示新添加的一行。

* + 1. 按 CTRL + C 停止执行 **tail -f** 并返回至 shell 提示符。
    2. 关闭两个终端窗口的其中一个。

1. 日志文件和系统日志

由于其重要性，通常的做法是将日志文件集中到一台监控计算机中。**系统日志**是旨在允许设备将其日志文件发送到集中式服务器（称为**日志记录**服务器）的系统。客户端使用**系统日志**协议与系统日志服务器进行通信。**系统日志**是常见的部署工具，并且支持几乎所有计算机平台。

CyberOps Worstation 虚拟机生成操作系统级别的日志文件，并将它们传递给**系统日志**。

* + 1. 使用 **cat** 命令作为 **root**，以列出 **/var/log/syslog** 文件的内容。此文件保留由 CyberOps Worstation 虚拟机操作系统生成并发送到**系统日志**服务的日志条目。

analyst@secOps ~$ **sudo cat /var/log/syslog**

[sudo] password for analyst:

Feb 7 13:23:15 secOps kernel: [ 5.458959] psmouse serio1: hgpk: ID: 10 00 64

Feb 7 13:23:15 secOps kernel: [ 5.467285] input: ImExPS/2 BYD TouchPad as /devices/platform/i8042/serio1/input/input6

Feb 7 13:23:15 secOps kernel: [ 5.502469] RAPL PMU: API unit is 2^-32 Joules, 4 fixed counters, 10737418240 ms ovfl timer

Feb 7 13:23:15 secOps kernel: [ 5.502476] RAPL PMU: hw unit of domain pp0-core 2^-0 Joules

Feb 7 13:23:15 secOps kernel: [ 5.502478] RAPL PMU: hw unit of domain package 2^-0 Joules

Feb 7 13:23:15 secOps kernel: [ 5.502479] RAPL PMU: hw unit of domain dram 2^-0 Joules

Feb 7 13:23:15 secOps kernel: [ 5.502480] RAPL PMU: hw unit of domain pp1-gpu 2^-0 Joules

Feb 7 13:23:15 secOps kernel: [ 5.672547] ppdev: user-space parallel port driver

Feb 7 13:23:15 secOps kernel: [ 5.709000] pcnet32 0000:00:03.0 enp0s3: renamed from eth0

Feb 7 13:23:16 secOps kernel: [ 6.166738] pcnet32 0000:00:03.0 enp0s3: link up, 100Mbps, full-duplex

Feb 7 13:23:16 secOps kernel: [ 6.706058] random: crng init done

Feb 7 13:23:18 secOps kernel: [ 8.318984] floppy0: no floppy controllers found

Feb 7 13:23:18 secOps kernel: [ 8.319028] work still pending

Feb 7 14:26:35 secOps kernel: [ 3806.118242] hrtimer: interrupt took 4085149 ns

Feb 7 15:02:13 secOps kernel: [ 5943.582952] pcnet32 0000:00:03.0 enp0s3: link down

Feb 7 15:02:19 secOps kernel: [ 5949.556153] pcnet32 0000:00:03.0 enp0s3: link up, 100Mbps, full-duplex

为什么 **cat** 命令必须作为 **root** 执行？

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* + 1. 请注意，**/var/log/syslog** 文件只存储最新的日志条目。为了将系统日志文件保持在较小的水平，操作系统会定期滚动更新日志文件，将较早的日志文件重命名为 **syslog.1**、**syslog.2**，以此类推。

使用 **cat** 命令列出较早的**系统日志**文件：

analyst@secOps ~$ **sudo cat /var/log/syslog.2**

analyst@secOps ~$ **sudo cat /var/log/syslog.3**

analyst@secOps ~$ **sudo cat /var/log/syslog.4**

同学们能想到为什么保持计算机的时间和日期的正确同步如此重要吗？

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. 日志文件和 Journalctl

另一个常用的日志管理系统称为 **journal**。该系统由 **journald** 守护程序管理，旨在集中管理日志，而不管消息来自何处。在本实验的背景下，**journal** 系统守护程序最明显的功能是使用仅能进行追加操作的二进制文件作为其**日志文件**。

* 1. 运行 journalct 且不使用任何选项。
     1. 要查看 **journald** 日志，请使用 **journalctl** 命令。**journalctl** 工具解释并显示之前存储在 **journal** 二进制日志文件中的日志条目。

analyst@secOps ~$ **journalctl**

-- Logs begin at Fri 2014-09-26 14:13:12 EDT, end at Tue 2017-02-07 13:23:29 ES

Sep 26 14:13:12 dataAnalyzer systemd[1087]: Starting Paths.

Sep 26 14:13:12 dataAnalyzer systemd[1087]: Reached target Paths.

Sep 26 14:13:12 dataAnalyzer systemd[1087]: Starting Timers.

Sep 26 14:13:12 dataAnalyzer systemd[1087]: Reached target Timers.

Sep 26 14:13:12 dataAnalyzer systemd[1087]: Starting Sockets.

Sep 26 14:13:12 dataAnalyzer systemd[1087]: Reached target Sockets.

Sep 26 14:13:12 dataAnalyzer systemd[1087]: Starting Basic System.

Sep 26 14:13:12 dataAnalyzer systemd[1087]: Reached target Basic System.

Sep 26 14:13:12 dataAnalyzer systemd[1087]: Starting Default.

Sep 26 14:13:12 dataAnalyzer systemd[1087]: Reached target Default.

Sep 26 14:13:12 dataAnalyzer systemd[1087]: Startup finished in 18ms.

Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopping Default.

Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopped target Default.

Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopping Basic System.

Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopped target Basic System.

Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopping Paths.

Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopped target Paths.

Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopping Timers.

Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopped target Timers.

Sep 26 14:14:24 dataAnalyzer systemd[1087]: Stopping Sockets.

<output omitted>

**注**：以 root 身份运行 journalctl 将显示更详细的信息。

* + 1. 使用 CTRL + C 退出显示。
  1. Journalctl 和一些选项。

使用 **journalctl** 的部分功能依赖于其选项。

* + 1. 使用 **journalctl-utc** 可显示 UTC 时间内的所有时戳：

analyst@secOps ~$ **sudo journalctl –utc**

* + 1. 使用 **journalctl -b** 显示上次启动时记录的日志条目：

analyst@secOps ~$ **sudo journalctl –b**

Feb 07 08:23:13 secOps systemd-journald[172]: Time spent on flushing to /var is

Feb 07 08:23:13 secOps kernel: Linux version 4.8.12-2-ARCH (builduser@andyrtr)

Feb 07 08:23:13 secOps kernel: x86/fpu: Supporting XSAVE feature 0x001: 'x87 fl

Feb 07 08:23:13 secOps kernel: x86/fpu: Supporting XSAVE feature 0x002: 'SSE re

Feb 07 08:23:13 secOps kernel: x86/fpu: Supporting XSAVE feature 0x004: 'AVX re

Feb 07 08:23:13 secOps kernel: x86/fpu: xstate\_offset[2]: 576, xstate\_sizes[2]

Feb 07 08:23:13 secOps kernel: x86/fpu: Enabled xstate features 0x7, context si

Feb 07 08:23:13 secOps kernel: x86/fpu: Using 'eager' FPU context switches.

Feb 07 08:23:13 secOps kernel: e820: BIOS-provided physical RAM map:

<output omitted>

* + 1. 使用 **journalctl** 指定日志条目的服务和时间范围。下面的命令显示了今天记录的所有 **nginx** 服务日志：

analyst@secOps ~$ **sudo journalctl -u nginx.service --since today**

* + 1. 使用 **-k** 选项只显示内核生成的消息：

analyst@secOps ~$ **sudo journalctl –k**

* + 1. 与上面描述的 **tail -f** 类似，使用 **-f** 选项可在写入日志时主动跟踪这些日志：

analyst@secOps ~$ **sudo journalctl –f**

1. 思考

比较系统日志和 Journald。二者各有何优缺点？

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_