Getting Started With Log Management

JOHN VESTER
LEAD SOFTWARE ENGINEER, MARQETA

CONTENTS

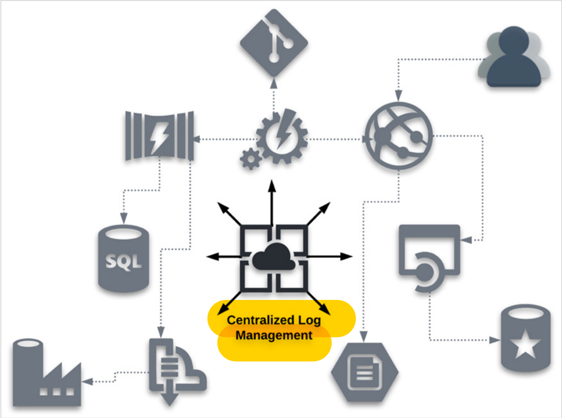
THE CURRENT STATE OF LOG MANAGEMENT

The landscape for developing applications has continued to evolve as technology matures into providing dedicated segments focused on handling aspects of the required processing. Aspects like microservices, JavaScript-based client frameworks, containerized strategies, cloud adoption, and configuration as code (or "* as code") have presented a new reality for the origin of log events and messages.

Each of these aspects can log events as they participate in some form of application service delivery. Without considering a centralized log management solution, those who rely on these very logs to support and maintain applications find themselves at a disadvantage — a disadvantage that can impact the bottom line of those components they support. Unfortunately, the format and structure of these logs vary from one system to the next.

This is where the concept of centralized log management can provide a great deal of assistance. In the diagram below, a centralized log management solution can ingest the log events from all components of an application to provide a single source to review and analyze.

Figure 1: Centralized log management workflow example



When the logs are combined and organized properly, the engineer assigned to the situation can easily walk through the event without having to toggle between log files contained within disparate and proprietary systems. In the last few years, new challenges have emerged as priorities within log management:

- The need to analyze and debug faster
- The requirement of ELK Stack integration
- The ability to adopt appropriate storage tiers to reduce periodic costs

Preferred log management solutions should account for all these needs, either directly or via some form of integration. Additionally, forward-thinking solutions will consider next-gen features such as:

- Recurring pattern identification
- Machine learning and crowdsourcing support
- Anomaly detection
- Data visualization

logz.io

Make log collection & analytics easier, faster, and more cost effective at scale.

- ✓ Search and analyze logs faster to debug problems sooner
- ✓ Centralize your logs on a high-powered ELK Stack
- ✓ Distribute log data across storage tiers to reduce costs
- ✓ Surface exceptions and errors automatically
- ✓ Full visibility into your environment in one place
- ✓ Stay notified of significant events with real-time alerting

save time and money

Focus on Data That Matters

Smart log management powered by ELK

Learn More

WHO USES CENTRALIZED LOG MANAGEMENT?

DEVOPS

When an unexpected situation emerges, a DevOps engineer often has a collection of information to analyze. Source logs from all the components in the application landscape are buried deep within a file system, most likely using proprietary logging formats. Even when all the logs can be gathered, stepping through each log chronologically to determine the cause of the issue is a tedious task, at best.

By contrast, if a centralized log management system is in place, the logs are not only consolidated, but also standardized and organized in a manner that will allow the DevOps engineer to play back the scenario that is under investigation. In doing so, the wider view provides a far greater opportunity to identify the root cause.

In most cases, justification for centralized log management is met by this benefit to DevOps staff. However, three other areas can become beneficiaries of a centralized log management solution: security, compliance, IT operations, and software engineering.

SECURITY

Security teams can benefit from a centralized log management solution when scanning for unauthorized access to a given application or service. The range of this benefit can include both anonymous external entities as well as internal accounts. Reports in the solution can be created within the centralized log management system to match logged events that may be indicators of suspect activity. Consider the following errors being logged by an API under management by the solution:

```
2022-02-14 03:07:15.824 ERROR 36209 --- [main]
AccessServiceImpl   : User id (someUserId) does not
have access to perform this operation

2022-02-14 05:27:07.212 ERROR 36209 --- [main]
OrderServiceImpl   : User id (null) does not have
access to review order (1001001)

2022-02-14 10:17:37.542 ERROR 36209 --- [main]
AccessServiceImpl   : Attempt to access API without
a proper token on IP 127.0.0.1
```

Once enriched and ingested into the centralized log management solution, the report can be set up to keep track of error type (36209), showing the date/time information, plus the message that was logged.

COMPLIANCE

As a part of the periodic compliance/audit tasks, a centralized log management solution can assist compliance efforts in making sure t he application or its users are following the expectations that have been established.

For applications required to comply to a regulatory guideline (SOx, FDA, HIPAA, etc.), employment of a centralized logging solution can provide a one-stop source for analysis and certification.

IT OPERATIONS

Monitoring and understanding the complexities of an IT infrastructure can become less intrusive when a centralized log management solution is adopted. IT operations staff can use the tool to gain an understanding as to how systems interact with each other — thus providing a tool to help make decisions when routine tasks (like system maintenance outages) are scheduled.

SOFTWARE ENGINEERING

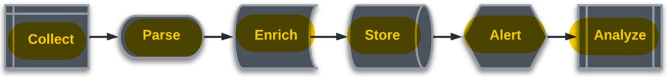
Software engineers who focus on building features, services, or integrations can benefit from centralized log management, regardless of the frameworks being employed. This is because the logs events needed for analysis and debugging during the development phase are sourced by frameworks, services, and containers called from proprietary services.

In a non-centralized model, the time required to locate and analyze independent logs quickly impacts the engineer's time to focus on delivering functionality planned for their current iteration. This time can lead to delivery delays, impacting the features and functionality requested by the business sponsor.

LOG MANAGEMENT: THE BASIC PROCESS AND TECHNIQUES

A centralized log management solution utilizes a flow like what is noted below:

Figure 2: Example log management workflow



COLLECT

Collect is the process of establishing a connection to a source system and ingesting the logs as they are natively created. A determination can be made regarding the log levels that are routed to the centralized log management solution, if necessary.

PARSE

Parse provides the ability to transform source log messages into a format that is standardized within the log management solution. This is an important aspect since logs produced by an API using an "extended" Apache format will be different than a log from database server (as an example). Correctly parsed, the following log events:

```
1550149377 INFO Userid (someUserId) successfully
logged in from IP 127.0.0.1
02/14/2022 03:07:15.824 ERROR 36209 --- [main]
AccessServiceImpl   : User id (someUserId) does not
have access to perform this operation
```

Could be updated to appear as shown below:

```
2022-02-14 08:02:57.000 (GMT) INFO Userid
(someUserId) successfully logged in from IP
127.0.0.1
2022-02-14 08:07:15.824 (GMT) ERROR 36209 -
AccessServiceImpl - User id (someUserId) does not
have access to perform this operation
```

The resulting parsed messages provide a standardized appearance for all messages, allowing the analyst to process the results of the logs more efficiently.

ENRICH

Enrich introduces the ability to further define the log event. As an example, enrichment functionality could perform the necessary logic to analyze a logged IP address to make it easier for the analyst to understand the system or service that is being referenced. Application or service-specific constants can be transcribed here as well to limit the need to cross-reference logged information.

The following event:

```
1550149377 ERROR 90215 - ServiceProviderCallOut
- Could not access service on 127.0.1.1 due to an
internal error code 10017.
```

Could be enriched as shown below:

```
2022-02-14 08:02:57.000 (GMT) - ERROR 90215
- SeviceProviderCallOut - Could not access
DocumentGenerationProcessor 4C (127.0.1.1) due to
an internal Socket Failure (error code 10017).
```

With the above enrichment, the timestamp was converted to a GMT date, the IP address was translated to include the host system, and the error code was looked up to return additional information.

STORE

Store persists the collected, parsed, and enriched logs into a data store utilized by the centralized log management solution. At this point, indexes and filters can be leveraged to provide greater insight into the native logs in the source system.

ALERT

With the necessary information stored in the centralized log management solution, alerts can be configured to catch events before they escalate to a higher severity level. At this point, centralized log management solutions can often send events to other systems to provide instant notification and escalation. Advanced log management solutions should take things a step further:

- **Real-time alerts** allow engineers the ability to be at the front end of the situation, receiving notification whenever an alert surfaces.
- **Live tail log monitoring** allows support teams to monitor the logs as they are written from the source system, avoiding the need to look at historical logs.

ANALYZE

Using an interface into the centralized log management solution, an analyst can search, filter, and review all the events related to a given situation without the need to review logs directly from the source system.

HOW TO GET STARTED WITH LOG MANAGEMENT

WHAT TO LOG (AND WHAT NOT TO LOG)

The time required to process and analyze logged information is important, especially when there is an urgent need to resolve an unexpected situation. As a result, a period of exploration should be taken to determine what information will be logged and what information will not be logged. In the example noted in the introduction, the DevOps engineer needed to review the consolidated events for an application that encountered an incident. In most cases, the situation needs to be resolved as soon as possible.

This effort could easily include logs from the microservices, databases, client application frameworks, and the security layer. If the logs included aspects that were not pertinent for analysis, additional time will be required to review and discard this type of information.

Consider the following example of logs that are ingested from the authentication/authorization service participating in the centralized log management strategy:

```
1550149377 INFO Userid (someUserId) successfully
logged in from IP 127.0.0.1
1550149382 INFO SSID for someUserId updated to
reflect last login
1550149385 WARN Password for someUserId will expire
in 26 hours
1550149415 INFO UserId (someUserId) granted access
to SomeApplication via token #tokenGoesHere
```

While the information being logged is important information, it might be best to filter out all the events, except for the following message:

```
1550149377 INFO Userid (someUserId) successfully
logged in from IP 127.0.0.1
```

In doing so, the number of log messages that would need to be reviewed during a crisis is minimized — especially in cases where hundreds (or thousands) of users are accessing the application.

SENSITIVE INFORMATION

Another aspect to consider is making sure that secretive information (access tokens, database connection strings, encryption keys, account information, user information, etc.) is not stored in the centralized log management solution. In the log example above, the `#tokenGoesHere` log message should be suppressed from ingestion into the centralized log management solution since that token could be considered sensitive information.

If the event is required, the message should be enriched to only ingest the following information:

```
1550149415 INFO UserId (someUserId) granted access
to SomeApplication
```

ESTABLISH GUIDELINES

The key is to establish guidelines that meet the needs of the entire user community that will utilize the solution. Think of this no differently than how any other application is architected — understanding the limits that are introduced by both not enough log events and too many log events. Once established, this information should be shared with teams who can create the events that are being captured by the centralized log management solution.

CENTRALIZED LOG MANAGEMENT CHECKLIST

When the decision is made to evaluate centralized log management solutions, the number of available offerings will certainly appear daunting. As a result, it is crucial to understand which features and functionality are important for your implementation.

Below are some high-level aspects to consider:

- □ **Zero to 60** – What is involved in getting started? How quickly can a new implementation span from setup and configuration to log analysis and debugging success?

  While the time required to get up to full speed is not an exclusive metric, there is some merit in knowing if the product under review gives the ability to get started quickly without a great deal of setup and configuration.

- □ **Ease of data exploration** – Can all types of users easily operate the system and locate data?

  Remember, the user is often more of a data explorer and not a data scientist. Any built-in reports or filters will lead to a better user experience when extracting results from the system.

- □ **Analyst efficiency** – How quickly does the system respond, including the ability to create complex searches or filters? Once data is returned, how easy are the results to comprehend and utilize?

  As noted above, time is often the driving component when trying to retrieve information from a centralized log management solution. Again, filters and reporting can help improve end-user efficiency.

- □ **Scalability** – How well does the solution work within your organization? Can all systems function within one centralized log management solution or would multiple instances be required? How about five years from now?

  It is important to understand how the centralized log management solution scales as more systems are introduced to the technology. While performance degradation is expected the larger the log data storage pool becomes, understanding the anticipated response time is a valuable metric to use during side-by-side comparisons.

- □ **Storage** – What are the storage expectations, where does the storage live, and what are the associated costs of a target implementation?

  It is also important to understand any boundaries around storage, especially with respect to system performance. Ideal log management systems should include the ability to leverage tiered storage for data — which is maintained for historical purposes over analytical purposes — at a lower price point.

- □ **Log completeness** – Does the information retained include everything necessary, or is extra effort required to retrieve data from an additional source?

  If you find yourself having to retrieve data from outside the centralized log management solution, there might be a gap in functionality with respect to your requirements.

- □ **Data enrichment functionality** – Does enrichment functionality exist? If so, how easy is it to utilize and maintain?

  It might be a good idea to review current log sources and understand more of the edge cases that could require data enrichment exceeding typical enrichment usage patterns.

- □ **Open/closed source** – Does the solution utilize an open-source approach? How does this approach line up with other solutions your entity employs?

- □ **Log collectors** – How are the log collectors defined? Are they proprietary or have they been created by third parties/vendors? Can they be centrally managed by the solution, or do they have to be independently managed at the log source level?

  Typically, proprietary developed connectors lag those created by the third party/vendor themself. If the connectors can be managed and configured by the centralized log management solution, there is less need for configuration to be maintained on the log source.

□ **Configuration as Code** – Does the solution employ a "* as code" approach, allowing the configuration of the centralized log management solution to be stored in a code repository?

> If your organization is embracing the "* as code" concept, centralized log management solutions that adopt this philosophy will be able to build and configure instances programmatically.

□ **Class-specific functionality** – Does the solution contain features that help a particular class of user (DevOps, security, compliance, ITOps) obtain common results? Do reports exist to locate regulatory exceptions like HIPAA, for example?

> Having functionality built in to assist specific use cases will lessen the time required for such groups to get up to speed and recognize value in the centralized log management solution.

□ **API functionality** – Is there a public API available for the solution?

> Gaining familiarity of any underlying application program interface (API) for the centralized log management solution could further justify or leverage the value returned from implementation.

□ **Anticipated costs** – How is the product licensed?

> Understanding the cost model will allow for product comparisons as time progresses and more components embrace centralized log management within your organization.

□ **CLM vs. SEIM** – Is the target solution a centralized log management (CLM) solution, or is it a security information and event management (SEIM) product? Do your current needs require one solution over the other… or perhaps both solutions?

> The requirements approved for the product will be a guide to understand what type of solution should be considered. It is important, however, to understand the differences between CLM and SEIM.

□ **ELK Stack** – Does the underlying solution leverage the ELK Stack? If not, what integrations exist to leverage these tools to improve the effectiveness and observability of the centralized logging solution?

> The ELK Stack is an ideal collection that works well with centralized log management solutions. It facilitates storing large amounts of data across multiple systems by leveraging products that scale naturally to align with logging needs. A purpose-driven search engine is coupled with a set of tools designed to analyze and observe the centralized log data.
>
> In every case, it is important to consider what steps are required to leverage the ELK Stack and any associated costs during product evaluations

NEXT-GEN FEATURES FOR LOG MANAGEMENT

While the centralized log management checklist section provided features and functionality that should be expected in an acceptable solution, below are a collection of next-generation (next-gen) features for products that offer leading-edge functionality. These features are intended to leverage technology in order to enrich the log management experience.

RECURRING PATTERN IDENTIFICATION

Having an option to quickly see recurring patterns across all log data allows analysts to isolate issues and unexpected behavior. Implementing the ability to present a patterns perspective will alter the analyst's view of a series of endless log entries to a categorized table, showing something like what is displayed below:

```
Count    Ratio   Pattern
-----    ------  ---------------------------
52,701   22.17%  License key has expired
19,457    7.99%  Invalid User ID
18,295    7.25%  New customer account created
```

In the example above, a large percentage of the messages match the pattern where a license key had expired, which may not be easy to locate in a large volume of individual log entries.

MACHINE LEARNING AND CROWDSOURCING

Analysts often find themselves looking for the root cause of an unexpected issue — which can resemble a needle in a haystack. Rather than spending hours trying to narrow down the endless sea of log entries, the concept of machine learning and crowdsourcing support can minimize the time required to identify the root cause. Next-generation solutions that employ machine learning can help present key terms found in the centralized log events, along with the number of occurrences for each term. This enables the analyst to quickly reduce the number of log entries to process, thus making the haystack much smaller.

With a smaller collection of logs to process, these same market leaders provide additional information from sources outside the centralized log management service. For example, opening a given log entry provides the expected information related to the captured log data, but it goes a step further by linking crowdsourced data like:

- Discussion threads related to the logged message or condition
- Blog entries focused on how to correct the situation
- Documentation for the method, function, or class being utilized
- Additional information related to the error itself

ANOMALY DETECTION

Issues that do not appear in the logs often, but that are extremely valuable in nature, are often referred to as anomalies. Next-generation solutions should provide the ability to detect anomalies so they can be surfaced and addressed. Anomaly detection leverages machine learning patterns to automatically isolate and investigate emergent problems based upon unusual behavior.

Once a baseline collection of fields has been specified and a query is established, the centralized log management service builds a model and begins to analyze and capture unusual behavior and events. Those detected events can be surfaced as new alerts in real time to provide instant access to emerging issues.

DATA VISUALIZATION

Visualizations enables analysts to view data in graphical format, which can allow them to better understand comparisons over time. By defining common elements for the X and Y axis, data in the next-generation centralized log management solution can be leveraged to communicate the state of the events being captured. These visualizations can be combined into a dashboard to communicate the overall status of the environment being monitored by the centralized log management service.

CONCLUSION

The value of a centralized log management solution can be justified by several groups within an entity's IT division. With proper guidelines and a successful implementation, the benefits from utilizing centralized log management can result in the ability to identify an event's root cause and to receive notification prior to issues escalating.

Implementation of a centralized log management solution should be treated no differently than any other tool being introduced into an organization. Requirements and understanding should drive what components are logged with the **right** level of logging ingested into the solution. Any sensitive information should be excluded from ingestion into the centralized log management solution, always adhering to any regulatory guidelines.

A major success factor for a centralized log management solution is the ability for an analyst to find the information needed, taking as little time as possible without having to span outside the solution itself. Centralized log management solutions that include pre-designed reports and functionality to assist all classes of users can provide a faster turnaround with performing routine requests.

Time should be taken to determine if a centralized log management solution is the right fit for your needs, or if your requirements fall into a security information and event management solution set. While some solutions may operate well as either, understanding your target end state is helpful in identifying the best solution for your corporation.

Systems that leverage the ELK Stack and machine learning approaches naturally offer a competitive advantage over those products that do not. Having next-generation features available will not only equip analysts with a complete set of tooling to leverage, but also minimize the potential downtime from an unexpected situation.

WRITTEN BY JOHN VESTER
LEAD SOFTWARE ENGINEER, MARQETA
FREELANCE WRITER | DZONE CORE MEMBER

Information Technology professional with 30+ years expertise in application design and architecture, feature development, project management, and team management. Currently focusing on enterprise architecture/application design utilizing object-oriented programming languages and frameworks.

Prior expertise in building (Spring Boot) Java-based APIs against React and Angular client frameworks. CRM design, customization, and integration with Salesforce. Additional experience using both C# (.NET Framework) and J2EE (including Spring MVC, JBoss Seam, Struts Tiles, JBoss Hibernate, Spring JDBC).

DZone
600 Park Offices Drive, Suite 300
Research Triangle Park, NC 27709
888.678.0399 | 919.678.0300

At DZone, we foster a collaborative environment that empowers developers and tech professionals to share knowledge, build skills, and solve problems through content, code, and community. We thoughtfully — and with intention — challenge the status quo and value diverse perspectives so that, as one, we can inspire positive change through technology.