How to use Parser & Strategies & Impl

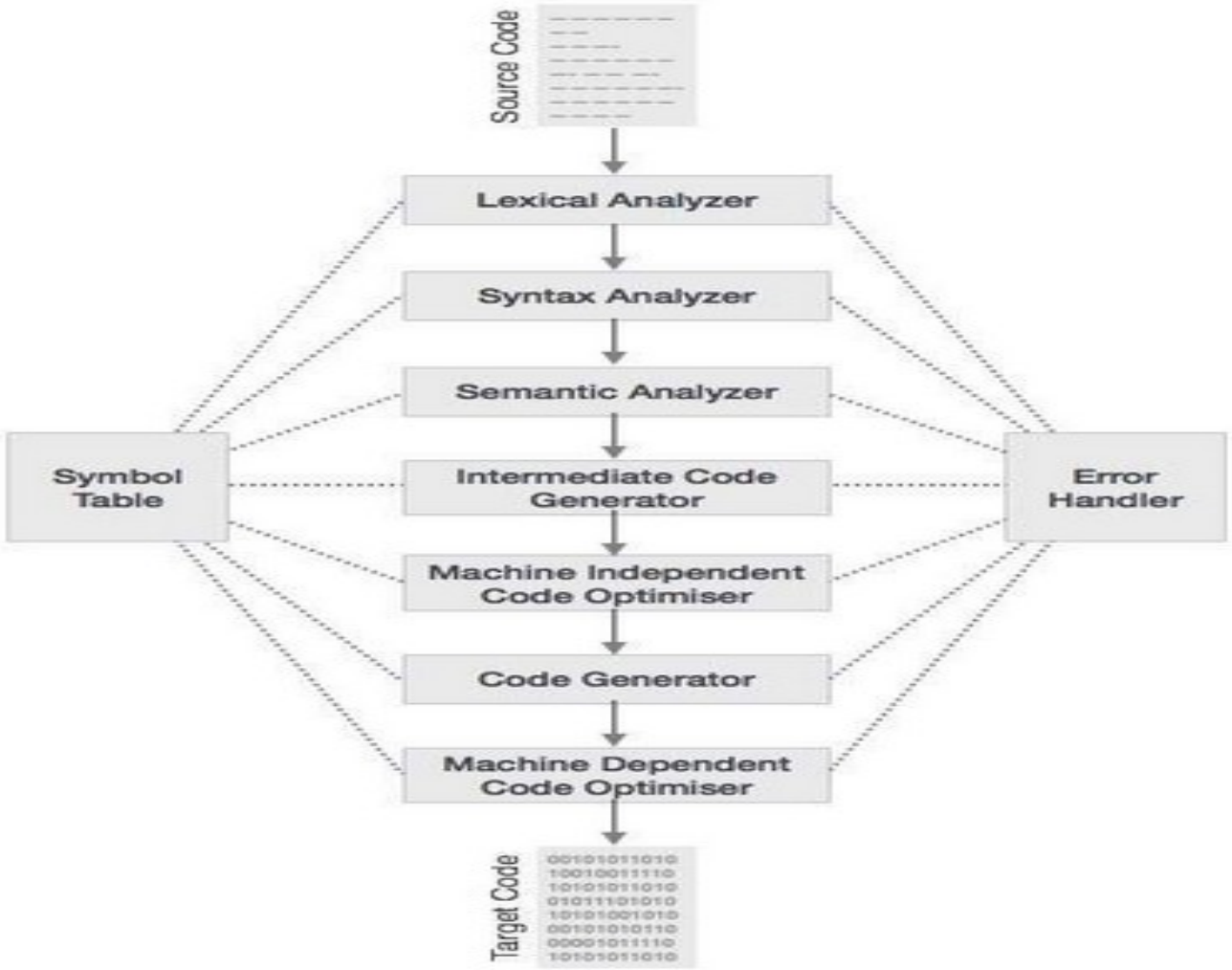
By - Mahesh Pavaskar

My Profile

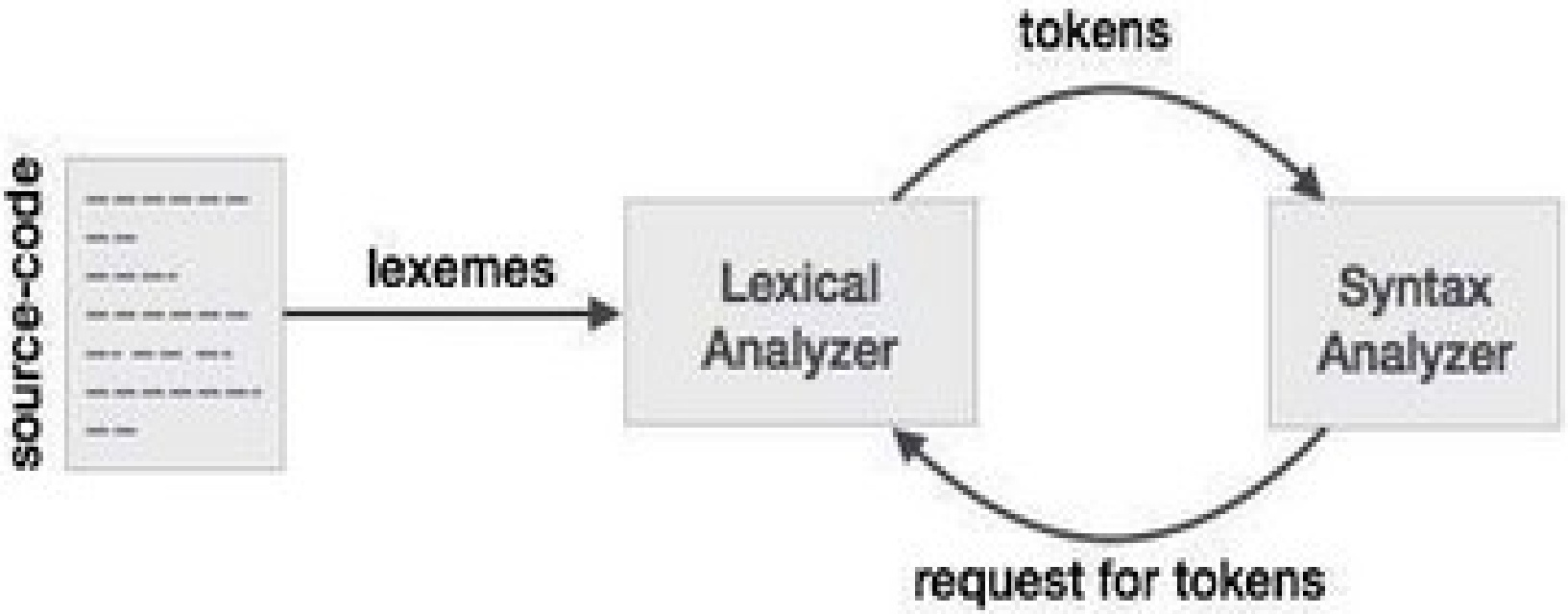
- Mahesh Pavaskar | SIEM Solution Engineer | 8.5 yrs Exp
- 4 years industry experience (1 year as System Admin + remaining - SIEM QRadar
- solution engineer)
- 4.5 years Teaching experience in Computer Science Subjects & Networking.
- Email ID : pavaskarmahesh@gmail.com
- Mobile Number : +91 8087114770
- Designation and role : Senior Software Engineer, IBM Practice
- LinkedIn Profile : https://www.linkedin.com/in/mahesh-pavaskar-88068263/

- Strong knowledge in Java, Linux, Network and SIEM domain.
- Worked as CCNA instructor trained by CISCO.
- Certification – 11 (RHCSA, RHCE, CCNA- ICND 1-4 ,MTA,MCP, DB2, RAD, Cyber Security)
- M.Tech (Computer Engineering-NIMS),VJTI University of Mumbai, 2013 | 9.3 CGPI
- Total Workshops = 34 (Still counting)
- Total Industrial training = 3

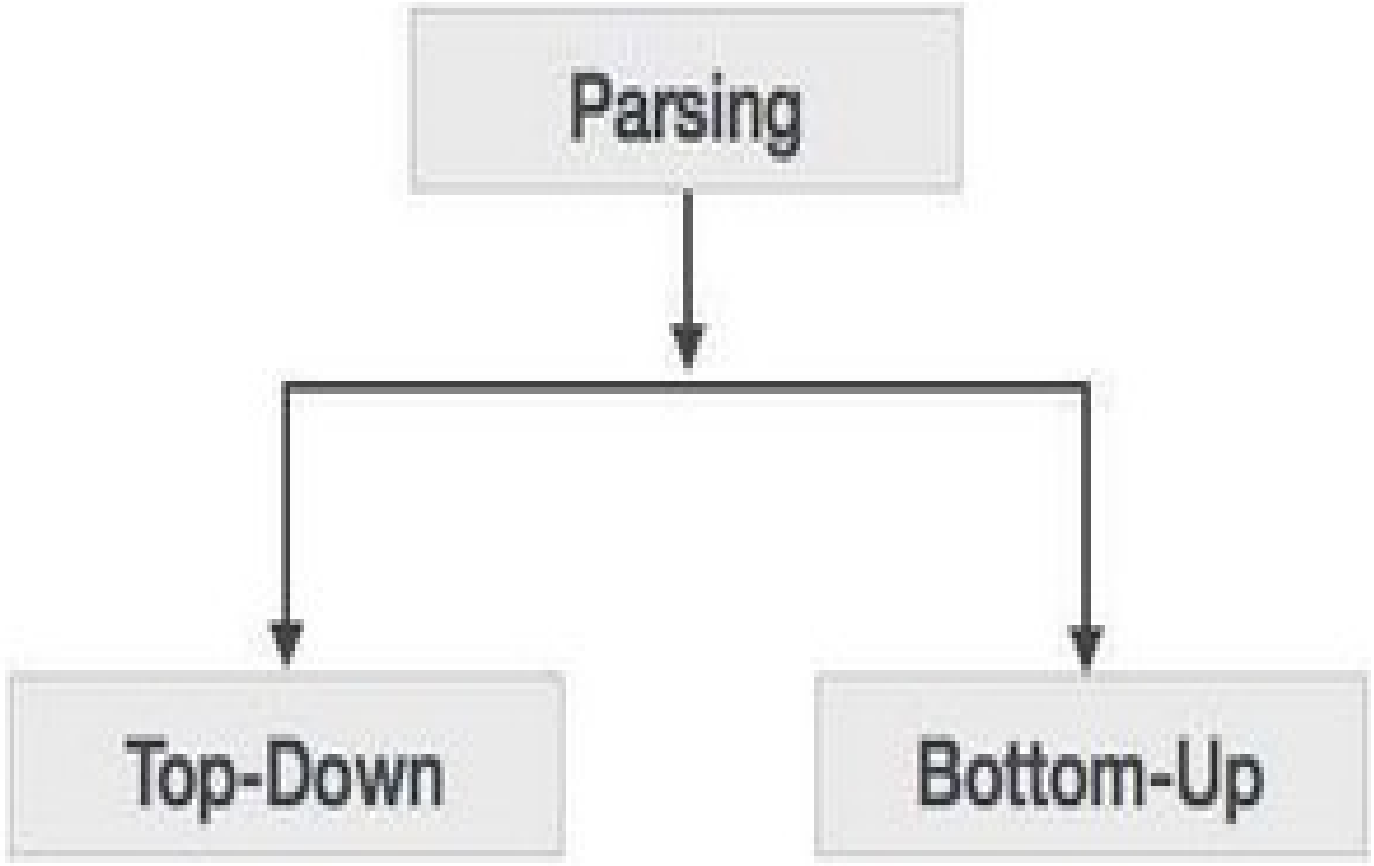
Compiler Phases



Compiler Design - Lexical Analysis



Parsing Types



Parser

- Developer would have to develop their own custom parsers in different languages like Java Programming, C++, etc.
- The reason behind developing custom parsers can be performance issue, complexity, flaws in parsing, not matching the requirement, etc.

Parsing & Parser

- Parsing can be defined as the process of breaking down a **block of data into smaller pieces** based on some **pre-defined set of rules**. And then interpret, modify or manage the small pieces of data as per the requirement.
- Parser is a software program that is used to break the data into smaller chunks.
- A parser can be written in any languages based on the requirement.

Types of parsers in Java

- Parsers can be categorized in different ways.
- A) Sequential
- B) Random
- In a sequential parser, only the current parsed data is accessible. It cannot go back or forward.

- In a random parser, parsed data can be accessed randomly, so moving back and forth is possible.

- SAX and StAX parsers are examples of sequential parser.
- XML DOM is an example of a random parser.

- In a different way, parsers can be classified as text parser or XML parser. A text parser parses textual data whereas XML parser parses XML/JSON data.

DOM vs SAX

- DOM (Document Object Model) defines an interface which can be used to manipulate XML documents. XML parsers are written by implementing this interface. DOM parsers are random parsers which are suitable when:
  1) Information about the structure of the document is important
  2) You need to move back and forth within the structure

- DOM parser provides several Java interfaces and methods to work with the XML data. It returns a tree structure of all the elements in a XML document. And the tree can be traversed to work with the data.

- SAX (Simple API for XML) is a sequential event-based parser. It parses the XML data in a sequential manner, starting from the root till the end. It does not form a tree structure to parse; rather it sends an event notification while parsing elements. SAX is suitable when

1) Linear and sequential processing is required
2) The XML document is too large
3) Complex nesting in XML is not there
4) Part of the XML document needs to be manipulated

Parser best practices

- DOM parser is best fit when the numbers of elements are under 1000 and you have a requirement of adding/deleting elements. But as DOM creates a tree structure before it starts processing, performance is an important parameter. So, for partial manipulation of an XML document, DOM is not recommended.
- SAX is best fit for large XML files with linear structure and unique elements. It is light weight and suitable for shallow xml document parsing. As it does not make any tree structure, the performance is better than DOM parser.

Parser best practices

When to Use SAX

- You want to process the XML document in a sequential manner from top to bottom.
- SAX requires much less memory than DOM because SAX does not create an in-memory tree of the XML data, as a DOM does. So if you want to process a very large XML document whose DOM tree would consume too much memory, you can shoose SAX over DOM.
- If the XML document is not deeply nested.
- SAX is fast and efficient and it is useful for state-independent filtering. SAX parser calls a method whenever an element tag is encountered and calls a different method when text or character is found.

JAXB

- Java Architecture for XML Binding
- JAXB tutorial provides concepts and API to convert object into XML and XML into object.
- It provides mechanism to marshal (write) java objects into XML and unmarshal (read) XML into object
- Program

JSON parsing in Java

- Program

XML vs JSON

http://www.devinline.com/2015/08/xml-vs-ison-tug-of-war.html

λ

JSON vs XML

- JSON is lightweight thus simple to read and write.
- JSON supports array data structure.
- JSON files are more human readable.
- JSON has no display capabilities .
- Provides scalar data types and the ability to express structured data through arrays and objects.
- Native object support.

- XML is less simple than JSON.
- XML doesn't support array data structure.
- XML files are less human readable.
- XML provides the capability to display data because it is a markup language.
- Does not provide any notion of data types. One must rely on XML Schema for adding type information.
- Objects have to be expressed by conventions, often through a mixed use of attributes and elements.

Similarities between JSON and XML

Both are simple and open.
Both supports unicode. So internationalization is supported by JSON and XML both.
Both represents self describing data.
Both are interoperable or language-independent.

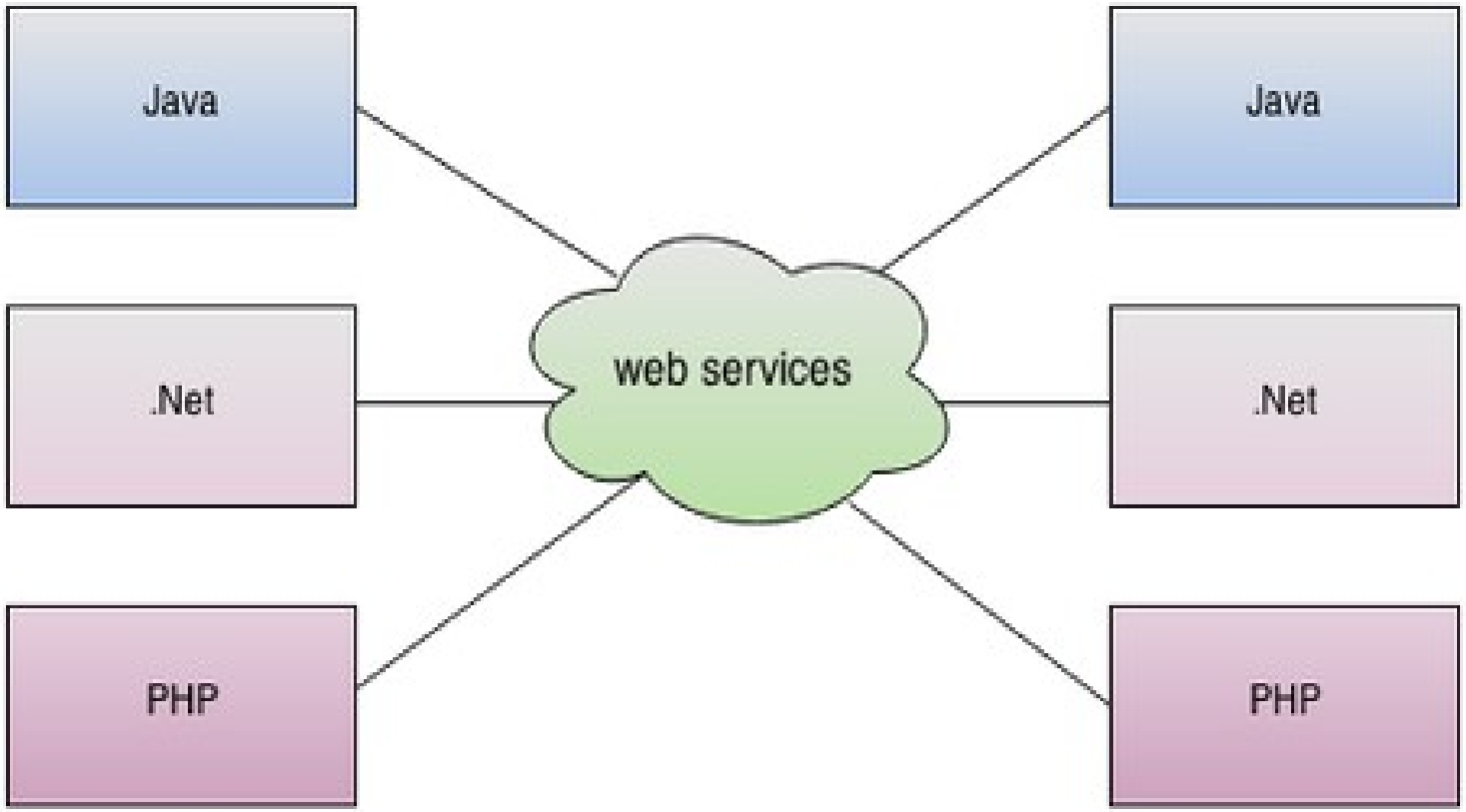
Web Services

A Web Service is can be defined by following ways:

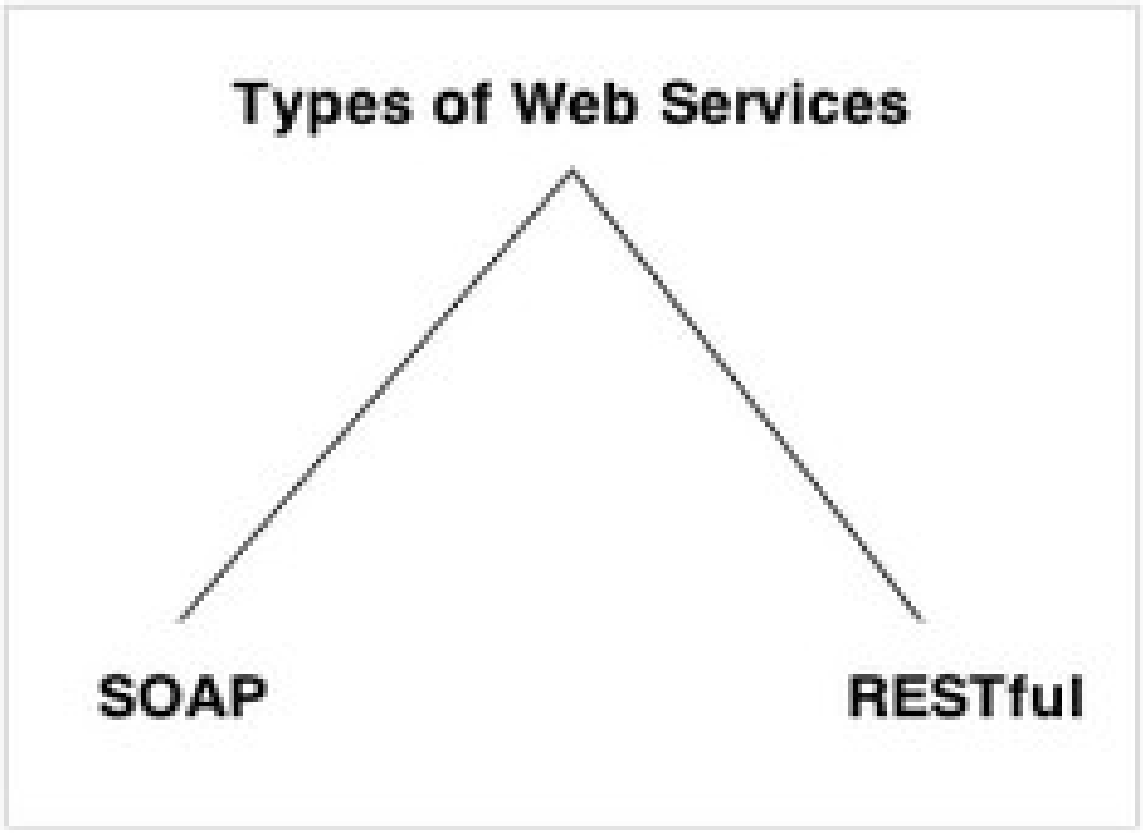
- is a client server application or application component for communication.
- method of communication between two devices over network.
- is a software system for interoperable machine to machine communication.
- is a collection of standards or protocols for exchanging information between two devices or application.

Web Services

Web service is a language independent way of communication.



SOAP vs RESTful services



SOAP vs RESTful services

No.	SOAP	REST
1)	SOAP is a **protocol**.	REST is an **architectural style**.
2)	SOAP stands for **Simple Object Access Protocol**.	REST stands for **REpresentational State Transfer**.
3)	SOAP **can't use REST** because it is a protocol.	REST **can use SOAP** web services because it is a concept and can use any protocol like HTTP, SOAP.
4)	SOAP **uses services interfaces to expose the business logic**.	REST **uses URI to expose business logic**.
5)	**JAX-WS** is the java API for SOAP web services.	**JAX-RS** is the java API for RESTful web services.
6)	SOAP **defines standards** to be strictly followed.	REST does not define too much standards like SOAP.
7)	SOAP **requires more bandwidth** and resource than REST.	REST **requires less bandwidth** and resource than SOAP.
8)	SOAP **defines its own security**.	RESTful web services **inherits security measures** from the underlying transport.
9)	SOAP **permits XML** data format only.	REST **permits different** data format such as Plain text, HTML, XML, JSON etc.
10)	SOAP is **less preferred** than REST.	REST **more preferred** than SOAP.

Task No - 1

<14>Feb 03 13:30:22 mahesh.com type=USER_LOGIN msg=audit(1422950418.736:39770): user pid=11590 uid=0 auid=0 ses=3 subj=unconfined_u:system_r:remote_login_t:s0-s0:c0.c1023 msg='op=login id=0 exe="/bin/login" hostname=10.35.34.213 addr=10.35.34.213 terminal=pts/3 res=failed'

<14>Feb 03 13:30:36 mahesh.com type=USER_LOGIN msg=audit(1422950431.355:39778): user pid=11590 uid=0 auid=510 ses=484 subj=unconfined_u:system_r:remote_login_t:s0-s0:c0.c1023 msg='op=login id=510 exe="/bin/login" hostname=10.35.34.213 addr=10.35.34.213 terminal=pts/3 res=success'

<14>Feb 03 13:30:36 mahesh.com type=USER_AUTH msg=audit(1422950431.161:39771): user pid=11590 uid=0 auid=0 ses=3 subj=unconfined_u:system_r:remote_login_t:s0-s0:c0.c1023 msg='op=PAM:authentication acct="punit" exe="/bin/login" hostname=10.35.34.213 addr=10.35.34.213 terminal=pts/3 res=success'

Task No - 1

<14>Feb 03 13:30:22 mahesh.com type=USER_LOGIN msg=audit(1422950418.736:39770): user pid=11590 uid=0 auid=0 ses=3 subj=unconfined_u:system_r:remote_login_t:s0-s0:c0.c1023 msg='op=login id=0 exe="/bin/login" hostname=10.35.34.213 addr=10.35.34.213 terminal=pts/3 res=failed'

- Event Name
- User
- Hostname
- Status

Regular Expressions

--> The lexical analyzer needs to scan and identify only a finite set of valid string/token/lexeme that belong to the language in hand. It searches for the pattern defined by the language rules.

--> Regular expressions have the capability to express finite languages by defining a pattern for finite strings of symbols. The grammar defined by regular expressions is known as regular grammar. The language defined by regular grammar is known as regular language.

--> Regular Expression practice : - Regex101

--Task : Write Java code to parse log and display and insert fields in Database.

Task 2: Accept Log file and count the events

Task 3: Create Log Collection and Correlation System

GSLAB

Thank You

www.gslab.com