NGINX (Ver 1.10+)

Background

The NGINX server logs analysis App automatically Collect - Read - Parse - Analyzes - Reports all machine's generated log data of the server and presents a comprehensive set of graphs and reports to analyze machine generated data. Use a predefined set of dashboards and gadgets to visualize and address the system software, code written, and infrastructure during development, testing, and production. This NGINX server logs analysis App helps you measure, troubleshoot, and optimize your servers integrity, stability and quality with visualization and investigation dashboards.

Steps

1. Add Log Data In XpoLog, When adding a log to XpoLog you can now select the Log Type (logtype) for NGINX with e the following logtypes:
    a. *nginx*
    b. *w3c*
    c. *webserver*
        i. In addition select not only httpd but also the log type - ***access*** or ***error***
        ii. *See error log definition at the bottom of this page*
2. Once all required information is set click next and edit the log pattern, this step is crucial to the accuracy and deployment of the Analytic App. Use the following conversion table to build the XpoLog pattern.

Example

In the NGINX configuration file, usually nginx.conf by default, located under the conf/ directory (Linux "NGINX ROOT DIR/conf/nginx.conf") search for the _____ directive:

Information from NGINX site:

"NGINX writes information about client requests in the access log right after the request is processed. By default, the access log is located at logs/access.log, and the information is written to the log in the predefined combined format. To override the default setting, use the `log_format` directive to change the format of logged messages, as well as the `access_log` directive to specify the location of the log and its format. The log format is defined using variables.

The following examples define the log format that extends the predefined combined format with the value indicating the ratio of gzip compression of the response. The format is then applied to a virtual server that enables compression.

access_log path [format [buffer=size] [gzip[=level]] [flush=time] [if=condition]];

access_log off;
Default:
access_log logs/access.log combined;

```
log_format main      '$remote_addr - $remote_user [$time_local] "$request" '
                     '$status $body_bytes_sent "$http_referer" '
                     '"$http_user_agent" "$http_x_forwarded_for"';
```

In XpoLog such pattern will be translated into:

{ip:RemoteIP,ftype=remoteip} - {text:Remote User,ftype=remoteuser} [{date:Date,dd/MMM/yyyy:HH:mm:ss z}] "{choice:Method,ftype=reqmethod;,GET;POST;HEAD} {url:URL,paramsFtype=querystring;ftype=requrl;paramsName=Query;,} {text:Request Protocol,ftype=reqprotocol;,}" {number:ResponseStatus,ftype=respstatus} {number:Bytes Sent,ftype=bytesent} "{string:RefererQuery,ftype=refererquery;,}{regexp:Referer,ftype=referer;refName=RefererQuery,^([\w-]+://[^?]+|/[^?]+)}" "{text:User-Agent,ftype=useragent}" "{ip:X-Forwarded-For,ftype=forwardforip}"{eoe}

for more information see below:

Apache Https Access Log Format Conversion Table

logtyep should be set to: *nginx, access*

Field	Appears as	Description	XpoLog Pattern
$arg_*name*		argument *name* in the request line	{text:Argument Name}
$args		arguments in the request line	{text:Query String,ftype=querystring}
$binary_remote_addr		client address in a binary form, value's length is always 4 bytes for IPv4 addresses or 16 bytes for IPv6 addresses	{ip:Binary Remote IP,ftype=binaryremoteip}
$body_bytes_sent		number of bytes sent to a client, not counting the response header; this variable is compatible with the "%B" parameter of the mod_log_config Apache module	{number:Bytes Sent,ftype=bytesent}
$bytes_sent		number of bytes sent to a client (1.3.8, 1.2.5)	{number:Bytes Sent,ftype=bytesent}
$connection		connection serial number (1.3.8, 1.2.5)	{text:Connection Serial Number,ftype=connser
$connection_requests		current number of requests made through a connection (1.3.8, 1.2.5)	{text:Connection Requests,ftype=numofreques
$content_length		"Content-Length" request header field	{text:Content Length,ftype=contentlength}
$content_type		"Content-Type" request header field	{text:Content Type,ftype=contenttype}
$cookie_*name*		the *name* cookie	{text:Cookie,ftype=cookie}
$document_root		root or alias directive's value for the current request	{text:Document Root,ftype=documentroot}
$document_uri		same as $uri	{text:Request URL,ftype=requrl}
$host		in this order of precedence: host name from the request line, or host name from the "Host" request header field, or the server name matching a request	{text:Server Name,ftype=servername}
$hostname		Host name	{text:Remotehost,ftype=remoteip}

$http_*name*		Arbitrary request header field; the last part of a variable *name* is the field name converted to lower case with dashes replaced by underscores	{text:HTTP Name,ftype=httpname}
$https		“on” if connection operates in SSL mode, or an empty string otherwise	{text:SSL,ftype=https}
$is_args		“?” if a request line has arguments, or an empty string otherwise	{text:Has Query Arguments}
$limit_rate		setting this variable enables response rate limiting; see li mit_rate	{text:Rate Limiting}
$msec		current time in seconds with the milliseconds resolution (1.3.9, 1.2.6)	{timestamp:yyyy-MM-dd HH:mm:ss.SSS}
$nginx_version		nginx version	{text:NGINX Version}
$pid		PID of the worker process	{text:ProcessID,ftype= processid}
$pipe		“p” if request was pipelined, “.” otherwise (1.3.12, 1.2.7)	{text:PIPE}
$proxy_protocol_addr		client address from the PROXY protocol header, or an empty string otherwise (1.5.12) The PROXY protocol must be previously enabled by setting the proxy_protocol parameter in the listendirective.	{text:X-Forwarded-For,ftype=forwardforip}
$proxy_protocol_port		client port from the PROXY protocol header, or an empty string otherwise (1.11.0) The PROXY protocol must be previously enabled by setting the proxy_protocol parameter in the listendirective.	{text:Client Port,ftype=clientport}
$query_string		same as $args	{text:Query String,ftype=querystring}

$realpath_root		an absolute pathname corresponding to the root or alias directive's value for the current request, with all symbolic links resolved to real paths	{text:Real Path}
$remote_addr		client address	{geoip:RemoteIP,ftype=remoteip}
$remote_port		client port	{number:Remote Port,ftype=remoteport}
$remote_user		User name supplied with the Basic authentication	{text:Remote User,ftype=remoteuser}
$request		Full original request line	1. {choice:Method,ftype=reqmethod;,GET;P {url:URL,paramsFtype=querystring;ftype= Protocol,ftype=reqprotocol;,}
$request_body		request body The variable's value is made available in locations processed by the proxy_pass , fastcgi_pass,uwsgi_pass, and scgi_pass directives when the request body was read to a memory buffer.	{text:Request Body}
$request_body_file		name of a temporary file with the request body At the end of processing, the file needs to be removed. To always write the request body to a file,client_body_in_file_only needs to be enabled. When the name of a temporary file is passed in a proxied request or in a request to a FastCGI/uwsgi/SCGI server, passing the request body should be disabled by the proxy_pass_request_body off, fastcgi_pass_request_body off, uwsgi_pass_request_body off, orscgi_pass_request_body off directives, respectively.	{text:Request Body File}
$request_completion		"OK" if a request has completed, or an empty string otherwise	{text:Request Completion}
$request_filename		file path for the current request, based on the root or alias directives, and the request URI	{text:Request File Name}
$request_id		unique request identifier generated from 16 random bytes, in hexadecimal (1.11.0)	{text:Request Identifier}

$request_length		request length (including request line, header, and request body) (1.3.12, 1.2.7)	{number:Request Length}
$request_method		request method, usually “GET” or “POST”	{choice:Method,ftype=reqmethod;,GET;POST;
$request_time		request processing time in seconds with a milliseconds resolution (1.3.9, 1.2.6); time elapsed since the first bytes were read from the client	{number:Request Time,ftype=reqtime}
$request_uri		full original request URI (with arguments)	{text:Request URL,ftype=requrl}
$scheme		request scheme, “http” or “https”	{text:Request Protocol,ftype=reqprotocol}
$sent_http_*name*		arbitrary response header field; the last part of a variable name is the field name converted to lower case with dashes replaced by underscores	{text:Sent Http Name,ftype=senthttp}
$sent_trailer_name		Arbitrary field sent at the end of the response (1.13.2); the last part of a variable name is the field name converted to lower case with dashes replaced by underscores	{text:Sent Trailer Name,ftype=trailername}
$server_addr		an address of the server which accepted a request Computing a value of this variable usually requires one system call. To avoid a system call, the listendirectives must specify addresses and use the bind parameter.	{ip:Local IP,ftype=localip}
$server_name		name of the server which accepted a request	{text:Server Name,ftype=servername}
$server_port		port of the server which accepted a request	{number:Server Port,ftype=serverport}
$server_protocol		request protocol, usually “HTTP/1.0”, “HTTP/1.1”, or “HTTP/2.0”	{text:Request Protocol,ftype=reqprotocol}
$status		response status (1.3.2, 1.2.2)	{number:Response Status,ftype=respstatus}
$tcpinfo_rtt, $tcpinfo_rttvar, $tcpinfo_snd_cwnd, $tcpinfo_rcv_space		information about the client TCP connection; available on systems that support the TCP_INFO socket option	1.{text:TCP Info RTT} 2. {text:TCP Info RTTVAR} 3.{text:TCP Info SND CWND} 4.{text:TCP Info RCV SPACE}

$time_iso8601		local time in the ISO 8601 standard format (1.3.12, 1.2.7)	{date:Date,yyyy-MM-dd'T' HH:mm:ss z}
$time_local		local time in the Common Log Format (1.3.12, 1.2.7)	{date:Date,dd/MMM/yyyy:HH:mm:ss z}
$uri		current URI in request, normalized The value of $uri may change during request processing, e.g. when doing internal redirects, or when using index files.	{text:Request URL,ftype=requrl}
`$http_user_agent`		The User Agent which is associated with the request	{text:User Agent,ftype=useragent;,}
`$http_referer`		The referer which is associated with the request	{string:RefererQuery,ftype=refererquery;,}{reg
$http_x_forwarded_for		Method for identifying the originating IP address of a client connecting to a web server through an HTTP proxy or load balancer.	{ip:X-Forwarded-For,ftype=forwardforip}
$upstream_addr		keeps the IP address and port, or the path to the UNIX-domain socket of the upstream server.	{text:Upstream Address,ftype=upstreamaddr}
$upstream_status		keeps status code of the response obtained from the upstream server.	{text:Upstream Status,ftype=upstreamstatus}
$upstream_response_time		keeps time spent on receiving the response from the upstream server; the time is kept in seconds with millisecond resolution.	{text:Upstream Response Time,ftype=upstrea
$geoip_country_name		The country from which the request was sent from.	{text:Country Name,ftype=countryname}
$geoip_country_code		The country code from which the request was sent from.	{text:Country Code,ftype=countrycode}
$geoip_region_name		The region from which the request was sent from.	{text:Region,ftype=region}
$geoip_city		The city from which the request was sent from	{text:City,ftype=city}

Error Log

Look for the ***error_log logs/error.log warn;*** directive the nginx configuration file.

YYYY/MM/DD HH:MM:SS [LEVEL] PID#TID: *CID MESSAGE

With PID and TID being the logging process and thread id and CID a number identifying a (probably proxied) connection, probably a counter. The *CID part is optional.

debug, info, notice,warn, error, crit, alert, or emerg.

Default XpoLog Pattern:

First Pattern:

{date:Date,yyyy/MM/dd HH:mm:ss} [{priority:Level,ftype=severity,debug;info;notice;warn;error;crit;alert;emerg}] {text:PID,ftype=processid}#{text:TID,ftype=threadid}: {text:CID,ftype=connectionid} {string:Message,ftype=message}, client: {geoip:Remote IP,ftype=remoteip}, server: {geoip:LocalIP,ftype=localip}, request: \"{choice:Method,ftype=reqmethod;,GET;POST} {url:URL,paramsFtype=querystring;ftype=requrl;paramsName=Query;,} {string:reqprotocol,ftype=reqprotocol;,}\", host: \"{text:Remotehost,ftype=host}\"{block,start,emptiness=true}, referrer: \"{text:RefererQuery,ftype=refererquery;,}{regexp:Referer,ftype=referer;refName=RefererQuery,^([\w-]+://[^?]+|/[^?]+)}\"{block,end,emptiness=true}

Second Pattern

{date:Date,yyyy/MM/dd HH:mm:ss} [{priority:Level,ftype=severity,debug;info;notice;warn;error;crit;alert;emerg}] {text:PID,ftype=processid}#{text:TID,ftype=threadid}:{block,start,emptiness=true}{text:CID,ftype=connectionid} {block,end,emptiness=true}{string:Message,ftype=message}