

Tencent Cloud EdgeOne

Log Service

Product Documentation



Copyright Notice

©2013-2024 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

Trademark Notice



All trademarks associated with Tencent Cloud and its services are owned by Tencent Cloud Computing (Beijing) Company Limited and its affiliated companies. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

Contents

Log Service

Real-time Logs

Overview

Ship Real-time logs

Real-time log field Description

Offline Logs

Log Service

Real-time Logs

Overview

Last updated : 2023-08-16 16:08:04

Function Overview

After adding your site to EdgeOne Service, EdgeOne provides you with a wealth of pre-built reports to help you monitor and analyze the operation of your business, including traffic analysis, cache analysis, L4 proxy, security analysis, etc. However, in data analysis, you may have more personalized data analysis demands, such as the following data analysis scenarios:

Scenario	Scenario Demands
Deep Data Analysis	<p>Need to specify one or more conditions to find logs that meet the conditions. For example: By specifying the client IP, query the access statistics (access URL, number of accesses, etc.) within a specified time range.</p> <p>Refine the analysis of status code distribution by filtering status codes, time, and URLs. By filtering logs with the action set to "observe", summarize the request header content and other request features carried, and adjust the security policy.</p>
Monitoring Service Metrics	Analyze the quality of EdgeOne Service and the access efficiency of users to detect exceptions in a timely manner. Access efficiency includes overall response time, download speed, origin-pull response time, etc.
Identifying Unauthorized Access	Identify client IPs with behaviors such as unauthorized access by analyzing traffic anomalies, access patterns, and access frequency.
Unified Monitoring of Data from Multiple Cloud Vendors	Build your own data dashboard to monitor application data from multiple cloud vendors.
Storing Logs	User-related access logs need to be retained for 30 days or longer.

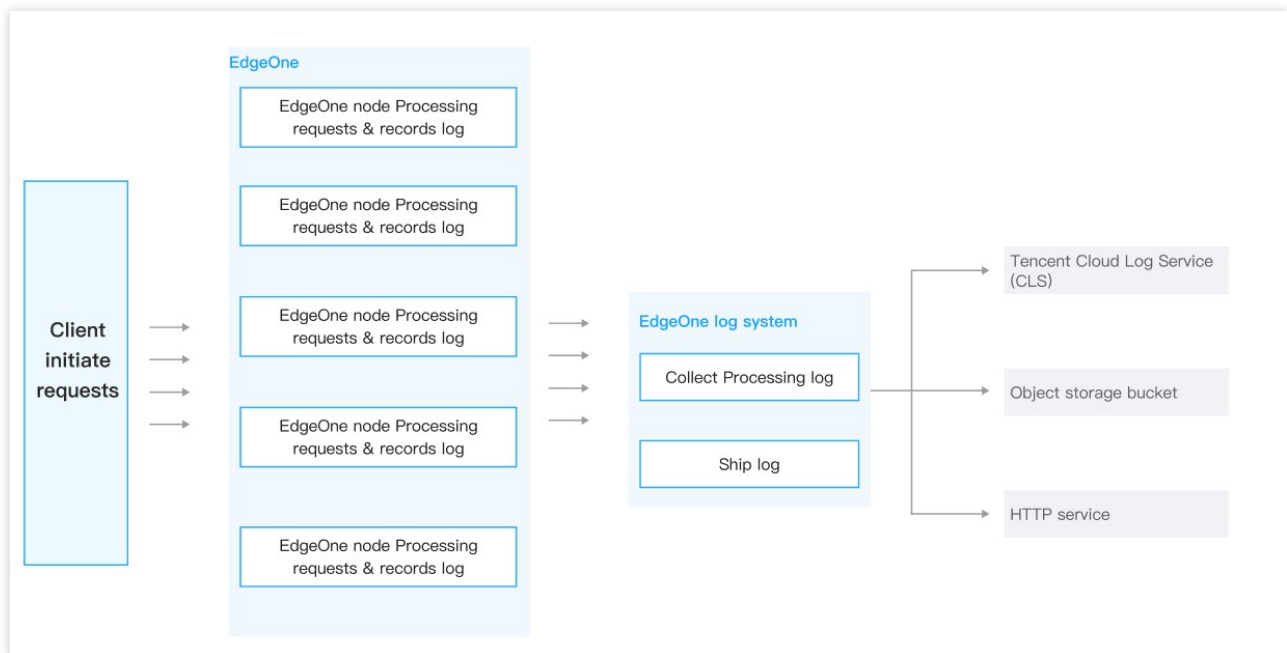
For the above scenario demands, EdgeOne real-time logging provides the ability to collect and ship logs in real-time, allowing you to ship your logs to Tencent Cloud Log Service (CLS) or your self-built data center, helping you to

implement flexible log data retrieval and analysis on your own. Currently, EdgeOne supports shipping site logs, L4 proxy logs, and security service logs to the following destinations:

Tencent Cloud Log Service (CLS): Ship logs to the one-stop log processing service (CLS) provided by Tencent Cloud for further log analysis on CLS.

Object Storage: Storage buckets compatible with AWS Signature V4 authentication method.

HTTP Service (POST): Ship logs to the specified backend server via HTTP POST requests.



Note :

1. In general, the log delivery delay is within 2-5 minutes. To ensure the real-time performance of log delivery, EdgeOne ships logs in fixed log quantities or fixed time periods as a batch to the corresponding destinations.
2. When shipping real-time logs to CLS service, traffic, storage, and other fees may be generated in CLS, and the related fees are charged by CLS product. For details, please view the [Log Service Billing Overview](#).

Billing and Quota Description

Billing Description: Real-time log shipping is a value-added service, and the billing method is based on the number of logs shipped. For details, please view the [VAU Fee \(Pay-as-You-Go\)](#) .

Quota Description: The number of real-time log shipping tasks varies depending on the plan, and the specific quota can be viewed in the [Comparison of EdgeOne Plans](#).

Ship Real-time logs

Last updated : 2024-01-02 10:25:33

This document will guide you on how to push logs to a specified service.

Step 1: Select the log source

1. Log in to the [EdgeOne console](#) and click Site List in the left sidebar. In the site list, click the target site to enter the site details page.
2. On the site details page, click **Log Service > Real-time logs**.
3. On the real-time log page, click **Create shipping Task**.
4. On the Select Log Source page, choose the log source information you want to push, configure the related parameters, and click **Next**.

The screenshot shows a configuration form for a log push task. It includes the following fields and options:

- Task name ***: A text input field with a placeholder and a note: "1-200 characters ([a-z], [A-Z], [0-9], [-])".
- Log type ***: A dropdown menu currently set to "Site acceleration". Below it, it says "Available task quota: 5".
- Service area ***: A dropdown menu currently set to "Global (MLC excluded)".
- Domain name ***: This section is split into two panels.
 - Select subdomain name**: A list box with a search bar "Enter Domain name" and a "Select all" button. It contains two items, the second of which is selected with a blue checkmark.
 - Selected (1)**: A list box showing the selected domain "ztstest.qcdntest.com.cn" with a close button (X).

Log Type: You can choose from site acceleration log, L4 proxy log, rate limiting log, CC attack defense log, Web attack defense log, custom rule log, and Bot management log;

Service Area: Select the log area you want to push. EdgeOne real-time log push tasks can push logs from the "Chinese mainland" or the "Global(MLC excluded)", but cannot directly push logs from the "Global". If you need to push logs from the "Global", please create two push tasks, one for the "Chinese mainland" and another for the "Global(MLC excluded)".

Domain: Select the subdomain or L4 instance for which you want to push logs. The same log does not support multiple push tasks, i.e., logs from subdomains/L4 proxy instances in the same region can only support one push task. For example, if the "Chinese mainland" site acceleration log of `www.example.com` has been configured with push task A, push task B cannot select `www.example.com`.

Step 2: Select log fields

1. In the Select Log Fields section, configure the fields you want to push. You can select them by checking the boxes in the field list; for a description of the related fields, please refer to [the real-time log field description](#).

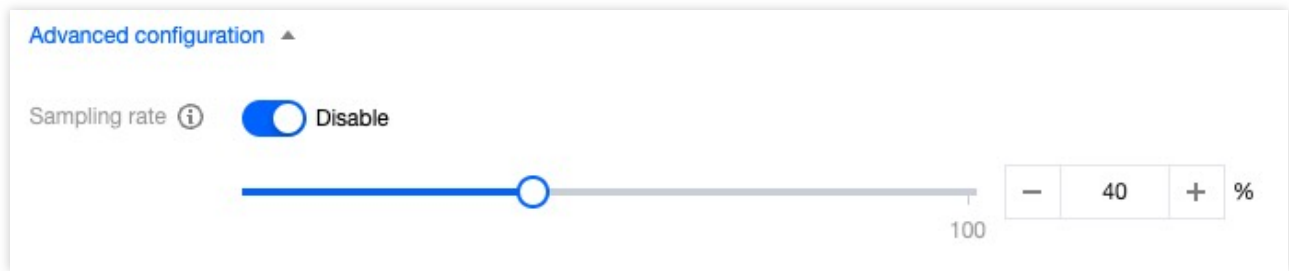
Note :

Currently, only **site acceleration logs** and **L4 proxy logs** support custom selection of logs to be pushed.

2. (Optional) If you need to push certain elements from the HTTP request header, HTTP response header, or Cookie for analysis, you can click Add Custom Field to configure the HTTP request header, HTTP response header, or Cookie name you want to push. You can record this information accurately in the log in key-value pair format. For example, the information corresponding to the `Accept-Language` header can be directly obtained from the `Accept-Language` field in the log.

Note :

1. Fields are case-sensitive by default, so they need to match the original fields exactly;
2. Currently, only **site acceleration logs** support adding custom fields.
3. (Optional) If you have a large volume of logs and only need to monitor and analyze the real-time log push data without requiring all log data, you can click Advanced Configuration to configure the sampling ratio to reduce the number of logs pushed. After configuration, EdgeOne will randomly extract logs according to the set percentage and push them to your specified destination.



4. After configuring the log fields, click Next Step to proceed to Step 3.

Step 3: Select the push destination

You can choose to push real-time logs to Tencent Cloud CLS, S3 compatible bucket, or a specified HTTP server according to your needs. Follow the steps below for configuration:

Ship to Tencent Cloud CLS

Ship to S3 compatible

Ship to specified HTTP server

If you have not yet built your own data analysis system, Tencent Cloud provides Log Service (CLS) to help you complete the collection, shipping, and search analysis of real-time logs in a one-stop manner, reducing your development and maintenance costs. You can follow the steps below to ship real-time logs to Tencent Cloud CLS service:

Prerequisites

You have already activated [Cloud Log Service \(CLS\)](#) and granted permission to Tencent Cloud EdgeOne to create a logset.

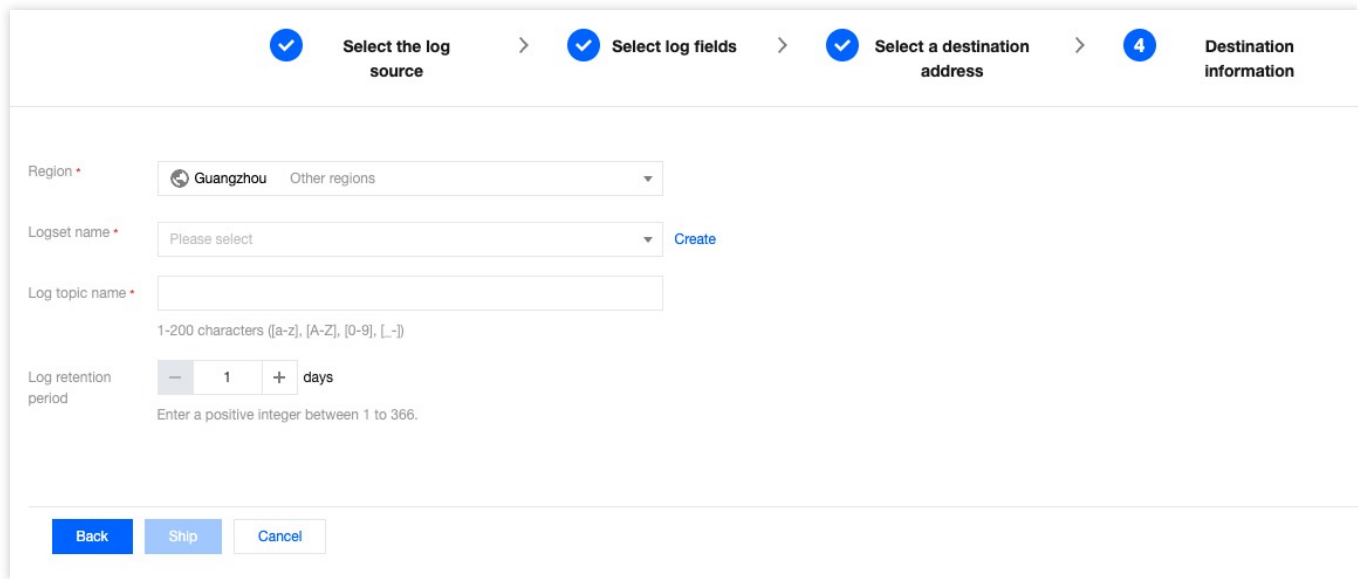
Note :

1. Log Service (CLS) is a paid service, for related fees, please refer to: [Log Service Billing Overview](#).
2. It is suggested to enable the service with the root admin account. If it is a sub-account or collaborator, you need to grant them the relevant permissions.

Directions

Create a shipping task

1. In Step ③, select the destination as **Tencent Cloud Log Service (CLS)** and click **Next**.
2. Fill in the relevant parameter information, the parameter explanation is as follows:



Region: Select the target region for shipping.

Target set name: Select the logset in the target region.

Note:

If this is empty or you need to create a new logset, click **Create** to create a logset in the selected region.

Log topic name: You can enter 1-200 characters, allowed characters are `a-z`, `A-Z`, `0-9`, `_`, `-`.

Log retention time: Please enter a positive integer between 1 and 366.

Related references

Log search

Log search supports various types of search analysis methods and chart analysis forms. For detailed explanations, please refer to [Log Search](#).

EdgeOne performs log search based on shipping tasks. On the Real-time logs page, select the shipping task you want to search, and click Search to enter the log search page.

You can later manage logsets and other modules through Tencent Cloud Log Service (CLS), such as modifying the logset name.

Logset

A logset (Logset) is a project management unit of Tencent Cloud Log Service (CLS), used to distinguish logs of different projects, and a logset corresponds to a collection. Tencent Cloud EdgeOne logset has the following basic attribute information:

Region: The [region](#) where the logset belongs.

Logset name: Logset naming.

Log retention time: The retention period of data in the current logset.

Creation time: Logset creation time.

Log topic

A log topic (Topic) is a basic management unit of Tencent Cloud Log Service (CLS). A logset can contain multiple log topics. A log topic corresponds to a type of application or service, and it is recommended to collect the same type of logs from different machines into the same log topic. For example, a business project has three types of logs: operation logs, application logs, and access logs, and each type can create a corresponding log topic.

The log service system manages different log data of users based on log topics, and each log topic can be configured with different data sources, different index rules, and delivery rules. Therefore, the log topic is the basic unit for configuring and managing log data in the log service. After creating a log topic, you need to configure the relevant rules to effectively collect logs and use search analysis and delivery functions as scheduled.

From a functional perspective, log topics mainly provide:

Collect logs to log topics.

Store and manage logs in units of log topics.

Search and analyze logs in units of log topics.

Deliver logs from log topics to other platforms.

Download and consume logs from log topics.

Note

The above information is excerpted from the [Cloud Log Service \(CLS\)](#) product documentation. Please refer to the explanations on the Log Service (CLS) side.

Each real-time log shipping task shipped to Tencent Cloud Log Service (CLS) will ship the logs of the selected subdomains to a corresponding log topic.

If you currently have your own built-in data source and need to ship real-time logs to a compatible Amazon Simple Storage Service bucket, you can refer to the following steps to continue the operation:

Note :

Currently, only support shipping site acceleration logs and L4 proxy logs to compatible Amazon S3 Storage Service buckets.

The format for log shipping is [JSON Lines](#).

Directions

1. In Step ③, Select the destination as **S3 compatible** and click **next**.

2. Fill in the corresponding destination parameters:

Endpoint URL: URL that does not contain the bucket name or path, such as:

`https://storage.googleapis.com` , `https://s3.ap-northeast-2.amazonaws.com` .

Bucket Region: The region where the bucket is located, such as: `ap-northeast-2` .

Bucket: The bucket name and the corresponding log storage path: for example, `your_bucket_name/EO-logs/` .

File Compression: Whether to use gzip compression for log files. If checked, the shipped log files will be compressed with gzip, and the file name will be changed to `filename.log.gz`.

SecretId: Access Key ID used to access the bucket.

SecretKey: Secret key used to access the bucket.

Note :

1. The bucket needs to be compatible with [AWS Signature Version 4 Authentication Algorithm](#). For specific compatibility, please refer to the instructions provided by your bucket provider.

2. File name description: Logs will be stored in the specified bucket path in the format of

`UploadTime_Random.log`, and logs will be archived in a folder by date (UTC +00:00), such as: `logs/20230331/20230331T185917Z_2aadf5ce.log`.

UploadTime: Log file upload time, using ISO-8601 format, UTC+00:00 timezone.

Random: Random characters. In cases where there are large log volumes, there may be multiple log files with the same upload time, and this random character string is used to identify different files.

3. Click **Push**. After issuing the real-time log shipping task, EdgeOne will ship a test file to the target bucket path to verify connectivity. For instance, a file named `1699874755_edgeone_push_test.txt` will be shipped with the fixed string `test`.

If you currently have a self-built data source, EdgeOne can call the backend interface address you provided by an HTTP POST request, transmitting the logs to your designated server within the HTTP body.

Note :

1. HTTP is plaintext transmission, so it is suggested that you use an encrypted HTTPS address for the API.

2. To further enhance the security verification of request sources, we provide a request authentication scheme. You can fill in the relevant authentication information in the push destination configuration, and the authentication algorithm can be found at: [Request Authentication Algorithm](#).
3. The log shipping format comprises an array of multiple JSON objects, and each JSON object is a log.

Operation Guide

Create a shipping task

1. In Step ③, select the destination as **HTTP service (POST)** and click **Next**.
2. Fill in the relevant destination and parameter information, with the following parameter descriptions:

API address: Enter your data source API address, e.g., `https://www.example.com/log`

File compression: To reduce the size of log files and save traffic costs, you can enable file compression by checking **"Compress log files with gzip"**. EdgeOne will use gzip format to compress logs before transmission and will add an HTTP request header `content-encoding = gzip` to indicate the compression format.

Origin authentication: When selecting encryption authentication, the shipping logs will carry authentication information for the origin to verify, ensuring the security of the data source identity.

Custom HTTP request headers: Add the HTTP headers that need to be carried when EdgeOne initiates a request. For example, if you need to identify the log source vendor as EdgeOne, you can add a header `log-source = EdgeOne` to identify the log source.

Progress bar: Select the log source > Select log fields > Select a destination address > 4

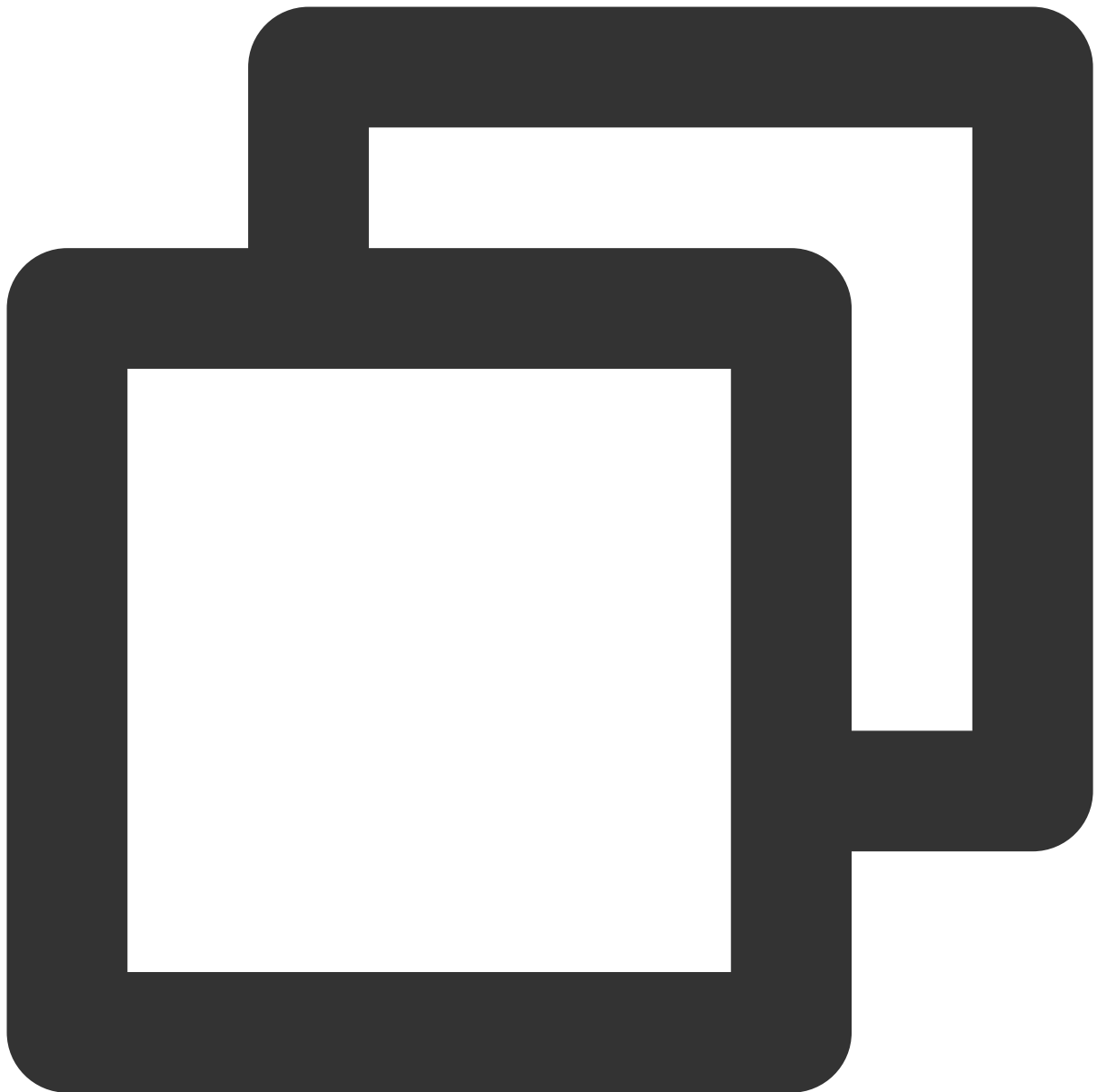
Address *
Enter the API address that supports POST requests

File compression ☐ Compress log files with gzip

Origin authentication ☒ None ☐ Signature ⓘ
It identifies the API caller with a 32-bit fixed length. For detailed signature verification methods, please [see here](#).

[Advanced settings](#) ▶

3. Click **"ship"** to issue a real-time log shipping task.
4. During the configuration phase of the real-time log shipping task, in order to verify the interface connectivity, an empty data will be sent to the interface address for verification. The data format is as follows:



```
.[.
  "BotClassAccountTakeOver": "-"/_
  "BotClassAttacker": "-"/_
  "BotClassMaliciousBot": "-"/_
  "BotClassProxy": "-"/_
  "BotClassScanner": "-"/_
  "ClientDeviceType": "-"/_
  "ClientIP": "-"/_
  "ClientISP": "-"/_
  "ClientRegion": "-"/_
  "ClientState": "-"/_
```

```

"EdgeCacheStatus": "-",
"EdgeEndTime": "-",
"EdgeInternalTime": "-",
"EdgeResponseBodyBytes": "-",
"EdgeResponseBytes": "-",
"EdgeResponseStatusCode": "-",
"EdgeResponseTime": "-",
"EdgeServerID": "-",
"EdgeServerIP": "-",
"EdgeSeverRegion": "-",
"LogTime": "-",
"OriginDNSResponseDuration": "-",
"OriginIP": "-",
"OriginRequestHeaderSendDuration": "-",
"OriginResponseHeaderDuration": "-",
"OriginResponseStatusCode": "-",
"OriginSSLProtocol": "-",
"OriginTCPHandshakeDuration": "-",
"OriginTLSHandshakeDuration": "-",
"ParentRequestID": "-",
"RemotePort": "-",
"RequestBytes": "-",
"RequestHost": "-",
"RequestID": "-",
"RequestMethod": "-",
"RequestProtocol": "-",
"RequestRange": "-",
"RequestReferer": "-",
"RequestSSLProtocol": "-",
"RequestTime": "-",
"RequestUA": "-",
"RequestUrl": "-",
"RequestUrlQueryString": "-"
}

```

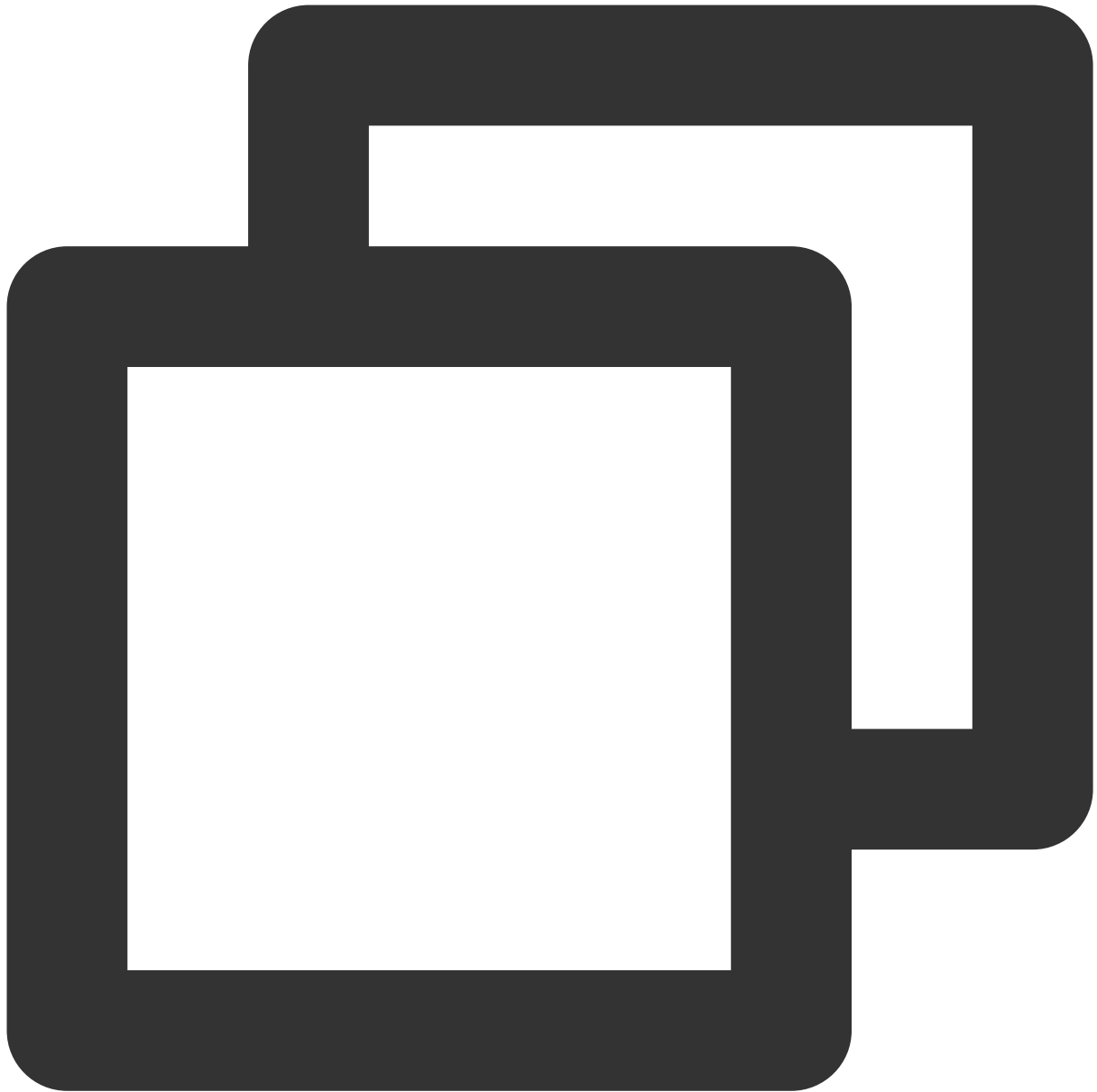
Related References

Request Authentication Algorithm

If you have selected encryption signature in the origin authentication of the push destination information, you can custom input your custom Configuration SecretId and SecretKey. EdgeOne will add the signature `auth_key` and `access_key` in the Request URL. The details of the signature algorithm are as follows:

1. Request URL composition

As shown below, the Request URL will carry `auth_key` and `access_key` after the "?".



```
http://DomainName[:port]/[uri]?auth_key=timestamp-rand-md5hash&access_key=SecretID
```

Parameter description:

timestamp: The current time of the request, using a Unix 10-digit second-level timestamp.

rand: random number

access_key: used to identify the identity of the API requester, that is, your custom Configuration SecretID.

SecretKey: fixed Length 32, that is, your custom Configuration SecretKey.

uri: resource identifier, for example: `/access_log/post` .

md5hash: `md5hash = md5sum(string_to_sign)` , where `string_to_sign = "uri-timestamp-rand-SecretKey"` . The verification string calculated by the md5 algorithm, a mixture of numbers 0-9 and lowercase English letters a-z, fixed Length 32.

2. Calculation example

Assuming the filled in parameters are:

API address: `https://www.example.com/access_log/post`

SecretID = `YourID`

SecretKey = `YourKey`

uri = `/access_log/post`

timestamp = `1571587200`

rand = `0`



```
string_to_sign = "/access_log/post-1571587200-0-YourKey"
```

Based on this string, calculate



```
md5hash=md5sum("/access_log/post-1571587200-0-YourKey")=1f7ffa7bff8f06bbfbe2ace0f14
```

The final push request URL is:



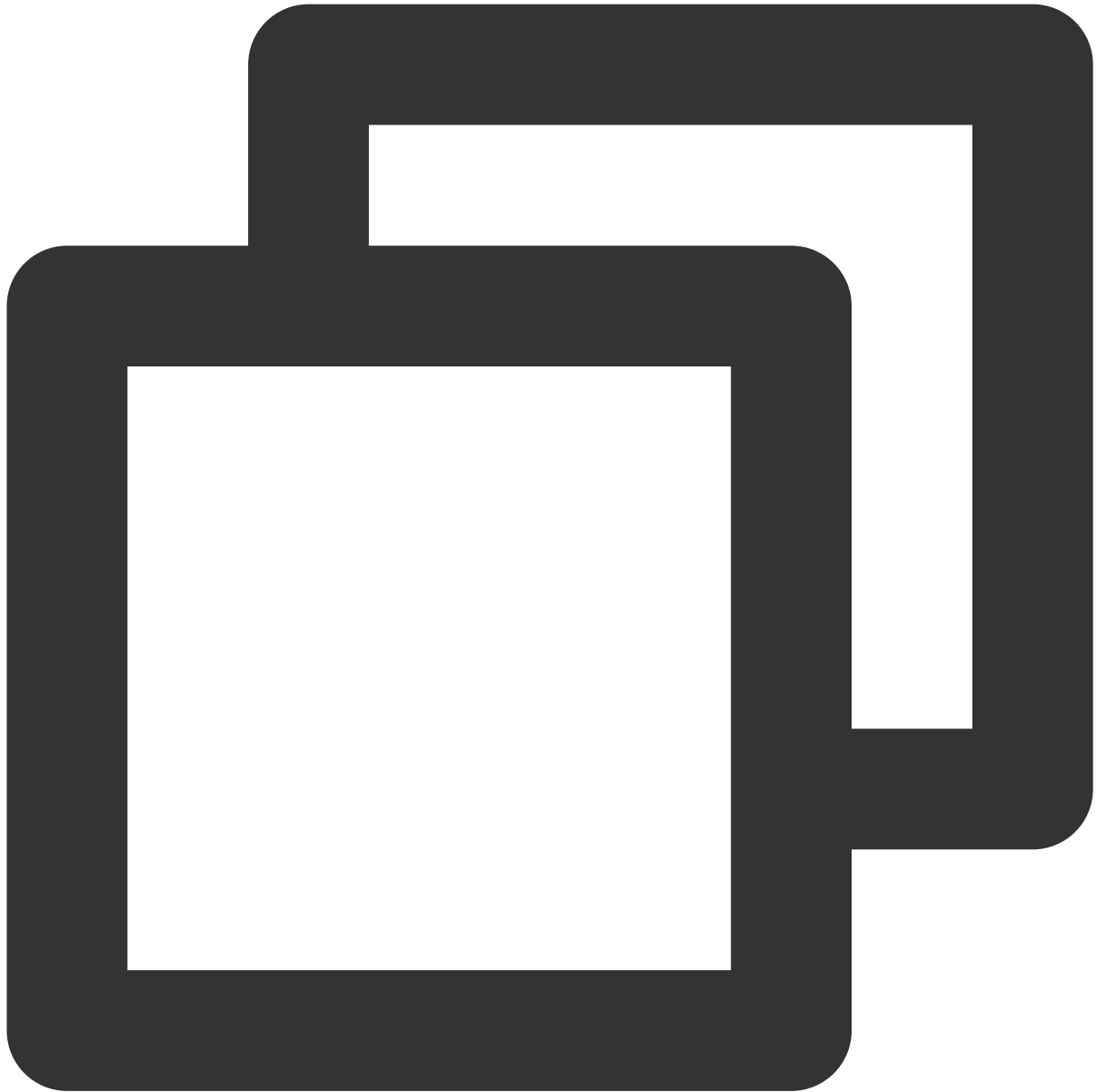
```
https://www.example.com/cdnlog/post?auth_key=1571587200-0-1f7ffa7bff8f06bbfbe2ace0f
```

After the service receives the push request, it extracts the value of `auth_key`. Split the value of `auth_key` to obtain `timestamp`, `rand`, and `md5hash`. You can first check whether the timestamp is expired, the suggested expiration time is `300s`, and assemble the encryption string based on the above rules. Use `SecretKey` to assemble the string to be encrypted, and compare the encrypted result with the `md5hash` value in `auth_key`. If they are the same, it means the authentication has passed.

3. Server-side authentication request resolution code example

Python

Goland



```
import hashlib

from flask import Flask, request

app = Flask(__name__)

def get_rsp(msg, result={}, code=0):
```

```
    return {
        "respCode": code,
        "respMsg": msg,
        "result": result
    }

def get_secret_key(access_key):
    return "secret_key"

@app.route("/access_log/post", methods=['POST'])
def access_log():
    if request.method == 'POST':
        if request.content_type.startswith('application/json'):
            current_time_ts, rand_num, md5hash = request.args.get("auth_key").split()
            # Judge whether the requests Time is within the Validity period
            if time.time() - int(current_time_ts) > 300:
                return get_rsp(msg="The request is out of time", code=-1)

            access_key = request.args.get("access_key")
            # collected secret_key through access_key(SecretID)
            secret_key = get_secret_key(access_key)
            raw_str = "%s-%s-%s-%s" % (request.path, current_time_ts, rand_num, secret_key)
            auth_md5hash = hashlib.md5(raw_str.encode("utf-8")).hexdigest()
            if auth_md5hash == md5hash:
                # Authentication Pass
                if request.headers['content-encoding'] == 'gzip':
                    # Decompression Data
                    pass
                # Data Processing
                return get_rsp("ok")
            return get_rsp(msg="Please use content_type by application/json", code=-1)
        return get_rsp(msg="The request method not find, method == %s" % request.method)

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8888, debug=True)
```



```
package main

import (
    "context"
    "crypto/md5"
    "fmt"
    "log"
    "net/http"
    "os"
    "os/signal"
    "strings"
```

```
"syscall"
)

func main() {
    mux := http.NewServeMux()
    mux.Handle("/access_log/post", &logHandler{})

    server := &http.Server{
        Addr:    ":5000",
        Handler: mux,
    }

    // Create system Signal receiver
    done := make(chan os.Signal)
    signal.Notify(done, os.Interrupt, syscall.SIGINT, syscall.SIGTERM)
    go func() {
        <-done

        if err := server.Shutdown(context.Background()); err != nil {
            log.Fatal("Shutdown server:", err)
        }
    }()

    err := server.ListenAndServe()
    if err != nil {
        if err == http.ErrServerClosed {
            log.Print("Server closed under request")
        } else {
            log.Fatal("Server closed unexpected")
        }
    }
}

type logHandler struct{}

func (*logHandler) ServeHTTP(w http.ResponseWriter, r *http.Request) {
    if r.Method == "POST" {
        query := r.URL.Query()
        authKey := query.Get("auth_key")
        accessKey := query.Get("access_key") //access_key is the SecretID you Provide
        authKeys := strings.Split(authKey, "-")
        if len(authKeys) == 3 {
            currentTimeTs := authKeys[0]

            //Carry out Timestamp Validity period judgment
            RandNum := authKeys[1]
            md5Hash := authKeys[2]
        }
    }
}
```



```
    secretKey := getSecretKey(accessKey)
    authStr := fmt.Sprintf("%s-%s-%s-%s", "/access_log/post", currentTimeTs
    data := []byte(authStr)
    has := md5.Sum(data)
    authMd5 := fmt.Sprintf("%x", has) //Conversion to String for Comparison
    if authMd5 == md5Hash {
        // todo Authentication successful
        if r.Header.Get("Content-Encoding") == "gzip" {
            //Decompression Data
        }
        //Data Processing
    }
} else {
    //exception handling
}
}

// collected SecretKey
func getSecretKey(accessKey string) string {
    if accessKey != "" {
        // collected Secret_Key through Access_key(SecretID)
        return "secret_key"
    }
    return ""
}
```

Real-time log field Description

Last updated : 2023-12-05 15:40:41

This article introduces the field explanation of site acceleration logs and L4 proxy logs in real-time logs.

Note:

When a field has no value:

If the data type of the field is String and the field has no data, the field value is: "-".

If the data type of the field is Integer and the field has no data, the field value is: -1.

Site Acceleration Log

Name	Data Type	Description
LogTime	Timestamp ISO8601	Time when the log is generated
RequestID	String	Unique ID of the client request
ClientIP	String	Client IP
ClientRegion	String	Country/region parsed from the client IP. Format standard: ISO-3166 alpha-2
ClientState	String	The client IP parses out the country's lower-level administrative divisions. Currently only data within mainland China is supported. Format standard: ISO-3166 alpha-2
ClientISP	String	ISP information resolved from client IP. Data within mainland China is recorded under the ISP's Chinese name. Global Availability Zones (excluding mainland China) data is recorded as Autonomous System Number (ASN)
RequestTime	Timestamp ISO8601	Client request time, time zone: UTC +00:00
RequestStatus	Integer	Status of the client request, if using Websocket protocol, EdgeOne will periodically print logs, this field can be used to determine the connection status. Value options: 0:Not ended 1:Request ended normally

		2:Ended abnormally
RequestHost	String	Host of the client request
RequestBytes	Integer	Size of the client request, unit: Byte
RequestMethod	String	HTTP Method of the client request, value options: GET POST HHEAD PUT DELETE CONNECT OPTIONS TRACE PATCH
RequestSSLProtocol	String	SSL (TLS) protocol used by the client, if the value is "-", there is no SSL handshake in the request; value options: TLS 1.0 TLS 1.1 TLS 1.2 TLS 1.3
ClientDeviceType	String	Device type of the client request, value options: TV Tablet Mobile Desktop Other
RequestUrl	String	URL of the client request
RequestUrlQueryString	String	Query parameter carried by the client request URL
RequestUA	String	User-Agent information of the client request
RequestRange	String	Range parameter information of the client request
RequestReferer	String	Referer information of the client request
RequestProtocol	String	Application layer protocol of the client request, value options: HTTP/1.0 HTTP/1.1 HTTP/2.0 HTTP/3

		WebSocket
RemotePort	Integer	Port for establishing a connection between the client and the node under the TCP protocol
EdgeCacheStatus	String	Whether the client request hits the node cache, value options: hit: Resource provided by the node cache miss: Resource can be cached, but provided by the origin dynamic: Resource cannot be cached
EdgeResponseStatusCode	Integer	Status code returned by the node response to the client
EdgeResponseBytes	Integer	Size of the node response returned to the client, unit: Byte
EdgeResponseBodyBytes	Integer	Body size of the node response returned to the client, unit: Byte
EdgeResponseTime	Integer	Time consumed from the start of receiving the client request by EdgeOne to the end of the client receiving the server response; unit: ms
EdgeInternalTime	Integer	Time consumed from the start of receiving the client request by EdgeOne to the first byte of the response to the client; unit: ms
EdgeServerIP	String	EdgeOne server IP address obtained by DNS resolution of Host
EdgeServerID	String	Unique identifier of the EdgeOne server accessed by the client
EdgeSeverRegion	String	Country of the responding EdgeOne node IP, format standard reference: ISO-3166 alpha-2
EdgeEndTime	Timestamp ISO8601	Time to complete the response to the client request
OriginDNSResponseDuration	Float	The duration taken to receive the DNS resolution response from the origin server. If there is no return to the origin, it is recorded as -1, unit: ms
OriginIP	String	Origin IP accessed by the origin-pull, if not origin-pull, record as "-"
OriginRequestHeaderSendDuration	Float	The duration taken to send the request header to the

		origin server is usually 0. If there is no return to the origin, it is recorded as -1, unit: ms
OriginSSLProtocol	String	SSL protocol version used for requesting the origin, if not origin-pull, record as "-"; value options: TLS 1.0 TLS 1.1 TLS 1.2 TLS 1.3
OriginTCPHandshakeDuration	Float	Time consumed to complete the TCP handshake when requesting the origin, if not origin-pull, record as "-1", unit: ms; Note: 0 when the connection is reused
OriginTLSHandshakeDuration	Float	Time consumed to complete the TLS handshake when requesting the origin, if not origin-pull, record as "-1", unit: ms; Note: 0 when the connection is reused
OriginResponseHeaderDuration	Float	Time consumed from sending the request header to the origin to receiving the response header from the origin, if not origin-pull, record as "-1", unit: ms
OriginResponseStatusCode	Integer	Origin response status code, if not origin-pull, record as "-1"
BotClassAttacker	String	Risk level of the client IP with attack behavior (such as DDoS, high-frequency malicious requests, site attacks, etc.) based on recent intelligence data, "-" corresponds to no historical data, other value options: high: corresponding to high risk medium: corresponding to medium risk low: corresponding to low risk
BotClassProxy	String	Risk level of the client IP with suspicious proxy ports open and used as network proxies (including Proxy) based on recent intelligence data, "-" corresponds to no historical data, other value options: high: corresponding to high risk medium: corresponding to medium risk low: corresponding to low risk
BotClassScanner	String	Based on recent intelligence data, the risk level of the client IP requesting scans for known vulnerabilities is as follows: "-" corresponds to no historical data, and other values are: high: corresponding to high risk medium: corresponding to medium risk

		low: corresponding to low risk
BotClassAccountTakeOver	String	Based on recent intelligence data, the risk level of the client IP requesting malicious account cracking and initiating account takeover attacks is as follows: "-" corresponds to no historical data, and other values are: high: corresponding to high risk medium: corresponding to medium risk low: corresponding to low risk
BotClassMaliciousBot	String	Based on recent intelligence data, the risk level of the client IP requesting malicious bots, hotlinking, and brute force cracking behaviors is as follows: "-" corresponds to no historical data, and other values are: high: corresponding to high risk medium: corresponding to medium risk low: corresponding to low risk

Note :

In the site acceleration log, using the WebSocket protocol for long connections, EdgeOne will periodically record logs and record a log at the end of the final request. You can identify requests by the `RequestID` field, and logs with the same `RequestID` represent the same connection; you can also determine the connection status at the time of log recording through the `RequestStatus` .

L4 Proxy Log

Name	Data Type	Description
ServiceID	String	Unique identifier ID for L4 proxy service
SessionID	String	Unique identifier ID for TCP connection or UDP session
ConnectTimeStamp	Timestamp ISO8601	Connection establishment time; default UTC +0 timezone
DisconnetTimeStamp	Timestamp ISO8601	Disconnection time; default UTC +0 timezone
DisconnetReason	String	Disconnection reason; Format is "direction: reason" Direction values: up: origin direction down: Client direction

		Reason values: net_exception_peer_error: read/write peer returns error net_exception_peer_close: peer has closed connection create_peer_channel_exception: failed to create channel to next hop channel_eof_exception: channel has ended (at the end of the request, the node that ends the request sends channel_eof to the adjacent node to inform that the request has ended) net_exception_closed: connection is closed net_exception_timeout: read/write timeout
ClientRealIP	String	Client real IP
ClientRegion	String	2-letter country/region code of the client, in accordance with ISO-3166 alpha-2 standard
EdgeIP	String	IP address of the accessed EdgeOne server
ForwardProtocol	String	TCP/UDP forwarding protocol configured by the customer
ForwardPort	Integer	Forwarding port configured by the customer
SentBytes	Integer	Inbound traffic generated from the last log record time to this log record time, unit: Byte
ReceivedBytes	Integer	Outbound traffic generated from the last log record time to this log record time, unit: Byte
LogTimeStamp	Timestamp ISO8601	Log generation time; default UTC +0 timezone

Note :

In the case of TCP long connections, EdgeOne will periodically record logs and record the last log when the connection ends. You can determine whether the connection is disconnected by whether the `DisconnetReason` field is empty; you can also use the `SessionID` to identify the connection, and logs with the same `SessionID` record the behavior of the same connection.

Offline Logs


Last updated : 2024-05-14 14:23:53


Edge Access Logs

Access logs are collected on an hourly basis and stored for 30 days. You can download the logs as need during the retention period.

Directions


1. Log in to the [EdgeOne console](#). Click **Log Service** > **Offline Logs** on the left sidebar.
2. On the page that displays, select a site or the log file of a subdomain name. You can also filter logs by time.


Offline logs 




- Provide node access log data of Site Acceleration subdomains, for data fields and more instructions, please check the [documentation](#)
- Because offline logs need to packages logs from all EdgeOne nodes, there is a certain delay to provide download links. Logs download links will be packaged (according to hourly granularity)

Today

2022-05-12 00:00:00 ~ 2022-05-12 09:41:00 

Site acceleration 

All 

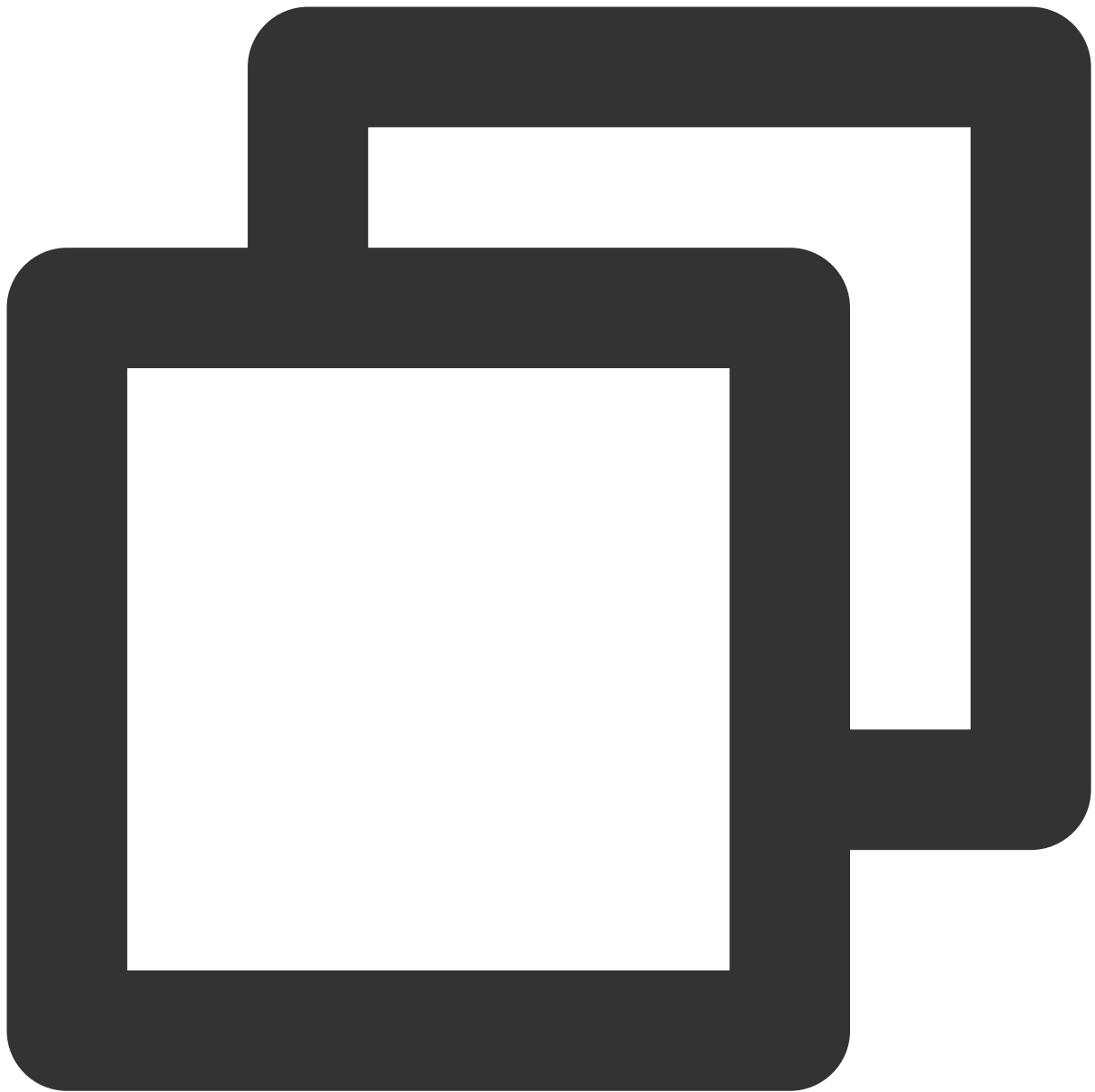
UTC+

3. Click **Download** in the **Operation** column.

Important

The access logs are collected on an hourly basis. If the selected domain name is not requested during the period, no logs are generated.

The access logs are compressed to a .gz file. Due to defects of the MacOS directory system, the .gz file may failed to be decompressed on MacOS by double-clicking it. In this case, you can run the following Terminal commands:



```
gunzip {your_file_name}.log.gz
```

EdgeOne nodes are distributed over the globe. To synchronize the time across time zones, logs are stored and queried in UTC+00:00 by default.

Generally, it takes around 30 hours to generate log data as it is collected from all EdgeOne nodes. The log data will be complete within 24 hours after being generated.

Field Description

Logs are stored in JSON format by default. The log fields are described as follows:

When a field is not specified:

For a string field, the field value is set to `-` if the field has no data.

For an integer field, the field value is set to `-1` if the field has no data.

Site acceleration logs

Name	Data type	Description
RequestID	String	Unique ID of the client request
ClientIP	String	Client IP
ClientRegion	String	Country/Region of the client IP. Format: ISO-3166 alpha-2
ClientState	String	The client IP parses out the country's lower-level administrative divisions. Currently only data within mainland China is supported. Format: ISO 3166-2 .
ClientISP	String	ISP information resolved from client IP. Data within mainland China is recorded under the ISP's Chinese name. Global Availability Zones (excluding mainland China) data is recorded as Autonomous System Numbers (ASN) .
RequestTime	String	The time that the client initiates a request, which is record in UTC +00:00 and defined in the ISO-8601 standard.
RequestStatus	int	Status of the client request. Values: <code>0</code> (not completed), <code>1</code> (completed successfully), <code>2</code> (completed abnormally)
RequestHost	String	Host of the client request
RequestBytes	int	Size of the client request, in bytes
RequestMethod	String	The HTTP method used by the client
RequestUrl	String	The URL for the client request
RequestUrlQueryString	String	The query string contained in the request URL
RequestUA	String	The User-Agent sent by the client
RequestRange	String	The Range parameter sent by the client
RequestReferer	String	The Referer parameter sent by the client

RequestProtocol	String	The application layer protocol used by the client. Values: <code>HTTP/1.0</code> , <code>HTTP/1.1</code> , <code>HTTP/2.0</code> , <code>HTTP/3</code> , <code>WebSocket</code>
RemotePort	int	The port that connects the client and node over the TCP protocol.
EdgeCacheStatus	String	Whether the client request results in a cache hit. Hit: Resources are served by node cache. Miss: Resources are served by the origin server and can be cached. Dynamic: Resources cannot be cached. other: Unidentifiable Cache Status.
EdgeResponseStatusCode	int	The status code that the node returns to the client
EdgeResponseBytes	int	Size of the response that the node returns to the client, in bytes
EdgeResponseTime	int	The amount of time elapsed between EdgeOne receiving a request from the client and waiting till the client receives the response from the server side. Unit: ms
EdgeInternalTime	int	The amount of time elapsed between EdgeOne receiving a request from the client and waiting till the client receives the response from the server side. Unit: ms
EdgeServerIP	String	IP address of the EdgeOne server, which can be resolved from the host using DNS.
EdgeServerID	String	The unique ID that identifies the EdgeOne server accessed by the client
SecurityAction	String	The rule action. Values: <code>Monitor</code> (observe), <code>JSChallenge</code> (JavaScript challenge), <code>Deny</code> (block), <code>Allow</code> (allow), <code>BlockIP</code> (block the IP), <code>Redirect</code> (redirect), <code>ReturnCustomPage</code> (return the custom page), <code>ManagedChallenge</code> (implement the managed challenge)
SecurityRuleID	String	ID of the security rule used
SecurityUserNote	String	The tag defined by the user
SecurityModule	String	Security feature of the hit security rule. Values: <code>CustomRule</code> (custom rules), <code>BotManagement</code> (bot management), <code>RateLimiting</code> (preset rate limiting rules), <code>RateLimitingCustomRule</code> (custom rate limiting rules), <code>ManagedRule</code> (managed rules), <code>BotClientReputation</code> (client reputation), <code>BotBehaviorAnalysis</code> (bot intelligence), <code>RateLimitingClientFiltering</code> (client filtering)

L4 proxy logs

Name	Data type	Description
ServiceID	String	Unique ID of the L4 proxy service
ConnectTimeStamp	String	The time that the connection is established, which is recorded in UTC +0 and defined in the ISO-8601 standard.
DisconnetTimeStamp	String	The time that the connection is disconnected, which is recorded in UTC +0 and defined in the ISO-8601 standard.
DisconnetReason	String	<p>Cause of disconnection Format: [Direction: Reason]. Direction: <code>up</code> (origin)/ <code>down</code> (client) Reason:</p> <ul style="list-style-type: none"> <code>Net_exception_peer_error</code> : Read/write peer error <code>Net_exception_peer_close</code> : Connection closed by the peer <code>Create_peer_channel_exception</code> : Failed to create the channel to the next hop <code>Channel_eof_exception</code> : Channel ended. When the quest ends, the related node sends <code>channel_eof</code> to neighbor nodes. <code>Net_exception_closed</code> : Connection closed <code>Net_exception_timeout</code> : Read/write timed out
ClientRealIP	String	Real client IP
ClientRegion	String	The 2-digit country/region code of the client in the ISO-3166 alpha-2 standard.
EdgeIP	String	IP address of the EdgeOne server accessed
ForwardProtocol	String	The TCP/UDP forwarding protocol configured by the client
ForwardPort	Int	The forwarding port configured by the client
SentBytes	Int	Outbound traffic produced when the log is generated, in bytes
ReceivedBytes	Int	Outbound traffic produced when the log is generated, in bytes
LogTimeStamp	String	The time that the log is generated, which is recorded in UTC +0 and defined in the ISO-8601 standard.

Notes

The traffic/bandwidth data (in bytes) recorded in the access log field "EdgeResponseBytes" may be different from the actual billing data for the following reasons:

Only application-layer data can be recorded in access logs. During actual data transfer, the traffic generated over the network is around 5-15% more than the application-layer traffic, including the following two parts:

Consumption by TCP/IP headers: in TCP/IP-based HTTP requests, each packet has a maximum size of 1,500 bytes and includes TCP and IP headers of 40 bytes, which generate traffic during transfer but cannot be counted by the application layer. The overhead of this part is around -4%.

TCP retransmission: During normal data transfer over the network, around 3% to 10% packets are lost on the internet, and the server will retransmit the lost ones. This type of traffic cannot be counted by the application layer, which accounts for 3% to 7% of the total traffic.

When smart acceleration is enabled, the traffic/bandwidth generated when the client sends a request to the EdgeOne node incurs charges. For more details, see [Billing Overview](#).