



GATE SCANNER

Security Dome

Development API Guide

Your gate to a safe system

Revision 1.0.24

July 2022

Sasa Software (C.A.S) Ltd.
Kibbutz Sasa
M.P. Merom Hagalil 1387000
Israel

Telephone: +972-4-8679959
Fax: +972-4-6918876
Email: info@sasa-software.com
Web: www.sasa-software.com

The information contained in this document, or any addendum or revision thereof is proprietary of Sasa Software Ltd. and is subject to all relevant copyright, patent and other laws and treaties protecting intellectual property, as well as any specific agreement protecting Sasa Software Ltd. rights in the aforesaid information. Any use of this document or the information contained herein for any purposes other than those for which it was disclosed is strictly forbidden. Sasa Software Ltd. reserves the right, without prior notice or liability, to make changes in equipment design or specifications. Sasa Software Ltd. assumes no responsibility for the use thereof nor for the rights of third parties, which may be affected in any way by the use thereof.

This document may contain flaws, omissions or typesetting errors; no warranty is granted nor liability assumed in relation thereto unless specifically undertaken in Sasa Software Ltd.'s sales contract or order confirmation. Information contained herein is periodically updated and changes will be incorporated into subsequent editions. If you have encountered an error, please notify Sasa Software Ltd. All specifications are subject to change without prior notice.

© Copyright by Sasa Software Ltd., 2021. All rights reserved worldwide.

Introduction

Thank you for choosing Gate Scanner® Security Dome (GSSD) by Sasa Software. Gate Scanner® treats every incoming file and email as suspicious, performing deep threat scans, transforming files into a safe, neutralized and harmless copy, that ensures security (D-CDR = Deep Content Disarm and Reconstruction). Gate Scanner® prevents unknown and undetectable malicious code attacks, including ransomware, while maintaining full usability, functionality and visibility of the files.

This document describes the GSSD Development API. The Development API enables an integration with the GSSD and use the GSSD functionality using external applications.

This document is intended for software developers who will be using the GSSD development API. This document provides the public interfaces necessary to use the features provided by the GSSD. A functional overview and information on leveraging the interface functionality are also provided. This document assumes that the user is familiar with web programming and specially with JS, AJAX and HTTP requests.

Document History

Version	Version Date	Author	Revision & Related Version
1.0.22	July 2021	Kobi Danone	v22
1.0.24	July 2022	Kobi Danone	v24

Table of Contents

- DEVELOPMENT APIS 4**
 - GetHello 4
 - GetGlobalValues..... 5
 - PostReceiveExtAPIToken 6
 - PostExtLogin 8
 - GetTokenActive 10
 - PostLogOut 11
 - GetUserRules..... 12
 - GetDestinations..... 13
 - PostUploadFiles..... 14
 - PostEmailFiles..... 16
 - GetUploadFilesQueue 19
 - PostClearFinishedFromQueue..... 20
 - GetFileList..... 21
 - GetScanHistory..... 23
 - PostDownloadFiles..... 25
 - PostShareFiles 27
 - PostDeleteFile 30
 - PostDeleteFolder..... 31
- CODE EXAMPLE (HTML & JS)..... 32**

Development APIs

GetHello

Description

Receives the simplest acknowledge from the server. The agent sends a simple request for Hello and the server responding hello.

URL Format

https://<Base Address>/api/GSPortal/GetHello

Response

Data Type	Value
String	"HELLO FROM API SERVER"

JS Code Example

```
1.   var settings = {
2.     "url": "https://<Base Address>/api/GSPortal/GetHello",
3.     "method": "GET"
4.   };
5.   $.ajax(settings).done(function (response) {
6.     console.log(response);
7.   });
8.
9.   //Response: HELLO FROM API SERVER
```

GetGlobalValues

Description

Returns a variety of general definitions.

URL Format

https://<Base Address>/api/GSPortal/GetGlobalValues

Response

Data Type	Value
Json/Dictionary	[DefaultSafeReplyCapacityMB, MaxSendSizeMB, DefaultSafeReplyMaxFiles, DefaultSafeReplyMaxReplies, SMSAvilable, EmailAvilable, ForceLogout, SecureFileResponse, TimeoutQueue, DefaultShareDaysLimit, DefaultShareDownloadLimit, MaxShareDaysLimit, MaxShareDownloadLimit, CompactUploadProfile, CompactUploadDestination, Proxy, ProxyAddress]

JS Code Example

```
1.  var settings = {
2.    "url": "https://<Base Address>/api/GSPortal/GetGlobalValues",
3.    "method": "GET"
4.  };
5.
6.  $.ajax(settings).done(function (response) {
7.    console.log(JSON.parse(response));
8.  });
9.
10. //Response:
11. /*CompactUploadDestination: "Vault"
12. CompactUploadProfile: "Default_Alias"
13. DefaultSafeReplyCapacityMB: "50"
14. DefaultSafeReplyMaxFiles: "100"
15. DefaultSafeReplyMaxReplies: "10"
16. DefaultShareDaysLimit: "4"
17. DefaultShareDownloadLimit: "3"
18. EmailAvilable: "True"
19. ForceLogout: "False"
20. MaxSendSizeMB: "50"
21. MaxShareDaysLimit: "7"
22. MaxShareDownloadLimit: "30"
23. Proxy: "False"
24. ProxyAddress: "https://<Proxy Base Address>/"
25. SMSAvilable: "True"
26. SecureFileResponse: "True"
27. TimeoutQueue: "3" */
```

PostReceiveExtAPIToken

Description

Receive Development API key using user and password.

The API key being used to login the GSSD.

The API key remains constant till erased or manually refreshed by the GSSD administrator.

URL Format

https://<Base Address>/api/GSPortal/PostReceiveExtAPIToken

Body Parameters

Data Type	Name	Example
String	Password	"my_password"

Header Parameters

Data Type	Name	Example
String	domain	"local"
String	user	"user1"
GUID	captchaID	"123e4567-e89b-12d3-a456-426614174000" or empty if no need
String	captchaCode	"123aB" or empty if no need
String	factorCode	"123456" or 0 if no need
String	ExtApp	"DefaultDomeApplication" or your External Application Name

Response

Data Type	Value
String	"123e4567-e89b-12d3-a456-426614174123" API key

More Response Options

Type	Value	Description
Captcha	"ERROR:Wrong CaptchaGSSDKEYGSSD" + captchaAnswerBit + "GSSDKEYGSSD" + captchaAnswerID	After 3 failed tries, return base64 captcha image + GUID id.
2FA	<ol style="list-style-type: none"> TwoFactorSMS: XXXX1234 TwoFactorEmail:*****@*****.*** 	Two Factor autentcation via SMS or Email, re-send the login with the 2FA code.

JS Code Example

```
1.   var settings = {
2.     "url": "https://<Base Address>/api/GSPortal/PostReceiveExtAPIToken",
3.     "method": "POST",
4.     "headers": {
5.       "user": "user1",
6.       "domain": "local",
7.       "captchaID": "",
8.       "captchaCode": "",
9.       "factorCode": "0",
10.      "ExtApp": "DefaultDomeApplication",
11.      "Content-Type": "application/json"
12.    },
13.    "data": JSON.stringify("my_password"),
14.  };
15.
16.  $.ajax(settings).done(function (response) {
17.    console.log(response);
18.  });
19.
20.  //Response:
21.  /*123e4567-e89b-12d3-a456-426614174123*/
```

PostExtLogin

Description

Login the GSSD using the Development API key and receive a session token to use for most requests.

A session token is valid from the last time the user used the site until the site Time Out configuration.

Note: The same session token can be used for multiple requests, no need to refresh it in every request.

URL Format

https://<Base Address>/api/GSPortal/PostExtLogin

Body Parameters

Data Type	Name	Example
GUID	Development API Key	"123e4567-e89b-12d3-a456-426614174000"

Header Parameters

Data Type	Name	Example
GUID	captchaID	"123e4567-e89b-12d3-a456-426614174000" or empty if no need
String	captchaCode	"123aB" or empty if no need
String	factorCode	"123456" or 0 if no need
String	ExtApp	"DefaultDomeApplication" or your External Application Name

Response

Data Type	Value
String	"session token, profilePicPath, privilege (1 admin, 2 read admin, 3 domain admin, 4 user), timeOutPeriodMS, EmailServerAvailable, userEmail, isQuotaMode, domainIsAD, allowSecureEmail, allowSecureEmailDesign, SecureEmailCharLimit, allowShareNote, allowShareNoteDesign, ShareNoteCharLimit, DefaultShareDaysLimit, DefaultShareDownloadLimit, MaxShareDaysLimit, MaxShareDownloadLimit, isCompactUploadMode"

More Response Options

Type	Value	Description
Captcha	"ERROR:Wrong CaptchaGSSDKEYGSSD" + captchaAnswerBit + "GSSDKEYGSSD" + captchaAnswerID	After 3 failed tries, return base64 captcha image + GUID id.
2FA	3. TwoFactorSMS: XXXX1234 4. TwoFactorEmail:*****@*****.***	Two Factor autentcation via SMS or Email, re-send the login with the 2FA code.

JS Code Example

```
22.  var settings = {
23.    "url": "https://<Base Address>/api/GSPortal/PostExtLogin",
24.    "method": "POST",
25.    "headers": {
26.      "captchaID": "",
27.      "captchaCode": "",
28.      "factorCode": "0",
29.      "ExtApp": "DefaultDomeApplication",
30.      "Content-Type": "application/json"
31.    },
32.    "data": JSON.stringify("123e4567-e89b-12d3-a456-426614174000"),
33.  };
34.
35.  $.ajax(settings).done(function (response) {
36.    console.log(response);
37.  });
38.
39.  //Response:
40.  /*123e4567-e89b-12d3-a456-426614174123, images\\users\\user.jpg, 4, 1800000, True,
41.  <User Email>, True, False, True, True, 10000, True, False, 300, 4, 3, 7, 30, False*/
```

GetTokenActive

Description

Checks if a session token is still active.

URL Format

https://<Base Address>/api/GSPortal/GetTokenActive

Header Parameters

Data Type	Name	Example
GUID	token	"123e4567-e89b-12d3-a456-426614174000"

Response

Data Type	Value
String	"OK"

JS Code Example

```
1.  var settings = {
2.    "url": "https://<Base Address>/api/GSPortal/GetTokenActive",
3.    "method": "GET",
4.    "headers": {
5.      "token": "123e4567-e89b-12d3-a456-426614174123"
6.    },
7.  };
8.
9.  $.ajax(settings).done(function (response) {
10.    console.log(response);
11.  });
12.
13.  //Response: OK
```

PostLogOut

Description

Logout and close the session token .

URL Format

https://<Base Address>/api/GSPortal/PostLogOut/{onlySetAbility}

URI Parameters

Data Type	Name	Description
Boolean	OnlySetAbility	False – Close the session. True – (Admins only) release the write permission for this session.

Header Parameters

Data Type	Name	Example
GUID	token	"123e4567-e89b-12d3-a456-426614174000"

Response

Data Type	Value
String	"true"

JS Code Example

```

1.   var settings = {
2.     "url": "https://<Base Address>/api/GSPortal/PostLogOut/false",
3.     "method": "POST",
4.     "headers": {
5.       "token": "123e4567-e89b-12d3-a456-426614174123"
6.     },
7.   };
8.
9.   $.ajax(settings).done(function (response) {
10.    console.log(JSON.parse(response));
11.  });
12.
13.  //Response: true

```

GetUserRules

Description

Checks if a session token is still active.

URL Format

https://<Base Address>/api/ApiRule/GetUserRules

Header Parameters

Data Type	Name	Example
GUID	token	"123e4567-e89b-12d3-a456-426614174000"

Response

Data Type	Value
JSON/Struct	{List<string> profileDescription; List<string> profileAlias; String profile; String capacity; String capacityUsed; String scanCapacity; String scanCapacityUsed; String avialble; String MaxDaysInStorage; String MaxFilesInJob; String MaxSingleFileSize; String MaxTotalFilesSizeInJob; String FileTypeBlackList; String FileTypeWhiteList}

JS Code Example

```

1.   var settings = {
2.     "url": "https://<Base Address>/api/ApiRule/GetUserRules",
3.     "method": "GET",
4.     "headers": {
5.       "token": "123e4567-e89b-12d3-a456-426614174123"
6.     },
7.   };
8.
9.   $.ajax(settings).done(function (response) {
10.    console.log(JSON.parse(response));
11.  });
12.
13.  //Response:
14.  /*FileTypeBlackList: "Unlimited" FileTypeWhiteList: "Unlimited"
15.  MaxDaysInStorage: "Unlimited" MaxFilesInJob: "Unlimited"
16.  MaxSingleFileSize: "Unlimited" MaxTotalFilesSizeInJob: "Unlimited"
17.  avialble: "Unlimited" capacity: "Unlimited" capacityUsed: "22.3396415710449"
18.  profile: "1,22,3,-1"
19.  profileAlias: (4) ["Default_Alias", "reconstructions", "fastprofile", "Engine Bypass"]
20.  profileDescription: (4) ["Default Engine Profile", "", "", "Engine Bypass"]
21.  scanCapacity: "Unlimited" scanCapacityUsed: "2894.74430274963"*/

```

GetDestinations

Description

Get a list of the destinations.

URL Format

https://<Base Address>/api/Destinations/GetDestinations/{all}

URI Parameters

Data Type	Name	Description
Boolean	all	False – Get the destination available to the user. True – (Admins only) Get all the destinations.

Header Parameters

Data Type	Name	Description
GUID	token	"123e4567-e89b-12d3-a456-426614174000"

Response

Data Type	Value
JSON/DataTable	[Alias, AllowDelete, AllowShare, AllowUpload, DestinationType, DestinationUser, IsShared, IsUNCVault]

JS Code Example

```

1.   var settings = {
2.     "url": "https://<Base Address>/api/Destinations/GetDestinations/false",
3.     "method": "GET",
4.     "headers": {
5.       "token": "123e4567-e89b-12d3-a456-426614174123"
6.     },
7.   };
8.
9.   $.ajax(settings).done(function (response) {
10.    console.log(JSON.parse(response));
11.  });
12.
13.  //Response:
14.  /*[{Alias: "UNCDest" AllowDelete: true AllowShare: true AllowUpload: true
15.  DestinationType: "unc" DestinationUser: "Scanner" IsShared: false IsUNCVault: false},
16.  {Alias: "myFTPS" AllowDelete: true AllowShare: true AllowUpload: true
17.  DestinationType: "ftps" DestinationUser: "FTPUser" IsShared: false IsUNCVault: false}]*/

```

PostUploadFiles

Description

Upload files to the portal.

URL Format

https://<Base Address>/api/HandleFiles/PostUploadFiles

Body Parameters

Data Type	Example
FormData	File: form_data.append('file', myFiles.File) With Folder: form_data.append('file', myFiles.File, FolderPath + "/" + myFiles.File.name)

Header Parameters

Data Type	Name	Description
GUID	token	"123e4567-e89b-12d3-a456-426614174000"
String	Destination	Destination alias.
String	Profile	Profile alias
String	FilePass	File password (in case the file is locked with a password)
String	ExtApp	External application name

Response

Data Type	Value
JSON/DataTable	[File Name, File Unique ID, Engine job ID]

JS Code Example

```
1.   var form = new FormData();
2.   form.append("file", fileInput.files[0], "test/test.pdf");
3.
4.   var settings = {
5.     "url": "https://<Base Address>/api/HandleFiles/PostUploadFiles",
6.     "method": "POST",
7.     "headers": {
8.       "token": "123e4567-e89b-12d3-a456-426614174123",
9.       "Destination": "Default",
10.      "Profile": "Default_Alias",
11.      "FilePass": "",
12.      "ExtApp": "DefaultDomeApplication"
13.    },
14.    "processData": false,
15.    "mimeType": "multipart/form-data",
16.    "contentType": false,
17.    "data": form
18.  };
19.
20.  $.ajax(settings).done(function (response) {
21.    console.log(JSON.parse(response));
22.  });
23.
24.  //Response: [{"test.pdf","123e4567-e89b-12d3-a456-426614174000","89012345-e89b-12d3-
a456-426614174111"}]
```

PostEmailFiles

Description

Upload and email the files/folders to the portal.

URL Format

https://<Base Address>/api/HandleFiles/PostEmailFiles

Body Parameters

Data Type	Example
FormData	Email's Body: <code>form_data.append('ShareNote', 'EMAIL HTML BODY ')</code> File: <code>form_data.append('file', myFiles.File)</code> File with Folder: <code>form_data.append('file', myFiles.File, FolderPath + "/" + myFiles.File.name)</code>

Header Parameters

Data Type	Name	Description
GUID	token	"123e4567-e89b-12d3-a456-426614174000"
String	Destination	Vault alias.
String	Profile	Profile alias
String	FilePass	File password (in case the file is locked with a password)
String	ExtApp	External application name
Long	size	Total files' size (bytes)
Int	maxSingleSize	The size of the biggest file in the job (bytes)
Boolean	NeedShare	False – Create a share, but do not activate it. True – Create an active share (most cases).
String	Subject	The email's subject.
String	ShareEmail	Email list separate by ';'. If you prefer the copy the link and send it yourself please use "Link Copied"
Boolean	SeparateEmail	False – Send the share URL linked with the share key in one email. True – Separate two emails, one with the share link and one with the share key.
JSON/DataTable	twoFA	Force 2FA to enter the share, leave empty for avoid this feature or add dictionary include the email, phone and sms limit for each address (f.e. <code>2FAArray.push({ email: info@sasa-software.com, phone: 0544444444, smsLimit: 10})</code>);

JSON/DataTable	SMSNum	Send the share key via SMS, leave empty for avoid this feature or add dictionary include the email and phone for each address (f.e. SMSArray.push({ email: info@sasa-software.com, phone: 0544444444}));
String	encryptPass	Protect your email's body with a password.
String	fileDownloadPass	Protect the shared file with a password (the file will be downloaded as a password protected zip).
Int	DownloadLimit	Max downloads for the share (Set 0 for unlimited).
Int	DaysLimit	Max days for the share (Set 0 for unlimited).
Boolean	DownloadAlert	False – Do not receive an email once the file was downloaded. True – Receive an email once the file was downloaded.
String	Portal	Put the base address to get the summary email or leave empty.
String	ShareLink	Put the base address for the share.
Boolean	AllowReply	Allow recipient to send a secure file response.
String	ProfileID	Profile alias to use by the secure reply
String	ReplyDestination	Vault or destination alias for the secure reply.
Int	AllocateCapacity	Max capacity for the secure reply (MB).
Int	ReplyMaxFiles	Max files for the secure reply.
Int	ReplyMaxReplies	Max replies for the secure reply.

Response

Data Type	Value
JSON/DataTable	[File Name, File Unique ID, Engine job ID]

JS Code Example

```
1.   var form = new FormData();
2.   form.append("file", fileInput.files[0], "test/test.pdf");
3.   form.append("ShareNote", "Hello From GSSD");
4.
5.   var settings = {
6.     "url": "https://<Base Address>/api/HandleFiles/PostEmailFiles",
7.     "method": "POST",
8.     "headers": {
9.       "token": "123e4567-e89b-12d3-a456-426614174123",
10.      "Destination": "Default",
11.      "Profile": "Default_Alias",
12.      "FilePass": "",
13.      "ExtApp": "DefaultDomeApplication",
14.      "size": 100,
15.      "maxSingleSize": 100,
16.      "NeedShare": "true",
17.      "Subject": "TEST EMAIL",
18.      "ShareEmail": "info@sasa-software.com",
19.      "SeparateEmail": true,
20.      "twoFA": "[{"email":"info@sasa-software.com","phone":"9725444","smsLimit":"10"}]",
21.      "SMSNum": [{"email":"info@sasa-software.com","phone":"9725444"}],
22.      "encryptPass": "1234",
23.      "fileDownloadPass": "1234",
24.      "DownloadLimit": 30,
25.      "DaysLimit": 7,
26.      "DownloadAlert": true,
27.      "Portal": window.location.origin,
28.      "ShareLink": window.location.origin,
29.      "AllowReply": true,
30.      "ProfileID": "Default_Alias",
31.      "ReplyDestination": "vault",
32.      "AllocateCapacity": 50,
33.      "ReplyMaxFiles": 100,
34.      "ReplyMaxReplies": 10
35.    },
36.    "processData": false,
37.    "mimeType": "multipart/form-data",
38.    "contentType": false,
39.    "data": form
40.  };
41.
42.  $.ajax(settings).done(function (response) {
43.    console.log(JSON.parse(response));
44.  });
45.
//Response: [{"test.pdf","123e4567-e89b-12d3-a456-426614174000","89012345-e89b-12d3-
a456-426614174111"}]
```

GetUploadFilesQueue

Description

Get the files in-process (waiting, in queue, in scan, saving, finished).

URL Format

https://<Base Address>/api/HandleFiles/GetUploadFilesQueue

Header Parameters

Data Type	Name	Description
GUID	token	"123e4567-e89b-12d3-a456-426614174000"

Response

Data Type	Value
JSON/DataTable	[CreateDate, Destination, DestinationType, EndPoint, ExternalApplicationName, FileGUID, FileName, FileSize, FolderPath, JOBID, Link, OneDriveUser, ProfileID, ScanInfo, ScanStatus, UpdateDate]

JS Code Example

```

1.   var settings = {
2.     "url": "https://<Base Address>/api/HandleFiles/GetUploadFilesQueue",
3.     "method": "GET",
4.     "headers": {
5.       "token": "123e4567-e89b-12d3-a456-426614174123"
6.     },
7.   };
8.
9.   $.ajax(settings).done(function (response) {
10.    console.log(JSON.parse(response));
11.  });
12.
13.  //Response:
14.  /*[{CreateDate: "2021-03-16T09:16:52.83" Destination: "Default" DestinationType: null
15.  EndPoint: "net.tcp://192.168.1.1:15810/GSDA_API"
16.  ExternalApplicationName: "DefaultDomeApplication"
17.  FileGUID: "123e4567-e89b-12d3-a456-426614174789" FileName: "test.pdf" FileSize: 1530018
18.  FolderPath: "test" JOBID: "123e4567-e89b-12d3-a456-426614174456" Link: null
19.  OneDriveUser: "ExFFY8vUDnCrNDZHxqnnRQ==" ProfileID: 1 ScanInfo: "KEYIPKEY127.0.0.1"
20.  ScanStatus: "In Scan [28%]" UpdateDate: "2021-03-16T09:17:01.197"}]*/

```

PostClearFinishedFromQueue

Description

Clear all the finished files from the in-process table.

URL Format

https://<Base Address>/api/HandleFiles/PostClearFinishedFromQueue

Header Parameters

Data Type	Name	Description
GUID	token	"123e4567-e89b-12d3-a456-426614174000"

Response

Data Type	Value
JSON/String	"true"

JS Code Example

```
1.   var settings = {
2.     "url": "https://<Base Address>/api/HandleFiles/PostClearFinishedFromQueue",
3.     "method": "POST",
4.     "headers": {
5.       "token": "123e4567-e89b-12d3-a456-426614174123"
6.     },
7.   };
8.
9.   $.ajax(settings).done(function (response) {
10.    console.log(JSON.parse(response));
11.  });
12.
13.  //Response: true
```

GetFileList

Description

Get a list of files and folders from a specific destination/path.

URL Format

`https://<Base Address>/api/HandleFiles/GetFileList/{secPeriod}/{limit}`

URI Parameters

Data Type	Name	Description
Int	secPeriod	Get only files created in the last X seconds (0 for limitless).
Int	limit	Get top X files (0 for limitless).

Header Parameters

Data Type	Name	Description
GUID	token	"123e4567-e89b-12d3-a456-426614174000"
String	vault	Vault alias.
String	folder	Sub folder name (empty for root).
String	searchKey	Search by file name (empty for all).

Response

Data Type	Value	Description
JSON/Struct	{ String[][] Files, Int FilesCount, Int FolderCount }	Folder: [folder name, created date, keygssdfolderkey]. File: [file name, created date, size, identifier, external application]

JS Code Example

```
1.   var settings = {
2.     "url": "https://<Base Address>/api/HandleFiles/GetFileList/0/0",
3.     "method": "GET",
4.     "headers": {
5.       "token": "123e4567-e89b-12d3-a456-426614174123",
6.       "vault": "vault",
7.       "folder": "test",
8.       "searchKey": ""
9.     },
10.  };
11.
12.  $.ajax(settings).done(function (response) {
13.    console.log(JSON.parse(response));
14.  });
15.
16.  //Response: ["test.pdf", "2021-03-16 09:17:16", "1530068", "123e4567-e89b-12d3-a456-426614174789", ""]
```

GetScanHistory

Description

Get all the information regarding the history of the scans in the system.

URL Format

https://<Base Address>/api/ScanHistory/GetScanHistory

Header Parameters

Data Type	Name	Exmple	Description
GUID	token	123e4567-e89b-12d3-a456-426614174000	
DateTime	fromDate	'YYYY-MM-DD HH:mm:ss'	
DateTime	toDate	'YYYY-MM-DD HH:mm:ss'	
Int	limit	10	
String	searchKey	"test.doc"	leave empty for all the results
String	column	"File Name"	leave empty for to search all columns

Response

Data Type	Value	Description
Json/DataTable[]	[Alias (destination), ClientHostName, ConcatErrorMessage, Saved, ConcatLogMessage, CreatedTimeStamp, DestinationFileType, DomainName, DurationSeconds, Endpoint, ExternalApplicationName, FileGuid, FileHASH, FileName, FileNewHASH, FileNewSizeBytes, FileSizeBytes, IPAddress, JOBID, OutputFullFilePath, Profileid, Results, Saved, ScanTime, SourceFileType, SourceTrueFileType, TimeInScan, UserName, isInfected]	Table – scan information Note: admin receives the information regarding all users, while normal user receives the information only regarding himself.
	[AVGTimeScan, TotalFiles, TotalsizeKB, cntClean, cntFixed, cntInfected, cntdeleted]	Table1 – summary

JS Code Example

```
1.  var settings = {
2.    "url": "https://<Base Address>/api/ScanHistory/GetScanHistory",
3.    "method": "GET",
4.    "headers": {
5.      "token": "123e4567-e89b-12d3-a456-426614174123",
6.      "fromDate": "2021-03-16 09:00:00",
7.      "toDate": "2021-03-16 10:00:00",
8.      "limit": "10",
9.      "searchKey": "",
10.     "column": ""
11.   },
12. };
13.
14. $.ajax(settings).done(function (response) {
15.   console.log(JSON.parse(response));
16. });
17.
18. //Response:
19. /*Table: Array(1) 0:
20.   [Alias: "Default" ClientHostName: "XXXX" ConcatErrorMessage: ""
21.   ConcatLogMessage: "Passed Description : <OK> [OK]"
22.   CreatedTimeStamp: "Mar 16 2021 9:17AM" DestinationFileType: "" DomainName: "XXX"
23.   DurationSeconds: 0 Endpoint: "net.tcp://192.168.1.1:15810/GSDA_API"
24.   ExternalApplicationName: "DefaultDomeApplication"
25.   FileGuid: "123e4567-e89b-12d3-a456-426614174789"
26.   FileHASH: "XXXX" FileName: "test.pdf" FileNewHASH: "" FileNewSizeBytes: 1530018
27.   FileSizeBytes: 1530018 IPAddress: "127.0.0.1"
28.   JOBID: "123e4567-e89b-12d3-a456-426614174456" OutputFullFilePath: "XXXX\XXXXXX.pdf"
29.   Profileid: 1 Results: "Ok" Saved: true ScanTime: "2021-03-16T09:16:58"
30.   SourceFileType: "pdf" SourceTrueFileType: "pdf" TimeInScan: 0 Username: "XXX"
31.   isInfected: false] ,
32. Table1: Array(1) 0:
33.   AVGTimeScan: 33 TotalFiles: 1 TotalsizeKB: 1494 cntClean: 1 cntFixed: 0
34.   cntInfected: 0 cntdeleted: 0]*/
```


PostDownloadFiles

Description

Download files/folder from the portal.

URL Format

https://<Base Address>/api/HandleFiles/PostDownloadFiles

Body Parameters

Data Type	Name	Example
GUID	File Identifier	<ol style="list-style-type: none"> “123e4567-e89b-12d3-a456-426614174000” For folder, send the folder path + name. For Zip with multiple files, separate the GUIDs/folders with *gssd*: “123e4567-e89b-12d3-a456-426614174000*gssd*123e4567-e89b-12d3-a456-426614174001*gssd* folder path + name”

Header Parameters

Data Type	Name	Description
GUID	token	“123e4567-e89b-12d3-a456-426614174000”
String	vault	Vault alias.

Response

Data Type	Value
StreamContent	Requested file

JS Code Example

```
1.   var settings = {
2.     "url": "https://<Base Address>/api/HandleFiles/PostDownloadFiles",
3.     "method": "POST",
4.     "headers": {
5.       "token": "123e4567-e89b-12d3-a456-426614174123",
6.       "vault": "vault",
7.       "Content-Type": "application/json"
8.     },
9.     "data": JSON.stringify("test/123e4567-e89b-12d3-a456-426614174789"),
10.  };
11.
12.  $.ajax(settings).done(function (response) {
13.    var blob = new Blob([response], { type: "application/octet-stream" });
14.    var link = window.document.createElement('a');
15.    link.href = window.URL.createObjectURL(blob);
16.    link.download = "test.pdf";
17.    document.body.appendChild(link);
18.    link.click();
19.    URL.revokeObjectURL(link.href);
20.    document.body.removeChild(link);
21.  });
```

PostShareFiles

Description

Share files/folder from the portal.

URL Format

`https://<Base Address>/api/Share/PostShareFiles/{separate}/{sendEmail}/{isActiveShare}`

URI Parameters

Data Type	Name	Description
Boolean	separate	False – Send the share URL linked with the share key in one email. True – Separate two emails, one with the share link and one with the share key.
Boolean	sendEmail	False – In case you prefer to send the share link yourself. True – Send the share via the GSSD.
Boolean	isActiveShare	False – Create a share, but do not activate it. True – Create an active share (most cases).

Body Parameters

Data Type	Name	Example
GUID	File Identifier	<ol style="list-style-type: none"> “123e4567-e89b-12d3-a456-426614174000” For folder, send the folder path + name. For multiple files, separate the GUIDs/folders with *gssd*: “123e4567-e89b-12d3-a456-426614174000*gssd*123e4567-e89b-12d3-a456-426614174001*gssd* folder path + name”

Header Parameters

Data Type	Name	Description
GUID	token	“123e4567-e89b-12d3-a456-426614174000”
String	vault	Vault alias.
String	Address	Email list separate by ‘;’. If you prefer the copy the link and send it yourself please use "Link Copied"
JSON/DataTable	twoFA	Force 2FA to enter the share, leave empty for avoid this feature or add dictionary include the email, phone and sms limit for each address (f.e. 2FAArray.push({ email: info@sasa-software.com, phone: 05444444444, smsLimit: 10}));

JSON/DataTable	SMSNum	Send the share key via SMS, leave empty for avoid this feature or add dictionary include the email and phone for each address (f.e. SMSArray.push({ email: info@sasa-software.com, phone: 0544444444}));
String	rootDir	The root folder of the files (leave empty in most cases).
String	encryptPass	Protect your email's body with a password.
String	fileDownloadPass	Protect the shared file with a password (the file will be downloaded as a password protected zip).
Int	DownloadLimit	Max downloads for the share (Set 0 for unlimited).
Int	DaysLimit	Max days for the share (Set 0 for unlimited).
Boolean	DownloadAlert	False – Do not receive an email once the file was downloaded. True – Receive an email once the file was downloaded.
Boolean	isDynamic	False – For static shares. True – For dynamic share a folder.
String	Link	Put the base address for the share.
String	Note	The email's body.
Boolean	AllowReply	Allow recipient to send a secure file response.
String	ProfileID	Profile alias to use by the secure reply
String	ReplyDestination	Vault or destination alias for the secure reply.
Int	AllocateCapacity	Max capacity for the secure reply (MB).
Int	ReplyMaxFiles	Max files for the secure reply.
Int	ReplyMaxReplies	Max replies for the secure reply.

Response

Data Type	Value
JSON/DataTable	[Address, Share Key, Message ID]

JS Code Example

```
1.   var settings = {
2.   "url": "https://<Base Address>/api/Share/PostShareFiles/true/true/true",
46.  "method": "POST",
47.  "headers": {
48.    "token": "123e4567-e89b-12d3-a456-426614174123",
49.    "vault": "vault",
50.    "Address": "info@sasa-software.com",
51.    "twoFA": "[{"email":"info@sasa-software.com","phone":"9725444","smsLimit":"10"}]",
52.    "SMSNum": [{"email":"info@sasa-software.com","phone":"9725444"}],
53.    "rootDir": "",
54.    "encryptPass": "1234",
55.    "fileDownloadPass": "1234",
56.    "DownloadLimit": 30,
57.    "DaysLimit": 7,
58.    "ProfileID": "Default_Alias",
59.    "ReplyDestination": "vault",
60.    "DownloadAlert": true,
61.    "isDynamic": false,
62.    "Link": window.location.origin,
63.    "Note": "Hello from GSSD",
64.    "AllowReply": true,
65.    "AllocateCapacity": 50,
66.    "ReplyMaxFiles": 100,
67.    "ReplyMaxReplies": 10
68.  },
69.  "data": JSON.stringify("123e4567-e89b-12d3-a456-426614174789"),
70. };
71.
72. $.ajax(settings).done(function (response) {
73.   console.log(JSON.parse(response));
74. });
75.
76. //Response: "[["info@sasa-software.com","F54B1ED","2807"]]"
```

PostDeleteFile

Description

Delete files from the vault.

URL Format

https://<Base Address>/api/HandleFiles/PostDeleteFile

Body Parameters

Data Type	Name	Example
GUID	File Identifier	<ol style="list-style-type: none"> “123e4567-e89b-12d3-a456-426614174000” For multiple files, separate the GUIDs with *gssd*: “123e4567-e89b-12d3-a456-426614174000*gssd*123e4567-e89b-12d3-a456-426614174001”

Header Parameters

Data Type	Name	Description
GUID	token	“123e4567-e89b-12d3-a456-426614174000”
String	vault	Vault alias.

Response

Data Type	Value
String	“true”

JS Code Example

```

3.   var settings = {
4.     "url": "https://<Base Address>/api/HandleFiles/PostDeleteFile",
5.     "method": "POST",
6.     "headers": {
7.       "token": "123e4567-e89b-12d3-a456-426614174123",
8.       "vault": "vault",
9.       "Content-Type": "application/json"
10.    },
11.    "data": JSON.stringify("test/123e4567-e89b-12d3-a456-426614174789"),
12.  };
13.
14.  $.ajax(settings).done(function (response) {
15.    console.log(JSON.parse(response));
16.  });
14.
15.  //Response: true

```

PostDeleteFolder

Description

Delete folders from the vault.

URL Format

https://<Base Address>/api/HandleFiles/PostDeleteFolder

Body Parameters

Data Type	Name	Example
String	Folder Identifier	<ol style="list-style-type: none"> 1. "folder path + name" 2. For multiple folders, separate the names with *gssd*: "folder path + name*gssd*folder path + name"

Header Parameters

Data Type	Name	Description
GUID	token	"123e4567-e89b-12d3-a456-426614174000"
String	vault	Vault alias.

Response

Data Type	Value
String	"true"

JS Code Example

```

1.   var settings = {
2.     "url": "https://<Base Address>/api/HandleFiles/PostDeleteFolder",
3.     "method": "POST",
4.     "headers": {
5.       "token": "123e4567-e89b-12d3-a456-426614174123",
6.       "vault": "vault",
7.       "Content-Type": "application/json"
8.     },
9.     "data": JSON.stringify("test"),
10.  };
11.
12.  $.ajax(settings).done(function (response) {
13.    console.log(JSON.parse(response));
14.  });
15.
16.  //Response: true

```

Code Example (HTML & JS)

IMPORTANT!

You can also find the example under [ROOT]/portal/GSSD_API_EXAMPLE.html.

```
<!DOCTYPE html>
<html>
<body>
  <!--<script
  src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
  </script>-->
  <script src="../../vendors/jquery/dist/jquery.min.js"></script>

  <div>Domain: <input type="text" id="inpDomain" /></div>
  <div>User: <input type="text" id="inpUser" /></div>
  <div>Password: <input type="text" id="inpPass" /></div>
  <div><input type="file" id="FileInput" /></div>

  <script>
    var BaseAddress = window.location.origin + "/api/";
    var async = false;
    var lastScanStatus;
    var HistoryFromDate = "2022-07-01 00:00:00";
    var HistoryToDate = "2030-01-01 00:00:00";

    $('#FileInput').change(function () {

      var domain = $('#inpDomain')[0].value;
      var user = $('#inpUser')[0].value;
      var password = $('#inpPass')[0].value;

      //START MAIN PROCESS
      console.log("Start: " + new Date());
      PrintHello();
      PrintGlobalValues();

      var ApiKey = PostReceiveExtAPIToken(domain, user, password);
      var SessionToken = PostExtLogin(ApiKey);
      PrintTokenActive(SessionToken);
      var ProfileAliasToUse = GetUserRules(SessionToken);
      var DestinationAliasToUse = GetDestination(SessionToken);
      console.log("Uploading File...");

      alert("START UPLOADING")
      PostUploadFiles(SessionToken, DestinationAliasToUse,
        ProfileAliasToUse, "", "DefaultDomeApplication");

      var FileGUIDandName = PrintUploadFilesQueueUntillDone(SessionToken);
      var FileGUID = FileGUIDandName.split(',')[0];
      var FileName = FileGUIDandName.split(',')[1];
      PostClearFinishedFromQueue(SessionToken);

      alert("START DOWNLOADING")
```



```

PostDownloadFiles(SessionToken, DestinationAliasToUse, FileGUID, FileName);
PrintFileList(SessionToken, 0, 0, DestinationAliasToUse, "", "");
PrintScanHistory(SessionToken, HistoryFromDate, HistoryToDate,
    10, "", "");
PostDeleteFile(SessionToken, DestinationAliasToUse, FileGUID);
PostLogout(SessionToken);

alert("FINISHED")
console.log("Done: " + new Date());

});

//FUNCTIONS
function PrintHello() {
    var settings = {
        "url": BaseAddress + "GSPortal/GetHello",
        "method": "GET",
        "async": async,
    };

    $.ajax(settings).done(function (response) {
        console.log(response);
    });
}

function PrintGlobalValues() {
    var settings = {
        "url": BaseAddress + "GSPortal/GetGlobalValues",
        "method": "GET",
        "async": async,
    };

    $.ajax(settings).done(function (response) {
        console.log("Global Settings:");
        console.log(JSON.parse(response));
    });
}

function PostReceiveExtAPIToken(domain, user, password) {
    var settings = {
        "url": BaseAddress + "GSPortal/PostReceiveExtAPIToken",
        "method": "POST",
        "async": async,
        "headers": {
            "domain": domain,
            "user": user,
            "captchaID": "",
            "captchaCode": "",
            "factorCode": "0",
            "ExtApp": "DefaultDomeApplication",
            "Content-Type": "application/json"
        },
        "data": JSON.stringify(password),
    };

    $.ajax(settings).done(function (response) {
        console.log("ApiKey: " + response);
        ApiKey = response;
    });
}

```

```

    return ApiKey;
}

function PostExtLogin(ApiKey) {
    var settings = {
        "url": BaseAddress + "GSPortal/PostExtLogin",
        "method": "POST",
        "async": async,
        "headers": {
            "captchaID": "",
            "captchaCode": "",
            "factorCode": "0",
            "ExtApp": "DefaultDomeApplication",
            "Content-Type": "application/json"
        },
        "data": JSON.stringify(ApiKey),
    };

    var SessionToken;

    $.ajax(settings).done(function (response) {
        SessionToken = response.split(',')[0];
        console.log("Session Token: " + SessionToken);
    });

    return SessionToken;
}

function PrintTokenActive(SessionToken) {
    var settings = {
        "url": BaseAddress + "GSPortal/GetTokenActive",
        "method": "GET",
        "async": async,
        "headers": {
            "token": SessionToken
        },
    };

    $.ajax(settings).done(function (response) {
        console.log("Session Token Is Active: " + response);
    });
}

function PostLogOut(SessionToken) {
    var settings = {
        "url": BaseAddress + "GSPortal/PostLogOut/false",
        "method": "POST",
        "async": async,
        "headers": {
            "token": SessionToken
        },
    };

    $.ajax(settings).done(function (response) {
        console.log("Log Out: " + JSON.parse(response));
    });
}

function GetUserRules(SessionToken) {
    var settings = {

```

```

        "url": BaseAddress + "ApiRule/GetUserRules",
        "method": "GET",
        "async": async,
        "headers": {
            "token": SessionToken
        },
    };

    var ProfileAliasToUse;

    $.ajax(settings).done(function (response) {
        console.log("User Rules:");
        console.log(JSON.parse(response));
        ProfileAliasToUse = JSON.parse(response).profileAlias[0];
    });

    return ProfileAliasToUse;
}
function GetDestination(SessionToken) {
    var settings = {
        "url": BaseAddress + "Destinations/GetDestinations/false",
        "method": "GET",
        "async": async,
        "headers": {
            "token": SessionToken
        },
    };

    var DestinationAliasToUse = "No vault available.";

    $.ajax(settings).done(function (response) {
        console.log("User Destinations:");
        var destinations = JSON.parse(response);
        console.log(JSON.parse(response));

        for (let step = 0; step < destinations.length; step++)
            if (destinations[step].DestinationType == "vault"
                && destinations[step].AllowUpload
                && !destinations[step].RequiredManagerApproval) {
                DestinationAliasToUse = destinations[step].Alias;
                break;
            }
    });

    return DestinationAliasToUse;
}

function PostUploadFiles(SessionToken, Destination, Profile, FilePass,
    ExtApp) {
    var form = new FormData();
    form.append("file", FileInput.files[0]);

    var settings = {
        "url": BaseAddress + "HandleFiles/PostUploadFiles",
        "method": "POST",
        "async": async,
        "headers": {
            "token": SessionToken,
            "Destination": Destination,
            "Profile": Profile,

```

```

        "FilePass": FilePass,
        "ExtApp": ExtApp
    },
    "processData": false,
    "mimeType": "multipart/form-data",
    "contentType": false,
    "data": form
};

$.ajax(settings).done(function (response) {
    console.log( JSON.parse(response));
});
}

function PrintUploadFilesQueueUntillDone(SessionToken) {
    var settings = {
        "url": BaseAddress + "HandleFiles/GetUploadFilesQueue",
        "method": "GET",
        "async": async,
        "headers": {
            "token": SessionToken
        }
    },
};

var FileGUIDandName;

$.ajax(settings).done(function (response) {
    if (lastScanStatus != JSON.parse(response)[0].ScanStatus)
        console.log("Scan Status: " +
            JSON.parse(response)[0].ScanStatus);

    lastScanStatus = JSON.parse(response)[0].ScanStatus;

    if (lastScanStatus.indexOf("Finished") == -1) {
        //delay
        var d1 = new Date(); var d2 = new Date();
        while (d2.valueOf() < d1.valueOf() + 3000) d2 = new Date();

        PrintUploadFilesQueueUntillDone(SessionToken);
    }

    FileGUIDandName = JSON.parse(response)[0].FileGUID + "," +
        JSON.parse(response)[0].FileName;
});

return FileGUIDandName;
}

function PostClearFinishedFromQueue(SessionToken) {
    var settings = {
        "url": BaseAddress + "HandleFiles/PostClearFinishedFromQueue",
        "method": "POST",
        "async": async,
        "headers": {
            "token": SessionToken
        }
    },
};

$.ajax(settings).done(function (response) {
    console.log("ClearFinishedFromQueue: " + JSON.parse(response));
});

```

```

    });
}

function PrintFileList(SessionToken, secPeriod, limit, vault, folder,
    searchKey) {
    var settings = {
        "url": BaseAddress + "HandleFiles/GetFileList/" + secPeriod + "/"
            + limit,
        "method": "GET",
        "async": async,
        "headers": {
            "token": SessionToken,
            "vault": vault,
            "folder": folder,
            "searchKey": searchKey
        },
    },
};

$.ajax(settings).done(function (response) {
    console.log("File List From The Vault:");
    console.log(JSON.parse(response));
});
}

function PrintScanHistory(SessionToken, fromDate, toDate, limit,
    searchKey, column) {
    var settings = {
        "url": BaseAddress + "ScanHistory/GetScanHistory",
        "method": "GET",
        "async": async,
        "headers": {
            "token": SessionToken,
            "fromDate": fromDate,
            "toDate": toDate,
            "limit": limit,
            "searchKey": searchKey,
            "column": column
        },
    },
};

$.ajax(settings).done(function (response) {
    console.log("Scan History:");
    console.log(JSON.parse(response));
});
}

function PostDownloadFiles(SessionToken, vault, FileGUID, FileName) {
    $.ajax({
        url: BaseAddress + "HandleFiles/PostDownloadFiles",
        cache: false,
        type: 'POST',
        headers: {
            token: SessionToken,
            vault: vault
        },
        data: JSON.stringify(FileGUID),
        contentType: "application/json",
        xhr: function () {
            var xhr = new XMLHttpRequest();
            xhr.open("POST", BaseAddress + "HandleFiles/PostDownloadFiles");

```

```

        xhr.responseType = 'blob';
        xhr.addEventListener("progress", function (evt) {
            if (evt.lengthComputable) {
                var percentComplete = ((evt.loaded / evt.total) * 100);
                console.log("Download Progress: " +
parseInt(percentComplete) + "%");
            }
        }, false);

        return xhr;
    },
    crossDomain: true,
    accept: "Application/octet-stream",
    responseType: "blob",
    async: true,
    success: function (response) {
        var blob = new Blob([response], { type: "application/octet-stream" });

        if (window.navigator && window.navigator.msSaveOrOpenBlob) {
            window.navigator.msSaveOrOpenBlob(blob, FileName);
        }
        else {
            var link = window.document.createElement('a');
            link.href = window.URL.createObjectURL(blob);
            link.download = FileName;
            document.body.appendChild(link);
            link.click();
            URL.revokeObjectURL(link.href);
            document.body.removeChild(link);
        }
    }
});

function PostDeleteFile(SessionToken, vault, FileGUID) {
    var settings = {
        "url": BaseAddress + "HandleFiles/PostDeleteFile",
        "method": "POST",
        "async": async,
        "headers": {
            "token": SessionToken,
            "vault": vault,
            "Content-Type": "application/json"
        },
        "data": JSON.stringify(FileGUID),
    };

    $.ajax(settings).done(function (response) {
        console.log("File Deleted: " + JSON.parse(response));
    });
}

</script>
</body>
</html>

```

END OF DOCUMENT