StrongKey FIDO2 Server 4.3.1 Release Notes

1.1—2020 Q4

1.1.1—Fixes and Features in 4.3.1

#	Explanation
DEV-1914	**Add an administration servlet for policy and configurations.** Add a new administration servlet to handle Create/Read/Update/Delete (CRUD) operations for policy and configurations. **Fix:** A new administration servlet (REST) has been added to perform CRUD operations on the policies and configurations.

DEV-1913	**Change API input to only accept JSON.**
	In the older builds, the API accepted a JSON input with the sub-JSONs being converted into strings. Update this to only have JSON objects and not JSON strings.
	
	Fix: The code has been updated and now all the sub elements in the input that would have been *jsonobjects* are not converted to strings.
	The variable `metadata` is now `strongkeyMetadata`, and the variable `response` is now `publicKeyCredential`.
	
	Example old input:
	
	`{`
	`"svcinfo": {`
	`"did": 1,`
	`"protocol": "FIDO2_0",`
	`"authtype": "PASSWORD",`
	`"svcusername": "svcfidouser",`
	`"svcpassword": "Abcd1234!"`
	`},`
	`"payload": {`
	`"metadata": "{\"version\": \"1.0\",\"create_location\": \"Sunnyvale, CA\",\"username\": \"johndoe\",\"origin\": \"https://demo4.strongkey.com\"}",`
	`"response": "{\"id\":\"79U433x2h\",\"rawId\": \"79U433x2h\",\"response\": {\"attestationObject\": \"o2N\",\"clientDataJSON\": \"ey\"},\"type\": \"public-key\"}"`
	`}`
	`}`

#	Explanation

Example new input:


```
{
"svcinfo": {
"did": 1,
"protocol": "FIDO2_0",
"authtype": "PASSWORD",
"svcusername": "svcfidouser",
"svcpassword": "Abcd1234!"
},
"payload": {
"strongkeyMetadata": {
"version": "1.0",
"create_location": "Sunnyvale, CA",
"username": "test123",
"origin": "https://fidoscatest.strongkey.com"
},
"publicKeyCredential": {
"id": "LGCun1USkhhpoB-p--
6cfowLmgbjweyvL0JSoKPqm8sYETPGv8yhkAx7RAJVQL4f4zvPpcuX7iB3VgpRN1Ccwl26DgBH
xki0bQecrEektnNOMrBmh_CCf04bGCusJuojUUXj9FjrDHM9DDzfNTbP4o7KtyoPAvvKYnXWOx
AArhPYfXoMCcnuyuZG52gwW_5VBwLmQLlRCpFTMR2H0Lq_x9Jl_dJQkMiqHz_ySLASCzg",
"rawId": "LGCun1USkhhpoB-p--
6cfowLmgbjweyvL0JSoKPqm8sYETPGv8yhkAx7RAJVQL4f4zvPpcuX7iB3VgpRN1Ccwl26DgBH
xki0bQecrEektnNOMrBmh_CCf04bGCusJuojUUXj9FjrDHM9DDzfNTbP4o7KtyoPAvvKYnXWOx
AArhPYfXoMCcnuyuZG52gwW_5VBwLmQLlRCpFTMR2H0Lq_x9Jl_dJQkMiqHz_ySLASCzg",
"response": {
"attestationObject":
"o2NmbXRmcGFja2VkZ2F0dFN0bXSjY2FsZyZjc2lnWEYwRAIgbCciJWRp5qK63yMoQdxsUqT
WUkPWsAL7E6dQHwuljikCIFtWmlRO6wLJTF60AZhW9ZJum07o_HaeFqFtZ719K-
qjY3g1Y4FZAeQwggHgMIIBg6ADAgECAgRsK1jyMAwGCCqGSM49BAMCBQAwZDELMAkGA1UEBh
MCVVMxFzAVBgNVBAoTDlN0cm9uZ0F1dGggSW5jMSIwIAYDVQQLExlBdXRoZW50aWNhdG9yIE
F0dGVzdGF0aW9uMRgwFgYDVQQDDA9BdHRlc3RhdGlvbl9LZXkwHhcNMTkwNzE4MTcxMTI3Wh
cNMjkwNzE1MTcxMTI3WjBkMQswCQYDVQQGEwJVUzEXMBUGA1UEChMOU3Ryb25nQXV0aCBJbm
MxIjAgBgNVBAsTGUF1dGhlbnRpY2F0b3IgQXR0ZXN0YXRpb24xGDAWBgNVBAMMD0F0dGVzdG
F0aW9uX0tleTBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABDH0hj6698S9n0dolffJpiY6pI
hhDFLc6LcR3uLDjNcZHkLhForI5B4i7WErZAKCirbXQqPA0VTdOmyoAxktmYWjITAfMB0GA1
UdDgQWBBQ0QtDgcEONNbOP0TRnvX0TgR4vGDAMBggqhkjOPQQDAgUAA0kAMEYCIQDtFtHY0K
3IxDCLIYY4APLysMeM0U-
Vkbwin2Sv2IAqKwIhAKLdw0AWNMvTf6yTsLWWKPslDqY7uuF90Mx8NdKN6DC5aGF1dGhEYXR
hWQE0WnTBrV2dI2nYtpWAzOrzVHMkwfEC46dxHD4U1RP9KKNFAAAAAAAAAAAAAAAAAAAAAAA
AAAAAsCxgrp9VEpIYaaAfqfvunH6MC5oG48Hsry9CUqCj6pvLGBEzxr_MoZAMe0QCVUC-H-
M7z6XLl-
4gd1YKUTdQnMJdug4AR8ZItG0HnKxHpLZzTjKwZofwgn9OGxgrrCbqI1FF4_RY6wxzPQw83z
U2z-KOyrcqDwL7ymJ11jsQAK4T2H16DAnJ7srmRudoMFv-
VQcC5kC5UQqRUzEdh9C6v8fSZf3SUJDIqh8_8kiwEgs4pQECAyYgASFYIBatK7Qi99KplJ9a
g_m1qSD73FsGvQfxkQAoOvfPpS5dIlggAnkPDx-
BfcYy51Qr3tI_vLdO3qnD4Zi6gltfQNuwEwA",
"clientDataJSON":
"eyJ0eXBlIjoid2ViYXV0aG4uY3JlYXRlIiwiY2hhbGxlbmdlIjoiaTc3ZHNkZmVraUVtUkFkQ
VY0dzN6dyIsIm9yaWdpbiI6Imh0dHBzOi8vZmlkb3NjYXRlc3Quc3Ryb25na2V5LmNvbSJ9"
},
"type": "public-key"
}
}
}
```

#	Explanation
DEV-1912	**Add Docker README.** Add a README file to the Docker folder to explain the files. **Fix:** A new README has been created that will be part of the Docker folder.
DEV-1911	**Update the command-line interface (CLI) client for the new web services.** Update the command-line interface (*skfsclient*) to add Create/Read/Update/Delete (CRUD) operations for policies and the new configuration table. **Fix:** SKFS client has now been updated to have the requested operations.

DEV-1910	**FIDO policy: allow AAGUIDs.**
	Add a new entry to the FIDO policy to restrict authenticators based on AAGUID.
	
	Fix: A new entry has been added in the policy JSON which can restrict authenticators based on AAGUIDs. By default it allows all, as shown below:
	
	`"allowedAaguids": ["all"]`
	
	To restrict specific AAGUIDs, just replace *all* with a comma-separated list:
	
	`"allowedAaguids": ["6d44ba9b-f6ec-2e49-b930-0c8fe920cb73"]`
	
	or:
	
	`"allowedAaguids": ["6d44ba9b-f6ec-2e49-b930-0c8fe920cb73","8876631b-d4a0-427f-5773-0ec71c9e0279"]`

DEV-1909	**Update FIDO policy JSON.**
	Update the current FIDO policy JSON to add more metadata and an additional item which can restrict authenticators based on AAGUID.
	
	Fix: The FIDO policy JSON structure has been updated to add more metadata and reorganize them.
	
	Example old JSON:
	
	`{`
	`"storeSignatures": false,`
	`"extensions": {`
	`"example.extension": true`
	`},`
	`"userSettings": true,`
	`"cryptography": {`
	`"attestation_formats": ["fido-u2f", "packed", "tpm", "android-key", "android-safetynet", "none"],`
	`"elliptic_curves": ["secp256r1", "secp384r1", "secp521r1", "curve25519"],`
	`"allowed_rsa_signatures": ["rsassa-pkcs1-v1_5-sha1", "rsassa-pkcs1-v1_5-sha256", "rsassa-pkcs1-v1_5-sha384", "rsassa-pkcs1-v1_5-sha512",`

#	Explanation

```
"rsassa-pss-sha256", "rsassa-pss-sha384", "rsassa-pss-sha512"],
"allowed_ec_signatures": ["ecdsa-p256-sha256", "ecdsa-p384-sha384",
"ecdsa-p521-sha512", "eddsa", "ecdsa-p256k-sha256"],
"attestation_types": ["basic", "self", "attca", "ecdaa", "none"]
},
"registration": {
"attestation": ["none", "indirect", "direct"],
"displayName": "required",
"authenticatorSelection": {
"authenticatorAttachment": ["platform", "cross-platform"],
"userVerification": ["required", "preferred", "discouraged"],
"requireResidentKey": [true, false]
},
"excludeCredentials": "enabled"
},
"counter": {
"requireIncrease": true,
"requireCounter": false
},
"rp": {
"name": "demo.strongauth.com:8181"
},
"authentication": {
"userVerification": ["required", "preferred", "discouraged"],
"allowCredentials": "enabled"
}
}
```

Example new JSON:

```
{
"FidoPolicy": {
"name": "DefaultPolicy",
"copyright": "",
"version": "1.0",
"startDate": "1606957205",
"endDate": "1760103870871",
"system": {
"requireCounter": "mandatory",
"integritySignatures": false,
"userVerification": ["required", "preferred", "discouraged"],
"userPresenceTimeout": 0,
"allowedAaguids": ["all"],
"algorithms": {
"curves": ["secp256r1", "secp384r1", "secp521r1", "curve25519"],
"rsa": ["rsassa-pkcs1-v1_5-sha256", "rsassa-pkcs1-v1_5-sha384", "rsassa-
pkcs1-v1_5-sha512", "rsassa-pss-sha256", "rsassa-pss-sha384", "rsassa-
pss-sha512"],
"signatures": ["ecdsa-p256-sha256", "ecdsa-p384-sha384", "ecdsa-p521-
sha512", "eddsa", "ecdsa-p256k-sha256"]
},
"attestation": {
"conveyance": ["none", "indirect", "direct", "enterprise"],
"formats": ["fido-u2f", "packed", "tpm", "android-key", "android-
safetynet", "none"]
},
```

#	Explanation

	<pre>"registration": {
	"displayName": "required",
	"attachment": ["platform", "cross-platform"],
	"residentKey": ["required", "preferred", "discouraged"],
	"excludeCredentials": "enabled"
	},
	"authentication": {
	"allowCredentials": "enabled"
	},
	"authorization": {
	"maxdataLength": 256,
	"preserve": true
	},
	"rp": {
	"name": "FIDOServer",
	"id": "strongkey.com"
	}
	},
	"extensions": {
	"example.extension": true
	}
	}
	}</pre>

DEV-1908	**Assign configurations to the FIDO2 Server.** Add a new configurations table to the FIDO2 server to enable domain-level configurations. **Fix:** A new configurations table has been added to the FIDO2 server which will allow for certain properties to be set on a domain basis. New operations have been added to the command line client to demonstrate the Create/Read/Update/Delete (CRUD) operations on the configurations table. **Mutable Configurations** ##SKCE - Domain-specific properties **ldape.cfg.property.service.ce.ldap.ldapadmingroup**=Identifies the Common Name (CN) for the *Administrator* group in LDAP/AD. Default value: cn=AdminAuthorized **ldape.cfg.property.service.ce.ldap.ldapservicegroup**=Identifies the Common Name (CN) for the *Services* group in LDAP/AD. Default value: cn=Services # LDAP Encryption-Authorized group **ldape.cfg.property.service.ce.ldap.ldapencryptiongroup**=Identifies the Common Name (CN) for the *file encryption authorized* group in LDAP/AD. This property is only used by the file encryption module. Default value: cn=EncryptionAuthorized # LDAP Decryption-Authorized group **ldape.cfg.property.service.ce.ldap.ldapdecryptiongroup**=Identifies the Common Name (CN) for the *file decryption authorized* group in LDAP/AD. This property is only used by the file encryption module. Default value: cn=DecryptionAuthorized

#	Explanation
	# LDAP CloudMove-Authorized group **ldape.cfg.property.service.ce.ldap.ldapcloudmovegroup**=Identifies the Common Name (CN) for the *file move authorized* group in LDAP/AD. This property is only used by the file encryption module. Default value: cn=CloudMoveAuthorized # LDAP Load-Authorized group **ldape.cfg.property.service.ce.ldap.ldaploadgroup**=Identifies the Common Name (CN) for the *Key Load authorized* group in LDAP/AD. This property is only used by the signing module. Default value: cn=LoadAuthorized # LDAP Remove-Authorized group **ldape.cfg.property.service.ce.ldap.ldapremovegroup**=Identifies the Common Name (CN) for the *Key remove authorized* group in LDAP/AD. This property is only used by the signing module. Default value: cn=RemoveAuthorized # LDAP Sign-Authorized group **ldape.cfg.property.service.ce.ldap.ldapsigngroup**=Identifies the Common Name (CN) for the *Sign authorized* group in LDAP/AD. This property is only used by the signing module. Default value: cn=SignAuthorized # LDAP FIDO-Authorized group **ldape.cfg.property.service.ce.ldap.ldapfidogroup**=Identifies the Common Name (CN) for the *FIDO authorized* group in LDAP/AD. This property is only used by the FIDO server to perform *patch* and *delete* operations. Default value: cn=FidoAuthorized # LDAP FIDO-REG Authorized group **ldape.cfg.property.service.ce.ldap.ldapfidoreggroup**=Identifies the Common Name (CN) for the *FIDO registration authorized* group in LDAP/AD. This property is only used by the FIDO server to perform *pre-register* and *register* operations. Default value: cn=FidoRegAuthorized # LDAP FIDO-SIGN Authorized group **ldape.cfg.property.service.ce.ldap.ldapfidosigngroup**=Identifies the Common Name (CN) for the *FIDO assertion authorized* group in LDAP/AD. This property is only used by the FIDO server to perform *pre-authenticate* and *authenticate* operations. Default value: cn=FidoSignAuthorized # LDAP FIDO-AUTHZ Authorized group **ldape.cfg.property.service.ce.ldap.ldapfidoauthzgroup**=Identifies the Common Name (CN) for the *FIDO authorizations authorized* group in LDAP/AD. This property is only used by the FIDO server to perform *pre-authorize* and *authorize* operations. Default value: cn=FidoAuthzAuthorized **ldape.cfg.property.service.ce.ldap.ldapfidoadmingroup**=Identifies the Common Name (CN) for the *FIDO admin authorized* group in LDAP/AD. This property is only used by the FIDO server to perform admin (policy and configurations) operations. Default value: cn=FidoAdminAuthorized **ldape.cfg.property.service.ce.ldap.ldapurl**=Identifies the LDAP/AD URL for the

#	Explanation
	authentication/authorization of service credentials. Default value: ldap://localhost:1389 **#ldape.cfg.property.service.ce.ldap.ldapbinddn**=Identifies the LDAP/AD bind Distinguished Name (DN) for the configured LDAP/AD. Default: CN=Directory Manager **#ldape.cfg.property.service.ce.ldap.ldapbinddn.password**=Identifies the password for the LDAP/AD bind Distinguished Name (DN) for the configured LDAP/AD. Default value: Abcd1234! **ldape.cfg.property.service.ce.ldap.ldapdnprefix**=Identifies the Distinguished Name (DN) prefix to be used for service credentials. Default value: cn= **ldape.cfg.property.service.ce.ldap.ldapdnsuffix**=Identifies the user suffix to be appended to the user Distinguished Name (DN). Default value: ,ou=users,ou=v2,ou=SKCE,ou=StrongAuth,ou=Applications,dc=strongauth,dc=com **ldape.cfg.property.service.ce.ldap.ldapgroupsuffix**=Identifies the groups suffix to be appended to the group Distinguished Name (DN). Default value: ,ou=groups,ou=v2,ou=SKCE,ou=StrongAuth,ou=Applications,dc=strongauth,dc=com ##APPL - Domain-specific properties **appl.cfg.property.service.ce.ldap.ldaptype**=Identifies what type of LDAP will be used for authenticating service credentials for the domain. Accepted values: LDAP | AD. Default value: LDAP ##SKFS - Domain-specific properties **skfs.cfg.property.fido2.user.sendfakeKH**=Identifies if fake keyhandles should be sent back to the calling application when they request *preauthentication* for unregistered users. Accepted values: TRUE | FALSE. Default value: FALSE
DEV-1904	**Support for custom Distinguished Names (DN) in LDAP for application service credentials** The FIDO server has a concept of cryptographic domains; the service credentials for every domain are separated by the domain ID in the Distinguished Name (DN). There is an RFE for it not to be tied to a specific DN. **Fix:** With the addition of the new configurations table, LDAP may be set for a specific domain, and once set, it will override any default values. This will allow a company to use a custom *dnsuffix* for the users and groups in LDAP, thereby removing the requirement for `"did=<did>"` or `"ou=<did>"` set by default.
DEV-1903	**Separate service credentials for registration and authentication.** Current StrongKey FIDO2 Server has only one LDAP/AD group (*FIDOAuthorized*) which allows service credentials to both register and authenticate users. Divide this up to have more granularity. **Fix:** The LDAP lookup has been updated and instead of just one group (*FIDOAuthorized)* to verify service credentials for all FIDO operations, there are now multiple groups: *FIDORegAuthorized* - Registrations *FIDOSignAuthorized* - Authentications *FIDOAdminAuthorized* - Admin operations such as Create/Read/Update/Delete (CRUD) on policies and configurations. All other operations still rely on the *FIDOAuthorized* group.

#	Explanation
DEV-1902	**Improve Signature Performance** Improve the signature performance for database-row-level signatures performed for StrongKey FIDO2 Server. **Fix:** With the old build one specific provider (Bouncy Castle Federal Information Processing System, a.k.a. BC FIPS) was used for creating the signature, which makes the process single-threaded in FIPS mode. The provider was changed to help improve signature performance. The signature input generation is now a JSON object instead of XML. The keystore has been updated to use an Elliptical Curve (EC) key instead of Rivest-Shamir-Adelman (RSA) key.
DEV-1897	**Username *keyhandle* combination does not exist.** It ignores the error condition. **Fix:** The new build, instead of ignoring the check, verifies if the combination already exists and returns an appropriate response: "Username and Key Handle combination exists."