CISCO
SECURE

CISCO The bridge to possible

Passwords are Dead:

WebAuthn for the security of webapps

Chris Volny (she/her)

Cisco :: Duo Security :: CloudSSO Services :: Sr. Software Engineer

About Speaker – Chris Volny

10+ years in infosec

NORIS (.NET)

MFA + TS Credentials

Reflective Dependency Graph

XML Transformation / ETL

GM / OnStar (Java)

DataStage / ETL

SSO EE w/ SAP ERP

Connected Vehicle Back-office

VES (C++, Java, Bash)

(libvirt) Android Profiles-MDM

Qt C++ Cross-Domain Bridge

Yocto Linux + CI

Duo Security (Python + JS)

Cloud SAML/M365/OIDC

Twisted + Async Python

JS + React

2008 - 2014

2014 - 2017

2017 - 2021

2021 -

Toledo, OH

Detroit, MI

Ann Arbor, MI

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

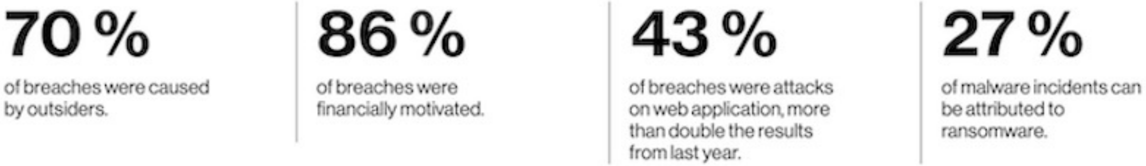
Agenda

- Why Passwordless?
- History, Auth Factors, and Cryptography
- FIDO and WebAuthn
- Demo and Usage
- System Design Considerations
- Q/A + Resources

Why Passwordless?

Over 80% of hacking breaches involve brute force or the use of lost or stolen credentials.

Verizon DBIR



70 %

of breaches were caused by outsiders.

86 %

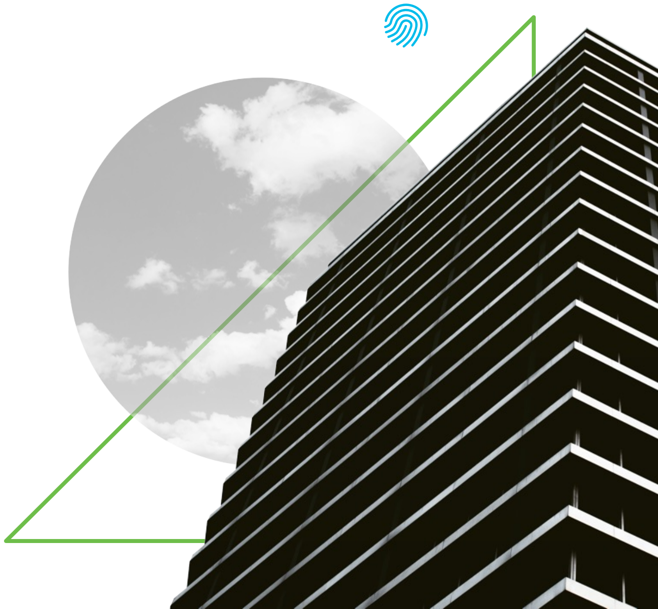
of breaches were financially motivated.

43 %

of breaches were attacks on web application, more than double the results from last year.

27 %

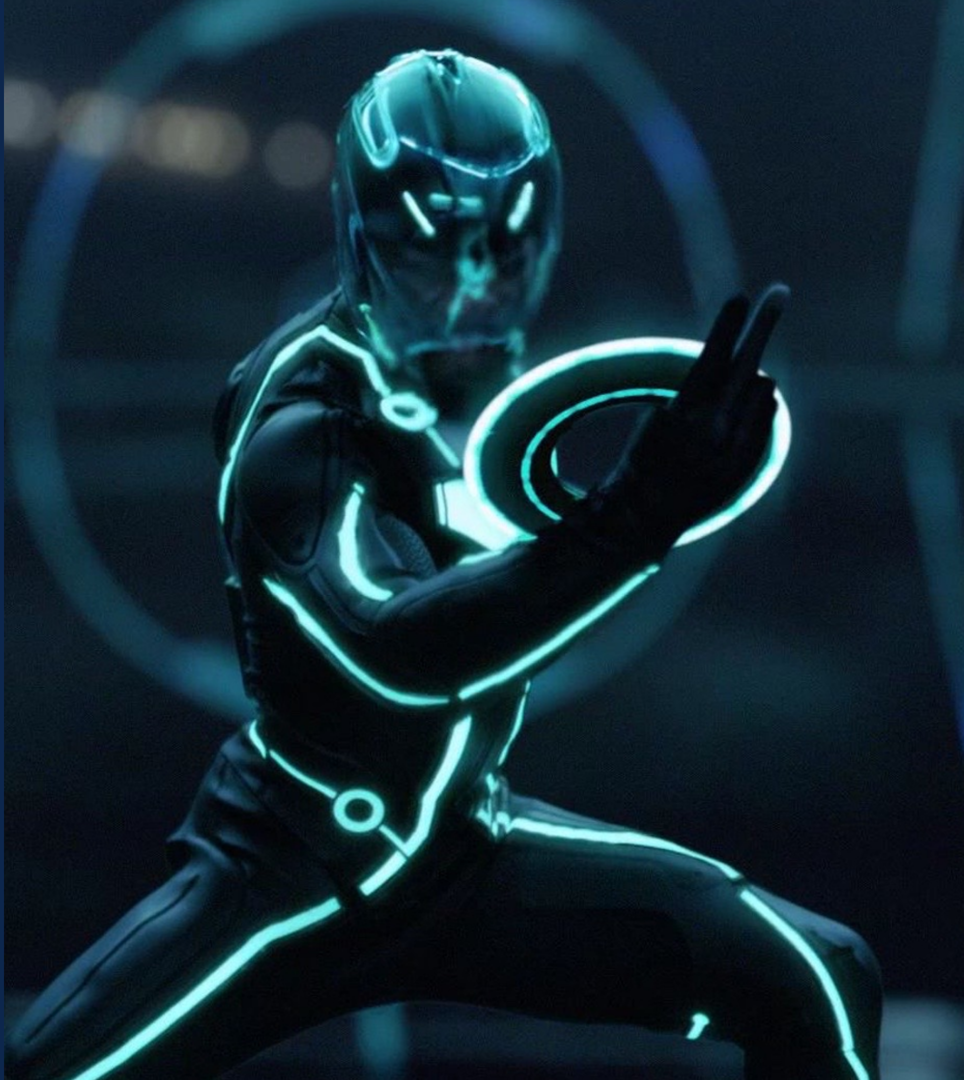
of malware incidents can be attributed to ransomware.



© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

“I Fight for the User.”

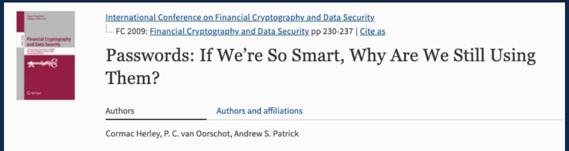
“I’m a user too!”



© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Passwords: a history

- “something you know”
- Roman Legion – friend from foe
- Defacto computer security since 1960s
  - Fernando Corbató, MIT CTSS
  - … also first leaks:
    - Spring 1962 printed password file
    - 1966 motd and password files swapped
- We’ve been asking this since 2009:



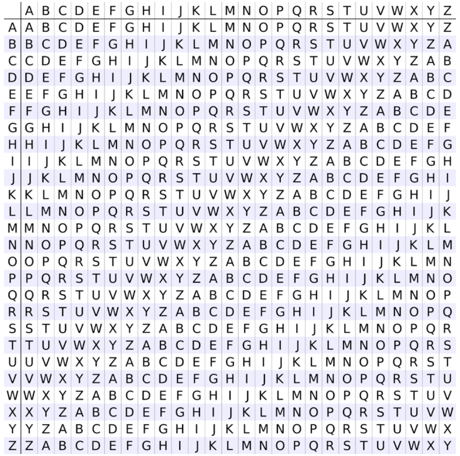
International Conference on Financial Cryptography and Data Security

FC 2009: Financial Cryptography and Data Security pp 230-237 | Cite as

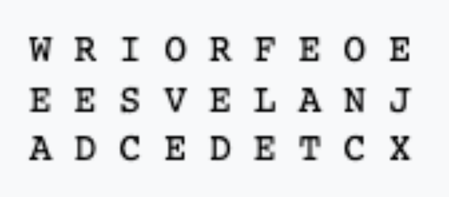
Passwords: If We’re So Smart, Why Are We Still Using Them?

Authors Authors and affiliations

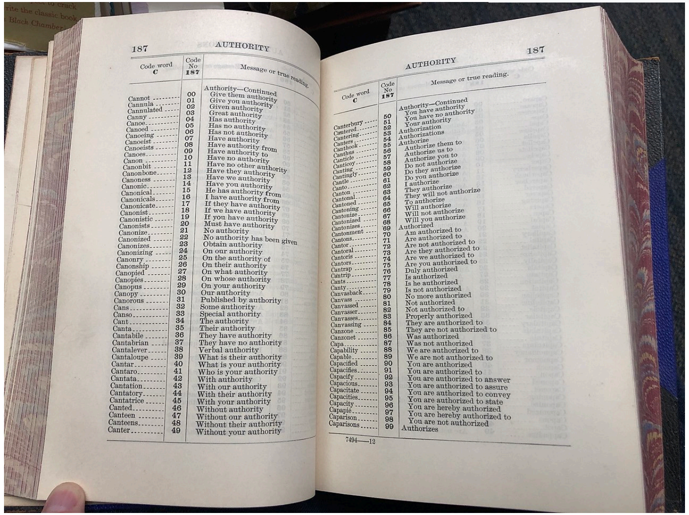
Cormac Herley, P. C. van Oorschot, Andrew S. Patrick



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y



```
W R I O R F E O E
E E S V E L A N J
A D C E D E T C X
```



187 AUTHORITY

Code word C	Code No 187	Message or true reading.
		Authority—Continued
Cannot	00	Give them authority
Cannula	01	Give you authority
Cannulated	02	Given authority
Canny	03	Great authority
Canoe	04	Has authority
Canoed	05	Has no authority
Canoeing	06	Has not authority
Canoeist	07	Have authority
Canoeists	08	Have authority from
Canoes	09	Have authority to
Canon	10	Have no authority
Canonbit	11	Have no other authority
Canonbone	12	Have they authority
Canoness	13	Have we authority
Canonic	14	Have you authority
Canonical	15	He has authority from
Canonicals	16	I have authority from
Canonicate	17	If they have authority
Canonist	18	If we have authority
Canonistic	19	If you have authority
Canonists	20	Must have authority
Canonize	21	No authority
Canonized	22	No authority has been given
Canonizes	23	Obtain authority
Canonizing	24	On our authority
Canonry	25	On the authority of
Canonship	26	On their authority
Canopied	27	On what authority
Canopies	28	On whose authority
Canopus	29	On your authority
Canopy	30	Our authority
Canorous	31	Published by authority
Cans	32	Some authority
Canso	33	Special authority
Cant	34	The authority
Canta	35	Their authority
Cantabile	36	They have authority
Cantabrian	37	They have no authority
Cantalever	38	Verbal authority
Cantaloupe	39	What is their authority
Cantar	40	What is your authority
Cantaro	41	Who is your authority
Cantata	42	With authority
Cantation	43	With our authority
Cantatory	44	With their authority
Cantatrice	45	With your authority
Canted	46	Without authority
Canteen	47	Without our authority
Canteens	48	Without their authority
Canter	49	Without your authority

AUTHORITY 187

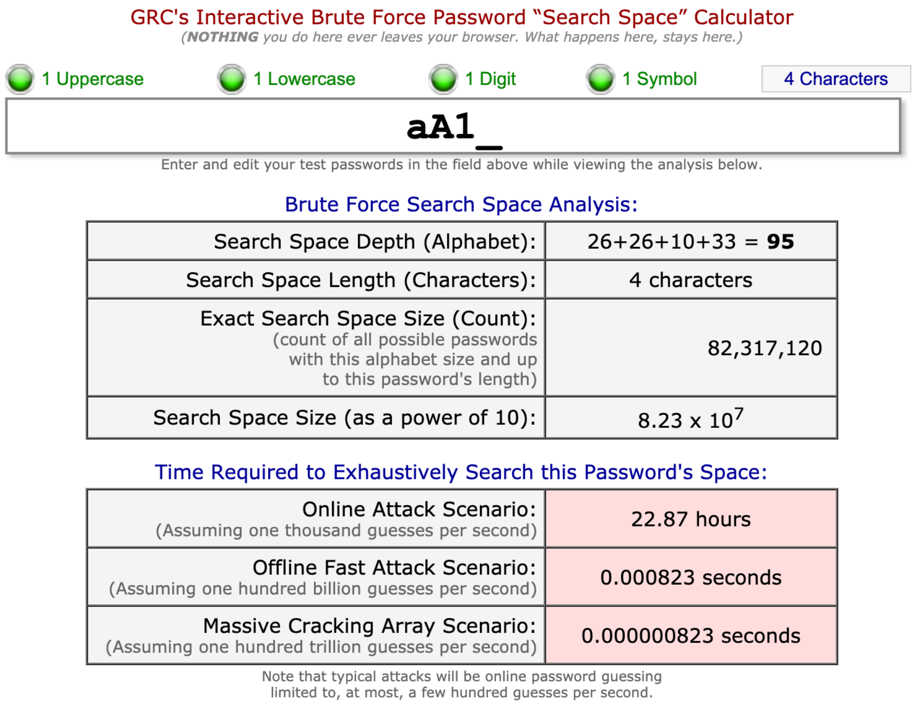
Code word C	Code No 187	Message or true reading.
		Authority—Continued
Canterbury	50	You have authority
Cantered	51	You have no authority
Cantering	52	Your authority
Canters	53	Authorization
Canthook	54	Authorizations
Canthus	55	Authorize
Canticle	56	Authorize them to
Canticoy	57	Authorize us to
Canting	58	Authorize you to
Cantingly	59	Do not authorize
Cantle	60	Do they authorize
Canto	61	Do you authorize
Canton	62	I authorize
Cantonal	63	They authorize
Cantoned	64	They will not authorize
Cantoning	65	To authorize
Cantonize	66	Will authorize
Cantonized	67	Will not authorize
Cantonizes	68	Will you authorize
Cantonment	69	Authorized
Cantons	70	Am authorized to
Cantor	71	Are authorized to
Cantoral	72	Are not authorized to
Cantoris	73	Are they authorized to
Cantors	74	Are we authorized to
Cantrap	75	Are you authorized to
Cantrip	76	Duly authorized
Cants	77	Is authorized
Canty	78	Is he authorized
Canvasback	79	Is not authorized
Canvass	80	No more authorized
Canvassed	81	Not authorized
Canvasser	82	Not authorized to
Canvasses	83	Properly authorized
Canvassing	84	They are authorized to
Canzone	85	They are not authorized to
Canzonet	86	Was authorized
Capa	87	Was not authorized
Capability	88	We are authorized to
Capable	89	We are not authorized to
Capacified	90	You are authorized
Capacifies	91	You are authorized to
Capacify	92	You are authorized to answer
Capacious	93	You are authorized to assure
Capacitate	94	You are authorized to convey
Capacities	95	You are authorized to state
Capacity	96	You are hereby authorized
Capapié	97	You are hereby authorized to
Caparison	98	You are not authorized
Caparisons	99	Authorizes

7494——12

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Password Complexity / Search Space

- Think: time lock
- Choose: random xor memorable
- Rinse, repeat
- www.grc.com/haystack



GRC's Interactive Brute Force Password "Search Space" Calculator

*(**NOTHING** you do here ever leaves your browser. What happens here, stays here.)*

● 1 Uppercase ● 1 Lowercase ● 1 Digit ● 1 Symbol 4 Characters

aA1_

Enter and edit your test passwords in the field above while viewing the analysis below.

Brute Force Search Space Analysis:

Search Space Depth (Alphabet):	26+26+10+33 = **95**
Search Space Length (Characters):	4 characters
Exact Search Space Size (Count): (count of all possible passwords with this alphabet size and up to this password's length)	82,317,120
Search Space Size (as a power of 10):	8.23×10^{7}

Time Required to Exhaustively Search this Password's Space:

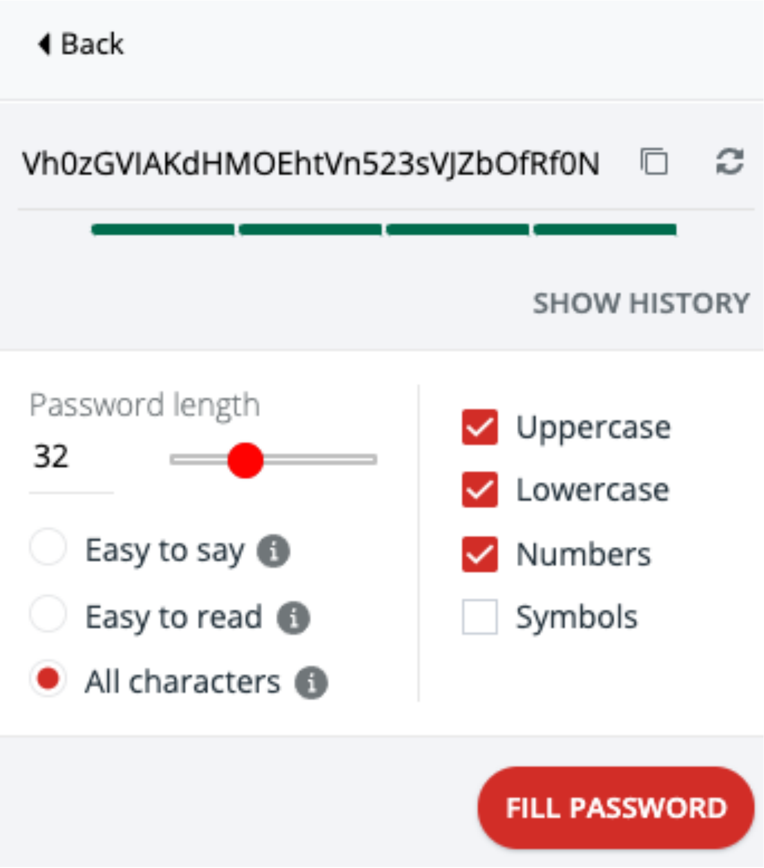
Online Attack Scenario: (Assuming one thousand guesses per second)	22.87 hours
Offline Fast Attack Scenario: (Assuming one hundred billion guesses per second)	0.000823 seconds
Massive Cracking Array Scenario: (Assuming one hundred trillion guesses per second)	0.000000823 seconds

Note that typical attacks will be online password guessing limited to, at most, a few hundred guesses per second.

cisco SECURE © 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Enter Password Managers

- Impossibly long passwords for everything
- Encrypted with one “strong” password
  - Hope you don’t forget it
  - Hope no one copies it
  - Hope that password was “strong”



Vh0zGVIAKdHMOEhtVn523sVJZbOfRf0N

Password length
32

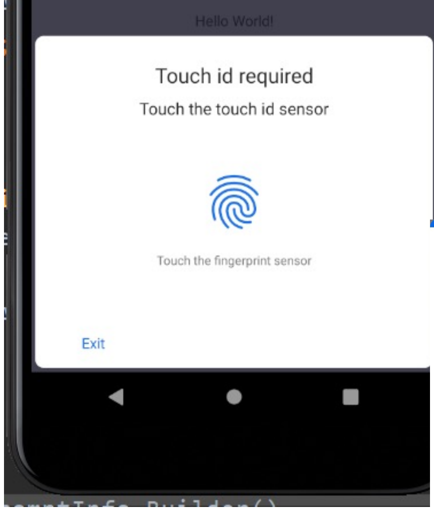
○ Easy to say
○ Easy to read
● All characters

☑ Uppercase
☑ Lowercase
☑ Numbers
□ Symbols

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Enter Multi-Factor

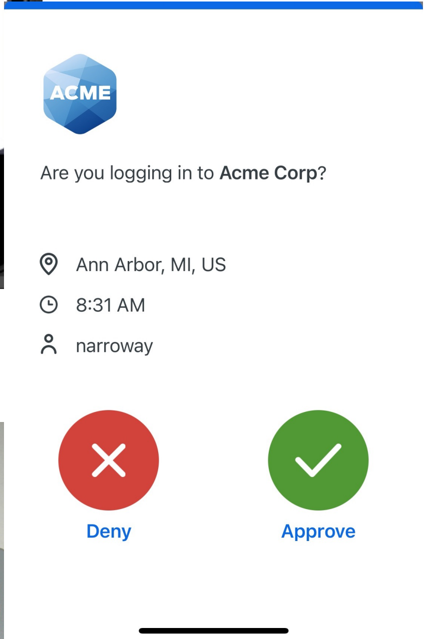
- Defense in depth
  - “something you know”
  - “something you have/are”
- “Quick, where’s my phone/YubiKey?”
- Variants:
  - Voice/SMS 2FA Hijack / Phishing
  - OTP Codes Exfiltrate / Phishing
  - Push Great - Duo
  - Certificates/PKI Great - WebAuthn
    - Maybe SE/TPM stored Excellent


Hello World!

Touch id required

Touch the touch id sensor

Touch the fingerprint sensor



Are you logging in to **Acme Corp**?

Ann Arbor, MI, US

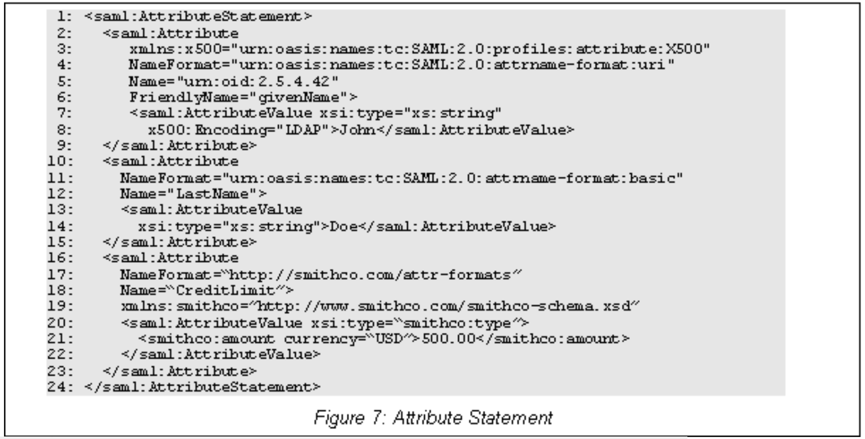
8:31 AM

narroway

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

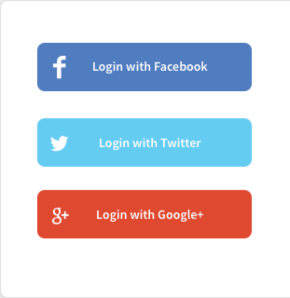
Enter Single Sign-on

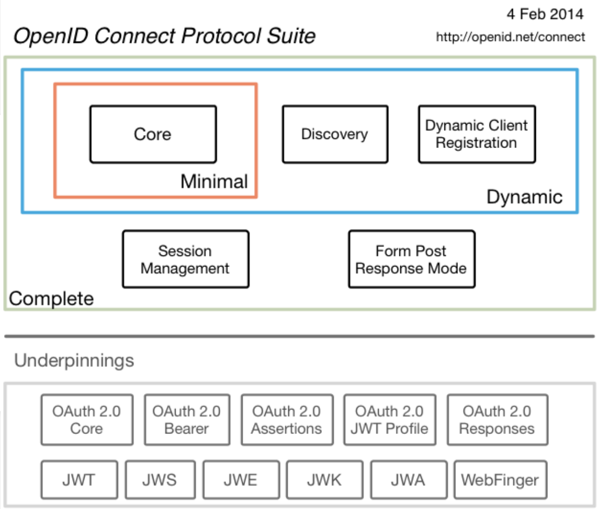
• Social Login	Wild West
• SAML/OIDC	Enterprise



```
<saml:AttributeStatement>
  <saml:Attribute
     xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
     NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
     Name="urn:oid:2.5.4.42"
     FriendlyName="givenName">
     <saml:AttributeValue xsi:type="xs:string"
       x500:Encoding="LDAP">John</saml:AttributeValue>
  </saml:Attribute>
  <saml:Attribute
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
    Name="LastName">
    <saml:AttributeValue
      xsi:type="xs:string">Doe</saml:AttributeValue>
  </saml:Attribute>
  <saml:Attribute
    NameFormat=“http://smithco.com/attr-formats”
    Name=“CreditLimit”>
    xmlns:smithco=“http://www.smithco.com/smithco-schema.xsd”
    <saml:AttributeValue xsi:type=“smithco:type”>
      <smithco:amount currency=“USD”>500.00</smithco:amount>
    </saml:AttributeValue>
  </saml:Attribute>
</saml:AttributeStatement>
```

Figure 7: Attribute Statement





© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

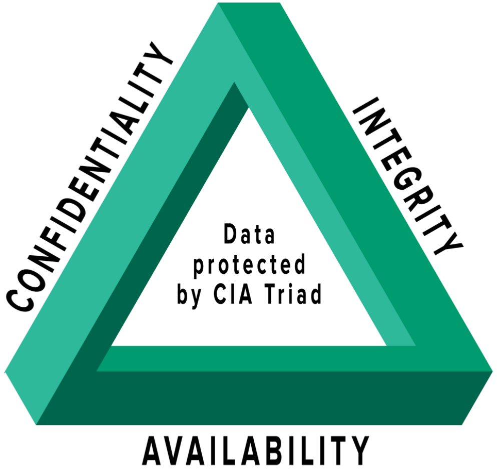
What are Security Keys?

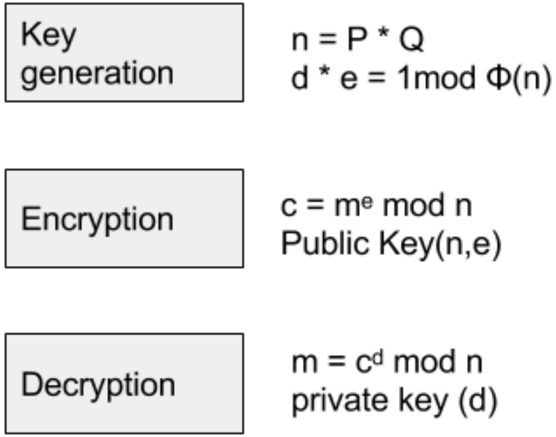
- We see these a lot in MFA
- What are they?
  - Secure Enclave (SE)
    - Tamper / extraction resistant
    - Asymmetric cryptography
  - Can embed PK Credentials
  - Sometimes built in (platform)
    - Touch ID
  - Sometimes a peripheral (external)
    - YubiKey
  - Can use to unlock bigger vaults

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

(A)Symmetric Cryptography

- Symmetric
  - same key for crypt/decrypt
  - Confidentiality
  - Secures data at rest + transport session
  - DES, AES, Blowfish
- Asymmetric
  - One key for crypt, one for decrypt
  - RSA 1977 (Ellis/Cocks ‘73)
  - Confidentiality and Integrity
  - Secures transport negotiation (‘web)
  - RSA, DE, EC
  - SSH, PGP, TLS





Key generation	$n = P * Q$ $d * e = 1 \bmod \Phi(n)$
Encryption	$c = m^e \bmod n$ Public Key(n,e)
Decryption	$m = c^d \bmod n$ private key (d)

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

FIDO Timeline/Philosophy

- Timeline
  - 2009 – PayPal and Validity Sensor talks
  - 2012 – FIDO Alliance Founded
  - 2014 – Samsung GS5 fingerprint e-shop
  - 2015 – FIDO1 published + BT/NFC
  - 2018 – FIDO2 published CTAP/WebAuthn
  - 2019 – Wide platform adoption
- Philosophy
  - Strong crypto
  - Limited scope Think: cookies + domain
  - Device attestation Which devices to trust

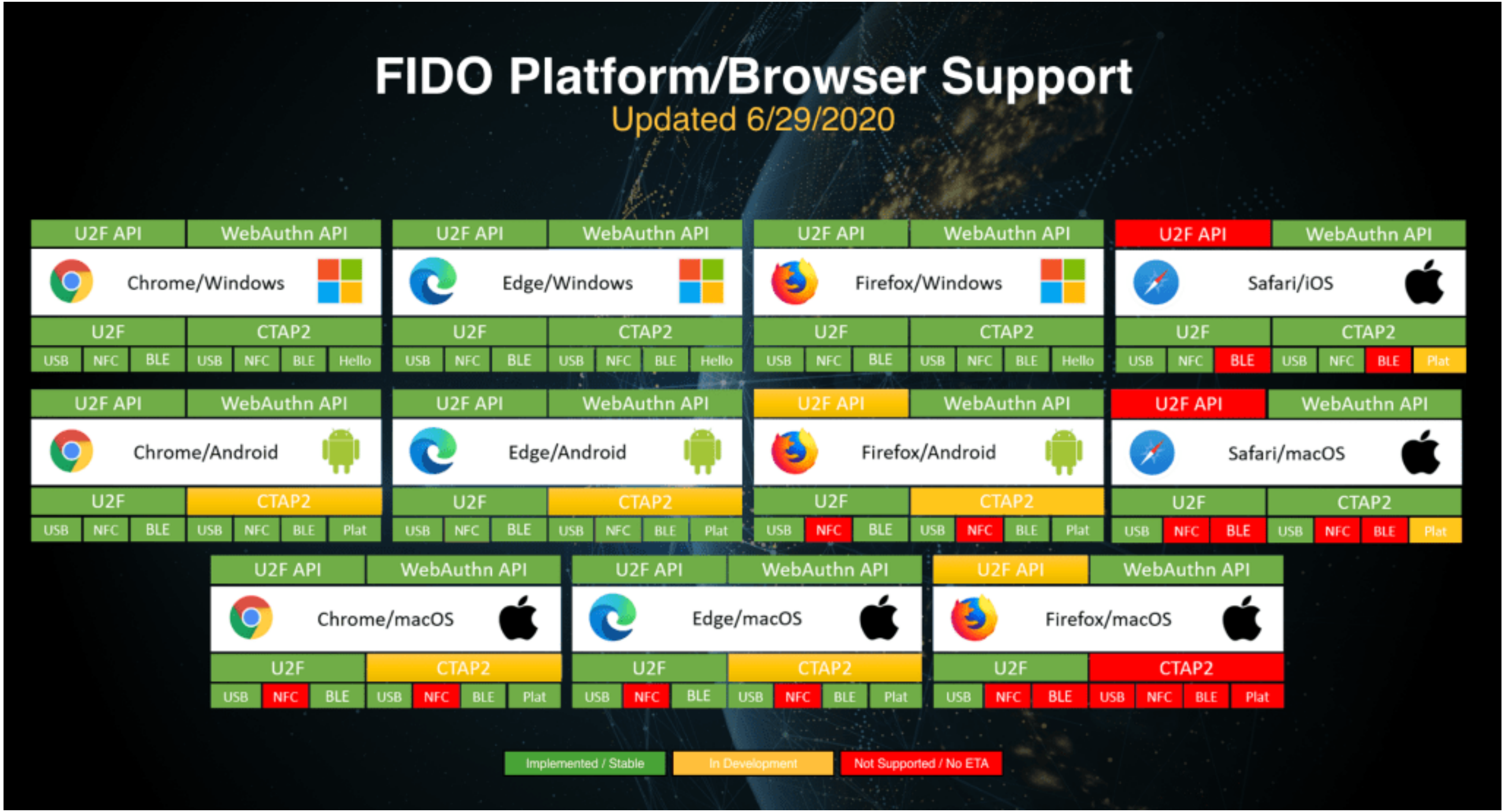
PayPal

validity™

fido™ ALLIANCE

simpler
stronger
authentication

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential



FIDO Platform/Browser Support

Updated 6/29/2020

Chrome/Windows
U2F API 🟩 | WebAuthn API 🟩
U2F 🟩: USB 🟩 NFC 🟩 BLE 🟩
CTAP2 🟩: USB 🟩 NFC 🟩 BLE 🟩 Hello 🟩

Edge/Windows
U2F API 🟩 | WebAuthn API 🟩
U2F 🟩: USB 🟩 NFC 🟩 BLE 🟩
CTAP2 🟩: USB 🟩 NFC 🟩 BLE 🟩 Hello 🟩

Firefox/Windows
U2F API 🟩 | WebAuthn API 🟩
U2F 🟩: USB 🟩 NFC 🟩 BLE 🟩
CTAP2 🟩: USB 🟩 NFC 🟩 BLE 🟩 Hello 🟩

Safari/iOS
U2F API 🟥 | WebAuthn API 🟩
U2F 🟩: USB 🟩 NFC 🟩 BLE 🟥
CTAP2 🟩: USB 🟩 NFC 🟩 BLE 🟥 Plat 🟨

Chrome/Android
U2F API 🟩 | WebAuthn API 🟩
U2F 🟩: USB 🟩 NFC 🟩 BLE 🟩
CTAP2 🟨: USB 🟩 NFC 🟩 BLE 🟩 Plat 🟩

Edge/Android
U2F API 🟩 | WebAuthn API 🟩
U2F 🟩: USB 🟩 NFC 🟩 BLE 🟩
CTAP2 🟨: USB 🟩 NFC 🟩 BLE 🟩 Plat 🟩

Firefox/Android
U2F API 🟨 | WebAuthn API 🟩
U2F 🟩: USB 🟩 NFC 🟥 BLE 🟩
CTAP2 🟨: USB 🟩 NFC 🟥 BLE 🟩 Plat 🟩

Safari/macOS
U2F API 🟥 | WebAuthn API 🟩
U2F 🟩: USB 🟩 NFC 🟥 BLE 🟥
CTAP2 🟩: USB 🟩 NFC 🟥 BLE 🟥 Plat 🟨

Chrome/macOS
U2F API 🟩 | WebAuthn API 🟩
U2F 🟩: USB 🟩 NFC 🟥 BLE 🟩
CTAP2 🟨: USB 🟩 NFC 🟥 BLE 🟩 Plat 🟩

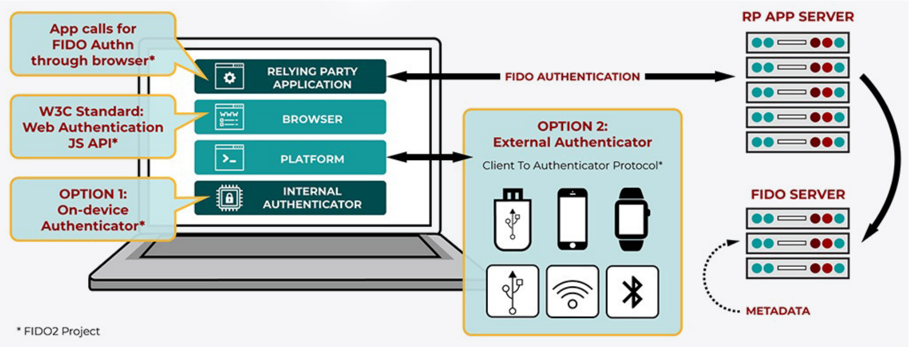
Edge/macOS
U2F API 🟩 | WebAuthn API 🟩
U2F 🟩: USB 🟩 NFC 🟥 BLE 🟩
CTAP2 🟨: USB 🟩 NFC 🟥 BLE 🟩 Plat 🟩

Firefox/macOS
U2F API 🟨 | WebAuthn API 🟩
U2F 🟩: USB 🟩 NFC 🟥 BLE 🟥
CTAP2 🟥: USB 🟥 NFC 🟥 BLE 🟥 Plat 🟥

🟩 Implemented / Stable | 🟨 In Development | 🟥 Not Supported / No ETA

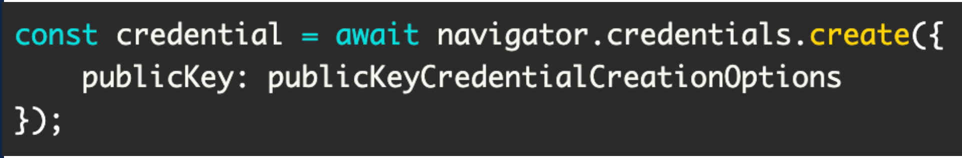
FIDO2 = CTAP + WebAuthn

- Client-to-Authenticator Protocols (CTAP)
  - Hardware to OS API / Transport
  - System calls Windows Hello, libfido2
  - USB, NFC, BLE, TPM Authenticators
- Web Authentication API (WebAuthn)
  - Just web applications
  - JavaScript API in browsers
  - Server-side libraries


App calls for FIDO Authn through browser*
W3C Standard: Web Authentication JS API*
OPTION 1: On-device Authenticator*
RELYING PARTY APPLICATION
BROWSER
PLATFORM
INTERNAL AUTHENTICATOR
FIDO AUTHENTICATION
OPTION 2: External Authenticator
Client To Authenticator Protocol*
RP APP SERVER
FIDO SERVER
METADATA
* FIDO2 Project

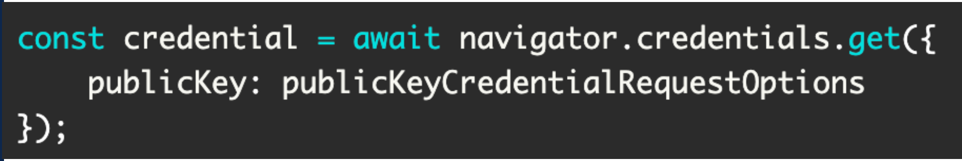
© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

FIDO2 = CTAP + WebAuthn



```
const credential = await navigator.credentials.create({
    publicKey: publicKeyCredentialCreationOptions
});
```

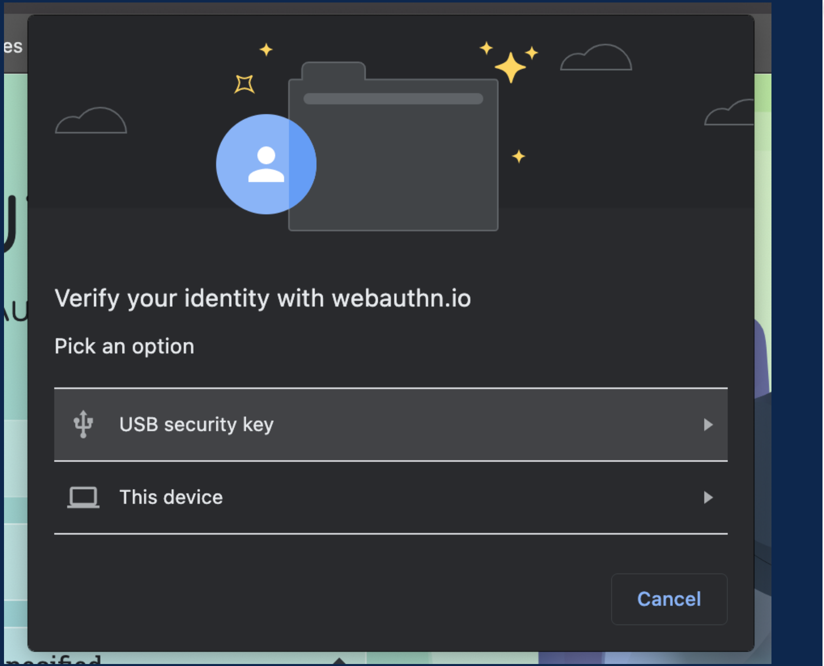
- Web Authentication API (WebAuthn)
  - JavaScript API in browsers



```
const credential = await navigator.credentials.get({
    publicKey: publicKeyCredentialRequestOptions
});
```

SECURE © 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Registration



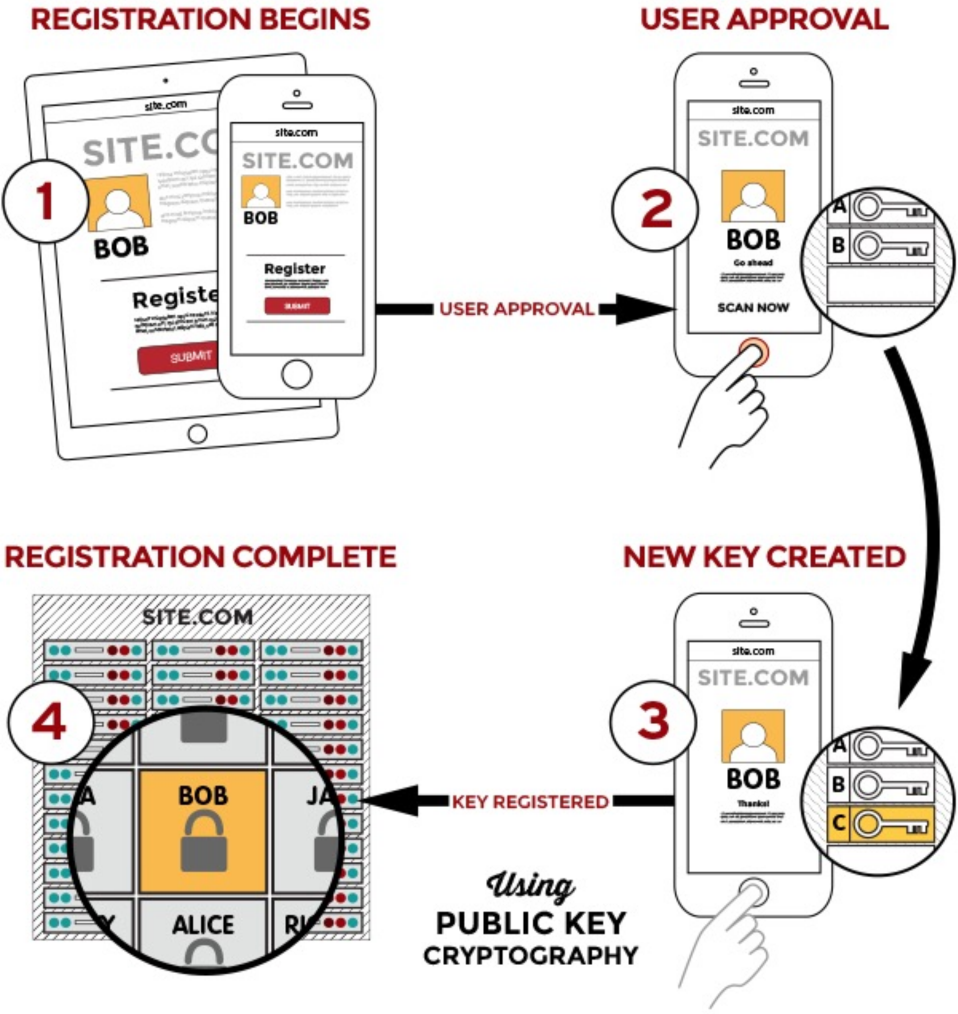
Verify your identity with webauthn.io

Pick an option

USB security key

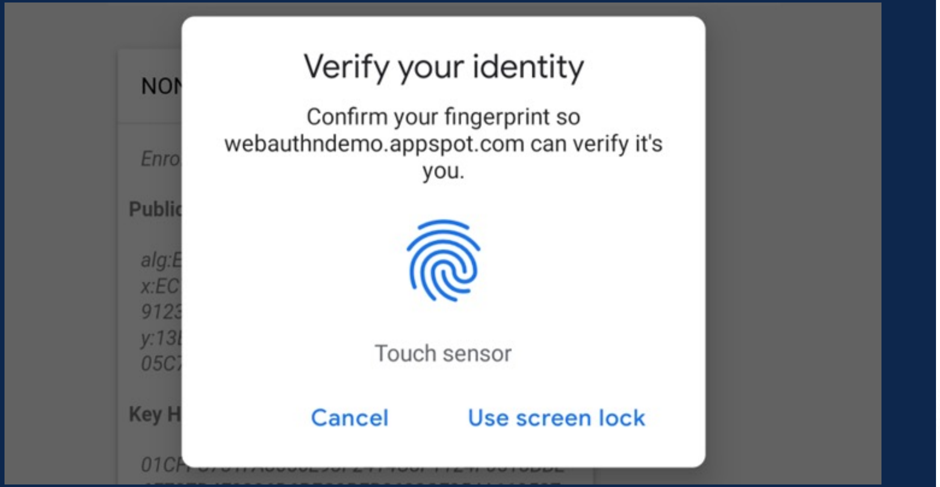
This device

Cancel



© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Login



NON

Enro

Public

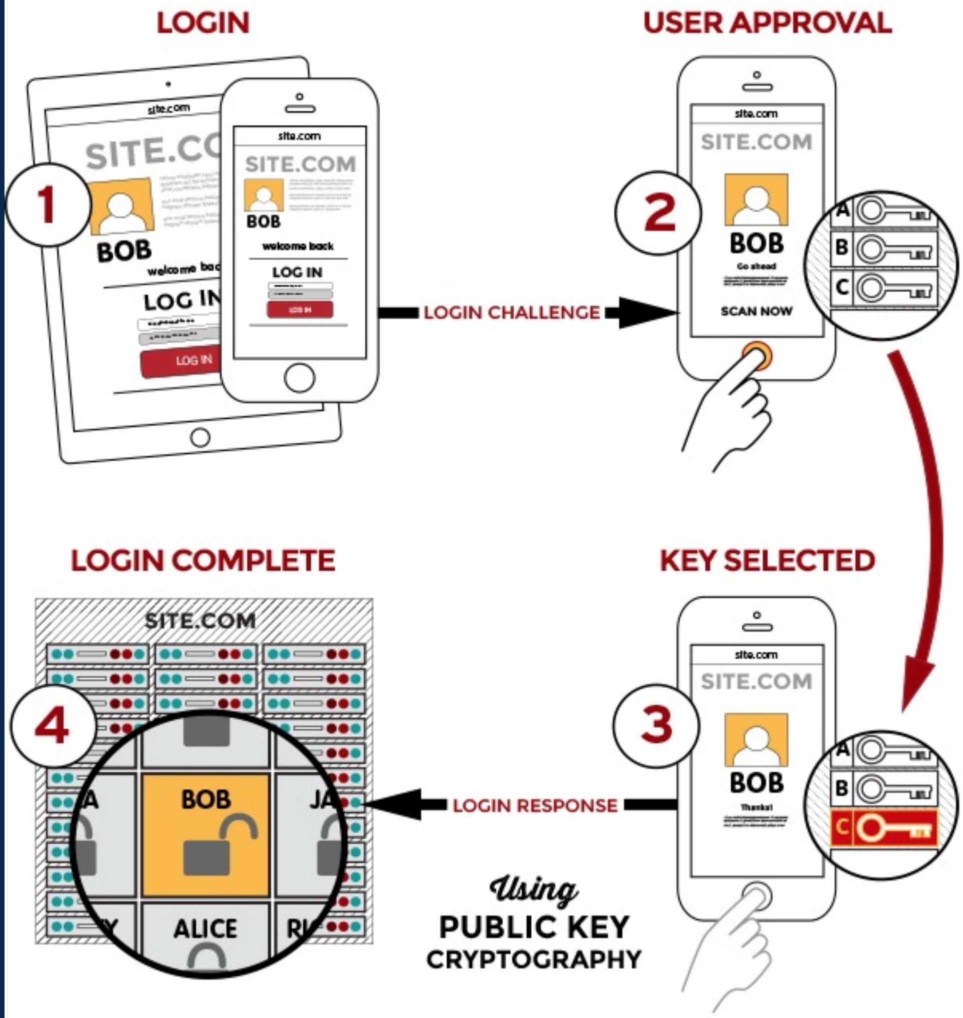
alg:E
x:EC
9123
y:13
05C

Key H

01CF

Verify your identity

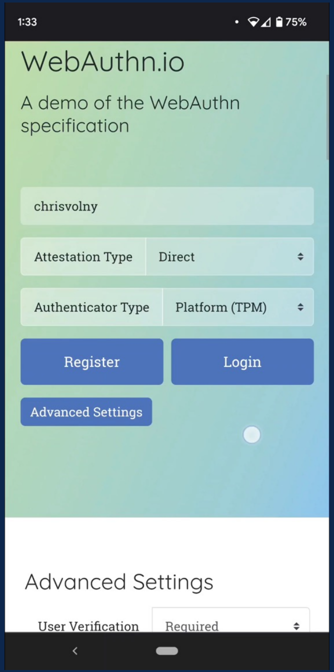
Confirm your fingerprint so webauthndemo.appspot.com can verify it's you.



© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Demonstrations

- https://webauthn.io (Duo Labs) (Android)



1:33 • 75%

WebAuthn.io

A demo of the WebAuthn specification

chrisvolny

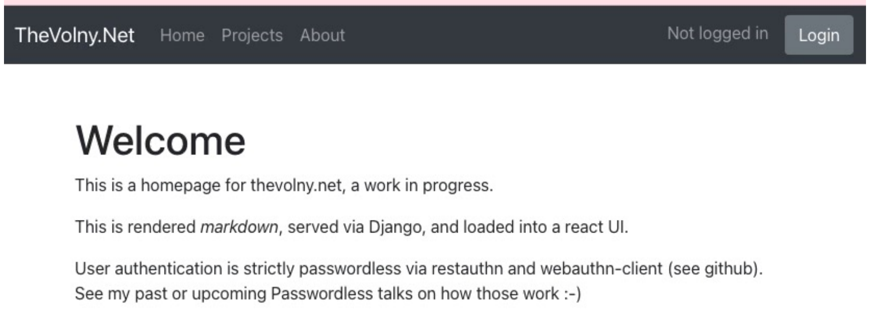
Attestation Type: Direct

Authenticator Type: Platform (TPM)

Advanced Settings

User Verification: Required

- https://www.thevolny.net/ (me) (OS X)



Welcome

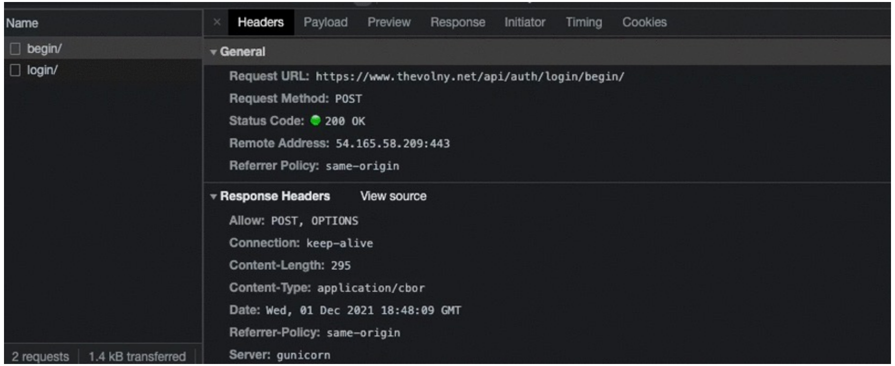
This is a homepage for thevolny.net, a work in progress.

This is rendered *markdown*, served via Django, and loaded into a react UI.

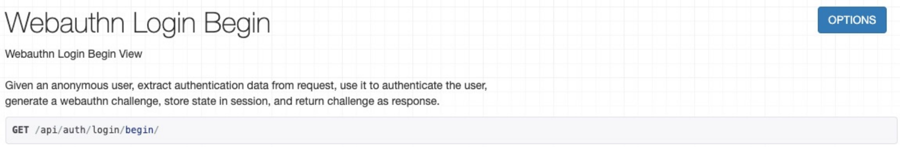
User authentication is strictly passwordless via restauthn and webauthn-client (see github). See my past or upcoming Passwordless talks on how those work :-)

My Demo – High Level

- Two Pieces:
  - JS Library
    - Axios with CBOR Interceptors
    - Login, Register functions
  - Django Module + App
    - Django REST Framework
    - CBOR render/parsers (some base64)
    - Authenticator, LoginToken models
    - ApiViews
    - Really simple templates for testing



```
▾ General
  Request URL: https://www.thevolny.net/api/auth/login/begin/
  Request Method: POST
  Status Code: ● 200 OK
  Remote Address: 54.165.58.209:443
  Referrer Policy: same-origin
▾ Response Headers
  Allow: POST, OPTIONS
  Connection: keep-alive
  Content-Length: 295
  Content-Type: application/cbor
  Date: Wed, 01 Dec 2021 18:48:09 GMT
  Referrer-Policy: same-origin
  Server: gunicorn
```

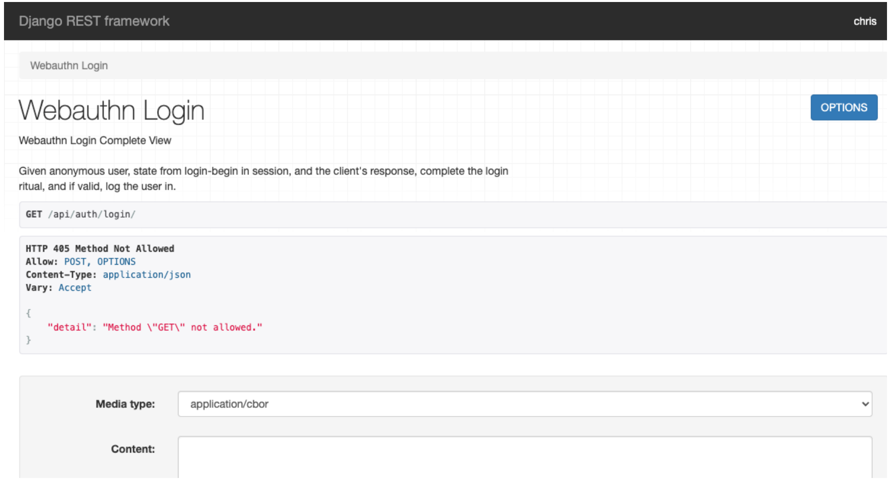


Webauthn Login Begin

Webauthn Login Begin View

Given an anonymous user, extract authentication data from request, use it to authenticate the user, generate a webauthn challenge, store state in session, and return challenge as response.

```
GET /api/auth/login/begin/
```



Webauthn Login

Webauthn Login Complete View

Given anonymous user, state from login-begin in session, and the client's response, complete the login ritual, and if valid, log the user in.

```
GET /api/auth/login/
```

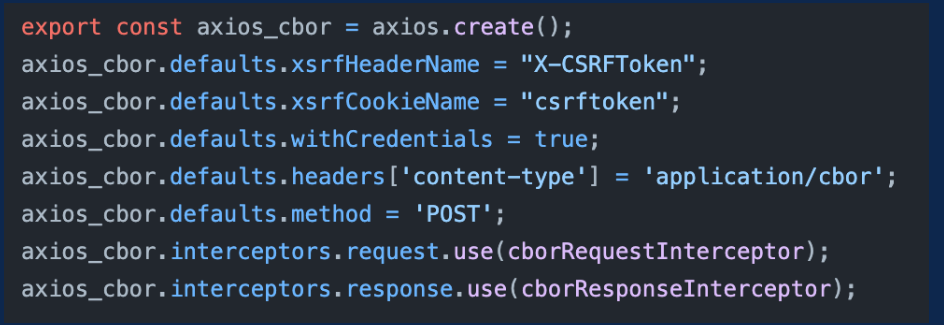
```
HTTP 405 Method Not Allowed
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "detail": "Method \"GET\" not allowed."
}
```

Media type: application/cbor

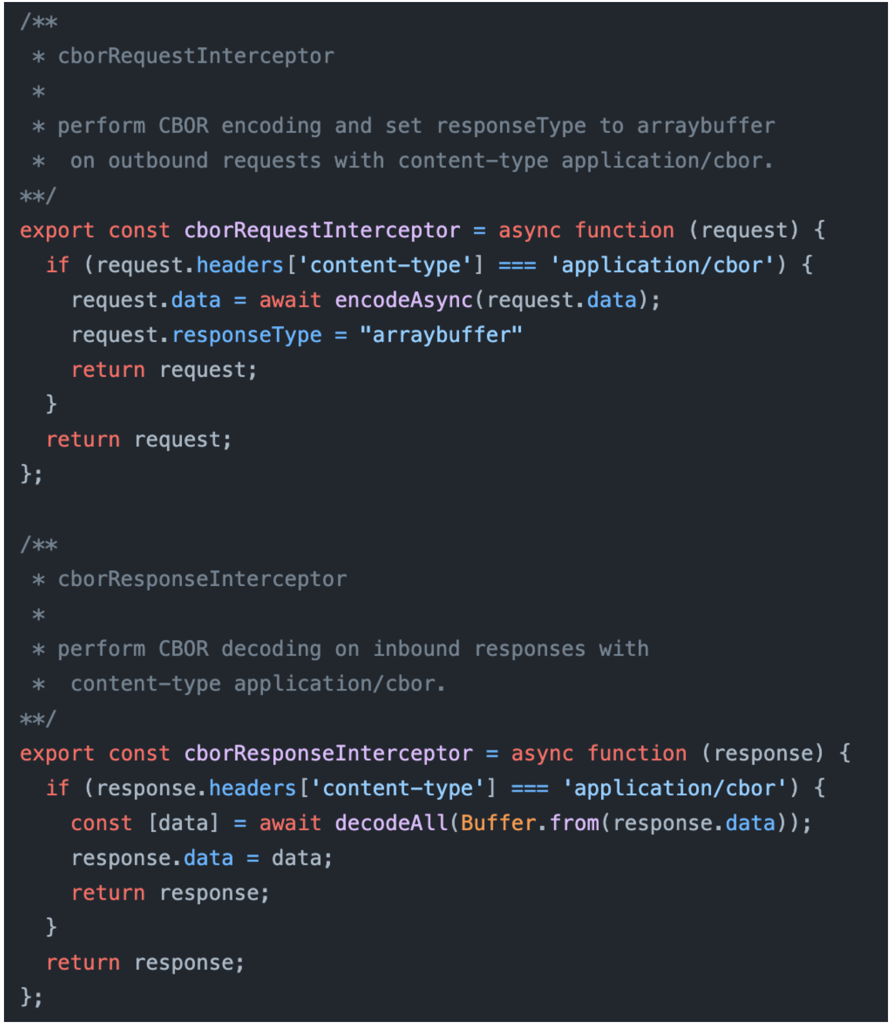
Content:

cisco SECURE © 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

My Demo – JS Axios



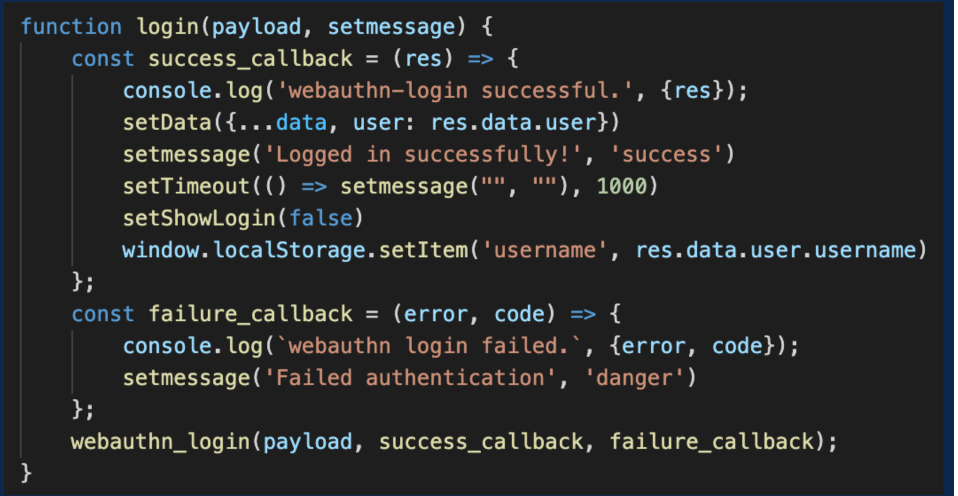
```
export const axios_cbor = axios.create();
axios_cbor.defaults.xsrfHeaderName = "X-CSRFToken";
axios_cbor.defaults.xsrfCookieName = "csrftoken";
axios_cbor.defaults.withCredentials = true;
axios_cbor.defaults.headers['content-type'] = 'application/cbor';
axios_cbor.defaults.method = 'POST';
axios_cbor.interceptors.request.use(cborRequestInterceptor);
axios_cbor.interceptors.response.use(cborResponseInterceptor);
```



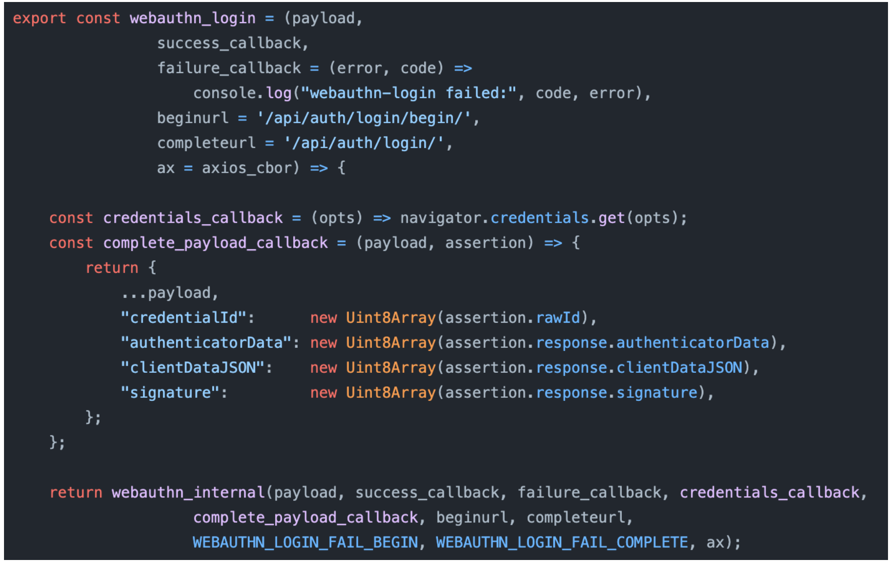
```
/**
 * cborRequestInterceptor
 *
 * perform CBOR encoding and set responseType to arraybuffer
 *  on outbound requests with content-type application/cbor.
**/
export const cborRequestInterceptor = async function (request) {
  if (request.headers['content-type'] === 'application/cbor') {
    request.data = await encodeAsync(request.data);
    request.responseType = "arraybuffer"
    return request;
  }
  return request;
};

/**
 * cborResponseInterceptor
 *
 * perform CBOR decoding on inbound responses with
 *  content-type application/cbor.
**/
export const cborResponseInterceptor = async function (response) {
  if (response.headers['content-type'] === 'application/cbor') {
    const [data] = await decodeAll(Buffer.from(response.data));
    response.data = data;
    return response;
  }
  return response;
};
```

SECURE © 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

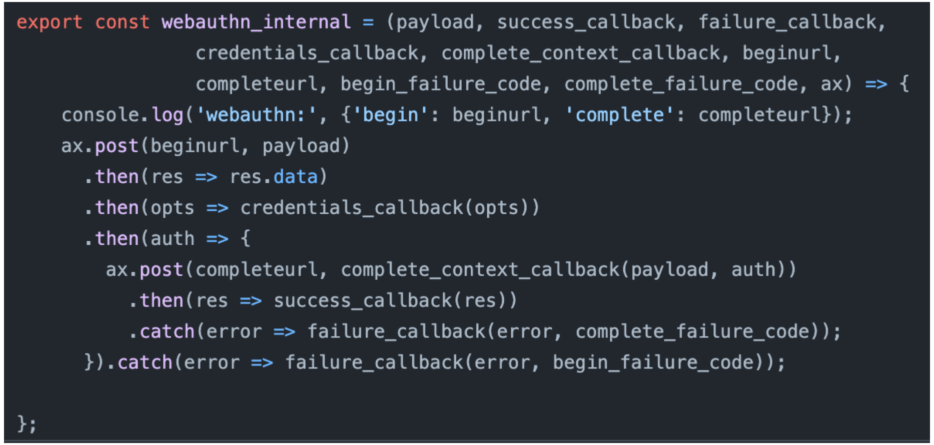
My Demo – JS Login



```
function login(payload, setmessage) {
    const success_callback = (res) => {
        console.log('webauthn-login successful.', {res});
        setData({...data, user: res.data.user})
        setmessage('Logged in successfully!', 'success')
        setTimeout(() => setmessage("", ""), 1000)
        setShowLogin(false)
        window.localStorage.setItem('username', res.data.user.username)
    };
    const failure_callback = (error, code) => {
        console.log(`webauthn login failed.`, {error, code});
        setmessage('Failed authentication', 'danger')
    };
    webauthn_login(payload, success_callback, failure_callback);
}
```



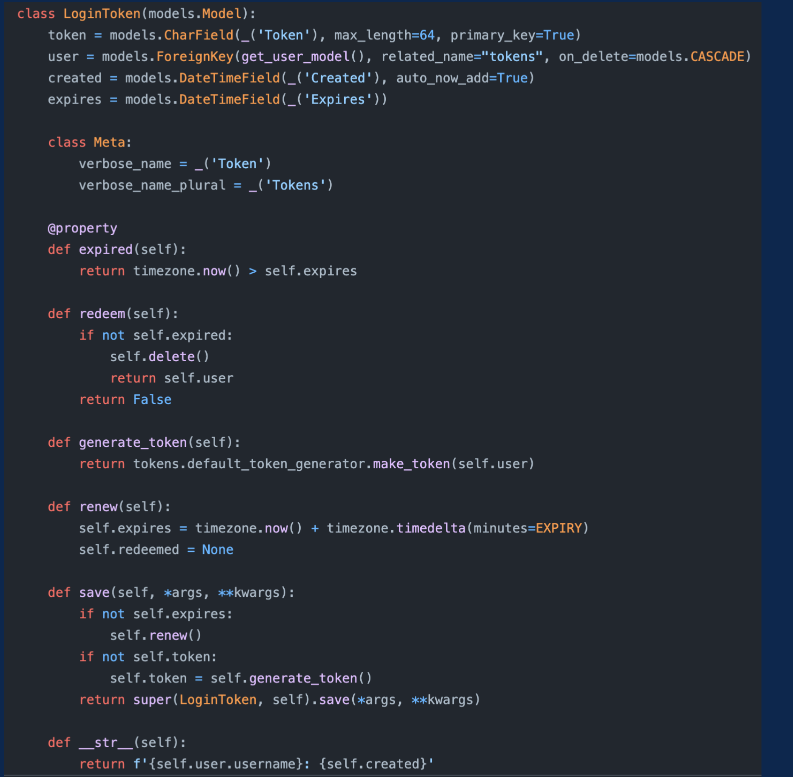
```
export const webauthn_login = (payload,
                success_callback,
                failure_callback = (error, code) =>
                    console.log("webauthn-login failed:", code, error),
                beginurl = '/api/auth/login/begin/',
                completeurl = '/api/auth/login/',
                ax = axios_cbor) => {

    const credentials_callback = (opts) => navigator.credentials.get(opts);
    const complete_payload_callback = (payload, assertion) => {
        return {
            ...payload,
            "credentialId":      new Uint8Array(assertion.rawId),
            "authenticatorData": new Uint8Array(assertion.response.authenticatorData),
            "clientDataJSON":    new Uint8Array(assertion.response.clientDataJSON),
            "signature":         new Uint8Array(assertion.response.signature),
        };
    };

    return webauthn_internal(payload, success_callback, failure_callback, credentials_callback,
                    complete_payload_callback, beginurl, completeurl,
                    WEBAUTHN_LOGIN_FAIL_BEGIN, WEBAUTHN_LOGIN_FAIL_COMPLETE, ax);
```



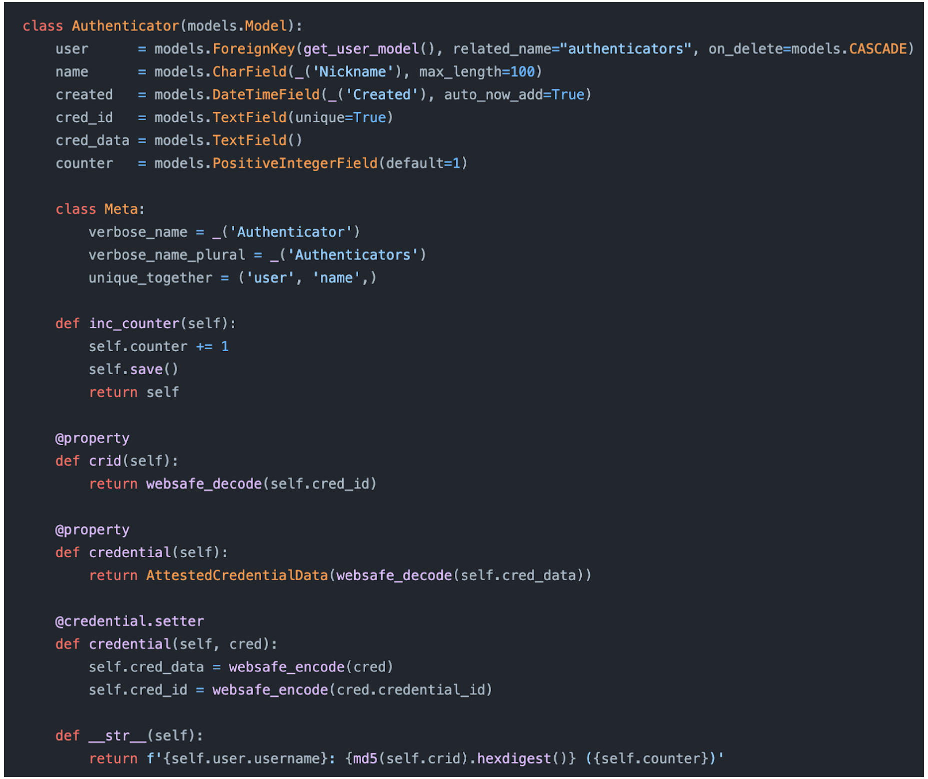
```
export const webauthn_internal = (payload, success_callback, failure_callback,
                credentials_callback, complete_context_callback, beginurl,
                completeurl, begin_failure_code, complete_failure_code, ax) => {
    console.log('webauthn:', {'begin': beginurl, 'complete': completeurl});
    ax.post(beginurl, payload)
      .then(res => res.data)
      .then(opts => credentials_callback(opts))
      .then(auth => {
        ax.post(completeurl, complete_context_callback(payload, auth))
          .then(res => success_callback(res))
          .catch(error => failure_callback(error, complete_failure_code));
      }).catch(error => failure_callback(error, begin_failure_code));

};
```

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

My Demo – Django Models



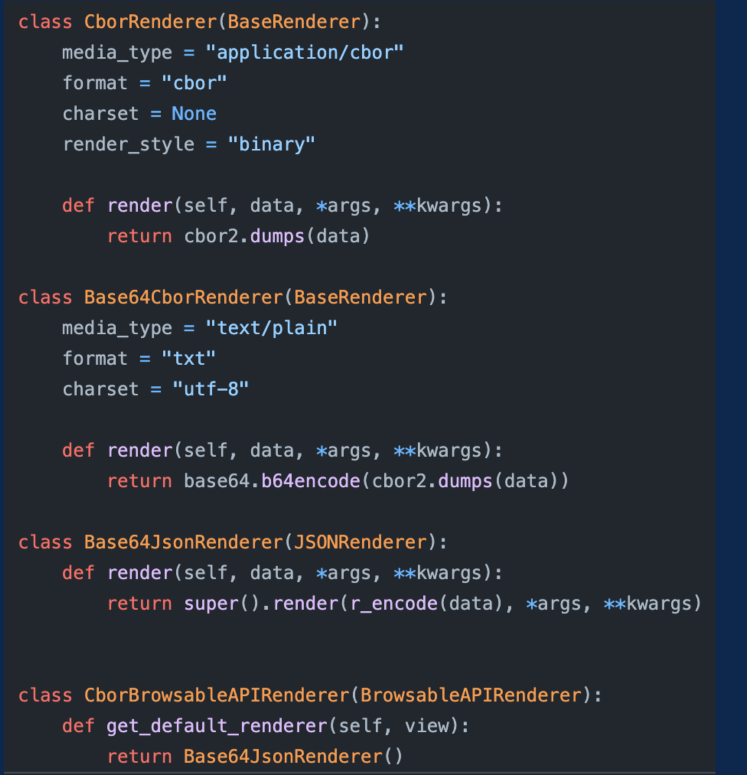
```
class LoginToken(models.Model):
    token = models.CharField(_('Token'), max_length=64, primary_key=True)
    user = models.ForeignKey(get_user_model(), related_name="tokens", on_delete=models.CASCADE)
    created = models.DateTimeField(_('Created'), auto_now_add=True)
    expires = models.DateTimeField(_('Expires'))

    class Meta:
        verbose_name = _('Token')
        verbose_name_plural = _('Tokens')

    @property
    def expired(self):
        return timezone.now() > self.expires

    def redeem(self):
        if not self.expired:
            self.delete()
            return self.user
        return False

    def generate_token(self):
        return tokens.default_token_generator.make_token(self.user)

    def renew(self):
        self.expires = timezone.now() + timezone.timedelta(minutes=EXPIRY)
        self.redeemed = None

    def save(self, *args, **kwargs):
        if not self.expires:
            self.renew()
        if not self.token:
            self.token = self.generate_token()
        return super(LoginToken, self).save(*args, **kwargs)

    def __str__(self):
        return f'{self.user.username}: {self.created}'
```



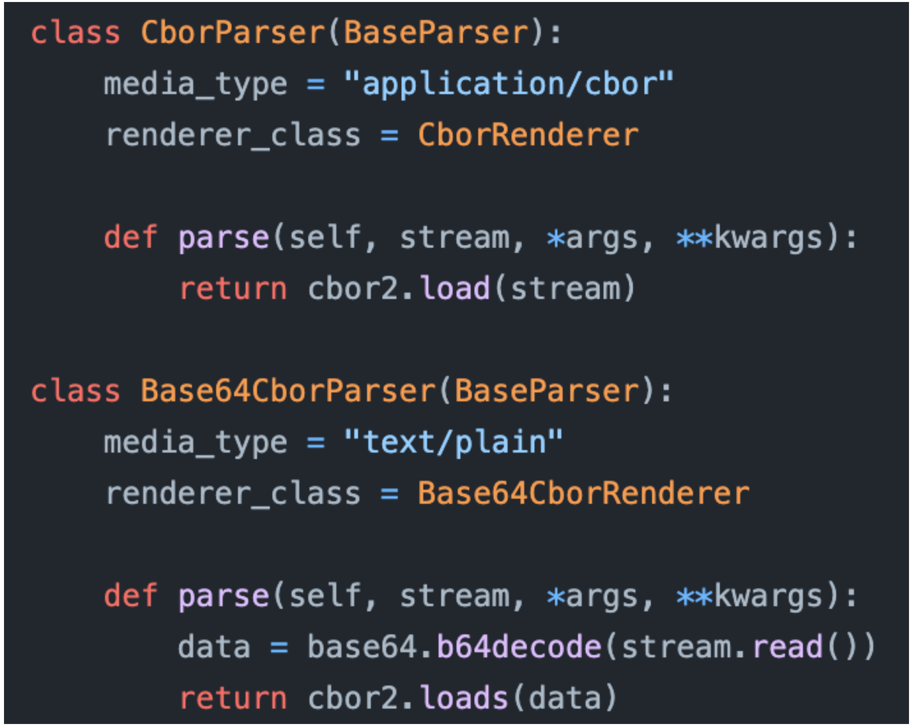
```
class Authenticator(models.Model):
    user      = models.ForeignKey(get_user_model(), related_name="authenticators", on_delete=models.CASCADE)
    name      = models.CharField(_('Nickname'), max_length=100)
    created   = models.DateTimeField(_('Created'), auto_now_add=True)
    cred_id   = models.TextField(unique=True)
    cred_data = models.TextField()
    counter   = models.PositiveIntegerField(default=1)

    class Meta:
        verbose_name = _('Authenticator')
        verbose_name_plural = _('Authenticators')
        unique_together = ('user', 'name',)

    def inc_counter(self):
        self.counter += 1
        self.save()
        return self

    @property
    def crid(self):
        return websafe_decode(self.cred_id)

    @property
    def credential(self):
        return AttestedCredentialData(websafe_decode(self.cred_data))

    @credential.setter
    def credential(self, cred):
        self.cred_data = websafe_encode(cred)
        self.cred_id = websafe_encode(cred.credential_id)

    def __str__(self):
        return f'{self.user.username}: {md5(self.crid).hexdigest()} ({self.counter})'
```

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

My Demo – REST Encoding



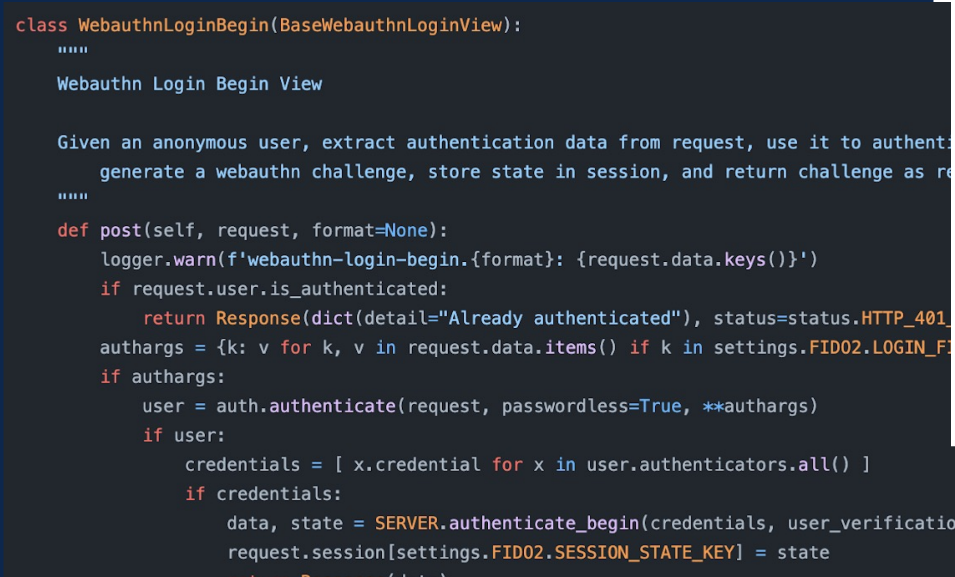
```
class CborRenderer(BaseRenderer):
    media_type = "application/cbor"
    format = "cbor"
    charset = None
    render_style = "binary"

    def render(self, data, *args, **kwargs):
        return cbor2.dumps(data)

class Base64CborRenderer(BaseRenderer):
    media_type = "text/plain"
    format = "txt"
    charset = "utf-8"

    def render(self, data, *args, **kwargs):
        return base64.b64encode(cbor2.dumps(data))

class Base64JsonRenderer(JSONRenderer):
    def render(self, data, *args, **kwargs):
        return super().render(r_encode(data), *args, **kwargs)


class CborBrowsableAPIRenderer(BrowsableAPIRenderer):
    def get_default_renderer(self, view):
        return Base64JsonRenderer()
```



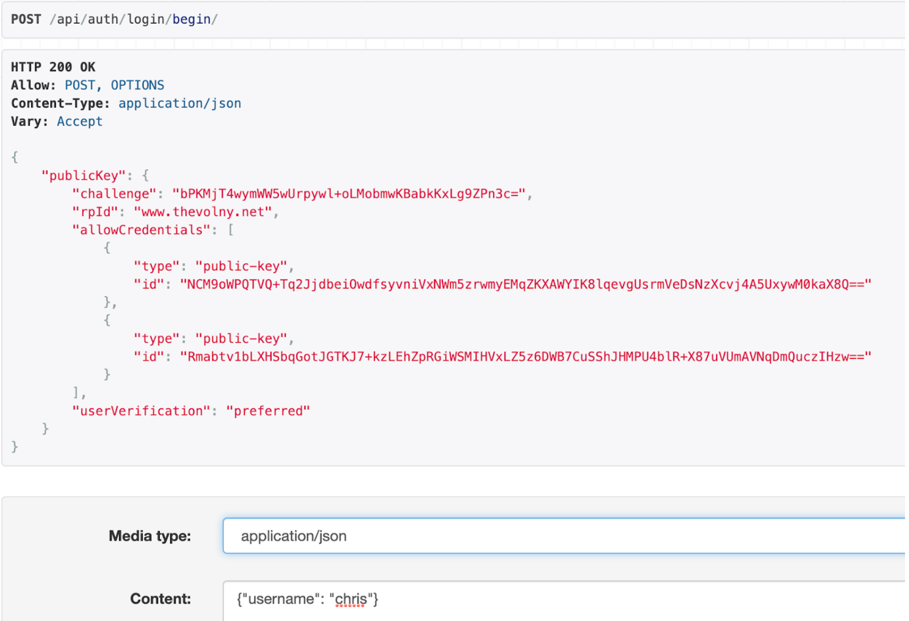
```
class CborParser(BaseParser):
    media_type = "application/cbor"
    renderer_class = CborRenderer

    def parse(self, stream, *args, **kwargs):
        return cbor2.load(stream)

class Base64CborParser(BaseParser):
    media_type = "text/plain"
    renderer_class = Base64CborRenderer

    def parse(self, stream, *args, **kwargs):
        data = base64.b64decode(stream.read())
        return cbor2.loads(data)
```

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

My Demo – Login Begin



```python
class WebauthnLoginBegin(BaseWebauthnLoginView):
    """
    Webauthn Login Begin View

    Given an anonymous user, extract authentication data from request, use it to authenti
        generate a webauthn challenge, store state in session, and return challenge as re
    """
    def post(self, request, format=None):
        logger.warn(f'webauthn-login-begin.{format}: {request.data.keys()}')
        if request.user.is_authenticated:
            return Response(dict(detail="Already authenticated"), status=status.HTTP_401_
        authargs = {k: v for k, v in request.data.items() if k in settings.FIDO2.LOGIN_FI
        if authargs:
            user = auth.authenticate(request, passwordless=True, **authargs)
            if user:
                credentials = [ x.credential for x in user.authenticators.all() ]
                if credentials:
                    data, state = SERVER.authenticate_begin(credentials, user_verification=settings.FIDO2.USER_VERIFICATION)
                    request.session[settings.FIDO2.SESSION_STATE_KEY] = state
                    return Response(data)
                logger.warn(f'No authenticators registered for
                return Response(dict(detail="No Authenticators
            logger.warn(f'Bad authargs {redact(authargs)}')
        else:
            logger.warn(f'Bad payload: {redact(request.data)}')
        return Response(dict(detail="Bad payload"), status=stat
```

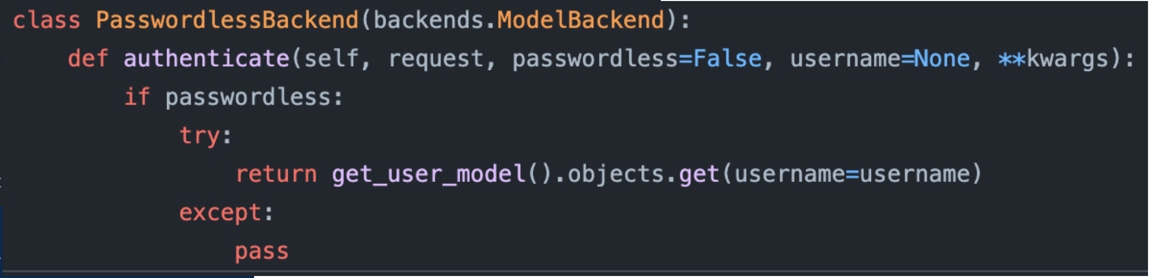


```
POST /api/auth/login/begin/
```

```
HTTP 200 OK
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "publicKey": {
        "challenge": "bPKMjT4wymWW5wUrpywl+oLMobmwKBabkKxLg9ZPn3c=",
        "rpId": "www.thevolny.net",
        "allowCredentials": [
            {
                "type": "public-key",
                "id": "NCM9oWPQTVQ+Tq2JjdbeiOwdfsyvniVxNWm5zrwmyEMqZKXAWYIK8lqevgUsrmVeDsNzXcvj4A5UxywM0kaX8Q=="
            },
            {
                "type": "public-key",
                "id": "Rmabtv1bLXHSbqGotJGTKJ7+kzLEhZpRGiWSMIHVxLZ5z6DWB7CuSShJHMPU4blR+X87uVUmAVNqDmQuczIHzw=="
            }
        ],
        "userVerification": "preferred"
    }
}
```

Media type: application/json

Content: {"username": "chris"}



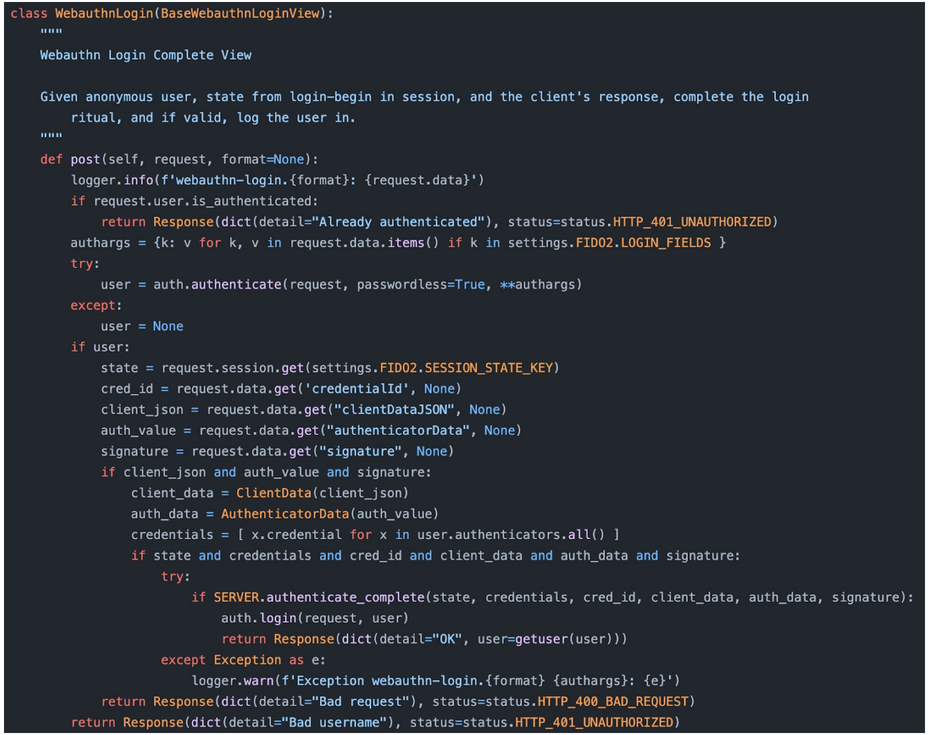
```python
class PasswordlessBackend(backends.ModelBackend):
    def authenticate(self, request, passwordless=False, username=None, **kwargs):
        if passwordless:
            try:
                return get_user_model().objects.get(username=username)
            except:
                pass
```

cisco SECURE

© 2021 Cisco and/or its affiliates. All rights reserved.

My Demo – Login Complete

- Gives:
  - credentialId
  - clientDataJSON
  - authenticatorData
  - Signature
- Gets:
  - Detail + UserInfo
    - username
    - full_name
    - is_staff

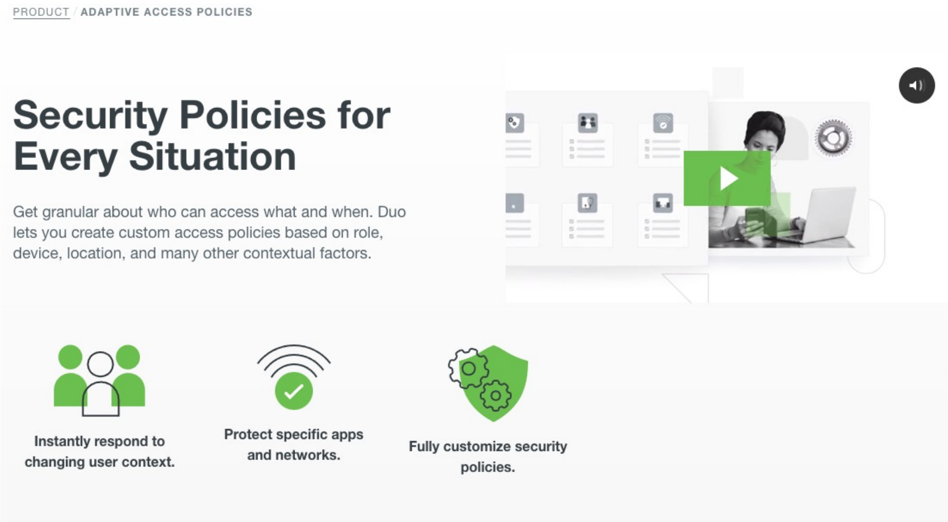


```python
class WebauthnLogin(BaseWebauthnLoginView):
    """
    Webauthn Login Complete View

    Given anonymous user, state from login-begin in session, and the client's response, complete the login
        ritual, and if valid, log the user in.
    """
    def post(self, request, format=None):
        logger.info(f'webauthn-login.{format}: {request.data}')
        if request.user.is_authenticated:
            return Response(dict(detail="Already authenticated"), status=status.HTTP_401_UNAUTHORIZED)
        authargs = {k: v for k, v in request.data.items() if k in settings.FIDO2.LOGIN_FIELDS }
        try:
            user = auth.authenticate(request, passwordless=True, **authargs)
        except:
            user = None
        if user:
            state = request.session.get(settings.FIDO2.SESSION_STATE_KEY)
            cred_id = request.data.get('credentialId', None)
            client_json = request.data.get("clientDataJSON", None)
            auth_value = request.data.get("authenticatorData", None)
            signature = request.data.get("signature", None)
            if client_json and auth_value and signature:
                client_data = ClientData(client_json)
                auth_data = AuthenticatorData(auth_value)
                credentials = [ x.credential for x in user.authenticators.all() ]
                if state and credentials and cred_id and client_data and auth_data and signature:
                    try:
                        if SERVER.authenticate_complete(state, credentials, cred_id, client_data, auth_data, signature):
                            auth.login(request, user)
                            return Response(dict(detail="OK", user=getuser(user)))
                    except Exception as e:
                        logger.warn(f'Exception webauthn-login.{format} {authargs}: {e}')
            return Response(dict(detail="Bad request"), status=status.HTTP_400_BAD_REQUEST)
        return Response(dict(detail="Bad username"), status=status.HTTP_401_UNAUTHORIZED)
```

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Design Considerations

- Single (passwordless) or Multifactor?
  - What's your env's posture?
  - Adaptive?
  - Username-less?
- Requirement Parameters
  - Authenticator type
    - Platform
    - Cross-platform
  - User Verification
    - Warm body, pin, biometric?
  - Attestation Level
    - Identity vs privacy



PRODUCT / ADAPTIVE ACCESS POLICIES

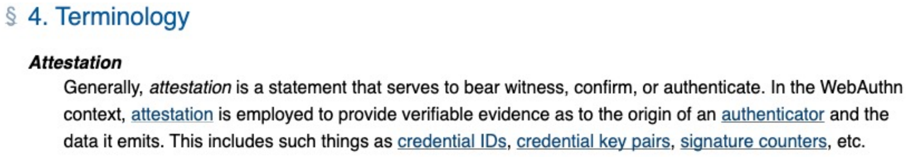
Security Policies for Every Situation

Get granular about who can access what and when. Duo lets you create custom access policies based on role, device, location, and many other contextual factors.

Instantly respond to changing user context.

Protect specific apps and networks.

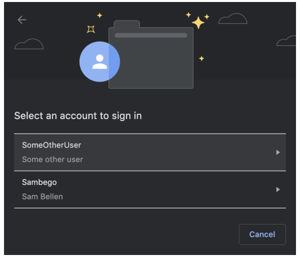
Fully customize security policies.



§ 4. Terminology

Attestation

Generally, *attestation* is a statement that serves to bear witness, confirm, or authenticate. In the WebAuthn context, attestation is employed to provide verifiable evidence as to the origin of an authenticator and the data it emits. This includes such things as credential IDs, credential key pairs, signature counters, etc.



Select an account to sign in

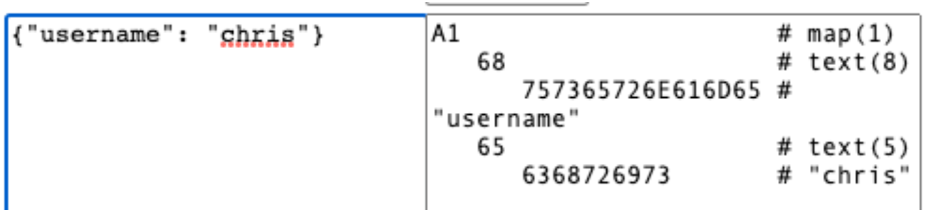
SomeOtherUser
Some other user

Sambego
Sam Bellen

cisco SECURE © 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

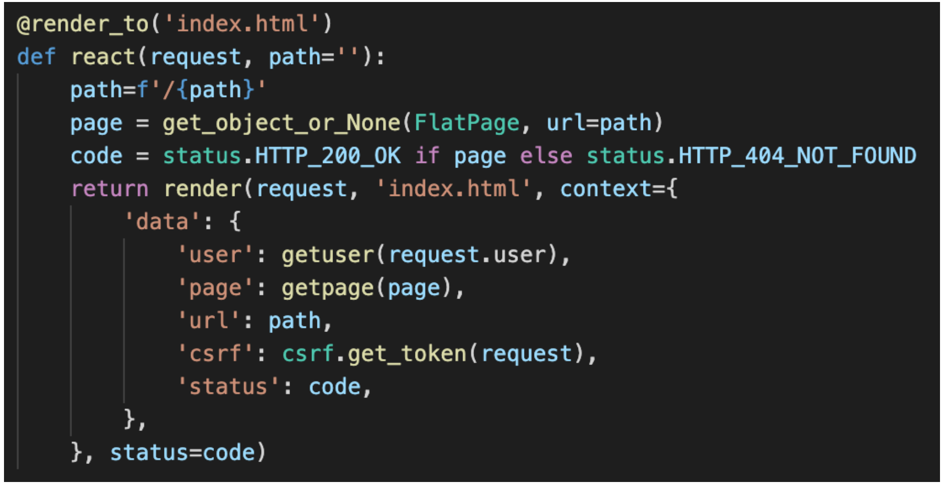
Additional Notes

- Hybrid Password/Passwordless?
  - Challenge for username enumeration
- Do not roll your own crypto/security
- CBOR vs Base64
- Django Views/API to React = awkward
  - JSON blobs?

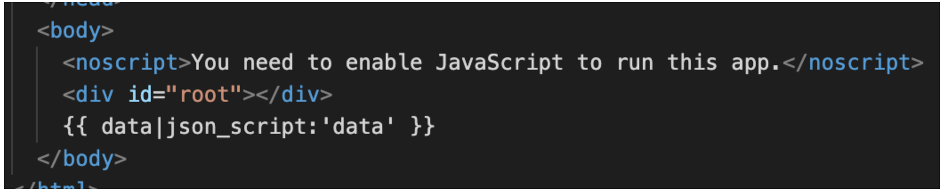


```
{"username": "chris"}
```

```
A1                              # map(1)
   68                           # text(8)
      757365726E616D65 #
"username"
   65                           # text(5)
      6368726973                # "chris"
```



```python
@render_to('index.html')
def react(request, path=''):
    path=f'/{path}'
    page = get_object_or_None(FlatPage, url=path)
    code = status.HTTP_200_OK if page else status.HTTP_404_NOT_FOUND
    return render(request, 'index.html', context={
        'data': {
            'user': getuser(request.user),
            'page': getpage(page),
            'url': path,
            'csrf': csrf.get_token(request),
            'status': code,
        },
    }, status=code)
```



```html
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
    {{ data|json_script:'data' }}
  </body>
</html>
```

© 2021 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Questions?

- WebAuthn 101 https://webauthn.guide/
- Duo WebAuthn Demo https://webauthn.io/
- FIDO Alliance https://fidoalliance.org/
- Django Extension https://github.com/cvolny/django-restauthn/
- React WebAuthn Client Library https://www.npmjs.com/package/@cvolny/webauthn-client

Thank You!

cisco SECURE

Chris Volny (she/her)

Cisco :: Duo Security :: CloudSSO Services :: Sr. Software Engineer