Algorithm Discovery API

WebCrypto API Proposal

Israel Hilerio & Vijay Bharadwaj, Microsoft

Why?

- There are scenarios that have dependencies between specific algorithms in order to create a secure process
  - Not having support for any one of these algorithms invalidates the flow
  - Not knowing which of these algorithms are supported ahead of time creates weird coding patterns
    - The creation of temporary keys to validate support for an algorithm
- There are scenarios where a user can use a subset of algorithms from a supported set to establish a secure connection
  - Not knowing which of these algorithms are supported ahead of time creates weird coding patterns
    - The creation of temporary keys to validate support for an algorithm

Scenario #1

- Imagine you are talking to a web service which has two modes. The server sends a challenge, and then:
  - If you run a PBKDF2 of the user's password to derive an HMAC-SHA512 key, then send the HMAC-SHA512 of the challenge, you are authenticated as a high security user and can see more content
  - Otherwise, you can browse the site in anonymous mode
- How do you know whether it makes sense to offer the user a password prompt or just tell them their browser is restricted to anonymous mode?
  - Do you create a garbage PBKDF2 key and a garbage HMAC-SHA512 key?
  - Or do you prompt the user for a password and then if HMAC-SHA512 is not supported you fail?

Scenario #2

- Imagine you are writing an OTR messaging app. The messaging protocol requires you to offer one or more cipher suites from a given set when you begin a session
  - How do you decide which cipher suites to use without creating a bunch of garbage keys / operations to try out all the various components of each suite?

Proposal

Create a synchronous API that check if a list of algorithms are supported or not. The API will return true if all of the algorithms on the list are supported. Otherwise, it will fail as soon as one of the algorithms on the array list is not supported.

Syntax

```
partial interface SubtleCrypto {
   static boolean areAlgorithmsSupported (AlgorithmIdentifier[] algorithms) ;
};
```

Usage

```
crypto.subtle.areAlgorithmsSupported( [ {name:“PBKDF2”}, {name:“AES-GCM”}] );
```