DRAFT - GitHub Smart Card Integration

Integration Overview

GitHub the website does not support Public Key or Certificate authentication for access. GitHub.com only supports Username/Password, while GitHub Enterprise can be integrated with a SSO system that already manages Smart Card Authentication.

GitHub supports remote repository access over ssh and https protocols. For ssh, it relies on ssh-key authentication, which is compatible with Certificates stored on a smart card. For the https protocol, it only supports username/password and authentication tokens, not certificates or ssh-keys.

PIV and CAC support for Windows

Hardware

Windows has good hardware support for smart card readers, so finding compatible options is trivial. Most Government agencies will already have the hardware on hand, including standalone readers and keyboards.

Outside of standard-issue hardware, I have had success with the IOGEAR GSR-202 Smart Card reader. For a somewhat complete list of the different options and their compatibility, see this webpage.

Middleware

ActivClient is the leader in Windows PIV/CAC enabling software. Many agencies provide licenses and installs for their users as part of their standard baseline. An alternative is the CACKEY software. This also has the ability to change the PIN on PIV cards.

Windows support for git over https with a Smart Card

Natively git does not support using a pkcs#11 certificate over an https request.

However, a process has been created that utilizes Cgywin on Windows (and native tools on other OS's) to enable this.

This isn't necessary for GitHub, as we do not currently support PKI authentication over https

Using Smart Card certificates with SSH

On Windows, Putty is the standard for SSH Clients as it provides a native, GUI experience. While openssh is available for cygwin, this guide assumes that's a bridge too far for most users.

Putty-CAC is a derivative of Putty that has support for pkcs#11 certificates and Smart Cards. Putty-CAC is an open source program, but the core functionality was last updated in 2014. You can find the download and configuration instructions on the Putty-CAC page maintained by Daniel Risacher.

For Putty-CAC to operate, it also needs a middleware to access the Smart Card. See the **Middleware** section for Windows compatible software.

Putty-CAC will also expose the certificate the SSH connection is going to use that you can share with GitHub and your account.

PIV and CAC support for Mac

Installation steps on a Mac

Hardware support

Mac has decent hardware support for Smart Card readers, but not as many options as Windows. All major devices should say on their packaging if they are compatible with MACs and Government issued smart cards.

The hardware list from militarycac.com is still the best resource.

Middleware

ActivClient has a MAC option, but it does not seem to be as well adopted or supported as it's Windows counterparts.
PKard is what I have used and it works well. It will read your CAC,

PIV and CAC Support for Linux

Installation steps on Ubunutu

Middleware

Red Hat (and Cent-based OS's) ship with CoolKey natively installed, however I've always had better luck and features reported with CACKEY.

SSH support

RHEL 7+ and Ubuntu 12.04+ all ship with a version of the openssh client that has native opensc/pcks11 support. Just ensure that OpenSC is installed, and ssh with your smart card with no other work.

OpenSSH-client's native support

Modern versions of openssh portable (5.4p and newer) include native integration with opensc and therefore native support for smart cards like the PIV and CAC. Valid versions are nthe ative SSH client for Macs 10.9+, ships with RHEL 7+ and Ubuntu 12.04+, and a cgwin version is available for Windows.

Accessing keys

I've found the easiest way to access keys is via the pkcs15-tool. This comes with OpenSC, and allows you to access the public keys (and private metadata) from your card. You can then copy them to whatever tool you wish.

Example: Store and verify GitHub Smart Card connectivity

This will take you through using the opensc and openssh toolset to extract your public key, add it to GitHub, and test connectivity.

Locate your Authentication key

List the public keys available on your smart card, identify which one is for your Authentication requests. Look for the name containing "AUTH", and note the ID for the next step.

`pcks15-tool --list-public-keys`

```
  Public RSA Key [PIV AUTH pubkey]
    Object Flags   : [0x0]
    Usage          : [0xD1], encrypt, wrap, verify, verifyRecover
    Access Flags   : [0x2], extract
    ModLength      : 2048
    Key ref        : 154 (0x9A)
    Native         : yes
    Auth ID        : 01
    ID             : 01
    DirectValue    : <absent>

Public RSA Key [SIGN pubkey]
    Object Flags   : [0x0]
    Usage          : [0x2C1], encrypt, verify, verifyRecover, nonRepudiatio
    Access Flags   : [0x2], extract
    ModLength      : 2048
    Key ref        : 156 (0x9C)
    Native         : yes
    Auth ID        : 01
    ID             : 02
    DirectValue    : <absent>

Public RSA Key [KEY MAN pubkey]
    Object Flags   : [0x0]
    Usage          : [0x11], encrypt, wrap
    Access Flags   : [0x2], extract
    ModLength      : 2048
    Key ref        : 157 (0x9D)
    Native         : yes
    Auth ID        : 01
    ID             : 03
    DirectValue    : <absent>
```

Read the Authentication key, store it on GitHub

Fetch the SSH-compatible key format for your Auth key, copy it to your clipboard and add it to your profile on GitHub. As you can see from the example above, the AUTH key is the first key. I can read the first key with the following command that will also format it in ssh-rsa fashion. `pkcs15-tool --read-ssh-key 1`

Use openssh to verify smart card based access

Attempt to create an SSH connection to GitHub to verify your cert is correct. If

working, this should identify you without any cert selection required. Modified from "Test the connection"

MAC Example

```
ssh -I /Library/OpenSC/lib/opensc-pkcs11.so -T git@github.com
```

Linux Example

```
ssh -I /usr/lib/pensc-pkcs11.so -T git@github.com
```