Bypassing Local Windows Authentication to Defeat Full Disk Encryption

black hat®
EUROPE 2015

Ian Haken

Who Am I?

- Currently a security researcher at Synopsys, working on application security tools and Coverity’s static analysis product.
- Previously received my Ph.D. in mathematics from UC Berkeley.
- Twitter: @ianhaken
- Email: ian.haken@synopsys.com

Full Disk Encryption

- A scheme for protecting data at rest. Encrypts an entire disk or volume.
- Mitigates the impact of a threat with physical access; generally does *not* provide protection against remote adversaries.
- Encrypts everything, often including the OS.

Microsoft BitLocker

- BitLocker is Microsoft's proprietary full-disk encryption feature.
- Built into all professional/enterprise versions of Windows since Vista.
- Uses the system's Trusted Platform Module (TPM) to store the master encryption key.

What is a TPM?

- A TPM is a hardware module responsible for performing cryptographic operations, performing attestation, and storing secrets.
- It has fairly general APIs, so how it is used is mostly up to applications.
- Example applications include remote attestation, and storing encryption keys.

Storing Secrets on a TPM

- A TPM contains several Platform Configuration Registers (PCRs).
- Starting with the BIOS (which is assumed to be trusted), the next part of the boot process (e.g. the MBR) is hashed and this value is stored in the a PCR.
- Each stage of the boot process is responsible for hashing the next and storing it in a PCR.

Storing Secrets on a TPM

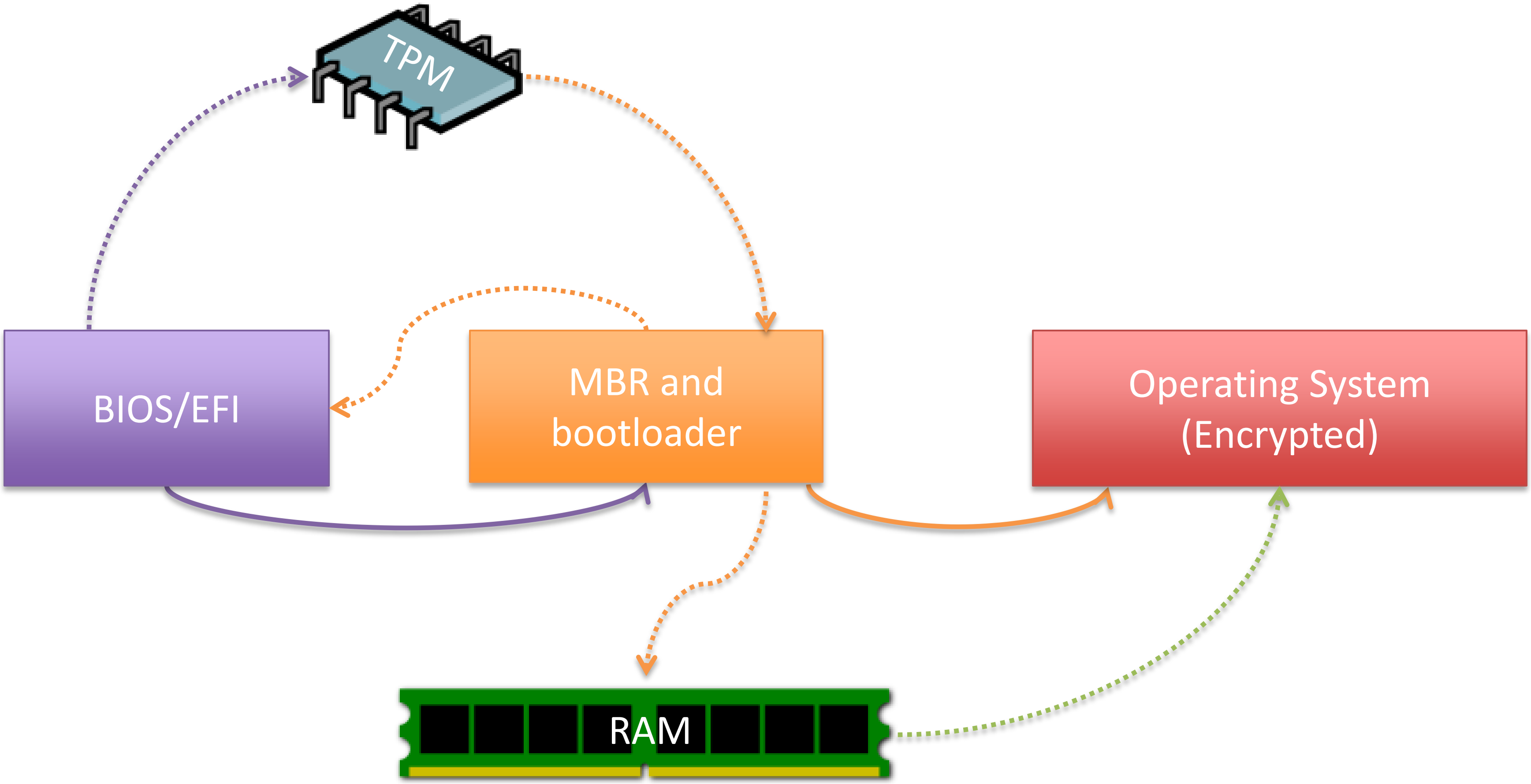
- A boot, the TPM has a zero in all PCR registers.
- Whenever the TPM is told to update a register *r* with a value *v*, it always sets: *r* = HASH (*r* | *v*)
- So PCR values can never get set directly, only appended to. Arbitrary PCR values cannot be spoofed.
- This means a set of values in the PCRs can only be replicated by having that same boot chain.

Storing Secrets on a TPM

- When the TPM stores a secret key, that key can be *sealed*. When a key is sealed, the TPM references the current value of the PCRs.
- An API call to unseal that key will fail unless the current PCR values match the original values from when the key was sealed.
- So effectively, only the original boot process will be able to retrieve that secret key.

Transparent BitLocker

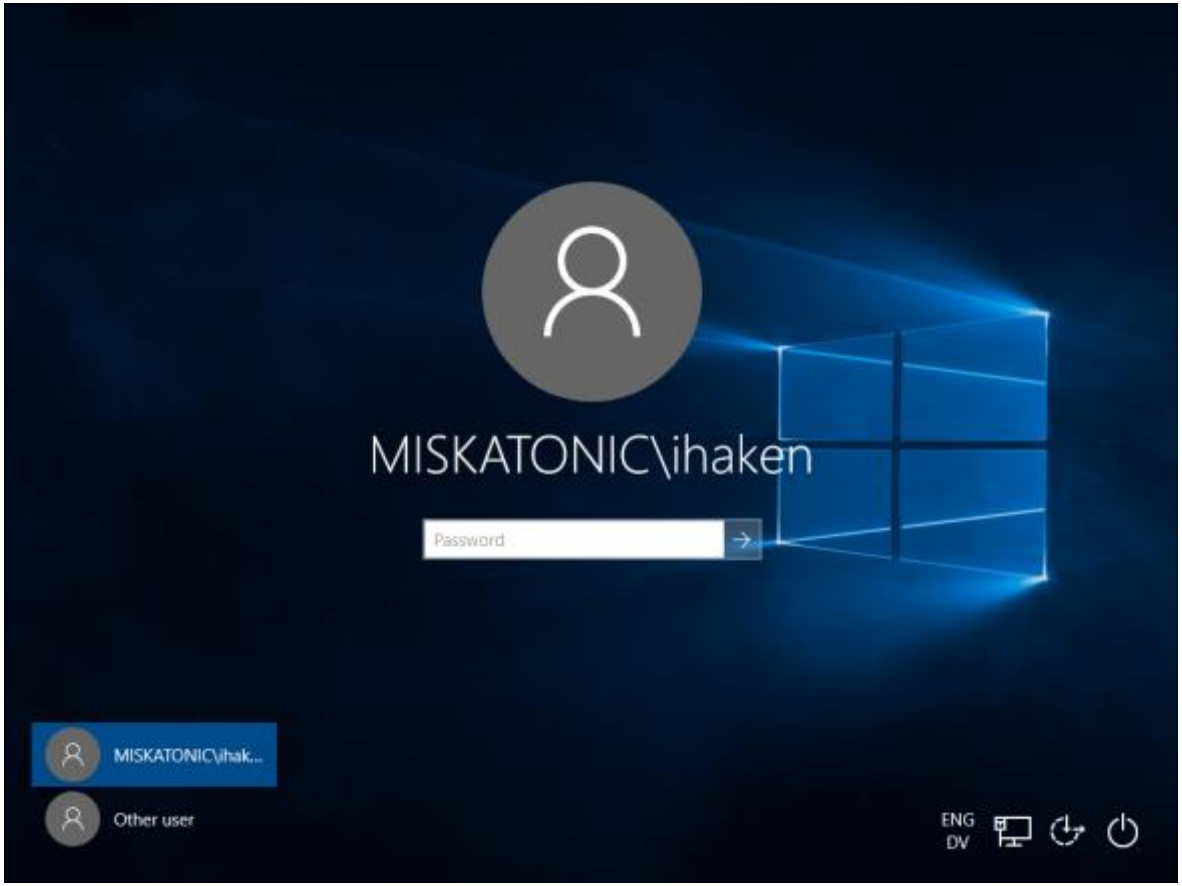
- BitLocker, in addition to the TPM, can optionally require a PIN or a key saved on a USB drive.
- However, it's recommended configuration works transparently. It seals the secret key in the TPM and only BitLocker can retrieve it.
- Your computer boots up to a login screen as usual, with no indication that FDE is enabled.


TPM
BIOS/EFI
MBR and bootloader
Operating System (Encrypted)
RAM

Attacks Given Physical Access

- Known Hardware Attacks
  - Attack the TPM (grounding control pins)
  - Do a cold-boot attack to get the key from RAM
- Attack an early part of the boot chain
  - Flash the BIOS/EFI with a custom image
  - Look for a defect in the BIOS, MBR, or boot loader
- Or see we can attack the OS itself and see if Windows will give us the key...

Booting Up With BitLocker



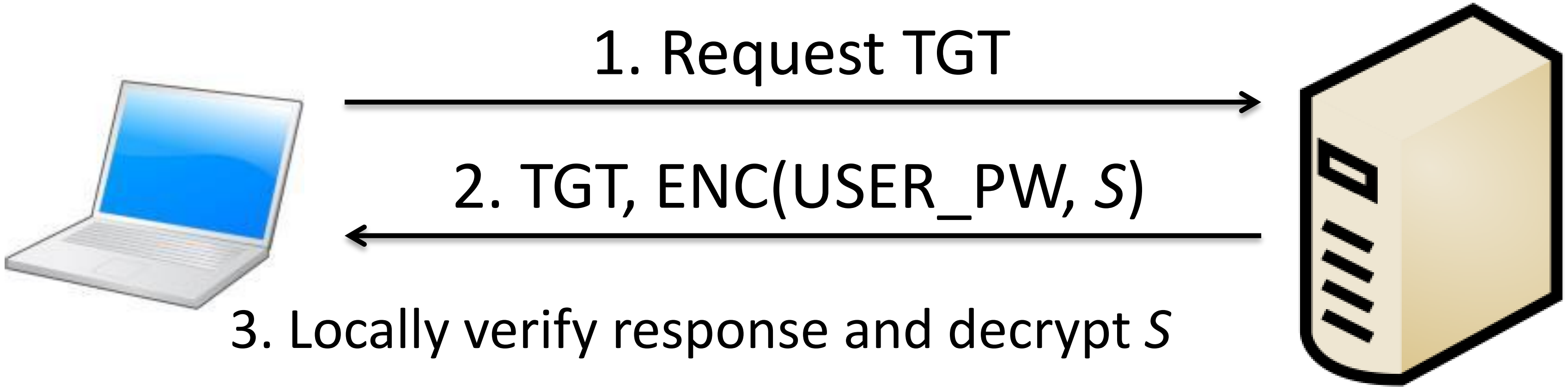
MISKATONIC\ihaken

Local Windows Authentication

- The Local Security Authority (LSA) manages authentication, usually using a Security Subsystem Provider (SSP).
- For a client-domain authentication, the Kerberos SSP exchanges messages with the Domain Controller (DC).
  - When attacking FDE, we have physical access. So we control the network and can run a “mock” DC.

Windows Domain Authentication

- Requests a session ticket (TGT) from the DC.
  - The TGT includes a secret key *S*, encrypted by the DC with the saved user password. Login screen decrypts *S* using the typed password.



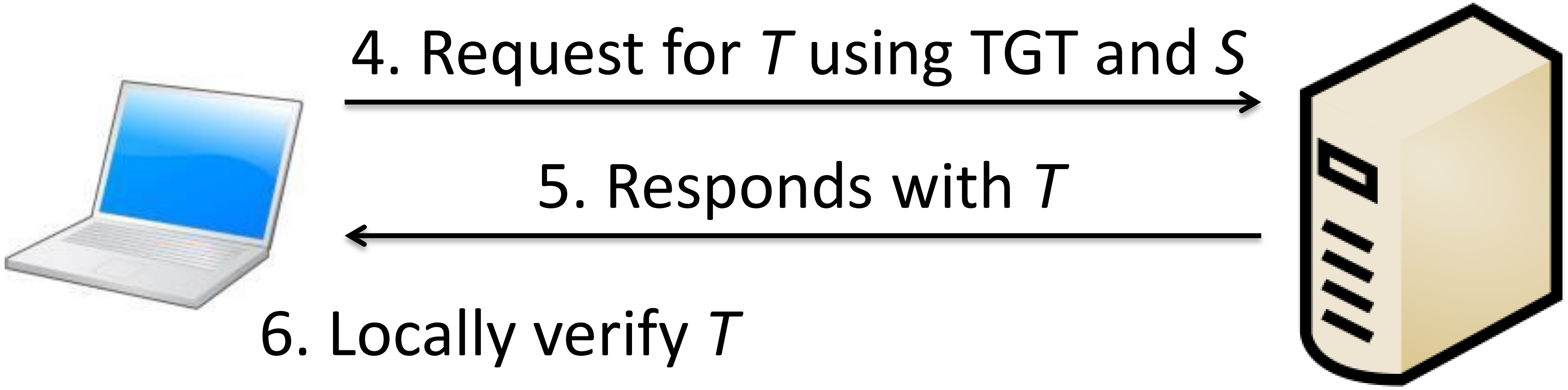
1. Request TGT

2. TGT, ENC(USER_PW, *S*)

3. Locally verify response and decrypt *S*

Windows Domain Authentication

- TGT and *S* are used to request a service ticket *T* from the DC for the target service (in this case, the local workstation).
  - The local workstation verifies *T*.



4. Request for *T* using TGT and *S*

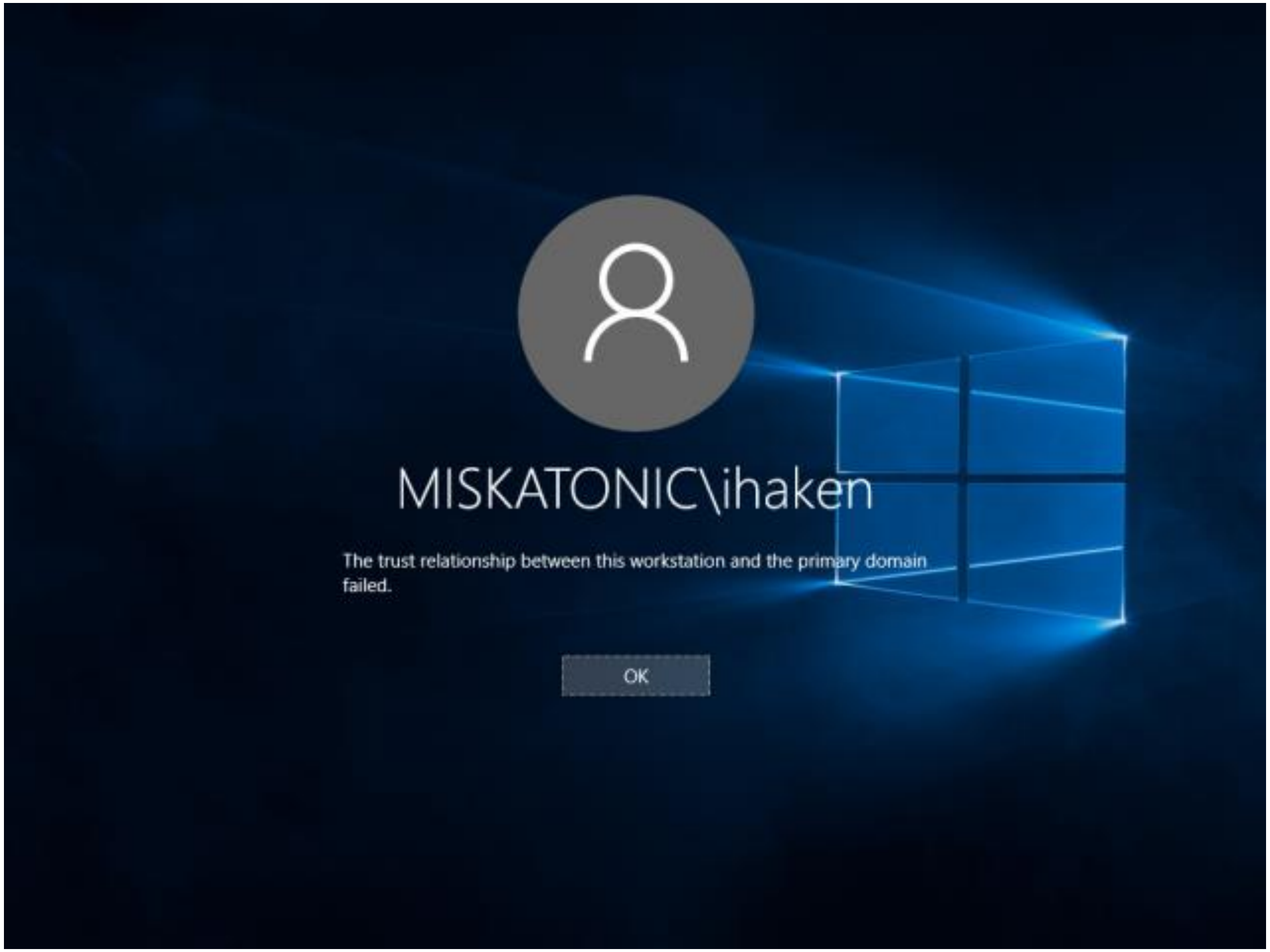
5. Responds with *T*

6. Locally verify *T*

Machine Passwords

- When a workstation first joins a domain...
  - A secret key is generated, called the machine password.
  - This password is sent to the DC, so they have a shared secret for future communication.
- To grant access to the workstation, the login process must present a valid service ticket *T*.
  - This ticket is signed using the machine password.
  - Which we don't have...

If the DC uses the wrong machine password



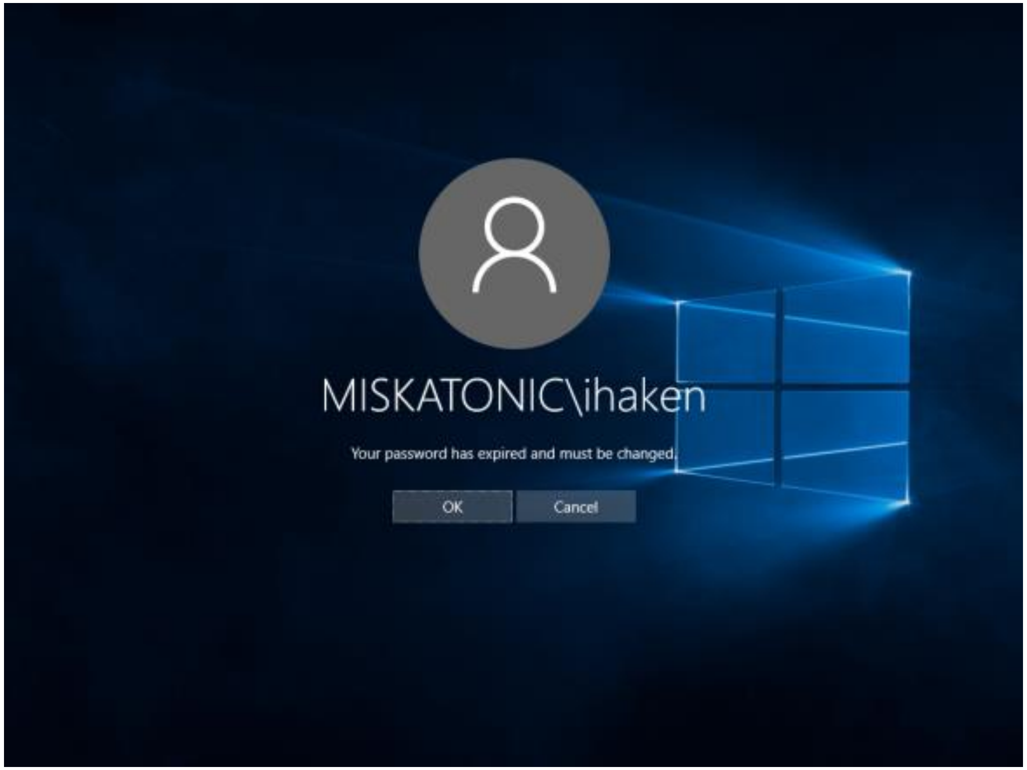
MISKATONIC\ihaken

The trust relationship between this workstation and the primary domain failed.

The Local Credentials Cache

- A user can login when the DC isn’t available
  - Like when you’re using your laptop at a conference during someone’s talk…
- The cache is usually updated whenever the workstation sees the credentials are changed.
  - So it's updated when you successfully login and were authenticating against the DC.
  - Also updated when you change your domain password.

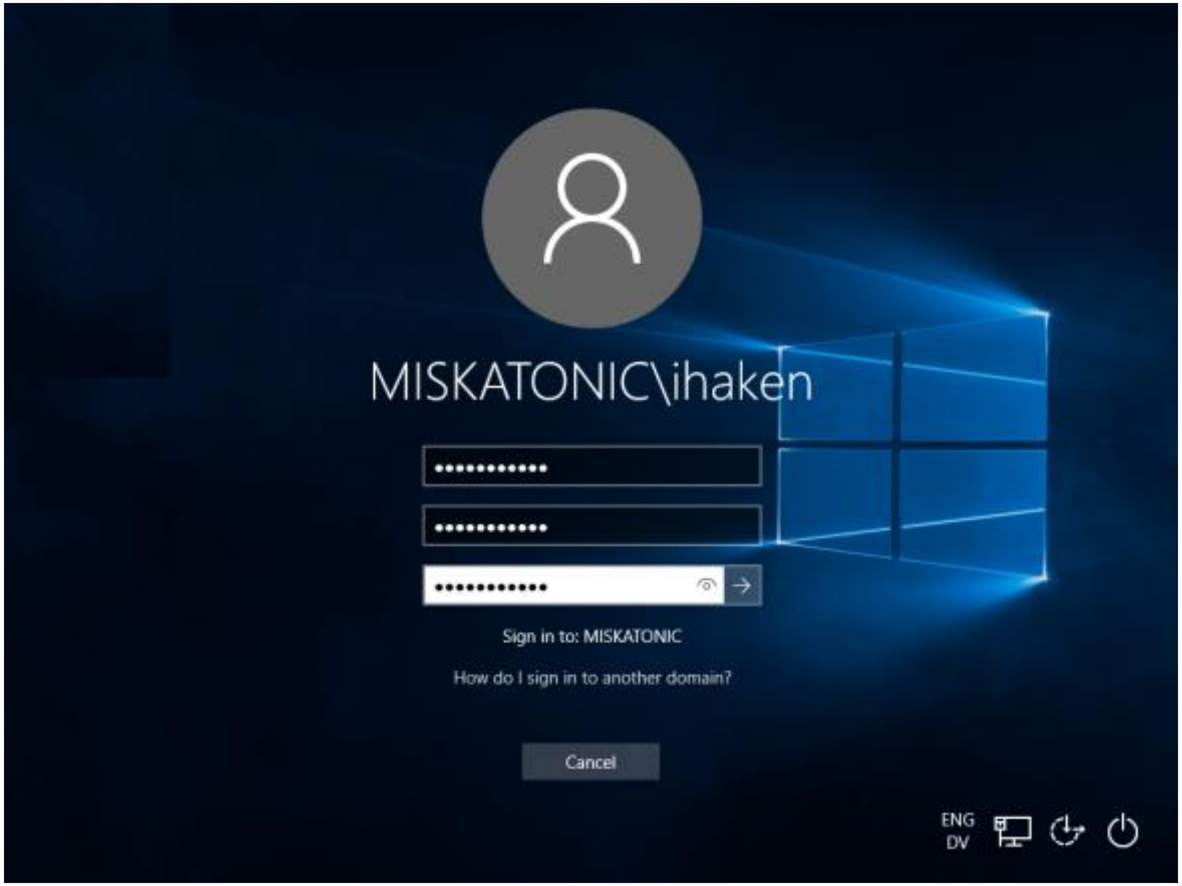
Too Bad We Can't Change the Password On the Login Screen



MISKATONIC\ihaken

Your password has expired and must be changed.

Password Reset



MISKATONIC\ihaken

••••••••••••

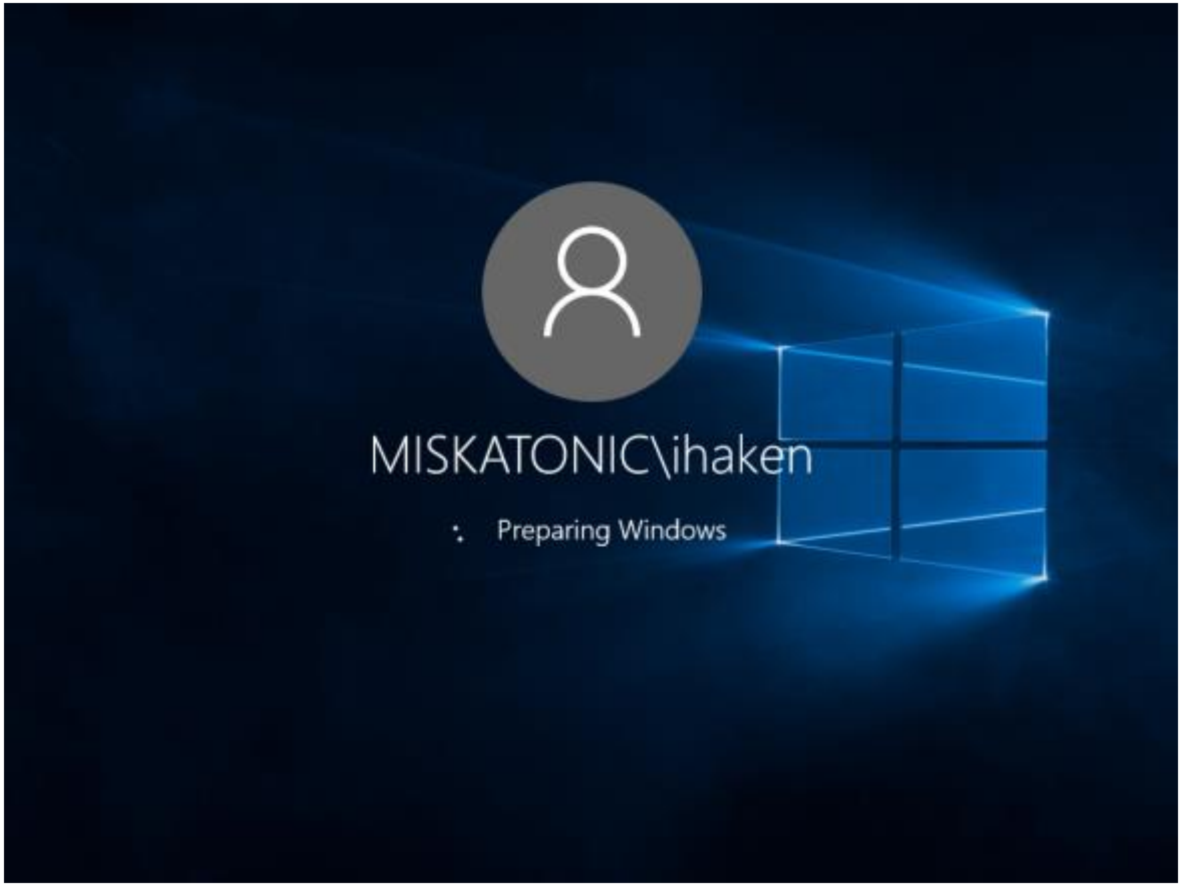
••••••••••••

••••••••••••

Sign in to: MISKATONIC

How do I sign in to another domain?

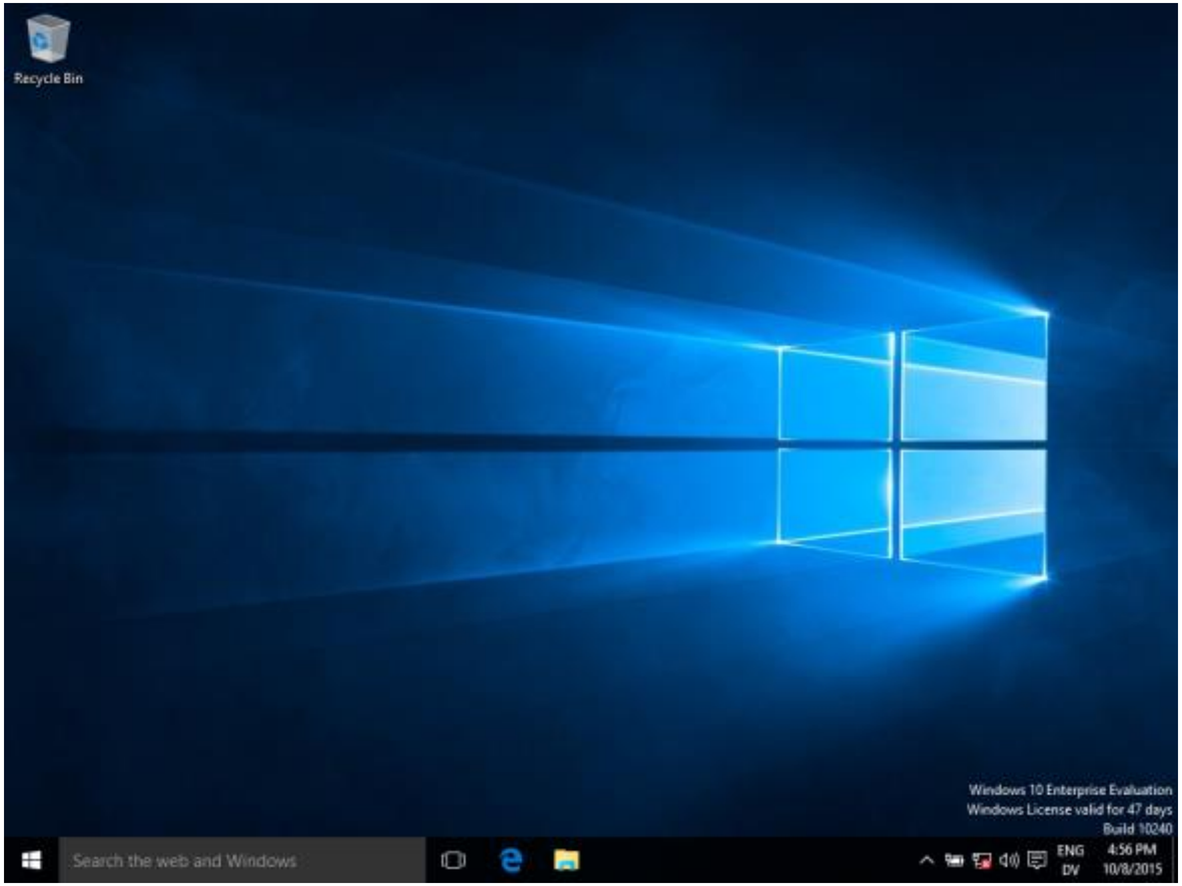
Poisoned Credentials Cache



MISKATONIC\ihaken

Preparing Windows

Poisoned Credentials Cache



Windows 10 Enterprise Evaluation
Windows License valid for 47 days
Build 10240

What Now?

- Dump the BitLocker key from kernel memory
  - As long as the domain account is a local admin
  - Although at this point you already have access to all the local user files, so it's pretty moot.
- Just dig through personal data
  - Saved passwords, Outlook emails, source code…
  - Drop in a trojan / backdoor, or whatever other malware you like.

Demo

System Configurations Effected

- Applies to any computer with:
  - BitLocker without pre-boot authentication
  - Attached to a domain
  - With a least one person having logged in with a domain account.
- Tested on Windows Vista, Windows 7, and Windows 8.1, Windows 10.
  - (Also Windows XP and Windows 2000)

How Else Does This Attack Apply?

- This isn't really BitLocker specific. More generally, this is an authentication bypass for domain accounts.
- If someone is logged in, locks their screen, and steps away, you could use this to unlock the PC.
  - Someone on their laptop at a coffee shop.
  - A computer in an office.

Impact and Mitigation

- This is 100% reliable attack, software-only, low sophistication, and takes a matter of seconds.
- You could use BitLocker with pre-boot authentication (i.e. using a PIN or USB key)
- You could use a BIOS password on boot
- Microsoft is releasing an update to address the issue. Expected release is November 10.
  - ACK to the Microsoft Security Response Center

Reflections: Why Does This Work?

- The protocol for password changes was written in RFC 3244 for Windows 2000, publish in 2002.
- At that point, local access was total access. Local access wasn't a valid threat model during protocol design.
- But local access is precisely the threat model under which FDE is applicable.

Black Hat Sound Bytes

- A defect in Windows domain authentication means BitLocker Full Disk Encryption can be bypassed; the attack is fast and non-technical.
- Microsoft is releasing a patch for the issue (expected November 10). *Make sure all your workstations are up-to-date!*
- Threat models change; when they do, you need to re-evaluate previous security choices.