**Dijital Hafızanın Sır Perdesini Aralayan Araç: Volatility Nedir ve Nasıl Kullanılır?**

Bir bilgisayar kapatıldığında, sabit diskteki verilerin çoğu korunur. Ancak o an bellekte (RAM) ne vardı? Hangi programlar çalışıyordu, hangi ağ bağlantıları aktifti, hangi kullanıcı komutları yazılmıştı veya panoya hangi şifre kopyalanmıştı? Bu "geçici" veya "uçucu" (volatile) veriler, bir siber saldırının en kritik kanıtlarını barındırır ve bilgisayar kapatıldığı anda sonsuza dek kaybolabilir. İşte bu paha biçilmez dijital delilleri, kaybolmadan önce yakalayıp analiz etmemizi sağlayan aracın adı **Volatility**'dir.

Bu yazıda, Volatility'nin ne olduğunu, dijital adli bilişim (forensics) ve olay müdahale (incident response) için neden bu kadar hayati olduğunu, nasıl kurulduğunu ve bu dijital mikroskop ile bir bellek dökümünden (memory dump) nasıl inanılmaz bilgiler çıkarabileceğinizi pratik örneklerle keşfedeceğiz.

**1. Volatility Aracı Nedir?**

**Volatility**, çalışan bir bilgisayarın belleğinin (RAM) anlık görüntüsünün ("memory dump" veya "memory image") analiz edilmesi için tasarlanmış, açık kaynaklı, komut satırı tabanlı bir dijital adli bilişim aracıdır. Python ile yazılmış olan Volatility Framework, bir bellek dökümü içindeki ham verileri anlamlı bilgilere dönüştürür.

Kısacası, bir siber saldırı anında veya şüpheli bir aktivite tespit edildiğinde, o anki sistem belleğinin bir kopyası alınır. Volatility, bu kopya üzerinde bir "dijital otopsi" gerçekleştirerek saldırının izlerini sürer. Bu izler arasında şunlar bulunabilir:

* Çalışan işlemler (processes) ve bu işlemlerin kullandığı DLL dosyaları.
* Aktif ve sonlandırılmış ağ bağlantıları (IP adresleri, portlar).
* Çalıştırılmış komutlar (komut istemi geçmişi).
* Açık dosyalar ve kayıt defteri (registry) anahtarları.
* Bellekte saklanan şifreler, hash'ler ve şifreleme anahtarları.
* Zararlı yazılımların (malware) gizlenmiş kodları veya işlemleri.

Volatility, bu analizleri yapabilmek için farklı işletim sistemlerinin (Windows, Linux, macOS) bellek yapısını tanıyan profillere sahiptir.

**2. Neden Kullanılır?**

Volatility, bir olay müdahale uzmanı veya adli bilişim analisti için vazgeçilmezdir, çünkü:

* **Disk Tabanlı Analizden Kaçan Delilleri Bulur:** Birçok modern zararlı yazılım "dosyasız" (fileless) olarak çalışır, yani kendilerini diske yazmadan doğrudan bellekte yaşarlar. Bu tür tehditler sadece bellek analizi ile tespit edilebilir.
* **Saldırının Anlık Resmini Sunar:** Saldırı anında sistemin durumunu tam olarak gösterir: Hangi zararlı işlem çalışıyordu, hangi komutla sisteme sızıldı, hangi C2 (Komuta-Kontrol) sunucusuna bağlandı?
* **Şifre ve Kimlik Bilgilerini Ortaya Çıkarır:** Kullanıcıların oturum açarken kullandığı şifrelerin hash'lenmiş halleri (örneğin, LSASS işlemi içinde) veya düz metin halindeki kimlik bilgileri bellekten çıkarılabilir.
* **Kök Neden Analizini Kolaylaştırır:** Bir zararlı yazılımın hangi işlem tarafından başlatıldığını (parent-child process relationship) göstererek saldırının başlangıç noktasını bulmaya yardımcı olur.
* **Hukuki Süreçlerde Kanıt Sağlar:** Adli bilişim standartlarına uygun şekilde yapıldığında, Volatility ile elde edilen bulgular mahkemede geçerli dijital delil olarak sunulabilir.

**3. Nasıl Kurulum Yapılır?**

Volatility'nin iki ana sürümü vardır: Volatility 2 (eski ama hala yaygın) ve Volatility 3 (modern ve aktif geliştirilen). Yeni başlayanlar için **Volatility 3** önerilir.

**Volatility 3 Kurulumu (Önerilen Yöntem)**

Volatility 3, Python 3.5+ gerektirir ve tek bir çalıştırılabilir dosya olarak tasarlandığı için kurulumu çok basittir.

# 1. Projeyi GitHub'dan klonlayın

git clone https://github.com/volatilityfoundation/volatility3.git

# 2. Proje dizinine gidin

cd volatility3

Kurulum bu kadar! Artık aracı python3 vol.py komutuyla kullanabilirsiniz. Volatility 3, bellek dökümünün işletim sistemi profilini otomatik olarak tanımaya çalışır, bu da kullanımı büyük ölçüde kolaylaştırır.

**Volatility 2 Kurulumu (Eski Projeler İçin)**

Eski eğitim materyalleri genellikle Volatility 2'yi referans alır. Python 2.7 gerektirir.

# 1. Projeyi indirin

git clone https://github.com/volatilityfoundation/volatility.git

# 2. Dizinine gidin

cd volatility

Kullanım için python2 vol.py komutunu çalıştırmanız gerekir.

**4. Kullanım Parametreleri Nedir?**

Volatility'nin gücü, sunduğu plugin (eklenti) adı verilen komutlarda yatar. Her plugin, bellekten farklı bir bilgi türünü çıkarmak için tasarlanmıştır.

**Volatility 3 Temel Parametreleri:**

* -f <DOSYA> veya --file <DOSYA>: Analiz edilecek bellek dökümü dosyasının yolunu belirtir. Bu temel parametredir.
* <plugin\_adı>: Çalıştırılacak olan eklentinin adıdır. (Örn: windows.pslist.PsList)

**Volatility 2 Temel Parametreleri:**

* -f <DOSYA>: Analiz edilecek bellek dökümü dosyasının yolu.
* --profile=<PROFİL>: Bellek dökümünün alındığı işletim sistemini belirtir. (Örn: --profile=Win7SP1x64). Doğru profili bulmak için önce imageinfo plug-in'i kullanılır.
* <plugin\_adı>: Çalıştırılacak eklentinin adı. (Örn: pslist)

**5. Etkili Kullanım İçin 5 Örnek ve Açıklamaları**

Şimdi, şüpheli bir Windows sisteminden alınmış memory.dmp adlı bir bellek dökümünü Volatility 3 ile nasıl analiz edeceğimize bakalım.

**Örnek 1: Çalışan İşlemleri Listeleme (İlk Adım)**

Bir analize başlarken ilk yapılacak iş, olay anında hangi işlemlerin çalıştığını görmektir.

python3 vol.py -f memory.dmp windows.pslist.PsList

* **Açıklama:** windows.pslist.PsList plug-in'i, bellek dökümü anında aktif olan tüm işlemleri (prosesleri) listeler. Çıktıda her işlemin adını, Process ID (PID), Parent Process ID (PPID) ve başlangıç zamanını görürsünüz. svchost.exe gibi normal bir sistem işlemi gibi görünen ama aslında bir zararlı yazılım olan veya alışılmadık bir ebeveyn-çocuk ilişkisine sahip (örneğin, explorer.exe'nin cmd.exe başlatması) işlemler şüpheli olarak işaretlenir.

**Örnek 2: Aktif Ağ Bağlantılarını Gösterme**

Bir zararlı yazılımın komuta-kontrol (C2) sunucusuyla iletişim kurup kurmadığını anlamak kritiktir.

python3 vol.py -f memory.dmp windows.netscan.NetScan

* **Açıklama:** windows.netscan.NetScan plug-in'i, bellek dökümü anındaki tüm aktif TCP ve UDP ağ bağlantılarını listeler. Çıktıda yerel IP/port, uzak IP/port ve bağlantıyı kuran işlemin PID'si bulunur. Burada, bilinmeyen veya şüpheli yabancı IP adreslerine yapılan bağlantılar, özellikle de svchost.exe veya powershell.exe gibi işlemler tarafından başlatılmışsa, bir C2 bağlantısına işaret edebilir.

**Örnek 3: Komut Satırı Geçmişini Ortaya Çıkarma**

Saldırganın sisteme sızdıktan sonra hangi komutları çalıştırdığını bilmek, saldırının kapsamını anlamamızı sağlar.

python3 vol.py -f memory.dmp windows.cmdline.CmdLine

**Açıklama:** windows.cmdline.CmdLine plug-in'i, çalışan işlemlerin hangi komut satırı argümanları ile başlatıldığını gösterir. Örneğin, bir powershell.exe işleminin -enc parametresi ile uzun ve anlamsız bir Base64 dizisiyle başlatıldığını görürseniz, bu büyük ihtimalle "dosyasız" bir zararlı yazılımın çalıştırıldığına dair güçlü bir kanıttır.

**Örnek 4: Gizlenmiş veya Sonlandırılmış İşlemleri Bulma (Malware Avı)**

Gelişmiş zararlı yazılımlar, standart işlem listelerinden (pslist) kendilerini gizlemeye çalışır.

python3 vol.py -f memory.dmp windows.malfind.Malfind

* **Açıklama:** windows.malfind.Malfind plug-in'i, "code injection" (kod enjeksiyonu) gibi tekniklerle başka işlemlerin içine gizlenmiş veya alışılmadık bellek bölgelerinde çalışan potansiyel zararlı kodları bulur. Ayrıca, bir işlemin bellekteki başlık bilgilerinin (header) disktedeki orijinal dosyadan farklı olup olmadığını kontrol eder. Bu, bir zararlı yazılımın kendini meşru bir işlem gibi (örneğin explorer.exe) gizlediği "process hollowing" saldırılarını tespit etmek için çok etkilidir.

**Örnek 5: Kullanıcı Şifre Hash'lerini Çıkarma (Credential Dumping)**

Saldırganların en büyük hedeflerinden biri, sistemdeki diğer hesaplara yayılmak için yönetici şifrelerini ele geçirmektir.

python3 vol.py -f memory.dmp windows.hashdump.Hashdump

* **Açıklama:** windows.hashdump.Hashdump plug-in'i, Windows'un SAM (Security Account Manager) veritabanının bellekteki kopyasını analiz ederek, sistemdeki tüm yerel kullanıcı hesaplarının NTLM şifre hash'lerini çıkarmaya çalışır. Bu hash'ler daha sonra Hashcat veya John the Ripper gibi araçlarla kırılarak orijinal şifreler elde edilebilir. Bu, saldırganın yanal hareket (lateral movement) yeteneklerini anlamak için kritik bir bulgudur.

**Sonuç**

Volatility, bir dijital dedektifin büyütecidir. Gözle görülmeyen, bilgisayar kapatıldığında yok olacak olan en kritik delilleri ortaya çıkararak bir siber olayın hikayesini anlatır. Dosyasız zararlı yazılımlardan gizli ağ bağlantılarına, çalınan şifrelerden çalıştırılan komutlara kadar, bir saldırının tüm anatomisini belleğin derinliklerinden çıkarmamızı sağlar. Olay müdahale ve dijital adli bilişim alanında kariyer yapmak isteyen herkes için Volatility, öğrenilmesi ve ustalaşılması gereken temel bir yetenektir.

Bu güçlü aracı kullanırken, analiz edilen verilerin hassasiyeti ve yasal süreçlerdeki önemi nedeniyle her zaman etik kurallar ve adli bilişim prosedürleri çerçevesinde hareket etmek esastır.