MANDIANT®
A FireEye® Company

The Audit Log Was Cleared

Stop, drop, shut ‘em down, open up shop

Austin Baker
Jacob Christie

SECURITY CONSULTING

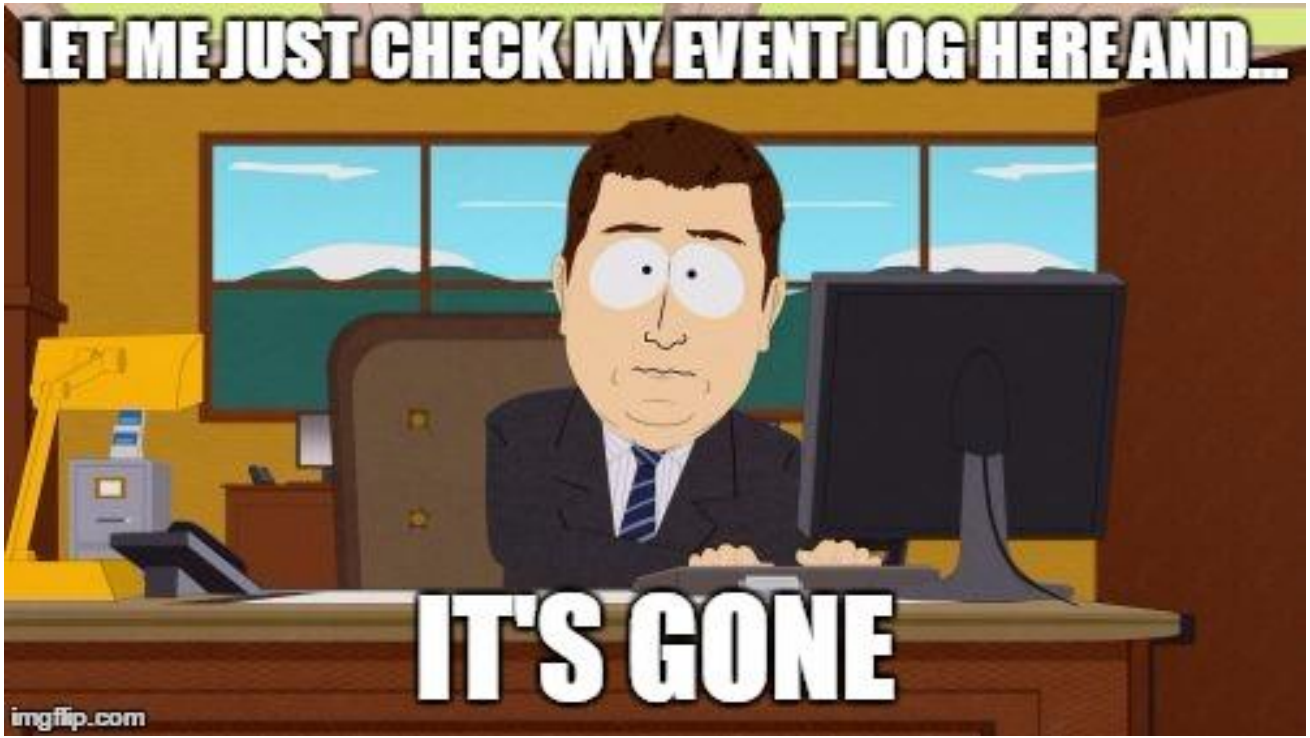
Intros

- Austin Baker
  - Former forensicator gone rogue
  - Semiprofessional Dungeon Master
  - A+ meme material

- Jacob Christie
  - Desktop supporter turned forensicator
  - Supporter of the coffee arts
  - Funny pictures

Agenda

- Event logs and the poisoned well dilemma
- Why we should care and what to do about it
- Event ID 1102: The Event Log has been cleared
- Event log configuration manipulation
- Editing event logs on disk
- Events rule everything around me, S.I.E.M.
- Remembering to forget
- Final thoughts



Event logs and the poisoned well dilemma

- Event logs are an incredibly valuable forensic resource for scoping compromises:
  - Authentication records
  - Service installations, stops, starts
  - If you're particularly cool, PowerShell auditing!

- One issue – trusting information provided by a system that's no longer yours
- Significant effort invested into file/disk antiforensics, not much thought given to event logs
- Why?
  - Other antiforensic methods are more readily available to attackers and red teams
  - Evidence of event log manipulation techniques is hard to find (more on this)
  - SIEMs' make local event log manipulation moot (or do they…)

Why we should care and what to do about it

- Given that, why care?
  - 1. We have evidence of advanced threat actors using event log manipulation techniques
  - 2. There is now a publicly available tool to remove events from logs
    - Without the usual suspects of evidence…
  - 3. Addressing these techniques may lead to discovery of more sophisticated attackers
    - Just knowing someone is playing with your logs reveals quite a bit

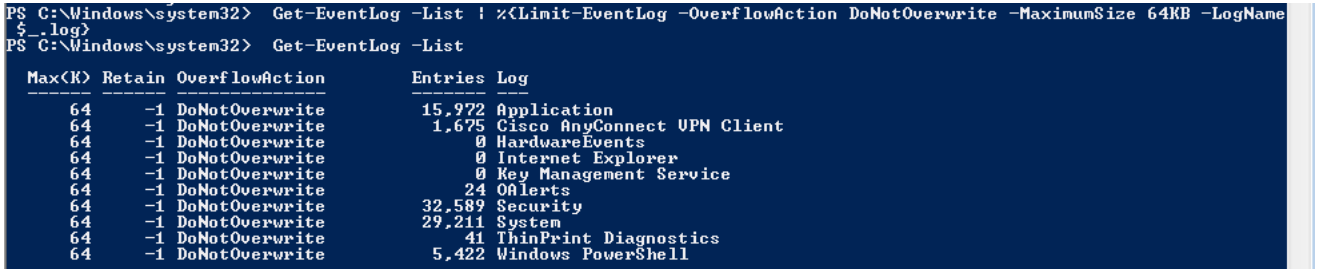
- What we intend to do
  - Address the techniques red teams and attackers use
  - Determine what (if any) forensic artifacts exist around them
  - Describe security posturing necessary to prevent it in the future

EID 1102: The Audit Log was Cleared

- The lazy person's antiforensics
- Noisy, but extremely common for all levels of threats
- Recommendation 1: Cross-correlate time of clearance with filesystem activities (obvi)
- Recommendation 2: Recover events from memory or attempt to carve them from disk

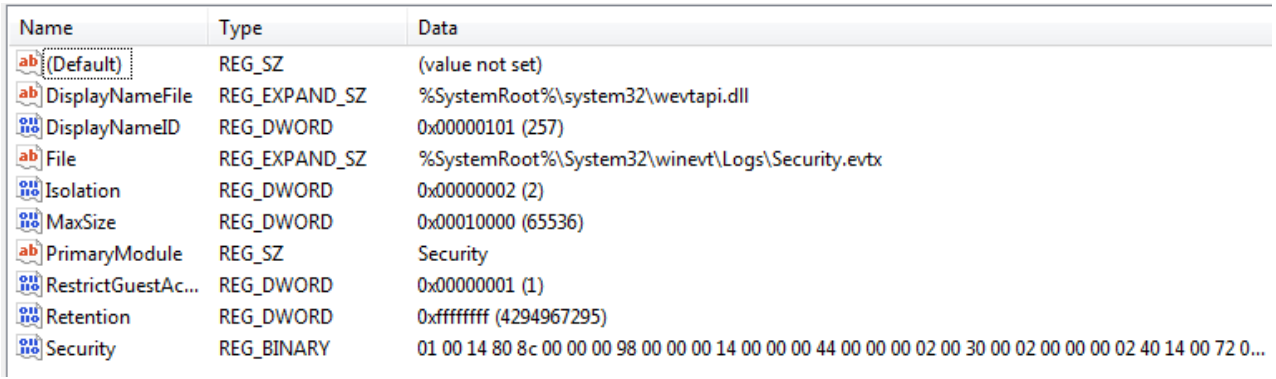
Reconfiguration FTW: A Brief How-To

- One step above clearing the entire log
- Reconfigure the targeted log to:
  - Drop new logs instead of overwriting old ones
  - Reduce the log to the minimum size
- After you're done, reinflate log size and poof, mischief managed



Reconfigure EVTX Size: WHAT NOW?

- How to detect actively:
  - Watch for modification to event log configuration registry keys, particularly Retention and MaxSize
- How to detect forensically:
  - Correlate registry modified timestamps to potential malicious activity
  - Carve EVTX chunks from slack space or recover from memory (as usual)



Name	Type	Data
(Default)	REG_SZ	(value not set)
DisplayNameFile	REG_EXPAND_SZ	%SystemRoot%\system32\wevtapi.dll
DisplayNameID	REG_DWORD	0x00000101 (257)
File	REG_EXPAND_SZ	%SystemRoot%\System32\winevt\Logs\Security.evtx
Isolation	REG_DWORD	0x00000002 (2)
MaxSize	REG_DWORD	0x00010000 (65536)
PrimaryModule	REG_SZ	Security
RestrictGuestAc...	REG_DWORD	0x00000001 (1)
Retention	REG_DWORD	0xffffffff (4294967295)
Security	REG_BINARY	01 00 14 80 8c 00 00 00 98 00 00 00 14 00 00 00 44 00 00 00 02 00 30 00 02 00 00 00 02 40 14 00 72 0...

EVTX Rewrite: The Gist

- Why waste all this time when you can just rewrite the file?
  - Remove event logs, or better yet, change key pieces of event log information
- Well, if this is so good, why hasn't anyone do-
  - 1. They have, but –
  - 2. It's not trivial
  - 3. Have to gain access to the file (usually locked)
- But fortunately, there are several great .evtx parsing utilities to give us insight into what we (or attackers) need!

EVTX Rewrite: EVTX Structure Refresher

- EVTX file separated into different “chunks”, containing binary XML
  - Each chunk defines first and last record numbers, headers, offsets, and…
  - Checksums! Header and data checksums, to be specific
- Checksum is CRC32 of data in question, intended for low-latency integrity guarantee
- Adding a little code onto a forensic parser yields a new tool:
  - 1. Iterate through chunked records
  - 2. Identify records of interest (maybe field matches, record numbers, or EIDs…)
  - 3. Scrub (zero out) or overwrite
- Unfortunately for us, a tool like this already exists and is publically available

EVTX Rewrite: What to do about it

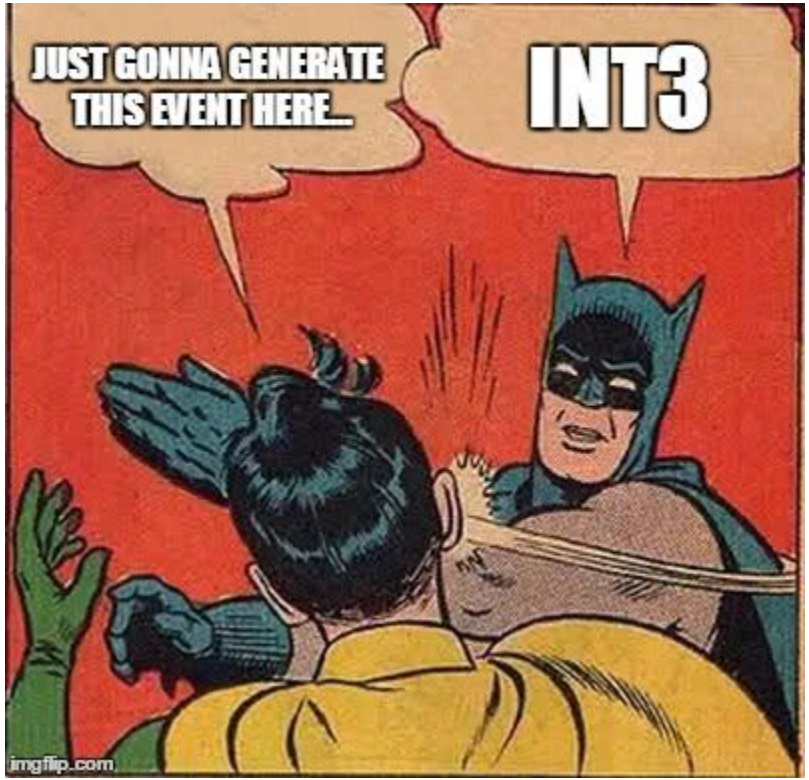
- How to detect actively
  - Look for attempts to release the lock on .evtx files, especially stopping the event logging service
- How to detect forensically
  - For systems of particular interest, cross-correlate SIEM logged events against the local cache
    - What you find (or don't find) may surprise you
  - Memory analysis (or historical backups, like VolumeShadowCopy) may provides some additional log differences to key in on
  - Carving far less likely to work ☹

Events rule everything around me, S.I.E.M.

- But doesn't a good SIEM address all of these issues
  - Well, yes actually
- Best practices tends to have that effect but:
  - Centralized logging will only make that difference if it is carefully preserved
- Self-assessment check list:
  - If someone stops my endpoint log forwarding service, do I know about it?
  - If someone changes the firewall to prevent my log forwarder from communicating, do I know about it?
  - If a system doesn't synchronize any logs for several days, do I know about it?
- These will get you extremely far, but reliance on log forwarding can create other issues…

Remembering to forget

- Rise of mechanisms like WMI and Event Tracing (ETW) have increased insight into how event logging really works in Windows
- Large series of data channels passed down to an eventual log write using APIs
  - A series of tubes…
- With a little bit of hooking, you can get event logging to do some interesting things



Remembering to forget: Looking upstream

- Simple example: I don't like people knowing when processes run
  - EID 4688 - A new process has been created.
- How do I prevent those EIDs from being written to my logs?
  - Fishing upstream!
- Example using a debugging style tool like FlareQDB

```
20  class EVENT_HEADER_ABBREV(vstruct.VStruct):
21      def __init__(self):
22          vstruct.VStruct.__init__(self)
23          self.Size = v_uint16()
24          self.HeaderType = v_uint16()
25          self.Flags = v_uint16()
26          self.EventProperty = v_uint16()
27          self.ThreadId = v_uint32()
28          self.ProcessId = v_uint32()
29          self.TimeStamp = v_uint64()
30          self.ProviderId = GUID()
31          self.EventDescriptor = EVENT_DESCRIPTOR()
32
33  class EVENT_RECORD_ABBREV(vstruct.VStruct):
34      def __init__(self):
35          vstruct.VStruct.__init__(self)
36          self.EventHeader = EVENT_HEADER_ABBREV()
37
38  def event_callback(p, q, trace, **kwargs):
39      evt = q.readstruct(EVENT_RECORD_ABBREV, q.vex('rcx'))
40
41      source = str(evt.EventHeader.ProviderId).upper()
42      id = evt.EventHeader.EventDescriptor.Id
43
44      # Process creation / Microsoft-Windows-Security-Auditing
45      if id == 4688 and source == '{54849625-5478-4994-A5BA-3E3B0328C30D}':
46          print('Process creation detected')
47          # Advance rip past logging
48          q.r('rip', 'wevtsvc+0x27B7')
```

Remembering to forget: Prevention and detection

- Detection?
  - Outside of memory forensics or hook detection, not much
  - No disk artifacts, no SIEM data to crosscheck ☹
- Better option: stop it from happening
  - Go further upstream yourself – ETW-based tampering or detection capabilities
  - Make it harder (but still feasible) with serious application whitelisting (e.g. DeviceGuard)

Final thoughts

- Primary goal is to shine a light on an oft-ignored area of antiforensics
  - Based on recent events, it won't be ignored much longer
- Developing IOCs to detect attempts at executing these bypasses is critical
- Robust SIEM logging and alerting helps a lot
  - Even more if you're watching it closely with analytics (gaps in logs, etc.)

Thanks and Questions

- Muchos gracias to Michael Bailey (@mykill and baileysoriginalirishtech.blogspot.com) for some reversing/debugging advice with EVTX
- Shout out to @williballenthin for his fantastic EVTX parser and EVTXtract