Refuting the Claim of Evidence Tampering

It would seem there are three explanations for how Google Map artifact showing the site where Nancy Cooper's body was found appeared on Brad Cooper's work computer:

1) The artifact was created at the date/time specified on the hard drive [i.e., Friday 7/11/2008 starting at 17:14:53 UTC (1:14:53 EDT)] by the defendant

2) The artifact was created at the date/time specified on the hard drive by someone else to frame the defendant

3) The artifact was created by someone else to frame the defendant at some point after Nancy's body was found and placed onto the computer before forensic analysis was conducted

Setting aside the original claim that this was a random act against Nancy while she was jogging, options 2 and 3 would require someone to have either physical or remote access to the defendant's computer in order to place the files onto the computer or cause them to be created by the operating system normally. In order to have remote access to the computer (i.e., across a network connection), the "hacker" would first have to be on the same network as the defendant's computer and secondly break into the computer or login as user bracoope (or the administrator account) to create the files locally on the defendant's computer.

The defense expert has submitted two videos showing the ease in which someone could capture a Wired Equivalent Privacy (WEP) security key for a wireless network and then use the key and knowledge of the wireless network name (Cooperhome) to gain access to the network. WEP was included as the privacy of the original IEEE 802.11 standard (the specification for transmitting data wirelessly) ratified in September 1999. WEP is a relatively insecure protocol – since 2001 information has existed describing how the protocol could be cracked and communications intercepted. It was superseded by Wi-Fi Protected Access (WPA) in 2003. Finally in 2004, with the ratification of the full 802.11i standard (i.e., WPA2), the IEEE declared that both WEP-40 and WEP-104 "have been deprecated as they fail to meet their security goals".

In order to break into a wireless network with WEP security, one must be within physical proximity to the wireless router for the network in question – i.e., be close enough to get a signal so that you could connect to the network if you had the proper key. Various tools are publicly available to capture

packets from a wireless router that are being broadcast, and once enough packets are obtained there are several tools that allow a user to determine the security key for the WEP network. This could be done over the course of a few minutes once someone using such a tool had verified which network they wanted to collect the packets (since only packets for the target network can be used to determine the key). After determining the network security key, the user would simply enter the network name and key and connect to the network as any computer on the network legitimately could.

Some caveats here. The intruder would have to be close enough to the home router to get the signal, capture the packets and then authenticate onto the network. If the home network had Media Access Control (MAC) address filtering enabled, just knowing the network name and security key would NOT allow an intruder onto the network. MAC filtering is an option that is enabled at the router level so that only authorized devices (based on their network connection MAC address) can be on a network. I don't have any of the routers and can't say if this feature was enabled, but it is a security feature for an additional layer of security. The intruder would have to be physically close enough to the wireless router once connected, to then break into the defendant's work computer, and then actually place whatever content on the computer that the intruder wants.

With Option 2, someone with physical access would login as user bracoope, open Internet Explorer, and visit Google Maps to place content on the defendant's computer. The content would be created in real time by the operating system, so the file creation timestamps would be legitimate. This would require prior knowledge that the "hacker" was going to kidnap Nancy the following morning (per the original alibi), kill her and dump her body in the location found on Google Maps.

With Option 3, someone would first have to create the Google Map content after the body is discovered. Those files would all have to be modified to make their file creation timestamps appear to be made on 7/11/2008. The files would then have to be saved to the defendant's hard drive in the folder where temporary Internet content is legitimately created and stored by Internet Explorer. The Master File Table (MFT) of the defendant's computer would reflect that these files were created out of sequence of the files legitimately created between 7/11 and 7/16 however, and the defendant's MFT shows no evidence of such tampering.

The file system in use on the defendant's computer, NTFS, stores time values in UTC format, so they are not affected by changes in time zone or daylight saving time. All files created on a hard drive formatted NTFS store file creation as a unique value Microsoft calls FILETIME. This is a 64-bit value representing the number of 100-nanosecond intervals since January 1, 1601 at 12:00:00 AM UTC. To put this into perspective, there are 10,000,000 100-nanosecond intervals in one second. If a person creates a file on January 01, 1601 at 00:00:01hrs, then the value stored on disk will be 0x989680 (10,000,000 decimal). That is extremely precise and I think most forensic examiners are aware of this precision, but forget about it since we never see a timestamp displayed in our tool of choice, with that amount of precision. More importantly, someone using a timestamp modification tool can only modify times to the second (of any of the ones I am aware), which would leave glaringly obvious gaps in the MFT when all the FILETIME stamps are retrieved. There was no such evidence in this case. While it may be possible to manually calculate the file timestamps and place them onto a file with a hexadecimal editor, it would require painstaking effort for every single file (there were 507 files related to the Google Maps content in this case), and the problem of the files still being out of sequence from real files created on the hard drive (as explained in the preceding paragraph) would still exist.

If we suspend our disbelief that someone would go to the trouble of manufacturing temporary Internet content and would then modify all the various files to make them look like they were really created on 7/11, there is still the problem of placing the files on the defendant's computer in the place where real temporary Internet content is stored by Internet Explorer. This would mean that someone would have to break into the computer as user bracoope or a user account with administrative rights to create temporary Internet files (TIF) in the account bracoope. That issue aside, the files would then have to be placed into multiple folders since TIF are not just stored in one folder.

Internet Explorer distributes TIF across multiple folders in a proprietary way. The folders are always created in multiples of four folders at a time; the folders have names that are always eight alphanumeric characters, and the content is distributed across the four folders until the system decides the folders are getting full and creates a set of four more folders, and so on. It would not be enough to just place TIF content manufactured on one computer onto another in the various four folders. One would also have to modify the various Internet Explorer history databases called index.dat files, to make it appear as though the manufactured content was made on the defendant's

computer. There are multiple index.dat files that would have to be altered and here again it would be glaringly obvious that there was content created out of sequence of the legitimate content on the computer. There was no evidence of such tampering found on the defendant's computer. In fact, we have an Index.dat history file the week of 7/11 that corroborates the visit to maps.google.com and a cookie for the visit. Since cookie times are recorded from the server (Google in this case) and not the local machine, we have an independent way of verifying time stamps.

How do we know someone didn't break into the computer? If the network intrusion occurred, I would want to ask the defense expert where that would have occurred. Presumably the intrusion happened while the defendant's computer was on his home network. Assuming none of the neighbors would have been the least bit concerned by a strange vehicle parked close enough to the Cooper home to capture the WEP key, crack it, get onto the Cooper home network, and access the defendant's laptop to place the files onto it, we have several Windows components that would have to be overcome.

Various Windows Event Logs are stored on Vista to record information about application, security and system events. There are also a number of additional logs that various applications can use to store information that can be used for troubleshooting and debugging. Things such as when a computer is started, put to sleep, rebooted, when a user logs in or out, when a wireless network is joined are all things that can be logged. Logging was enabled on the defendant's computer and each time an event is recorded, the event is given an event ID and the time and date of the event is noted in the log. Information noted in the event logs included activities that are successful, activities that fail or are prevented, and warning messages.

The Security log contains events such as valid and invalid logon attempts, as well as events related to resource use, such as creating, opening, or deleting files or other objects. Administrators can specify what events are recorded in the security log. There was no entry in this log for any unusual activity. We would expect to see failed logon attempts if someone tried to guess or brute force a user password on the system. If someone logged in as user bracoope on 7/16 when the computer was seized, there would have been an entry. There was not. The administrator account could have been used to place files onto the computer. The last time this account was used was in April.

User Account Control (UAC) is a security component in Windows Vista. UAC enables users to perform common tasks as non-administrators, called standard users in Windows Vista, and as administrators without having to switch users, log off, or use Run As. A standard user account is synonymous with a user account in Windows XP. User accounts that are members of the local Administrators group will run most applications as a standard user. By separating user and administrator functions while enabling productivity, UAC is an important security enhancement for Windows Vista. With previous versions of Windows if a user had administrative rights on a computer, any process started under that user's logon would run with administrative privileges. This access control model did not include any failsafe checks to ensure that users truly wanted to perform a task that required their administrative access token. As a result, malicious software could install on users' computers without notifying the users. (This is sometimes referred to as "silent" installation.) Even more damaging, because the user is an administrator, the malicious software could use the administrator's access control data to infect core operating system files and, in some instances, to become nearly impossible to remove.

To help prevent malicious software from silently installing and causing computer-wide infection, Microsoft developed the UAC feature. Unlike previous versions of Windows, when an administrator logs on to a computer running Windows Vista, the user's full administrator access token is split into two access tokens: a full administrator access token and a standard user access token. During the logon process, authorization and access control components that identify an administrator are removed, resulting in a standard user access token. The standard user access token is then used to start the desktop, the Explorer.exe process. Because all applications inherit their access control data from the initial launch of the desktop, they all run as a standard user as well. The takeaway from this is demonstrated in the defense expert's second video.

At the point in which the "intruder" gains access to the computer being represented as the defendant's computer, the UAC dialog box appears and asks for the user's permission to allow the program to run. In such a scenario, Brad would have been at home working on his computer or returned to his computer to find the UAC box on his screen. Until the user clicks to allow the program to work, nothing would happen. So Brad would have been tricked into clicking to allow the malicious program to run?

All of this is a moot point if the defense expert claims the intrusion happened on 7/11. Entries from the defendant's computer Wireless Local Area Network Auto Configuration Event log (WLAN-AutoConfig.evtx) show that the computer connected wirelessly to the Cisco domain via a network access point called Blizzard on 7/11/2008 at 12:28:52 PM EDT. It also shows that this network used the extremely robust security protocol WPA2 Enterprise, which also requires a secondary authentication feature such as a RADIUS server or EAP certificate on the machine in order for it to connect to the network (sort of like having MAC filtering enabled as a secondary security method as described on a home network). Cisco would have to tell you what is required in order for someone to connect wirelessly to their network, but suffice as to say it is no trivial matter to break into such a network. Brad's cell phone records could show the tower his phone was using around that time to also confirm his location.

If the intruder was able to break into the Cisco network, they would still have to find the defendant's specific computer on that network, and would still have to gain access to the computer by some malware. Again, there was no evidence of any malicious program run on the computer when we checked the Windows Prefetch folder, and when we did a virus/malware check against the computer's hard drive. Each time you turn on your computer, Windows keeps track of the way your computer starts and which programs you commonly open. Windows saves this information as a number of small files in the Prefetch folder. The next time you turn on your computer, Windows refers to these files to help speed the start process. Each time a program is run, a counter is incremented. By examining this folder we can tell what programs are run, how often, and the last time.

In addition to the lack of positive indication that a malicious program had run, there was also an absence of indicators that something was run and that an intruder had deleted numerous files and logs that would have captured malicious activity. We normally see evidence of something being there that shouldn't or big gaps of data that should be there but have been removed (since they contain evidence of the intrusion). I have never seen anyone who could break into a network and/or a computer and leave absolutely no evidence they had been there. There are just too many files, logs and ways of checking for such activity.