



BoneSaw

Exploitation of the
Open Source Application 'Bonsoir'

gammah@covert-operations.org

What is Bonsoir?

- vCard Trading Application using Bonjour / Multicast DNS and Cocoa's NSDistributedObject
- Opensource app written in Obj-C and Apple's Cocoa Frameworks
- Clients see service advertisements from peers for Bonsoir service
- Clients may initiate a download of a peer's advertised vCard
- No authentication or verification required

Full Disclosure

- The application creates a temporary card when downloading from a peer
- The filename is predictable “/tmp/Firstname Lastname.vcf,” and is controlled by the peer (including embedding 0x00)
- The file contents are controlled by the peer
- The downloaded file is opened using `-[NSWorkspace openfile:]` method, which causes the file to be opened according to Launch Services settings for the file’s extension

Full Disclosure

```
if ([proxyObject respondsToSelector:@selector(name)])
{
    NSData *theirVCardData = [proxyObject vCard];
    NSString *tempFilePath = [NSString stringWithFormat:@"%tmp/%%.vcf", [proxyObject name]];
    [theirVCardData writeToFile:tempFilePath atomically:FALSE];

    NSLog(@"Writing the vCard out to: %@", tempFilePath);

    if ([[NSWorkspace sharedWorkspace] openFile:tempFilePath] == FALSE)
        NSLog(@"Attempted to open file failed (%%)", tempFilePath);
}
else
```

What is BoneSaw?

- BoneSaw is a rogue client
- Has 2 modes: 'Patched' Bonsoir mode, and BoneSaw mode
- In Bonsoir mode, behaves just like the real client
- In BoneSaw mode, the attacker can specify the remote path to write to, and a local file as a payload

Exploit Development

- Used the original code as a platform for development
- Renamed all the project resources from Bonsoir to BoneSaw (good learning process); Patched the GUI, created some pointers between objects in the app
- Created a new object in the app to patch method calls to the existing code when the DO button is pressed
- Threw in a Quartz Composition for some serious eye kandy

Discoveries

- Write to any file in an existing directory that the victim has write access to
- For a specifically targeted user, you could write files to paths you know exist, `/Users/gammah/.ssh/authorized_keys`
- You can backdoor the application by writing to `/Applications/Bonsoir.app/Contents/MacOS/Bonsoir`

Problems

- Code execution is difficult - At this point, I haven't gotten arbitrary code execution by any of these payload types - compiled code, shell/perl/ruby scripts, applescript, automator action
- No code means the path is hardcoded, and the same for all victims (e.g. no ability to do discover the victim's \$HOME)
- Cannot create new directories
- No one uses this POS application anyways, and exploitation requires local network proximity to victim

Resources

- Bonsoir - <http://opensource.bleepsoft.com/>
- Bonesaw - AHA Wiki

A patch to Bonsoir has been made available, but let's face it, no one uses this app anyways.