Carrier-level Permissions, Smartphone Hacks, and the World's most Resilient Solution

How to leverage 6 existing Android capabilities to build one of the most-resilient smartphones ever

The Genesis of phone hacking can arguably be traced back to 1956 and a seven-year-old boy named Josef Carl Engressia, Jr. Engressia was born blind and had the enviable gift of perfect pitch. As a precocious five-year-old, Engressia discovered that he could dial phone numbers by clicking the hang-up switch on rotary phones. For those of you not old enough to know what a rotary phone is, it was essentially a clicking cylinder dial with a hang up switch. The number of clicks the rotary dialer completed as it rotated around the phone base "told" the telephone system what number it wanted to dial – much like a computer taking a keyboard instruction then going to an IP address. After a few rotations of the dial, the caller was connected to the number dialed.

Blind since birth, Engressia developed such a keen sense of hearing that he had mastered perfect pitch by age seven. And it was at this ripe young hacking age of seven, when he accidentally discovered that he could activate a phone switch – effectively make a phone call – by whistling 2600 hertz several times in a row into a telephone. 2600 hertz was the single frequency mode used in 1960s-era telephone dialing systems.

Engressia discovered that he could disconnect a long distance call with a whistle, then "hook switch" the call just as if he were the phone company switchboard. By the time Engressia reached college, long-distance calls were just a whistle away and classmates marveled, many paying $1 to him to serve as human switchboard for long distance calls that cost 3X that at the time.

The emergence of Engressia in computer hacking lore stands alone if only for the fact that it is extremely interesting. But two additional landmark moments emerged from Engressia's whistling mastery: 1) It was one of the first communications systems breaches that led to the emergence of phreaking (phone hacking), which in turn, led to computer

hacking. And, 2) it set a precedent for an enterprise communications company (Florida Bell Telephone) to illegally monitor phone conversations and provide details to a U.S. Government agency (The FBI).

The years following Engressia's arrest for "obtaining phone service by fraud" saw the rise of personal computers (Steve Jobs and Steve Wozniak were known phreaks) and the breakup of the phone company monopolies. Naturally, Phreaking evolved into computer hacking and now that smartphones and tablets have become effective network workstations; we have come full circle with a new type of Phreaker (hacker). And, some of today's hackers are still looking for free long distance service much like Engressia was 50 years ago. However, the hackers with the greatest entrepreneurial spirit seek a bigger payout: your data and your company's intellectual property, for sale, and to the highest bidder. And in most cases, the victim never knows they were hacked.

The fact that hackers are finding it relatively easy to access enterprise systems can be attributed to perspective. From the users' perspective mobile devices are serving as functional network workstations, but from most enterprises' security perspectives, these devices are just the latest technology distraction purchased from a mall cart. With this mentality, senior IT managers are setting themselves up for a future filled with chasing security problems through those enterprise systems which allow mobile devices to connect to them. Some organizations understand the severity of mobile device vulnerabilities. These "security aware" organizations are taking measures to stave off the threat, but they are the minority and their efforts are often hamstrung by carrier and OEM security apathy. As we've see in the past, the majority vulnerable enterprises react to advanced persistent threats (APTs) only taking action after a significant (and often very costly) breach.

Tracking Us then Hacking Us with "Permission" from our Mobile Carrier

In 2012, the IntegriCell team conducted a series of experimental tests after several anomalies were discovered on the team's mobile devices while they were traveling internationally. The experiments focused on manipulating consumer mobile devices with the intent of installing persistent malicious software or information-intercept capabilities without the user's permission or knowledge.

Using a series of attack platforms, ranging from easily-obtained Open BTS-type systems

(base transceiver station system for mobile communications) to purpose-built cellular intercept platforms available for purchase worldwide, the team proved that carrier-level permissions allow attackers to manipulate consumer mobile devices in ways that can persist until the device is effectively wiped either by a hard reset or through the installation of a new ROM image. Through further testing IntegriCell discovered that the only mobile platform resilient in the face of these attacks was a BlackBerry 7

handset paired with a BlackBerry Enterprise Server 5.0 back end. That handset had the “Disallow Patch Download over Roaming WAN” IT policy rule setting. Without this OS setting, the carrier had full access all the time and software updates (and malware) could be sent to BlackBerry handsets without the user ever knowing.

Both Android and iOS platforms were also evaluated in these tests. The Android devices were vulnerable to a wide variety of attacks, ranging from operating system manipulation to radio software compromise. iOS devices while resilient to many of the operating system attacks, were still vulnerable to attacks which exploit the carriers’ abilities to configure particular settings within the radio software. Radio software or antenna stack attacks are those which focus on changing settings of or injecting software into the software layer which interfaces with the carrier network. One persistent attack IntegriCell discovered took place when a malicious carrier made a change to the Access Point Name (APN) which directs all TCP/IP traffic from the device to an attacker’s APN (instead of the default APN which is used by the owner’s local carrier). While much of the traffic that flows to/from mobile devices is protected via SSL, there are many vulnerabilities in mobile phone SSL implementations that would allow an attacker to intercept nearly all SSL communications, decrypt them and gain access to the contents of those previously-protected SSL packets.

This was a significant discovery! Mobile carrier-based intrusion? The foundational layer of trust on our cellular architecture, always open to malicious threat? Why haven’t we heard about this before from the mobile carriers? Whatever the case, the fix would never come from a carrier-based source. There would be too many parties involved – hardware manufacturer, carrier, software distribution channel, retailer, enterprise, enterprise user, etc… If a fix were to be developed to plug this hole, it would have to be market-driven and led by a truly objective player in the mobile ecosystem – enter IntegriCell.

Given the information IntegriCell gathered and what was learned about this threat type, IntegriCell had the subject matter expertise and now the research data to deliver a solution. Based on the research gathered, the IntegriCell team began designing an architecture for a mobile device that would be

resilient to carrier attack, while allowing a user to have a normal consumer mobile device experience. Such a device would allow organizations to provide devices to employees which would be resilient to some of the most aggressive hostile-carrier attacks, while also enjoying the technology advances we continue to see in the consumer mobile device market.

A Significant Mobile Device Intrusion Discovery, now what?

Where to start? Due to the limitations of the iOS platform (specifically the lack of hardware-separated boot-time integrity checking), these devices were eliminated as a possibility for a potential development platform. The Android ecosystem was a much better fit for this project due to its open architecture and because of the wide variety of hardware

platforms that are able to support the Android operating system.

The core of the IntegriCell architecture had to be a hardware root of trust (HROT). An HROT is essentially an independent set of hardware and software which can verify the integrity of smartphone firmware and software components. Based upon the research, this HROT could not be merely a memory partition or a set of software instructions. This foundation of the IntegriCell solution has to be a trusted entity independent from all other hardware and software, performing authentication and encryption, etc… This HROT had to be a device which could withstand malicious carrier attacks; therefore it could not be a partition of the SIM card. The team eventually settled on a microSD-form-factor HROT. The design could be implemented on a FIPS-accredited (Federal Information Processing Standards) multi-domain contact smartcard in a microSD form factor. This would allow for the deployment of machine certificates, user certificates and application certificates, all of which are maintained in their own FIPS-accredited partitions.

The second component of the design was a flexible hardware platform. Further research showed the LG Optimus handset as one of the highest-performing handsets available with a microSD slot. The IntegriCell team scoped the creation of a custom-built bootloader for the Optimus handset and began developing a custom bootloader to interact with the HROT. This was a significant improvement in smartphone integrity, allowing real-time verification of the bootloader which prevents malicious, carrier-injected bootloaders and antenna stack attacks. In the future, this high-resiliency architecture of the solution could be expanded to any Android-capable device which has a microSD slot.

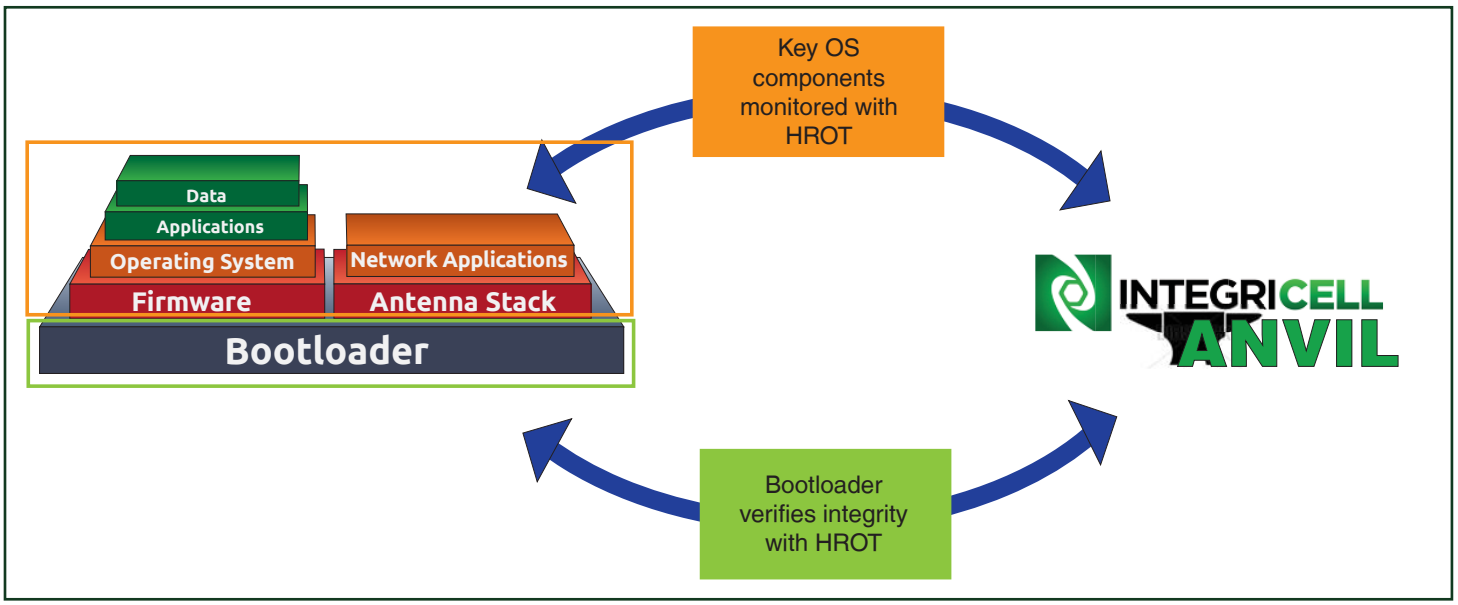
The third major component of the design is the operating system. The IntegriCell team identified the Nexus class of ROM's as the best implementations of the Android operating system from a security perspective. The direct connection that the Nexus ROM's allow to receive security updates directly from Google eliminates the patch madness that is caused by carrier and OEM custom software (and the need to perform regression testing on every security update to see if they break carrier and OEM bloatware). As all Android development is open source, the Nexus ROM's could be re-purposed for the IntegriCell smartphone project with some fairly minor modifications.

ANVIL: Forging a high-resiliency device solution – 6 concepts Anvil leverages to manage advanced smartphone threats

The only smartphone capable of withstanding a carrier-based intrusion is Anvil, from IntegriCell. Anvil combines high-performance smartphone hardware with the latest Android operating system security features, while leveraging the benefits of an HROT architecture. IntegriCell can now enable enterprises to build Anvil-hardened smartphones that stand up against nearly any type of advanced smartphone attack.

I Hardware Root of Trust

Any sort of logical memory partition which attempts to provide integrity-checking capabilities can be easily bypassed either through local manipulation (such as "rooting" or "jailbreaking" through USB connections) or through carrier network manipulation (whether done through platforms like OpenBTS or by production carrier equipment programmed to do so by "malicious



carriers"). Within the global system for mobile communications architecture, or GSM, some have suggested that SIM-cards could be used as roots of trust, but in situations where "malicious carriers" are manipulating devices, the GSM Operator Specification clearly provides that carriers may make changes to SIM-card-embedded components. In order to prevent USB-caused or carrier manipulation from happening, it is necessary to have a hardware-separated root of trust which can serve as the basis upon which certain Android system components can verify their relative integrity. IntegriCell discovered that the microSD card serves as a high-performance interface for a Hardware Root of Trust (HROT).

2 Boot-time Integrity

The bootloader on Android devices is one of the key components that must be monitored in order to avoid that certain classes of vulnerabilities are exploited, thereby reducing the integrity of the overall Android system. The HBoot components are critical to monitor, and developing high-integrity HBoot monitoring technology is critical to locking down mobile device data. Due to the requirements of many organizations with whom IntegriCell has worked, it has not been acceptable to completely disable devices in the event that the HBoot processes are compromised; the user still needs some limited functionality with security, especially while traveling overseas. IntegriCell has worked to develop policy fail-over mechanisms which can either allow the phone to boot only into "911 mode" which allows for certain phone numbers to be dialed, or into "Corporate Data Destruction" mode in which all of the enterprise data is destroyed.

Specific vulnerabilities that are introduced through bootloader compromise include the installation of soft-keystroke-loggers, anti-malware process downgrades (in which anti-virus programs report they are effective but they are not), and persistent kernel modifications which copy voice, SMS and data communications and send copies to attackers.

3 Antenna Stack Integrity

very-poorly documented and even more-poorly understood component of all smartphones is the Antenna Stack or "radio" as some may refer to it. This is software which is used to allow communications from the device to the carrier network. Per the GSM Standard (which even CDMA-LTE devices now follow), the carrier has full control over all updates to the Antenna Stack. When

vulnerabilities are exploited within Antenna Stack components, attackers can gain access to voice, SMS and data streams directed to and originating from the device. Other attacks which IntegriCell has observed are certain reverse-911 capabilities which can be compromised to enable a microphone on a mobile device to become a remote listening and recording device.

The Antenna Stack is probably the most-vulnerable set of components within the Android architecture. Because of the resources that cyber criminals have at their disposal, it is possible for entire carrier infrastructures within hostile international locations to be leveraged to conduct attacks against the Antenna Stack. IntegriCell recently worked with other branches of the U.S. Government to identify how Mexican carriers manipulated their cell tower signals close to the U.S. border to exploit Antenna Stack vulnerabilities for the benefit of criminal organizations.

Operating System integrity

The greatest number of vulnerabilities that can be remotely exploited within the Android ecosystem is within the operating system itself. As has been observed and documented from decades of experience, patching known vulnerabilities is the first step to managing the risks of compromise. IntegriCell has a structured and proven patch process for the Anvil platform.

Additional to patching, the Android system has native anti-exploit capabilities that can assist with locking down the smartphone including:

- Address Space Layout Randomization (ASLR) which prevents certain classes of memory attacks, but can be vulnerable in un-patched situations
- Position Independent Executable (PIE) which provides greater randomization for binary execution vulnerabilities, thereby reducing the likelihood of full arbitrary code attacks

The ASLR and PIE capabilities can be strengthened with Anvil's HROT. For example, for certain high-criticality kernel functions, a "load from trusted source" function can be implemented on a periodic basis (such as every few hours). By reloading key kernel-dependent components, this drives down the mean-time-of-persistence of any attacker which may exploit an un-patched vulnerability.

In conjunction with upcoming versions of Android, the IntegriCell is building capabilities to support mandatory code-signing within certain operating system and application components.

Application integrity

There are two significant vulnerability sets that must be managed for applications within the Android system. The first set of vulnerabilities exists within the application code that is compiled and runs on the handset. Most organizations perceive this threat as "Android malware," but realistically it is much more nuanced than the "viruses" of past decades.

For example, there can be malicious code developed to deliver a payload to a vulnerable application which is installed on an Android smartphone, but the actual exploit code must be so well-developed and tested that it creates a barrier to entry for most organized cyber criminals. Much more common are simple "malicious apps." Malicious applications are those which are delivered to users for the explicit purpose of stealing information, compromising smartphone components or abusing payment accounts linked to the line of service which the smartphone relies on.

Another class of vulnerabilities is driven by the back-end systems upon which the application relies. For example, freely-available applications like popular games have a back-end system dependency to push advertising from. Those back-end systems are generally Linux servers with PHP/JSON components installed. If the underlying Linux system is not properly configured or properly

maintained, there are scores of exploitable vulnerabilities which can be leveraged to gain a persistent foothold on the back-end system. This results in the attacker being able to successfully capture all information from the connected devices which have the endpoint application installed. Within this class of back-end system vulnerabilities are traditional exploitable problems such as SQL-injection and Cross-site-scripting. IntegriCell addresses this with a toolkit which can be rapidly deployed to evaluate both endpoint application code as well as back-end system configuration/ code for exploitable vulnerabilities.

6 User identity integrity

Within most high-integrity mobile platform architectures which IntegriCell has delivered to customers, the User generally creates the most vulnerabilities for themselves. Whether through social engineering or through the theft of credentials, once attackers successfully gain access to user credentials, most mobile security architectures are highly compromised. IntegriCell's policy for Anvil users is to define a series of attacks which can compromise smartphone user identity, from social engineering attacks to credential brute-forcing.

IntegriCell will then use that data to deploy hardening to User access:

- Hardware-separated user credential implementation
- High-entropy user data protection
- Anti-social-engineering application capabilities

ANVIL
by IntegriCell

The only smartphone capable of withstanding a carrier-based intrusion is Anvil, from IntegriCell. Anvil leverages 6 concepts of an HROT architecture to forge a highly resilient smartphone.

1. **Hardware Root of Trust**
2. **Boot-time Integrity**
3. **Antenna stack integrity**
4. **Operating system integrity**
5. **Application Integrity**
6. **User identity integrity**

www.integricell.com

Conclusion

IntegriCell's high-integrity Android system, Anvil, is designed to exceed users' expectations in terms of flexibility and reliability, as well as maintain the rigorous security standards needed to prevent advanced smartphone attacks. As the capabilities of consumer mobile devices rapidly increase – whether in processing, storage, video rendering or communications capabilities – Anvil will be flexible enough to leverage new hardware options in a relatively short amount of time. By allowing users to leverage the latest in hardware capabilities, users will not be as frustrated by restrictive mobile device deployments that arose out of large enterprises' previous BlackBerry deployment policies.

Anvil will deliver capabilities to enterprise users and operation staff which allows for the use of the latest consumer device hardware in some of the most-hostile mobile network environments. Based upon our experience deploying mobile devices in a diverse range of jurisdictions for clients with some of the most-demanding security requirements, Anvil will exceed customer expectations consistently and deliver unique high-integrity Android systems which can be maintained by clients in a scalable and repeatable manner.

For more information on IntegriCell and Anvil, please visit http://integricell.com, or email to contact@integricell.com.

IntegriCell, Inc. | **www.integricell.com**

455 Massachusetts Ave. NW
Suite 430
Washington, DC 20001

Contact:
Phone: +1-208-360-3746
E-mail: contact@integricell.com

Fax: +1-208-621-3947
Support: support@integricell.com