

# Zabezpieczanie i Analiza Danych z Urządzeń Mobilnych

## Wykład #7 – Przełamywanie zabezpieczeń



# Plan wykładu

- Definicje
- Karta SIM
- Model bezpieczeństwa
  - iOS
  - Android
- Znane podatności
  - iOS
  - Android
- Blokada ekranu
- Root/Jailbreak
- Kryptografia
- Ataki MITM (Man In The Middle)
- Tracking
- Uzyskiwanie informacji ze zdalnych lokalizacji



# Definicje - code execution

Doprowadzenie do uruchomienia na atakowanym systemie własnego programu/zestawu instrukcji - najbardziej krytyczna podatność

Analogia:

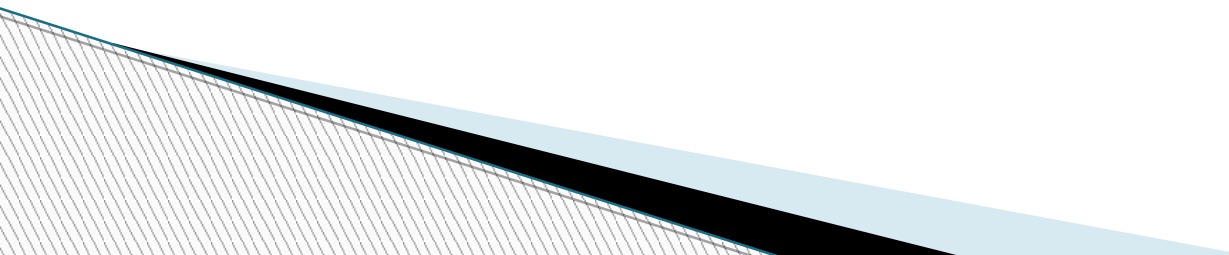
- Grupa osób ma zostać zaangażowana do pracy nad projektem. Przełożony wydaje w tej sprawie pisemne polecenie, pod którym widnieje lista pracowników.
- Przełożony wręcza nam podpisany dokument i prosi, byśmy dopisali swoje nazwisko na końcu listy i zanieśli przekazali dokument do dyrektora
- Zamiast uczciwie wpisać "*Jan Kowalski*", wpisujemy "*Jan Kowalski. Proszę również o natychmiastowe zwolnienie dyscyplinarne Andrzeja Nowaka.*" - i przekazujemy dokument do dyrektora
- Część podanych przez nas danych zostanie zinterpretowana jako instrukcja wydana przez kogoś do tego uprawnionego

# Definicje - code execution

Atak code execution wykorzystywany jest do:

- Uzyskiwania nieautoryzowanego dostępu do systemu
- Przeprowadzania nieautoryzowanej eskalacji uprawnień (privilege escalation - uzyskanie większej kontroli, włączając w to pełną kontrolę)

Problemy bezpieczeństwa prowadzące do code execution:

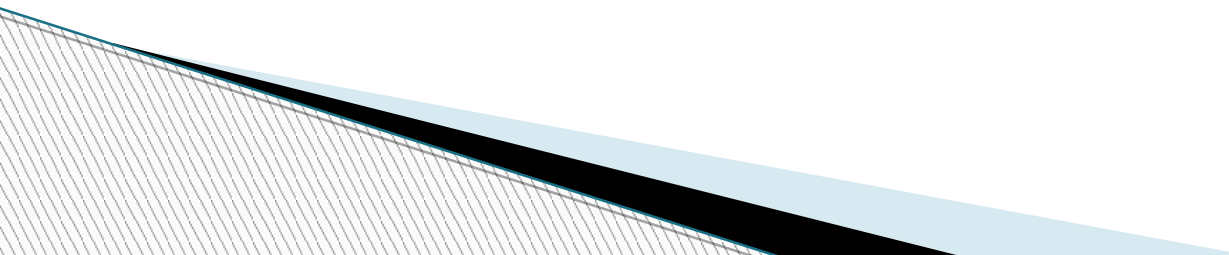
- Luki w autoryzacji lub uwierzytelnianiu funkcji pozwalających na wykonanie kodu
  - Interpretacja danych jako kodu (buffer overflow, command injection)
  - Naruszenie integralności kodu (możliwość nieautoryzowanego zapisu)
  - Konie trojańskie umieszczone przez twórców/dystrybutorów oprogramowania
  - Socjotechnika
- 

# Definicje - information disclosure

Uzyskanie dostępu do informacji, do której odczytu nie zostaliśmy uprawnieni

- Analogia:
  - Zgubienie na terenie uczelni listy pytań na test zaliczeniowy
  - Nieprzemyślane wygadanie tajemnicy w obecności osoby niewtajemniczonej

## Problemy bezpieczeństwa prowadzące do information disclosure:

- Luki w autoryzacji (authorization bypass)
  - Luki w uwierzytelnianiu (authentication bypass)
  - Code execution
  - Podatność na socjotechnikę
  - Podatności typu side channel
  - Ataki Man In The Middle (MITM)
  - Inne?
- 

# Definicje - privilege escalation

Nieautoryzowane zwiększenie swoich obecnych uprawnień w danym systemie.

- Analogie:
  - Szeregowy pracownik przedstawia się jako manager i wydaje innemu pracownikowi polecenie, do którego nie ma uprawnień
  - Szeregowy pracownik potajemnie modyfikuje polecenie wydane przez przełożonego (przykład z Code Execution), dopisując nakaz własnego awansu *"Jan Kowalski."*  
*Ponadto Jan Kowalski ma zostać natychmiastowo awansowany na managera."*

## Przykłady z systemów IT:

- Zwykły użytkownik uzyskuje uprawnienia administratora (w górę)
- Użytkownik z grupy Marketing uzyskuje uprawnienia grupy HR (w bok)

## Problemy bezpieczeństwa prowadzące do privilege escalation:

- Luki w autoryzacji lub uwierzytelnianiu funkcji administracyjnych
  - Code Execution
- 

# Definicje - Man In The Middle

Przechwycenie transmisji pomiędzy legalnymi użytkownikami /podmiotami i

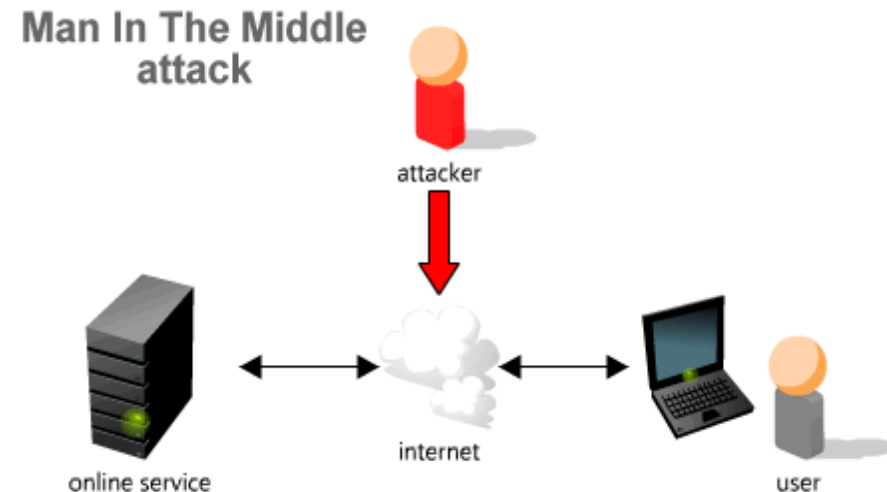
- odczytanie jej (podśluch)
- ingerencja w tę treść przed przesłaniem jej do prawowitego odbiorcy

Analogia:

- Listonosz czytający naszą korespondencję

Problemy bezpieczeństwa prowadzące do MITM:

- Błędy logiczne w uwierzytelnianiu między stronami komunikacji
- Brak zastosowania kryptografii i kontroli integralności podczas komunikacji

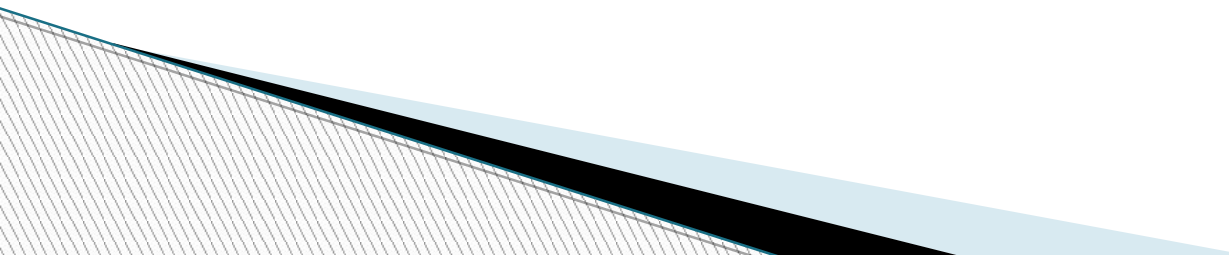


# Definicje - information disclosure

Uzyskanie dostępu do informacji, do której odczytu nie zostaliśmy uprawnieni

- Analogia:
  - Zgubienie na terenie uczelni listy pytań na test zaliczeniowy
  - Nieprzemyślane wygadanie tajemnicy w obecności osoby niewtajemniczonej

## Problemy bezpieczeństwa prowadzące do information disclosure:

- Luki w autoryzacji (authorization bypass)
  - Luki w uwierzytelnianiu (authentication bypass)
  - Code execution
  - Podatność na socjotechnikę
  - Podatności typu side channel
  - Ataki Man In The Middle (MITM)
  - Inne?
- 



# Definicje - side channel

- Znany również jako *covert channel/inference attack*
- Specyficzna forma information disclosure

Podatność pozwalająca na uzyskanie dostępu do poufnej informacji za pośrednictwem dostępu do jawnej informacji, która jest wywiedziona z informacji poufnej (pozwala zatem na jej ustalenie)

## Analogia:

- Nie mamy dostępu do listy plików znajdujących się na serwerze, ale możemy wywnioskować obecność poszczególnych plików/jej brak poprzez wysyłanie żądań odwołujących się do tych plików, gdyż odpowiedź serwera (404/200) zależy od tego, czy dany plik istnieje (można tak odtworzyć pełną listę istniejących plików, choć zajmuje to nieco czasu)
- Nie mamy dostępu do pamięci procesu innego użytkownika, ale widzimy podane przez niego jako parametr uruchomieniowy hasło, gdyż mamy dostęp do listy wszystkich procesów - UNIX)
- Tajna diagnoza + jawna lista leków
- Itd..

## Problemy bezpieczeństwa prowadzące do side channel:

- Błędy logiczne
- Brak dostatecznych analiz z całościowym spojrzeniem na system (podatności side channel są często bardzo trudne do przewidzenia)


# Definicje - Denial of Service (DoS)

Nieautoryzowane doprowadzenie aplikacji/urządzenia/systemu do zaprzestania/zaburzenia funkcjonowania

Przykłady:

- Uderzenie kogoś tak, że traci przytomność □
- Wysłanie milionów maili na czyjąś skrzynkę pocztową
- Odcięcie zasilania
- Uruchomienie na atakowanym systemie ogromnej liczby procesów; doprowadzenie do wyczerpania zasobów
- itd

Problemy bezpieczeństwa prowadzące do DoS:

- Błędna interpretacja danych
  - Brak limitów na poszczególne operacje
- 

# Definicje - ataki na kryptografię

Uzyskanie dostępu do jawnej postaci danych zaszyfrowanych bez znajomości klucza szyfrującego

## Typy ataków na kryptografię

- Przechwycenie klucza przechowywanego w jawnej postaci (np. w RAM)
- Brute force (metoda prób wszystkich możliwych wartości klucza szyfrującego); metoda żmudna i bez wykorzystania dodatkowych podatności/usprawnień mało skuteczna
  - Kolizje, tablice tęczowe
  - Słowniki
  - Dobrze jest dysponować jakąś znaną/przewidywalną wartością szyfrogramu, by móc poprawnie rozpoznać ustalony klucz
- Kryptoanaliza (w tym ataki statystyczne)
- Groźby/tortury wobec osoby znającej klucz □

## Problemy bezpieczeństwa pozwalające na ataki kryptograficzne:

- Przewidywalne/krótkie klucze szyfrujące
- Słabości algorytmu szyfrującego/implementacji
- Jawnie przechowywane klucze

# Definicje - threat modelling

Modelowanie zagrożeń polega na zestawieniu i analizie wszystkich podatności bezpieczeństwa w danym systemie i stworzenie na ich podstawie wszystkich możliwych wektorów ataku

- Na wektor ataku może składać się tylko jedna podatność (np. wysokouprzywilejowany remote code execution)
- Kilka niekrytycznych podatności wykorzystanych razem może doprowadzić do podatności krytycznej:
  - Np. Information Disclosure + Buffer Overflow = Remote Code Execution
  - Remote Code Execution + Local Privilege Escalation = Full Remote Control)

# Karta SIM - kod PIN

- Wykonanie trzech dozwolonych prób, jeśli żadna się nie powiedzie - pozyskanie PUK od operatora
- Włamanie się do systemu operacyjnego karty SIM poprzez OTA [NOHL BLACKHAT]
- Odzyskanie PIN-u w jawnej postaci z miejsca, w którym jest przechowywany, np.:
  - pamięć w iOS [CHRISTIAN FOREST]
  - opakowanie zestawu startowego (startera), które użytkownik otrzymał wraz z kupnem karty SIM

# Włamanie się do OS karty SIM przez OTA



Karsten Nohl

W roku 2013 (BlackHat, USA) Karsten Nohl przedstawia krytyczną lukę bezpieczeństwa w kartach SIM **[NOHL BLACKHAT]**

- Mechanizm OTA (Over The Air) służy operatorom GSM na zdalne aktualizowanie aplikacji zainstalowanych na karcie SIM poprzez SMS-y (specjalny typ SMS-ów, o których użytkownik nie jest zawiadamiany), jak również ustawień sieci GSM (roaming, WAP itd.)
- Uwierzytelnienie takiego SMS-a odbywa się poprzez podpis/zaszyfrowanie kluczem DES (symetrycznym) współdzielonym przez operatora i kartę SIM (shared secret)
- Często ten sam klucz używany jest do uwierzytelniania OTA jak i do szyfrowania komunikacji GSM
- Wiele kart SIM stosowało/stosuje DES (zamiast bezpieczniejszej wersji 3DES)
- Wiele kart SIM okazało się podatnych na atak pozwalający na zdobycie przewidywalnej treści (komunikat błędu) podpisanej poprawnym kluczem przez SMS (co pozwala z kolei na relatywnie łatwy atak kryptograficzny z pomocą tablic tęczowych)
- Odtworzenie klucza operatora pozwalało na zdalną instalację apletów Javy na karcie SIM poprzez SMS
- Odkryta luka w sandboxingu apletów w JavaCard pozwoliła na pełną kontrolę nad systemem operacyjnym (i została wykorzystana przez operatorów do załatania siebie samej)

# Odtworzenie $K_i$ przy dostępie fizycznym

- COMP128 – algorytm uwierzytelniania kart SIM w sieci GSM
- Karty SIM w wersji 1 używają wersji COMP128-1, podatnej na atak typu side channel, który pozwalał na odtworzenie klucza w ciągu 24 h [**EPRINT**], do ataku potrzebny jest czytnik kart SIM i dostępne w sieci darmowe oprogramowanie [SIMCRACK]
- W kartach SIM w wersjach 2+ używany jest algorytm COMP128-2 i COMP128-3, niepodatny na ten atak
- Uzyskanie dostępu do  $K_i$  pozwala na „sklonowanie SIM”, tzn. zalogowanie się do sieci GSM bez udziału oryginalnej karty, której klucz odtworzono (kradzież tożsamości/przejęcie numeru)

# iOS - bezpieczeństwo



## APLIKACJE

- Każda aplikacja ma osobny katalog przypisany przy instalacji (/var/mobile/app\_name)
- Aplikacje mogą uzyskiwać dostęp do danych innych aplikacji wyłącznie za pośrednictwem serwisów dostarczanych przez system operacyjny (sandbox)
- Aplikacje nie mają dostępu do plików systemu operacyjnego
- Brak możliwości instalacji niepodpisanych aplikacji
- Uprawnienia dla aplikacji przydzielane są selektywnie i na żądanie

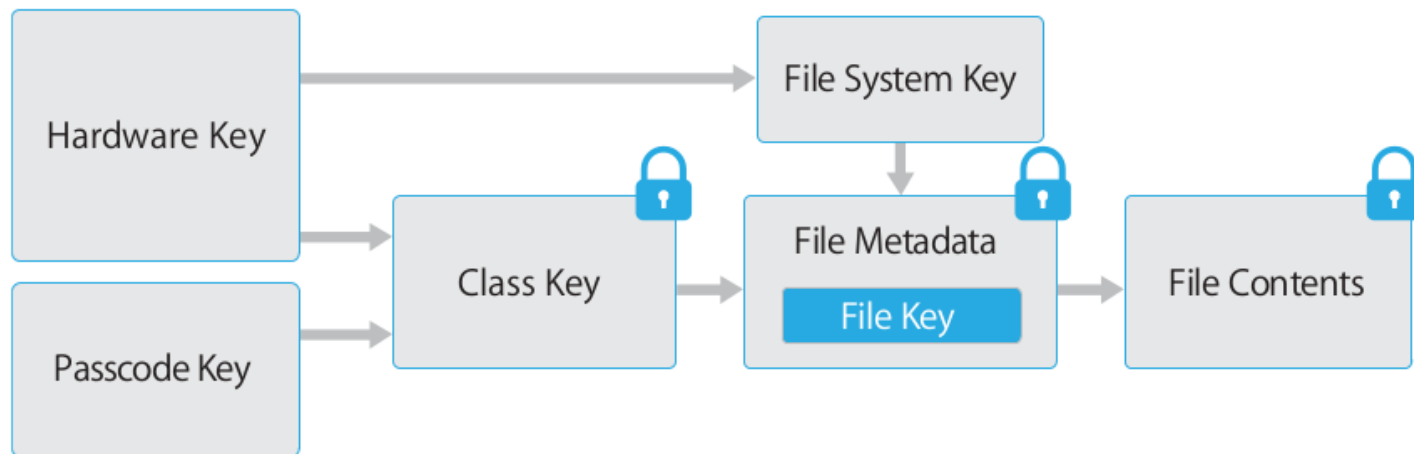




# iOS - bezpieczeństwo

## POUFNOŚĆ I INTEGRALNOŚĆ DANYCH

- Pełne szyfrowanie (Full Disk Encryption) partycji z danymi (bez możliwości wyłączenia)
- Dodatkowo szyfrowanie każdego pliku z osobną indywidualnym kluczem (bez możliwości wyłączenia) – Data protection:



[iOS security]

# iOS - bezpieczeństwo



## POUFNOŚĆ I INTEGRALNOŚĆ DANYCH

- Effaceable storage – rozwiązanie wymazujące wszystkie kopie bloków usuwanego pliku (kopie tworzą w wyniku działania mechanizmu wear leveling w pamięciach flash)
- Secure Enclave – osobny system (dedykowany chip, pamięć, ROM oraz system operacyjny SEPOS) do przechowywania wrażliwych danych (hasła, PIN-y, wzory odcisków palców wykorzystywane przez Touch ID) – dzięki temu nawet w przypadku przejęcia pełnej kontroli nad iOS, nie ma możliwości odczytania tych danych

# iOS - bezpieczeństwo



## BEZPIECZEŃSTWO SIECIOWE

- Randomizacja adresów MAC podczas skanowania w poszukiwaniu zapamiętanych sieci WiFi
- Adres MAC jest losowo zmieniany po każdym odłączeniu się od sieci WiFi, przez co nie może być wykorzystywany do rozpoznawania (śledzenia) danego urządzenia

# iOS - znane podatności



← → ↺ ⓘ www.cvedetails.com/product/15556/Apple-Iphone-Os.html?vendor\_id=49

## CVE Details

The ultimate security vulnerability datasource

(e.g.: CVE-2009-1234 or 2010-1234 or 20101234)

[Log In](#) [Register](#)

Vulnerability Feeds

[Switch to https://](#)

[Home](#)

Browse :

[Vendors](#)

[Products](#)

[Vulnerabilities By Date](#)

[Vulnerabilities By Type](#)

Reports :

[CVSS Score Report](#)

[CVSS Score Distribution](#)

Search :

[Vendor Search](#)

[Product Search](#)

[Version Search](#)

[Vulnerability Search](#)

[By Microsoft References](#)

Top 50 :

[Vendors](#)

[Vendor Cvss Scores](#)

[Products](#)

[Product Cvss Scores](#)

[Versions](#)

Other :

[Microsoft Bulletins](#)

[Bugtraq Entries](#)

[CWE Definitions](#)

[About & Contact](#)

[Feedback](#)

[CVE Help](#)

### Apple » iPhone Os : Vulnerability Statistics

[Vulnerabilities \(1091\)](#)

[CVSS Scores Report](#)

[Browse all versions](#)

[Possible matches for this product](#)

[Related Metasploit Modules](#)

[Related OVAL Definitions](#) :

[Vulnerabilities \(127\)](#)

[Patches \(93\)](#)

[Inventory Definitions \(0\)](#)

[Compliance Definitions \(0\)](#)

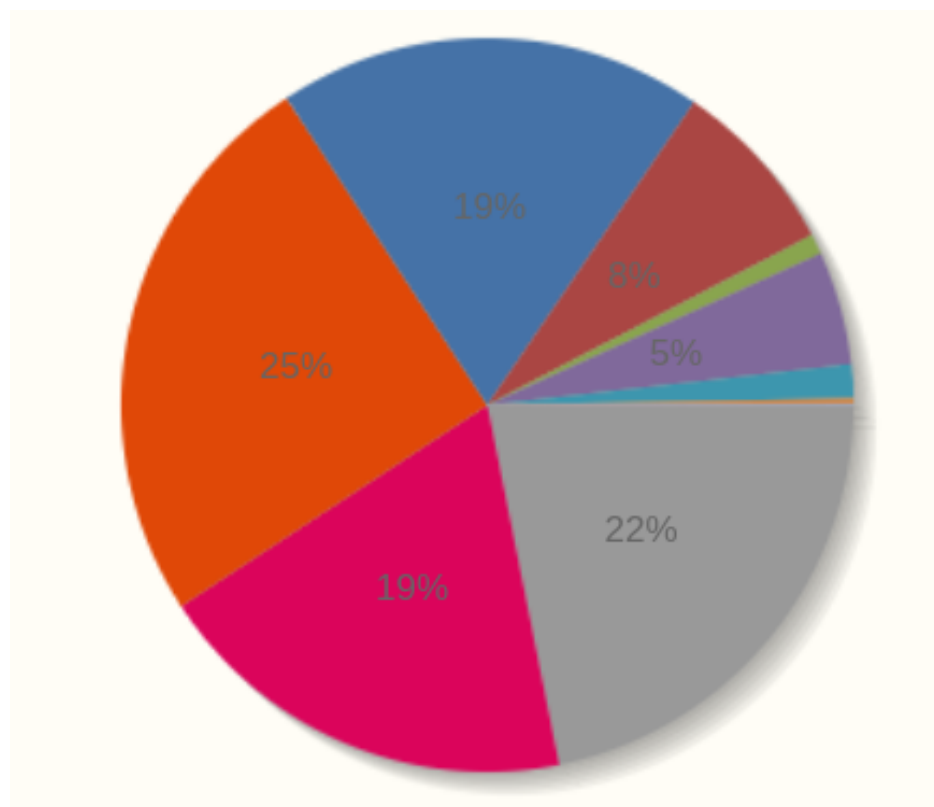
[Vulnerability Feeds & Widgets](#)

### Vulnerability Trends Over Time

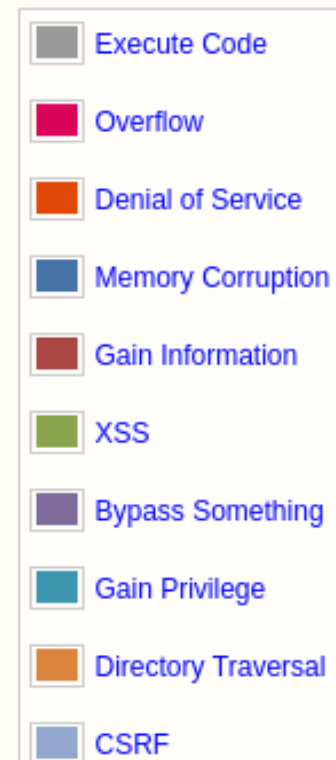
Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
<a href="#">2007</a>	<a href="#">1</a>		<a href="#">1</a>	<a href="#">1</a>											
<a href="#">2008</a>	<a href="#">9</a>	<a href="#">3</a>	<a href="#">2</a>		<a href="#">1</a>						<a href="#">2</a>				
<a href="#">2009</a>	<a href="#">27</a>	<a href="#">10</a>	<a href="#">6</a>	<a href="#">2</a>	<a href="#">4</a>		<a href="#">2</a>			<a href="#">3</a>	<a href="#">7</a>				<a href="#">3</a>
<a href="#">2010</a>	<a href="#">32</a>	<a href="#">14</a>	<a href="#">14</a>	<a href="#">9</a>	<a href="#">6</a>					<a href="#">5</a>	<a href="#">3</a>	<a href="#">2</a>			<a href="#">3</a>
<a href="#">2011</a>	<a href="#">37</a>	<a href="#">13</a>	<a href="#">10</a>	<a href="#">5</a>	<a href="#">6</a>		<a href="#">3</a>			<a href="#">2</a>	<a href="#">11</a>	<a href="#">1</a>			
<a href="#">2012</a>	<a href="#">112</a>	<a href="#">74</a>	<a href="#">69</a>	<a href="#">63</a>	<a href="#">60</a>		<a href="#">7</a>			<a href="#">13</a>	<a href="#">9</a>	<a href="#">1</a>			
<a href="#">2013</a>	<a href="#">96</a>	<a href="#">58</a>	<a href="#">50</a>	<a href="#">42</a>	<a href="#">47</a>		<a href="#">4</a>			<a href="#">17</a>	<a href="#">9</a>	<a href="#">1</a>			
<a href="#">2014</a>	<a href="#">122</a>	<a href="#">50</a>	<a href="#">51</a>	<a href="#">35</a>	<a href="#">33</a>		<a href="#">1</a>	<a href="#">1</a>		<a href="#">20</a>	<a href="#">25</a>	<a href="#">4</a>			
<a href="#">2015</a>	<a href="#">387</a>	<a href="#">232</a>	<a href="#">211</a>	<a href="#">183</a>	<a href="#">191</a>			<a href="#">5</a>		<a href="#">44</a>	<a href="#">63</a>	<a href="#">13</a>	<a href="#">1</a>		<a href="#">1</a>
<a href="#">2016</a>	<a href="#">161</a>	<a href="#">107</a>	<a href="#">78</a>	<a href="#">85</a>	<a href="#">75</a>		<a href="#">3</a>			<a href="#">8</a>	<a href="#">39</a>	<a href="#">11</a>			
<a href="#">2017</a>	<a href="#">107</a>	<a href="#">59</a>	<a href="#">50</a>	<a href="#">44</a>	<a href="#">43</a>		<a href="#">2</a>			<a href="#">12</a>	<a href="#">25</a>	<a href="#">4</a>			
Total	1091	<a href="#">620</a>	<a href="#">542</a>	<a href="#">469</a>	<a href="#">466</a>		<a href="#">22</a>	<a href="#">6</a>		<a href="#">124</a>	<a href="#">193</a>	<a href="#">37</a>	<a href="#">1</a>		<a href="#">7</a>
% Of All		56.8	49.7	43.0	42.7	0.0	2.0	0.5	0.0	11.4	17.7	3.4	0.1	0.0	

[CVEDetails iOS]

# iOS - znane podatności



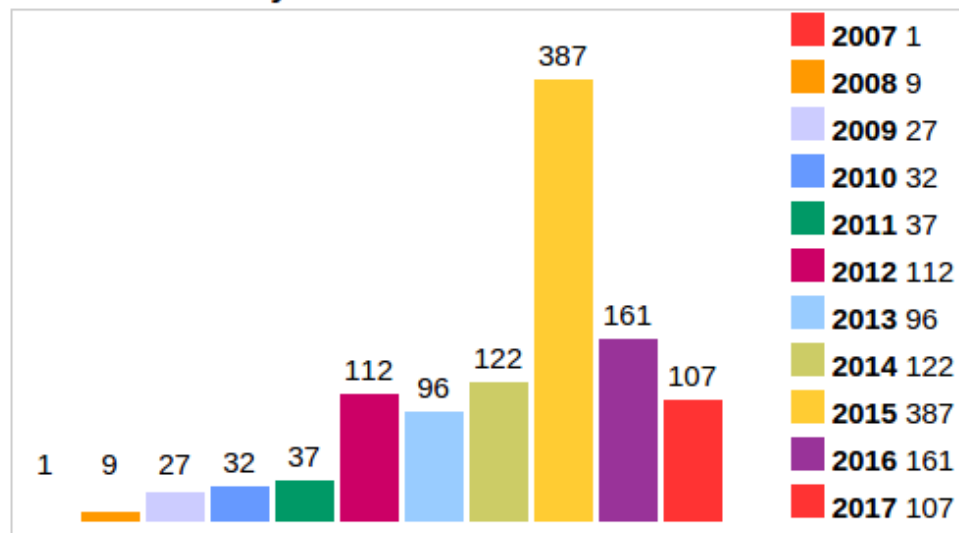
[CVEDETAILS iOS]



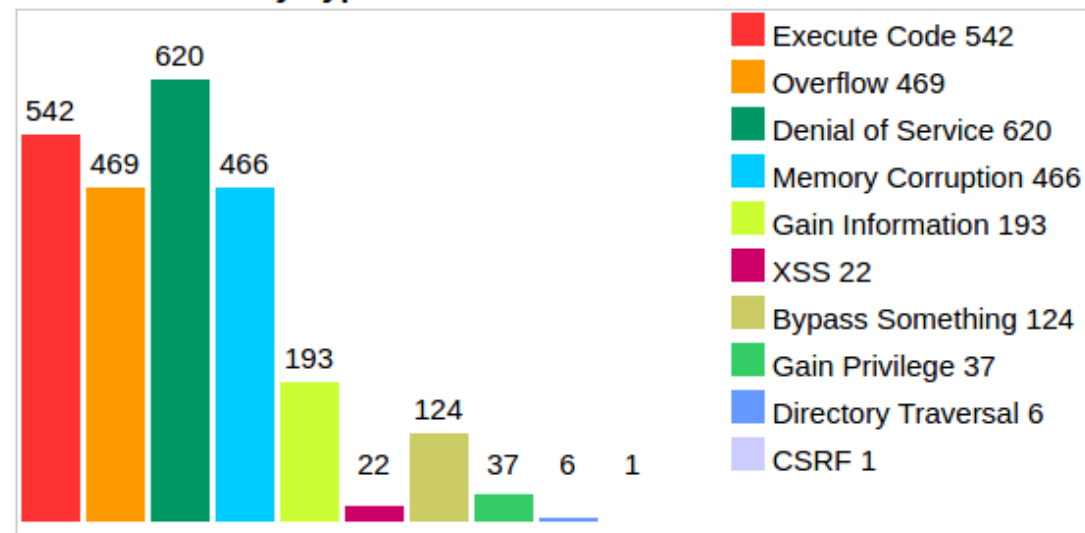
# iOS - znane podatności



Vulnerabilities By Year



Vulnerabilities By Type



[CVEDETAILS iOS]

# iOS - znane podatności - przykład(y) - Trident



- Webkit memory corruption [CVE-2016-4655] → Potential Remote Code Execution
- Information disclosure [CVE-2016-4656] → + CVE-2016-4655 = Remote Code Execution
- Kernel memory corruption [CVE-2016-4657] → Local Privilege Escalation

**CVE-2016-4655 + CVR-2016-4656 + CVE-2016-4657 → High Privileged Remote Code Execution**

# iOS - podatności ujawnione w #Vault7



## iOS Exploits Data

Name	Type	Access Granted	Born Date & iOS Version	Modification Date	Death Date	Found by	Description
Archon	technique	Remote Architecture Detection				(came with purchase)	
Dyonedo	macho-parsing	Codesign Defeat				JDW - GCHQ	
Earth/Eve		Remote Exploit				Purchased by NSA Shared with CIA Ported by GCHQ	
Elderpiggy		Sandbox Escape				Peppermint (NSA VR Contract) Implemented by GCHQ at JDW	
Ironic		Kernel ASLR Defeat			iOS 8	Public vulnerability researcher: Steffan Esser (i0nic)	
Nandao	Heap overflow corruption?	Kernel Exploit				GCHQ	
Juggernaut						Purchase- Baitshop	



# iOS - podatności ujawnione w #Vault7



WikiLeaks

[Leaks](#) [News](#) [About](#) [Partners](#)

[Search](#)

## Persistence

Execution via  
symbolic links

Reboot  
Persistence

June 2013,  
JDW XXXX

June 2014, JDW  
XXXX

CIA

By selecting specific executables on the system partition that are run with root privileges, a symbolic link can be created (on iOS 7.x) or an existing file can be overwritten (iOS 8.x) that will run our bootstrapper, giving use initial execution on every boot.

### Sandbox Profiles:

**Available for:** iPhone 4S and later, iPod touch (5th generation) and later, iPad 2 and later

**Impact:** A malicious application may be able to launch arbitrary binaries on a trusted device

**Description:** A permissions issue existed with the debugging functionality for iOS that allowed the spawning of applications on trusted devices that were not being debugged. This was addressed by changes to debugserver's sandbox.

**Publicly discovered by the Chinese Jailbreak team, Pangu**

**CVE: 2014-4457**

## Redux

Sandbox  
misconfiguration

Close Access

June 2012,  
iOS 6

7/15, workaround for  
missing  
vpnagent in iOS 8 dev  
dmgs

11/17/14,  
iOS 8.1.1

GCHQ

## Rhino

API misuse

Kernel ASLR  
Defeat

April 2013,  
iOS 7

June 2014,  
iOS 8 Beta  
1

GCHQ

Reads KEXT info that reveals the KASLR values by calling the OSKextCopyLoadedKextInfo function.

# iOS - podatności ujawnione w #Vault7



Leaks News About Partners

Search

Sal	Abnormal code path in the kernel	Codesign Defeat	DATE???, iOS 7	2/15, bugfix	FBI, ROU	<p>Copies non-paged sized chunks so that the <code>vm_map_copy_overwrite_unaligned()</code> path is taken in the kernel.</p> <p>This abnormal code path results in pages of memory not being paged in, so the <code>cs_tainted</code> flag is never set on the pages in memory, causing no signature checks.</p>
Saline	Buffer Overflow caused by deserialization parsing error in Foundation library	ROP execution	DATE???, iOS 8	2/15, Productized at TRICLOPS workshop	Purchase - Baitshop	<p>Sending a crafted <code>NSArchiver</code> object to any process that calls <code>NSArchive unarchive</code> method will result in a buffer overflow, allowing for ROP.</p>
Wintersky	Size Mismatch between user and kernel structures	Kernel ASLR Defeat	DATE???, iOS 8		NOCTURNALFEARS	<p>WinterSky leaks the kernel address of the <code>ipc_port</code> struct of a user provided mach port.</p>
Xiphos	Validation Issue	Kernel Exploit	March 2014, iOS 7	11/14, iOS 8.1.1	CIA	<p><b>Available for:</b> iPhone 4S and later, iPod Touch 5th gen and later, iPad 2 and Later.</p> <p><b>Impact:</b> A malicious application may be able to execute arbitrary code with system privileges.</p> <p><b>Description:</b> A validation issue existed in the handling of certain metadata fields of <i>IOSharedDataQueue</i> objects.</p> <p><b>Publicly discovered by the Chinese Jailbreak team, Pangu.</b></p>

# iOS - malware ujawniony w #Vault7



- Już w 2008 CIA dysponowało malware na iOS o nazwie NightSkies w wersji 1.2
- Malware przeznaczony jest do infekowania świeżo wyprodukowanych urządzeń (jeszcze zanim trafią do sklepów)

# Android - bezpieczeństwo

Uprawnienia do plików są zrealizowane przez standardowy model uniksowy, gdzie możliwymi podmiotami są:

**owner, group, others**, zaś możliwymi operacjami, na które osobno nadaje się uprawnienia są **read, write** i **execute** [CHMOD]. Superużytkownik (root) ma wszelkie uprawnienia; w producenckich wersjach systemu jest niedostępny.

Separacja między aplikacjami:

- Każda instalowana aplikacja (.apk) posiada dedykowanego użytkownika i grupę w systemie (plik .apk to archiwum z m. in. plikiem .dex, który można przekonwertować do .jar a następnie zdekompilować)
- Każda aplikacja otrzymuje osobny proces działający w dedykowanej instancji maszyny wirtualnej (Dalvik VM)
- Każda aplikacja posiada dedykowany katalog /data/data/<app> z prawami dostępu (chmod) wyłącznie dla powiązanego użytkownika systemowego
- 99% malware-u dotyczy Androida [ANDMALWARE STATS]



# Android - znane podatności (CVE)



Vulnerability Trends Over Time

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
<a href="#">2007</a>	1		<a href="#">1</a>	<a href="#">1</a>											
<a href="#">2008</a>	9	<a href="#">3</a>	<a href="#">2</a>		<a href="#">1</a>						<a href="#">2</a>				
<a href="#">2009</a>	27	<a href="#">10</a>	<a href="#">6</a>	<a href="#">2</a>	<a href="#">4</a>		<a href="#">2</a>			<a href="#">3</a>	<a href="#">7</a>				<a href="#">3</a>
<a href="#">2010</a>	32	<a href="#">14</a>	<a href="#">14</a>	<a href="#">9</a>	<a href="#">6</a>					<a href="#">5</a>	<a href="#">3</a>	<a href="#">2</a>			<a href="#">3</a>
<a href="#">2011</a>	37	<a href="#">13</a>	<a href="#">10</a>	<a href="#">5</a>	<a href="#">6</a>		<a href="#">3</a>			<a href="#">2</a>	<a href="#">11</a>	<a href="#">1</a>			
<a href="#">2012</a>	112	<a href="#">74</a>	<a href="#">69</a>	<a href="#">63</a>	<a href="#">60</a>		<a href="#">7</a>			<a href="#">13</a>	<a href="#">9</a>	<a href="#">1</a>			
<a href="#">2013</a>	96	<a href="#">58</a>	<a href="#">50</a>	<a href="#">42</a>	<a href="#">47</a>		<a href="#">4</a>			<a href="#">17</a>	<a href="#">9</a>	<a href="#">1</a>			
<a href="#">2014</a>	122	<a href="#">50</a>	<a href="#">51</a>	<a href="#">35</a>	<a href="#">33</a>		<a href="#">1</a>	<a href="#">1</a>		<a href="#">20</a>	<a href="#">25</a>	<a href="#">4</a>			
<a href="#">2015</a>	387	<a href="#">232</a>	<a href="#">211</a>	<a href="#">183</a>	<a href="#">191</a>			<a href="#">5</a>		<a href="#">44</a>	<a href="#">63</a>	<a href="#">13</a>	<a href="#">1</a>		<a href="#">1</a>
<a href="#">2016</a>	161	<a href="#">107</a>	<a href="#">78</a>	<a href="#">85</a>	<a href="#">75</a>		<a href="#">3</a>			<a href="#">8</a>	<a href="#">39</a>	<a href="#">11</a>			
<a href="#">2017</a>	107	<a href="#">59</a>	<a href="#">50</a>	<a href="#">44</a>	<a href="#">43</a>		<a href="#">2</a>			<a href="#">12</a>	<a href="#">25</a>	<a href="#">4</a>			
<b>Total</b>	1091	<a href="#">620</a>	<a href="#">542</a>	<a href="#">469</a>	<a href="#">466</a>		<a href="#">22</a>	<a href="#">6</a>		<a href="#">124</a>	<a href="#">193</a>	<a href="#">37</a>	<a href="#">1</a>		<a href="#">7</a>
<b>% Of All</b>		56.8	49.7	43.0	42.7	0.0	2.0	0.5	0.0	11.4	17.7	3.4	0.1	0.0	

# Android - podatności ujawnione w #Vault7



WikiLeaks

[Leaks](#) [News](#) [About](#) [Partners](#)

Weaponized/Delivered Name	Proof-of-Concept Name	Contract/Partner	Description	Affected Devices	Type
B12	SwampMonkey	Fangtooth	System->Root Priv Used in conjunction with NightMonkey		Priv
	BaronSamedi	Anglerfish	remote access (libxml2)		Remote access
Chronos	Chronos	Anglerfish (originally purchased via partner)	User->Root Priv	Certain MSM devices with Adreno GPUs	Priv
Creatine (crt)	Colobus	Fangtooth	Shell->Root Priv (Framebuffer/graphics stack vuln)	devices equipped with particular Adreno GPUs ie. Adreno 225 and 320 Nexus 7 OS 4.4.2	Priv
Dugtrio (da)	Dugtrio	Anglerfish	Browser/Javascript bridge Doesn't require porting	4.0 - 4.1.2 newer Samsung devices might have the vulnerability, but it is not guaranteed.	Remote Access
	EerieBatter				Priv
EggsMayhem	EggsMayhem	GCHQ, NSA		Chrome version 32 - 39 (present)	Remote Access

# Android - podatności ujawnione w #Vault7



WikiLeaks

[Leaks](#) [News](#) [About](#) [Partners](#)

Freedroid (fd3)					
EerieIndiana (ei)	Freedroid/EerieIndiana	Fangtooth	Kernel/user mem vuln	subset 2.3.6 - 4.2, unreliable in 4.3 - 4.4	Priv
Galago	Galago			SM-N910 (KTU84P.N910HXXU1ANK5), SM-N910S (KTU84P.N910SKSU1ANK8)	Priv
Glutamine (glt)	Bonobo	Fangtooth	Shell->Root Priv (Framebuffer/graphics)		Priv
Remote Code Execution (RCE) Exploits - Helios	Dragonfly/Beracuda	Purchased via partner org.			Remote Access
Flameskimmer					
HGH				Broadcom WiFi chipset devices	
(Note: HGH never deployed, will carry forward FS name in future angry priv framework)	Flameskimmer	SurfsUp	User->Root Priv (WiFi driver vuln) requires WiFi to be enabled	4.3—4.4.2 4.4.4 (updated on July 2015)	Priv
Levigator	Levigator	Public		pre 2.3 - 2.3.5	Priv
Livestrong	Totodile	Anglerfish	Library load via property	Kitkat devices	Persistence
LugiaLight (lgl)	Lugia	Peppermint		MSM devices until ~4.4	Priv
NightMonkey	NightMonkey	Fangtooth	User->System Priv, physical access required (Dex repack/MTP vuln)		Priv

# Android - podatności ujawnione w #Vault7



WikiLeaks

Leaks News About Partners

Search

Salamander	Salamander		browser	Chrome version 28.0.1500.94	Remote Access
			Requires porting if not listed in "Affected Devices"		
Salazar	Salazar	Anglerfish	Works on Chrome, Opera, and Samsung Browser's sbrowser	Chrome version 35.0.1916.141, 37.0.2062.117),	Remote Access
			Requires porting if not listed in "Affected Devices"	Opera version 21.0.1437.75510),	
Simian	Simian	Fangtooth	User->Root Priv, KGSL driver	MSM8974 devices	Priv
Skor	Skor		Requires porting per device	2.2 - 2.3.6	Remote Access
Snubble	Snubble/Snubull	Anglerfish	User->System Priv (with Absolute LoJack software)	Samsung Galaxy S5 (KOT49H.G900HXXU1ANCD) Samsung Galaxy Note 3 (KOT49H.N900W8UBUCNC1) Samsung Galaxy S4 (KOT49H.I9500UBUFNB3)	Priv
Sparrow (sp)	Sparrow	Anglerfish		4.1.2?	Remote Info Leak
Starmie (st)	Starmie	Anglerfish	Requires porting for each ROM -> suggest using Helios	4.0 - 4.3 Samsung Galaxy Tab 2 10-inch, GT-P5100 Epic 4G Touch, SPH-D710	Remote Access



# Przegląd niektórych podatności Androida - code execution

- [EUSECWEST2012] NFC -> Code execution -> Privilege escalation (memory corruption)
- E-Z-2-use (kontrolka WebView) [E-Z-2-use] (do wersji 4.2 wyłącznie, API 17), dostęp na poziomie uprawnień podatnej aplikacji
- [BIN4RY] Root, niedopatrzenie w kontroli uprawnień do customowego skryptu restore wypuszczonego przez Sony
- [RAGE AGAINST THE CAGE] DoS na adb, ponowne włączenie bez zrzucenia uprawnień z roota
- One To Root Them All (bluebox) [ONE TO ROOT THEM ALL] [KITKAT INCLUDED]; błąd logiczny (rozbieżna interpretacja danych między komponentem zabezpieczanym a komponentem zabezpieczającym), możliwość przemycania własnego kodu w zmodyfikowanych plikach .apk podpisanych zaufanymi certyfikatami dostawców (google, samsung) pozwalająca na uzyskanie uprawnień *system* , co z kolei pozwala na eskalację do roota.
- Stagefright (FroYo 2.2+, CVE-2015-1538) [Stagefright in action]

[CVEDETAILS ANDROID]

# Blokada ekranu

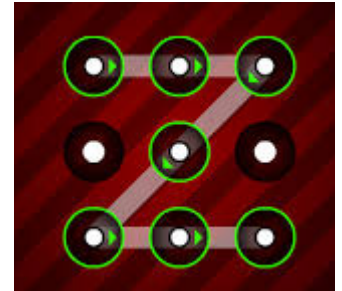
## Rodzaje

- PIN (Android, iOS)
- Pattern lock (Android)
- Rozpoznawanie twarzy (Android)
- Knock code (Android (Samsung Galaxy))
- Touch ID (odcisk palca – iOS)
- Obecność zaufanych urządzeń:
  - SmartLock (Android 5.0 i nowsze)
  - iPhone (zaufane instalacje iTunes na Mac/PC)



# Blokada ekranu - pattern lock

- Pattern lock – wzorem, który należy odtworzyć ciągłym ruchem palca na ekranie dotykowym
- Wymaga wzorca składającego się z co najmniej 4 punktów
- Po kilkunastu/kilkudziesięciu nieudanych próbach do odblokowania konieczne jest zalogowanie się do konta google
- Ze względu na częstotliwość wykonywania przez użytkownika powoduje pozostawienie na ekranie dotykowym śladów fizycznych:
  - Smugi od wodzenia palcem
  - Rys na wyświetlaczach niższej jakości



# Blokada ekranu - pattern lock - Smudge attack - tłusty paluch

- Wykonanie kilku fotografii pod różnymi kątami
- Połączenie ich w celu rozpoznania najbardziej wytartej ścieżki
- Utworzenie zestawu możliwych kombinacji

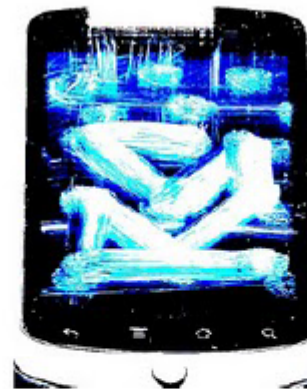
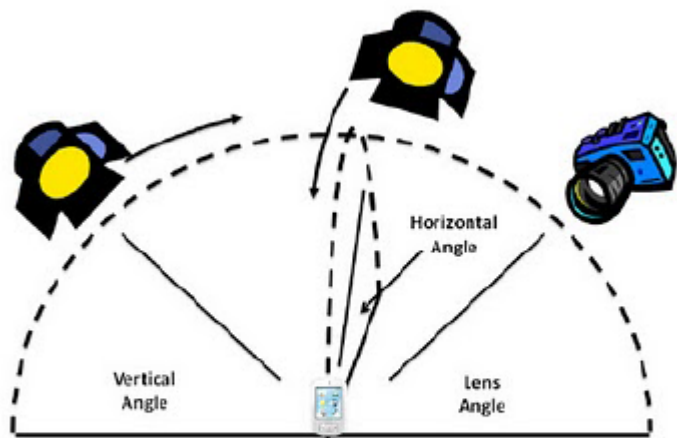


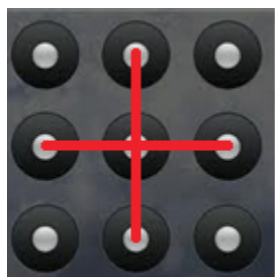
Figure A1: A phone from Experiment 2: The pattern contrasts greatly with the background noise; a grid of dots. The contrast on this image has been adjusted.

<http://bcove.me/7ozhp9u4>

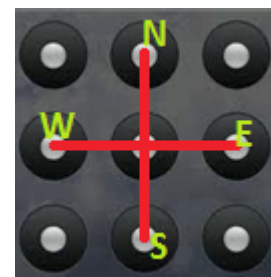
[OPENSECURITY]

# Blokada ekranu - pattern lock - Smudge attack (tłusty paluch)

Ustalenie możliwych wzorów - założmy, że smuga wygląda następująco (przypadek z życia ☐)



**1. Nadajemy możliwym początkom wzorca litery (oznaczenie):**



**2. Ustalamy zbiór możliwych wariacji:**

1. WENS
2. WESN
3. EWNS
4. EWSN
5. NSW
6. NSEW
7. SNWE
8. SNEW

**3. Próbujemy po kolei**  
(w najbardziej pesymistycznym wariacie trafimy za 8 razem ☐)

Metodą zabezpieczającą przed tłustym paluchem jest wyświetlanie wzorca w różnych lokalizacjach ekranu i z różnymi kątami obrotu, co randomizuje obszar, po którym należy przesunąć palec, randomizując też smugę ☐

# Blokada ekranu - pattern lock - usunięcie gesture.key



Potrzebne: USB debugging/tryb recovery

```
# cd /data/system
# ls
batterystats.bin
cache
called_pre_boots.dat
device_policies.xml
dropbox
entropy.dat
gesture.key
inputmethod
locksettings.db
locksettings.db-shm
locksettings.db-wal
netpolicy.xml
netstats
packages.list
packages.xml
password.key
registered_services
shared_prefs
sync
throttle
uiderrors.txt
usagestats
users
# ls -la gesture.key
-rw----- system system      20 2014-03-05 06:26 gesture.key
# cat gesture.key
A00:0000_0f0J&_00#
# rm gesture.key
# exit
root@kali:~#
```

Kroki:

- podłączenie urządzenia
- adb shell
- rm  
/data/system/gesture.key

(testowane na 4.2.2 Jelly Bean)

Efekt: pattern lock nadal widnieje na ekranie, ale odblokowuje go dowolne dotknięcie

# Blokada ekranu - pattern lock - odblokowanie przy znajomości hasła do konta google



Potrzebne:

- Hasło do skojarzonego konta google (jest szansa zdobyć je np. z innego urządzenia używanego przez tę samą osobę)
- Połączenie z Internetem

Kroki:

- Pięciokrotne błędne wprowadzenie wzorca
- Kliknięcie opcję “Forgot password”, która pojawi się po pięciu nieudanych próbach
- Wprowadzenie hasła do konta google

# Blokada ekranu - PIN/hasło - odblokowanie przy znajomości hasła do konta google

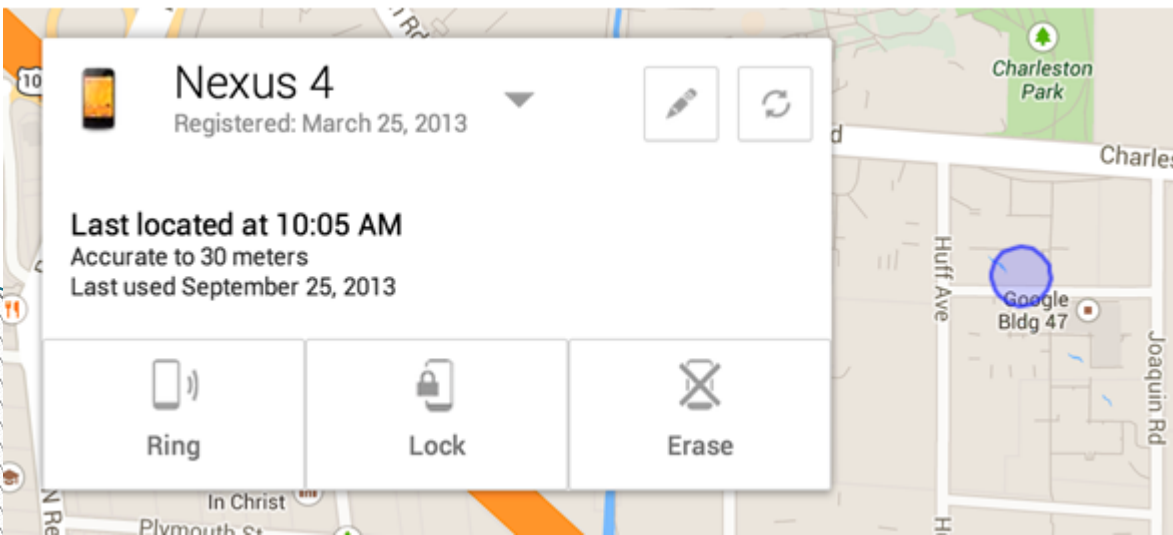


Potrzebne:

- Hasło do skojarzonego konta google (jest szansa zdobyć je np. z innego urządzenia używanego przez tę samą osobę)
- Połączenie z Internetem
- Uprzednio włączony Android Device Manager na urządzeniu

Kroki:

- Wchodzimy na <https://google.com/android/devicemanager>
- Wybieramy urządzenie
- Wybieramy opcję "Lock"





# Blokada ekranu - złamanie gesture.key



Potrzebne: USB debugging/tryb recovery

Kroki:

- podłączenie urządzenia
- `adb pull /data/system/gesture.key`
- Odtworzenie ciągu numerycznego pasującego do sumy kontrolnej
  - Z pomocą brute force (generowanie wszystkich możliwych wariacji numerycznych za każdym razem generując z wyniku sumę kontrolną sha1 i porównując ją z tą znajdującą się w pliku `gesture.key`)
  - Z pomocą rainbow tables

# Blokada ekranu - ingerencja w ustawienia



Potrzebne: USB debugging

Wyłączenie ustawienia bezpośrednio w bazie danych sqlite:

- adb shell
- cd /data/data/com.android.providers.settings/databases
- sqlite3 settings.db **update system set value=0 where name='lock\_pattern\_autolock'; update system set value=0 where name='lockscreen.lockedoutpermanently';**

Efekt: pattern lock dezaktywowany

# Blokada ekranu - „przyjazny trojan” SMS unlock app



Potrzebne: „Przyjazny trojan” SMS unlock app (wymaga roota)

Istnieją aplikacje pozostawiające tylną furtkę aktywowaną SMS-em o ustalonym wcześniej hasle, np. *secret 1234*, którego otrzymanie SMS-em powoduje odblokowanie ekranu

Metoda średnio użyteczna dla mobile forensics; utrzymanie telefonu w stanie pozwalającym na otrzymanie SMS-a jest niezgodne z przyjętym procesem zabezpieczania danych z urządzeń mobilnych; kod taki należałoby złamać metodą brute force, co spowodowałoby wypełnienie bazy sqlite przechowującej SMS-y i nadpisanie dużej ilości potencjalnie cennych informacji znajdujących się na urządzeniu.

[XDA SCREEN CRACK]

# Blokada ekranu - wykorzystanie różnych podatności

- Remote Code Execution [Over the air RCE and jailbreak] [Trident iOS vulnerabilities]
- Wykorzystanie protokołów kontrolonych/serwisowych (np. OMA-DM [NIA unlock])
- Użycie wbudowanych backdoorów
- Wykorzystanie braku mechanizmów chroniących przed brute force
- Wykorzystanie luk w mechanizmach chroniących przed brute force (e.g. OTG w Android – Samsung i Sony wspierają domyślnie [Lollipop Lockscreen Hack])

Wektor przeprowadzenia ataku jest specyficzny dla danej podatności; może to być:

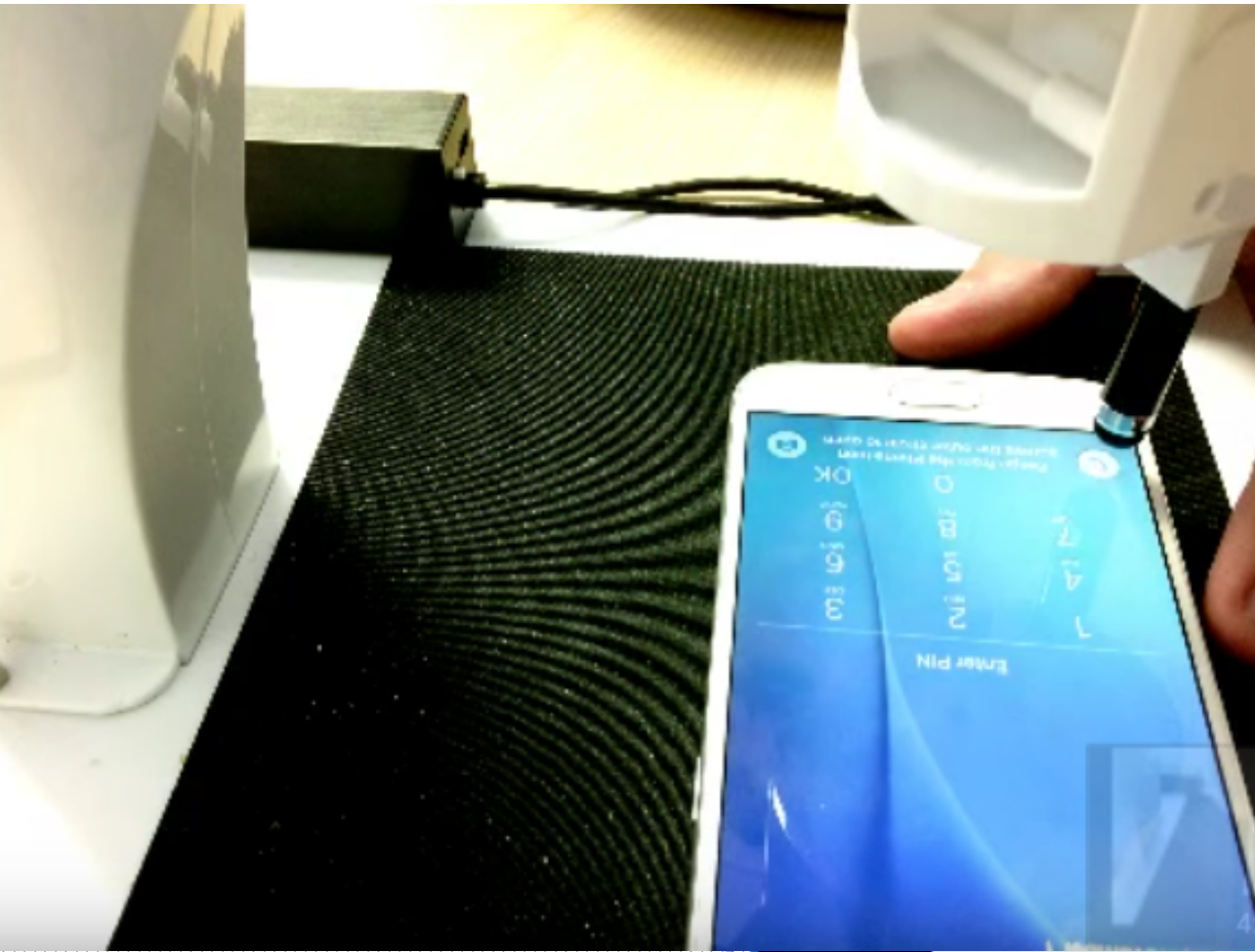
- połączenie kablem (zwykłym lub OTG)
- połączenie z siecią (WiFi, Bluetooth)
- połączenie z siecią GSM (można postawić fałszywą sieć GSM, aby zapobiec zalogowaniu się telefonu do prawdziwej sieci operatora) [OMA hacking 2]

# Blokada ekranu - robot mechanicznie wykonujący brute force



Secure View Strike – automatyzacja ataków metodą prób i błędów

[SV Strike]



- zdolność kalibracji (robot uczy się, gdzie na ekranie znajdują się poszczególne klawisze)
- kamera obserwuje ekran i reaguje na zmiany (udane odblokowanie)
- z powodzeniem stosowany na dowolnych urządzeniach bez aktywnej opcji wymazywania danych po przekroczeniu limitu nieudanych prób

# Blokada ekranu - gdy inne opcje zawiodą

- Wykorzystanie faktu, że uwierzytelnienie i autoryzacja są egzekwowane tylko przez system operacyjny urządzenia
- Należy uzyskać bezpośredni dostęp do nośnika z pominięciem OS (UWAGA: oznacza wyłączenie urządzenia + raczej niewykonalne przy iOS):
  - JTAG/chip off + ewentualne łamanie kryptografii
  - Reboot do trybu recovery (Android)
    - Opcjonalne dokonanie akwizycji fizycznej (backup)
    - Ingerencja w system plików (wyłączenie/usunięcie/przełamanie klucza)
    - Jeśli program recovery nie jest dostępny/ma ograniczone możliwości, instalujemy clockworkmod z pomocą fastboot [CMWIKI] (tryb fastboot jest dostępny w każdym urządzeniu z Androidem, w niektórych przypadkach jednak dostęp do niego wymaga odblokowania bootloadera)
      - Niektóre dystrybucje (ROM-y) nadpisują przy starcie partycję recovery swoją wersją, dlatego po instalacji należy zbootować urządzenie bezpośrednio do trybu recovery

# Blokada ekranu - uniwersalne sposoby/metodologia testowania

Historia problemów z bezpieczeństwem mechanizmów blokujących ekran pokazuje, że bardzo często są one:

- Źle zaprojektowane (możliwe są inne kanały interakcji z urządzeniem), blokada jest na poziomie ekranu, a nie warstwy autoryzacji w OS
- Źle zaimplementowane (możliwe jest ich wyłączenie/wywołanie nieobsługiwanego wyjątku prowadzącego do „wysypania” się procesu)

Częste metody to wywołanie innych aplikacji i funkcji, np.:

- Podłączenie do komputera (samo zdarzenie sprzętowe na USB)
- Podłączenie się protokołem po kablu USB/protokołem sieciowym
- Zresetowanie
- Zadzwonienie na numer/wysłanie SMS-a
- Spowodowanie uaktywnienia innej aplikacji (przesłanie innego sygnału na któryś z czujników)
- Uzyskanie dostępu poprzez aplikacje dozwolone do uruchomienia przy zablokowanym ekranie
  - Ustawienia alarmów
  - Rozpoznawanie mowy (np. Siri) [HAKOWANIE SIRII]
  - Połączenie alarmowe (które zawsze jest dostępne)
  - Kamera (której menu np. pozwala na przeglądanie plików)



# Blokada ekranu - iOS - zestawienie metod i wersji

iOS Device Dashboard								Reference Key		
iOS v1.0 -> v3.1.1		iOS v4	iOS v5	iOS v6	iOS v7	iOS v8	iOS v9		Commercial solutions available (XWays, Cellebrite, Lantern, etc)	
iPhone									Unallocated space may be carved from	
iPhone 3g									File level encryption	
iPhone 3gs									<a href="#">Cellebrite Advanced Investigative Services</a>	
iPhone 4									Pairing File from synced computer may be used to bypass lock	
iPad									Pairing File from synced computer may be used to bypass lock provided	
iPhone 4s									iCloud backup may contain snapshots	
iPhone 5									<a href="#">Law Enforcement/Legal Process options available</a>	
iPhone 5c									<a href="#">Cellebrite Unlock Tool</a>	<a href="#">IP</a>
iPad 2									Apple Inc. may assist in unlocking of device with proper legal authority	
iPad 3										
iPad 4										
iPad Mini										
iPhone 5s										
iPhone 6										
iPhone 6+										
iPad Air										
iPad Air 2										
iPad Mini 2										
iPad Mini 3										
iPad Mini 4										
iPad Pro										
iPhone 6s										
iPhone 6s+										

Źródło: [Ways to unlock iPhone]



# Android - root



Uzyskanie uprawnień superużytkownika (root, uid=0) w systemie Android może nastąpić w wyniku:

- Code Execution - exploit; wstrzyknięcie kodu do działającego z wysokimi uprawnieniami programu (aplikacja działająca z uprawnieniami roota (lub polecenie z bitem suid) lub sam kernel/moduł, podatny na dany typ ataku; buffer overflow, null pointer dereference itp. [NULLPTR])
  - Nadużycia błędnej konfiguracji wysoko uprzywilejowanego programu
  - Nadpisanie/stworzenie pliku wykonywalnego, który później zostanie wykonany z wysokimi uprawnieniami (wykorzystanie błędnej konfiguracji/ingerencja w system plików wykonana na nośniku z pominięciem OS - np. poprzez recovery)
- Tymczasowy root (najlepszy do zastosowań w forensics, mało inwazyjny)
  - Trwały root

# Android - tymczasowy root

Warto sprawdzić, czy telefon nie jest już zrootowany ☐:

```
adb shell su
```

Jeśli wynikiem jest *permission denied*, szukamy rozwiązania na xda-developers.com ☐

Instalacja i wykonanie przykładowego exploita :

```
adb devices
```

```
adb push psneuter /data/local/tmp
```

```
adb shell
```

```
$ cd /data/local/tmp
```

```
$ chmod 777 psneuter
```

```
$ ./psneuter
```

```
adb restart-server
```

Ten exploit wykorzystuje fakt, iż adb startuje (wersje?) jako root i zrzuca uprawnienia w zależności od ustawienia wartości ro.secure w pliku /default.prop (jeśli nie uda się go odczytać, nie rzuci uprawnień) [PSNEUTER]

\*Analogicznie w local.prop jest ustawienie informujące o emulatorze, pozwala na eskalację uprawnień z system do root



# Android – permanentny root

Wykonujemy z trybu recovery (powinno jednak również zadziałać na żywym systemie po wykonaniu exploita) – instalacja busybox [BUSYBOX] (nazwa urządzenia blokowego może się różnić)

```
# mount -o remount,rw -t rfs /dev/block/st19 /system
# exit
adb push busybox /system/bin
adb push su /system/bin
adb install Superuser.apk
adb shell
# chmod 4755 /system/bin/busybox
# chmod 4755 /system/bin/su
# mount -o remount,ro -t rfs /dev/block/st19 /system
# exit
adb reboot
```



# Android - SuperOneClick



SuperOneClick to uniwersalny pakiet oprogramowania przeznaczony do zdobywania uprawnień superużytkownika na urządzeniach z systemem Android, zawiera standardowo:

- busybox (spakowany w jeden plik zestaw podstawowych komend unixowych)
- su (aplikację z bitem suit przeznaczoną do elewacji uprawnień na żądanie (root))
- SuperUser.apk
- Zestaw exploitów (jeśli nie działa, warto szukać na xda-developers.com, luki są ciągle odkrywane i łatane ☐)

[SUPER ONE CLICK][UP TO DATE]

# Ataki na kryptografię

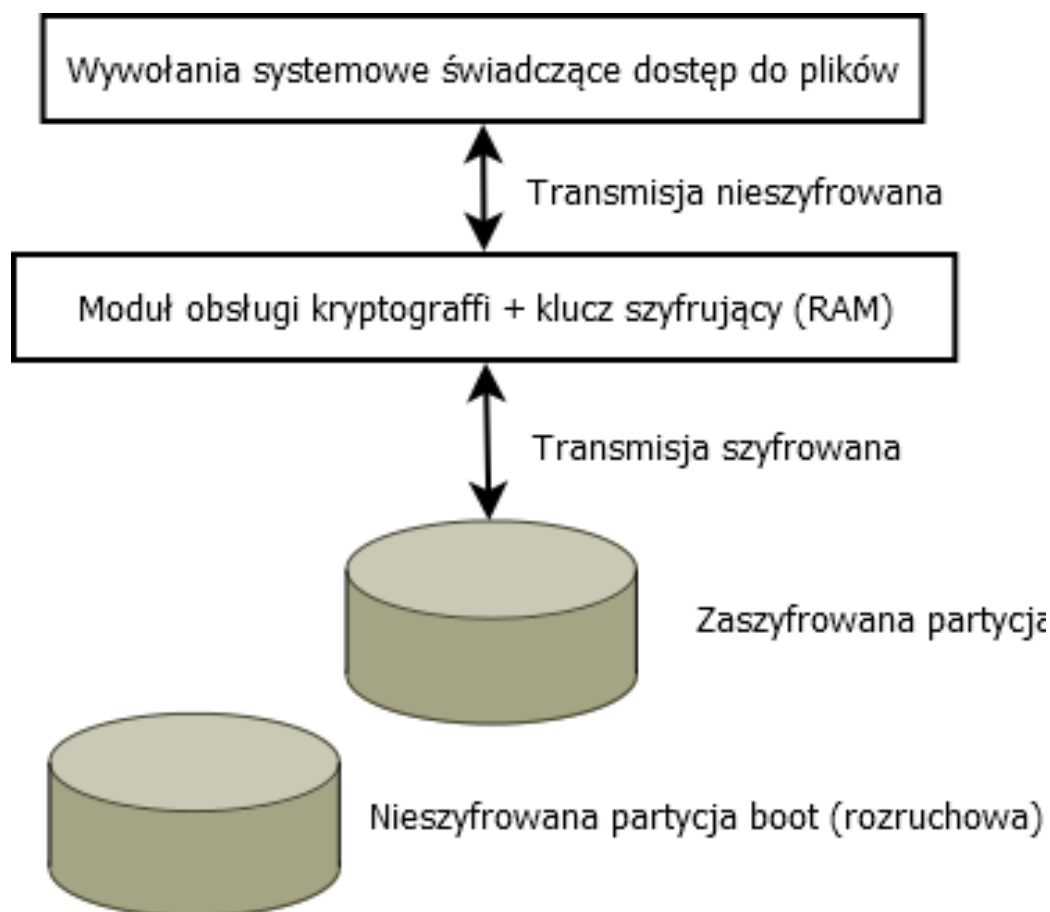
Metody realizacji kryptografii do celu przechowywania danych:

- Zaszyfrowany system plików (Full Disk Encryption (FDE)), np. Bitlocker, Linux LuKs/dmccrypt, File Vault) [FULL DISK ENCRYPTION]
- Zaszyfrowane poszczególne pliki (Encrypted File System), stosowane w Blackberry oraz iOS (ma tę zaletę w stosunku do FDE, że klucz może być usuwany z pamięci przy zablokowaniu ekranu □)

Obydwa mechanizmy mogą być zastosowane jednocześnie (tzn. zaszyfrowany plik w zaszyfrowanym systemie plików).

Do kryptografii służącej poufności przechowywania danych, podobnie jak w przypadku ich transmisji, stosowane są algorytmy symetryczne (wydajność); jeden klucz szyfrujący/desyfrujący (czasami składa się na niego więcej poświadczeń; np. plik klucza + hasło)  
AES, 3DES, Blowfish, Two-fish, RC4

# Ataki na kryptografię - FDE



- Klucz przechowywany jest w pamięci operacyjnej
- Nie nawet potrzeby go stamtąd wydobywać, skoro urządzenie jest włączone i mamy dostęp do danych w jawnej postaci
- W przypadku smartfonów/tabletów bardzo rzadko urządzenie jest wyłączone, co w większości przypadków czyni FDE bezużytecznym
- Jediną ochroną pozostaje blokada ekranu

# Ataki na kryptografie

Najczęściej stosowane ataki

- W przypadku FDE jedną z metod jest niewyłączanie urządzenia
- Wyciąganie klucza z pamięci włączonego urządzenia
- Wyciąganie klucza z pamięci wyłączonego urządzenia [FREEZER ATTACK], teoretycznie możliwy również na urządzeniach mobilnych
- Odgadywanie klucza (brute force + weryfikacja poprzez zaszyfrowanie znanego obszaru danych)

Weryfikacja poprzez zaszyfrowanie znanego obszaru danych: jeśli część szyfrogramu jest przewidywalna (np. wiemy, że wystąpi w nim określone słowo/ciąg bajtów), poprawność generowanego klucza weryfikujemy poprzez zaszyfrowanie nim znanej porcji danych i sprawdzenie, czy postać wynikowa występuje w szyfrogramie

W przypadku FDE przewidywalną informacją są np. dane znajdujące się na początku partycji

[BRUTEFORCE ANDROID CRYPTO]

[BRUTEFORCE ANDROID CRYPTO PY]



# Blackberry 10 + szyfrowana karta SD

*„Układ zabezpieczający zniszczy układ zabezpieczany”  
Edward A. Murphy*

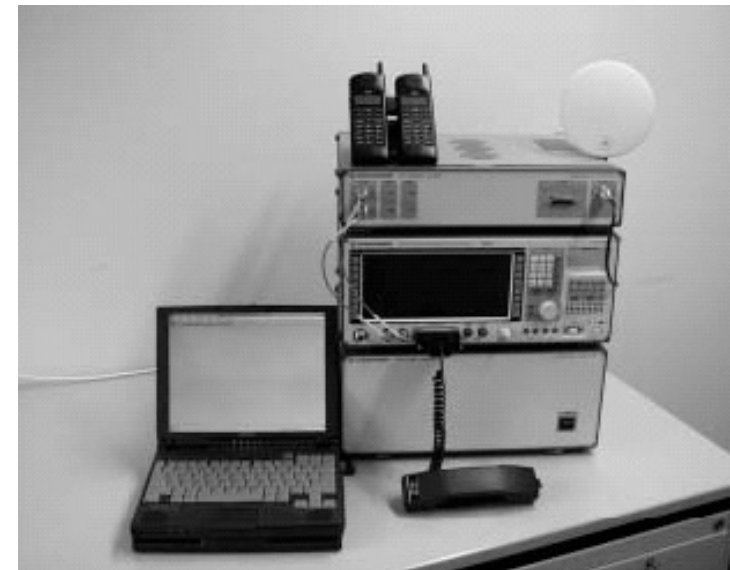
Sposób szyfrowania zawartości karty SD w Blackberry 10 pozwala na odtworzenie z niej hasła użytkownika [ELCOMSOFT]



# Man In The Middle - GSM (IMSI catcher)

- Problem: GSM nie wymaga obustronnego uwierzytelnienia; tzn:
  - Klient musi się uwierzytelnić wobec sieci GSM
  - Sieć GSM nie musi uwierzytelnić się wobec klienta
- Do przeprowadzenia ataku MITM wykorzystać można fałszywy BTS (IMSI catcher), znany również jako “jaskółka”,
- Stacja musi znajdować się w zasięgu atakowanego telefonu komórkowego
- Fałszywy BTS przedstawia się jako BTS operatora
- Fałszywy BTS negocjuje z telefonem brak szyfrowania transmisji (praktycznie żadne z urządzeń nie poinformuje o takim fakcie użytkownika) [IMSI-CATCHER] [IMSI CATCHER3] [Fake BTS]

[IMSI-CATCHER2]



# Man In The Middle - GSM (IMSI catcher)

- IMSI catchera można zamówić z Chin już za kilkaset USD



**tracking suspect, location device IMSI catcher**

1 Piece (Min. Order)

Color: Black  
Type: IMSI catcher  
Place of Origin: Guangdong,China (M...  
Brand Name: ETROSS  
Model Number: MD-1  
Working frequency: GSM 900/1800...

☐ Add to Compare



**SF-M200 Series Modem**

**MARCH low price gsm modem with rj45 imei changer imsi catcher 8 port multi gsm sim modem**

Free Inspection

US \$50-300 / Set

1 Piece (Min. Order)

Applications: Bulk sms,mms,email,fa...  
Type: Wireless  
Weight: 1KG  
Place of Origin: Guangdong,China (M...  
Brand Name: ShiFang  
Model Number: SF-GMS2G-108

☐ Add to Compare [Similar Products](#)



**17 Years In Telecom Products**

**Imsi catcher 32 port gsm/cdma/wcdma interceptor asterisk voip goip gateway avoid sim block**

US \$780-1980 / Piece

1 Piece (Min. Order)

Type: VoIP Gateway,asterisk voip goi...  
Place of Origin: Guangdong,China (M...  
Brand Name: NICEUC  
Model Number: NC-MG640W  
Frequency: Quad-band 850/900/180...  
SIM slot: 32 SIM cards

☐ Add to Compare [Similar Products](#)

[Fake BTS]

# Man In The Middle - Wi-Fi - Open SSID broadcast vulnerability

## Problem:

- wiele urządzeń mobilnych przy zapamiętywaniu ustawień sieci Wi-Fi zachowuje jedynie jej nazwę (SSID)
- gdy urządzenie nie jest podłączone do sieci Wi-Fi, rozsyła ramki odpytujące o obecność tej sieci (beacon) posługując się zachowanym SSID
- atakujący może odpowiedzieć na taką ramkę i automatycznie skonfigurować swój access point pod poznany SSID
- Klient łączy się z kontrolowaną przez atakującego siecią Wi-Fi traktując ją jako tę samą, którą miał zapisaną w ustawieniach (ten sam poziom zaufania)
- Atakujący ma pełen dostęp do transmitowanych danych (oczywiście szyfrowanie WEP/WPA nie ma na to żadnego wpływu, gdyż atakujący kontroluje access point, który ma dostęp do rozszyfrowanych ramek)
- Rozwiązaniem jest zapamiętywanie i weryfikacja BSSID (założenie, że atakujący nie będzie go znał)

[OPENSSID]



# Tracking - ustalanie aktualnej fizycznej lokalizacji bez aktywnego udziału operatora

- Skylock [skylock]
- TMSI paging [bezpieczeństwo polskiego gsm]
- HRL tracking [bezpieczeństwo polskiego gsm]

Wymagające dostępu do telefonu:

- Znane sieci WiFi
- Pobór mocy z baterii :) [pobor mocy]

# Uzyskiwanie danych ze zdalnych lokalizacji

Odzyskiwanie haseł/ciastek/tokenów uwierzytelniających do zdalnych lokalizacji, w których znajdują się dane wytworzone/przetworzone przez urządzenie mobilne

- Zdalne backupy
  - iCloud
  - google
- Historie konwersacji
  - Skype
- Konta shell/ftp/email
- Dane przechowywane przez operatora – dostępne przez serwisy operatora

# Dane przechowywane przez operatora – samodzielne odzyskiwanie billingu

- Często istnieje możliwość przeszukania billingu online poprzez serwis webowy operatora
- Niniejsza metoda wymaga znajomości (poznania :)) hasła uwierzytelniającego w portalu, co wiąże się z koniecznością odebrania hasła przesłanego w SMS – nie jest więc zgodna ze stosowanym procesem (ten wymaga odłączenia telefonu od sieci – ale w takim przypadku zwracamy się o te dane do samego operatora)
- Założenie: dysponujemy telefonem i mamy możliwość odbierania SMS

# Dane przechowywane przez operatora – samodzielne odzyskiwanie billingu

Krok 1 – rejestrujemy/odzyskujemy konto w serwisie webowym dostarczonym przez operatora

The screenshot shows the Play registration page. At the top, there's a purple header with the 'PLAY' logo. Below it, the title 'Rejestracja użytkownika' is centered. A progress bar indicates three steps: '1. Dane konta' (active, marked with a checkmark), '2. Potwierdzenie SMS', and '3. Zakończenie'. The registration form includes fields for 'Telefon \*' (with a green checkmark icon), 'E-mail \*' (with a green checkmark icon), 'Hasło \*' (with a strength indicator: 'Przynajmniej 8 znaków, w tym 4 litery i 2 cyfry'), and 'Powtórz Hasło\*'. There's a checkbox for 'Akceptuje regulamin' and a 'Dalej' button. At the bottom, a link says 'Masz już konto? Zaloguj się'.

Secure | <https://konto.play.pl/user-gui/registrationPage>

**PLAY**

Rejestracja użytkownika

✓

1. Dane konta 2. Potwierdzenie SMS 3. Zakończenie

Telefon \* 576 [redacted] ✓ ?

E-mail \* [redacted]@il.com ✓ ?

Hasło \* [redacted]

Przynajmniej 8 znaków, w tym 4 litery i 2 cyfry

Powtórz Hasło\* [redacted]

☒ Akceptuje regulamin


Dalej

Masz już konto? [Zaloguj się](#)

# Dane przechowywane przez operatora – samodzielne odzyskiwanie billingu

Krok 1 – rejestrujemy/odzyskujemy konto w serwisie webowym dostarczonym przez operatora – posiadamy taką możliwość, gdyż dysponujemy kartą SIM. Jesteśmy więc w stanie odbierać SMS z hasłem:

Secure | <https://konto.play.pl/user-gui/registrationPage>



## Rejestracja użytkownika

✓

✓

1. Dane konta2. Potwierdzenie SMS3. Zakończenie

Wpisz hasło które otrzymałeś na podany przez siebie numer telefonu

576 [REDACTED]

Nie otrzymałeś hasła? [Wyślij ponownie](#)

Hasło jednorazowe \*

Dalej



# Dane przechowywane przez operatora - samodzielne odzyskiwanie billingu

Krok 2 – logujemy się i korzystamy z wyszukiwarki

Secure | <https://24.play.pl/Play24/Billing>

Wyszukaj >

Podsumowanie 01.10.2016 - 31.10.2016

Pokaż statystyki ▾

Połączenia wychodzące	Wiadomości SMS / MMS	Transfer danych 	Zakupy mobilne	Łączny koszt
Łącznie		46,05 MB		
03:31 min.	4 / 0 szt.	Konto i pakiet zł	0 szt.	64,94 zł
		46,05 MB 57 zł		

Szczegóły połączeń

☐ Pokaż tylko połączenia z salda Konto

Pobierz:  

Data ▾	Godzina ▾	Kierunek ▾	Nr telefonu ▾	Wykorzystane ▾	Koszt ▾	Nazwa salda ▾	Opis ▾
08.10.2016	11:35:11	☎	Internet	1396 kB	1,68 zł	Konto	Transfer danych w strefie taniego internetu w Play
08.10.2016	11:17:18	☎	Internet	5803 kB	7,08 zł	Konto	Transfer danych w strefie taniego internetu w Play
08.10.2016	09:19:14	☎	4879 76 55 02	00:34 min.	0,16 zł	Konto	GŁOSOWE
08.10.2016	08:57:27	☎	4878 05 55 19	00:43 min.	0,21 zł	Konto	GŁOSOWE
07.10.2016	18:53:12	☎	Internet	2425 kB	2,88 zł	Konto	Transfer danych w strefie taniego internetu w Play
06.10.2016	18:35:13	☎	Internet	1377 kB	1,68 zł	Konto	Transfer danych w strefie taniego internetu w Play
06.10.2016	18:35:01	☎	4477 12 80 04	1 szt.	00,00 zł	Połączenie bezpłatne	SMS
06.10.2016	18:34:51	☎	80473	1 szt.	00,00 zł	Połączenie bezpłatne	SMS
06.10.2016	18:34:15	☎	80473	1 szt.	00,00 zł	Połączenie bezpłatne	SMS
06.10.2016	18:34:12	☎	80473	1 szt.	00,00 zł	Połączenie bezpłatne	SMS

Pokaż na stronie   

< 1 2

## Odnośniki

Lista odnośników pod adresem:

[https://github.com/ewilded/mobile/W7\\_URLS.txt](https://github.com/ewilded/mobile/W7_URLS.txt)