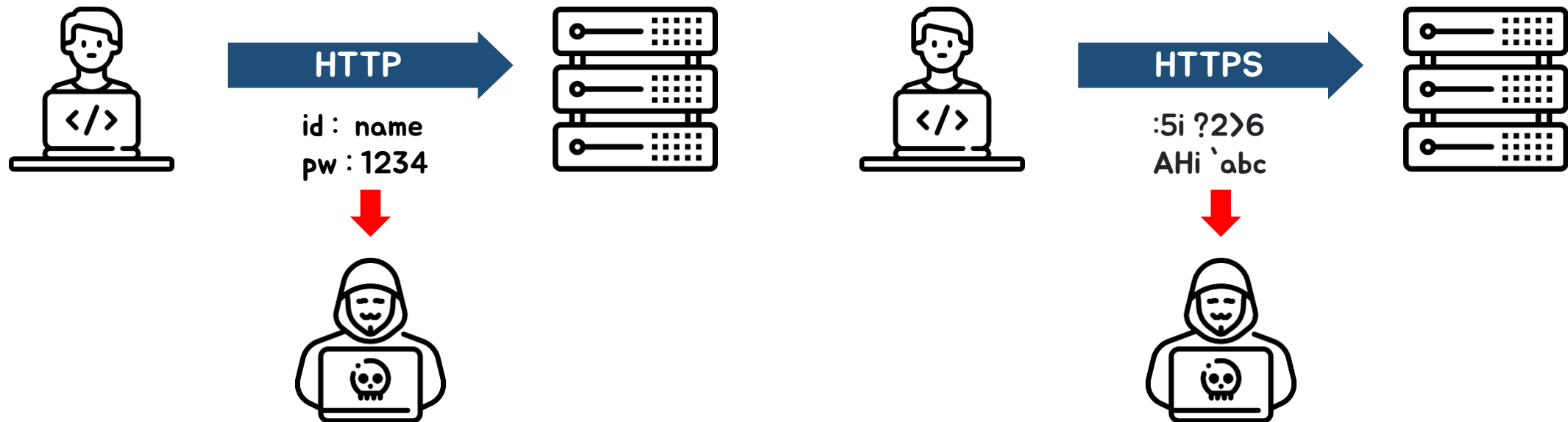




SSL Pinning in

HyperText Transfer Protocol Secure (HTTPS)

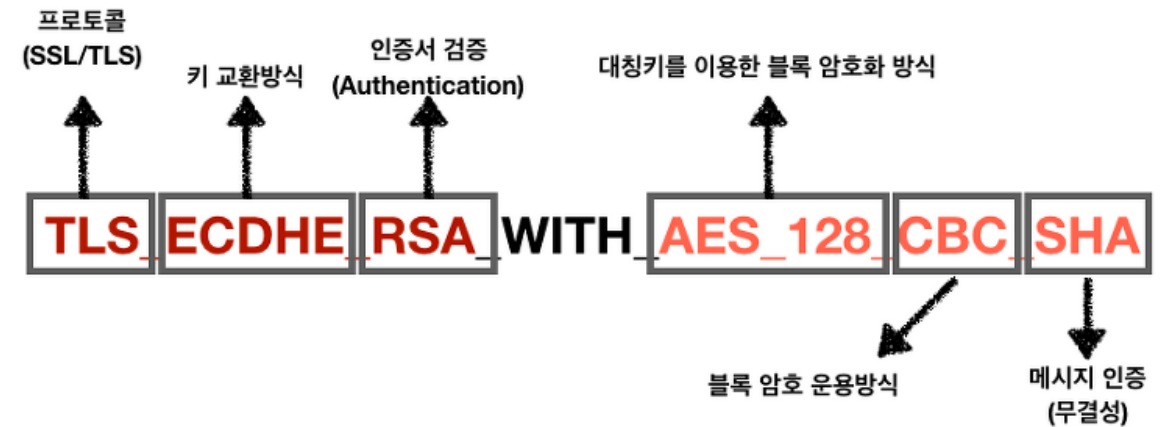
- HTTP 프로토콜에서 취약한 부분을 보완한 프로토콜
- 통신을 진행하기 이전 교환한 정보를 통해 이후 통신을 암호화하여 중간자 공격을 방지한다.



HTTPS



**Certificate
Authorities(CA)**



Cipher Suite

Transport Layer Security (TLS) / Secure Socket Layer (SSL)

- 클라이언트/서버 응용 프로그램이 네트워크로 통신을 하는 과정에서 도청, 간섭, 위조를 방지하기 위해서 설계
- SSL 규약은 1995년 Netscape에서 2.0 버전이 공개된 것이 처음이고 이후 3.0, 이를 베이스로 하는 TLS 1.0 ... 1.3 출시가 되고 있다.
- 통신 중 SSL 인증서를 사용하여 클라이언트와 서버간의 통신을 제 3자(CA)가 보증한다.

프로토콜	연도
SSL 1.0	-
SSL 2.0	1995
SSL 3.0	1996
TLS 1.0	1999
TLS 1.1	2006
TLS 1.2	2008
TLS 1.3	2018

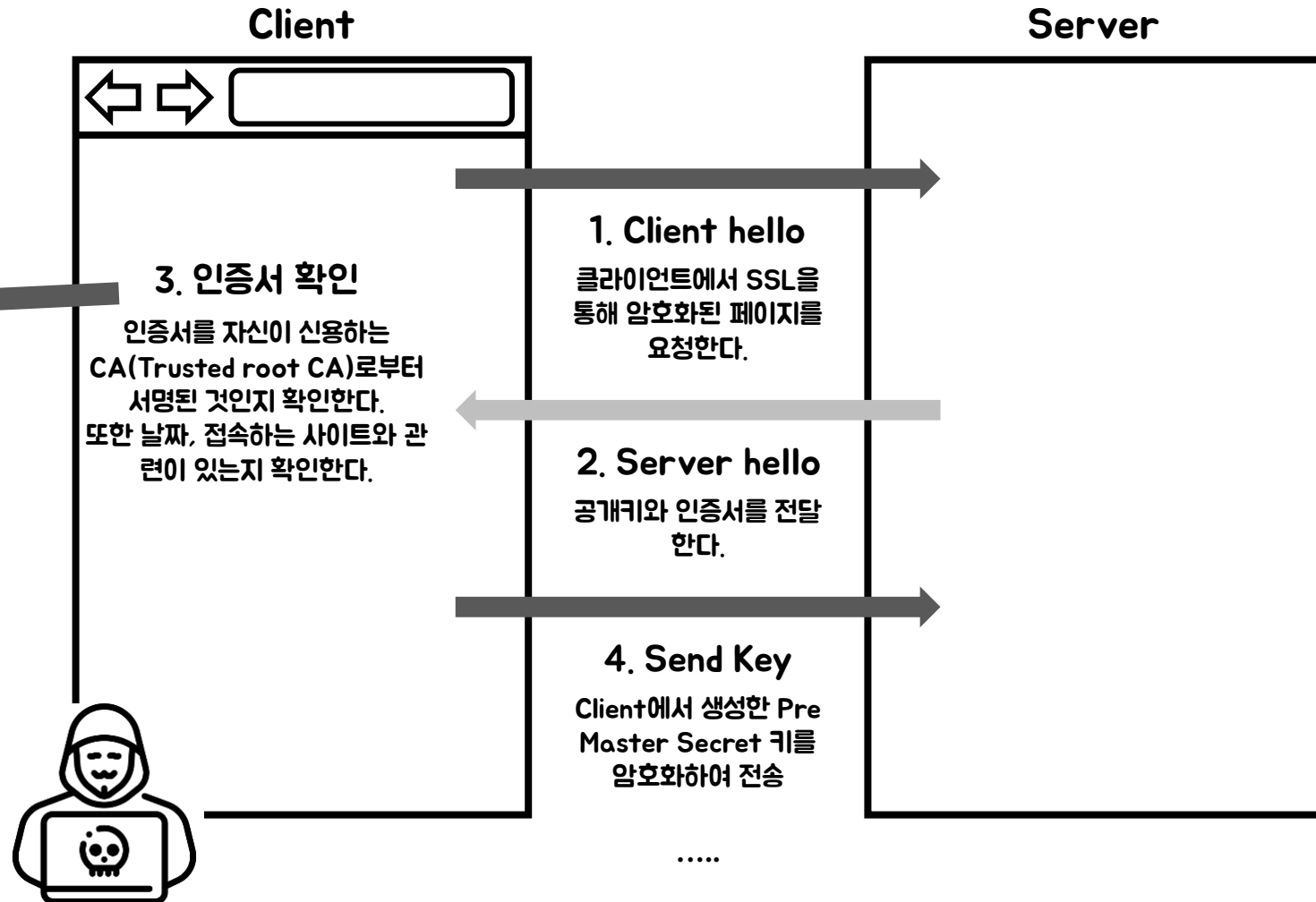
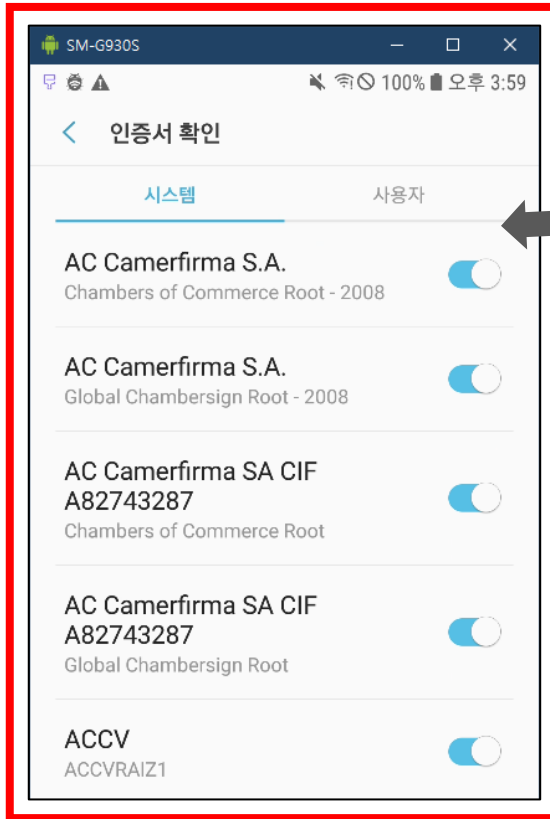
TLS/SSL, CA

Pinning

Androgoat

Bypass

Q & A



What is SSL Pinning...?

Pinning

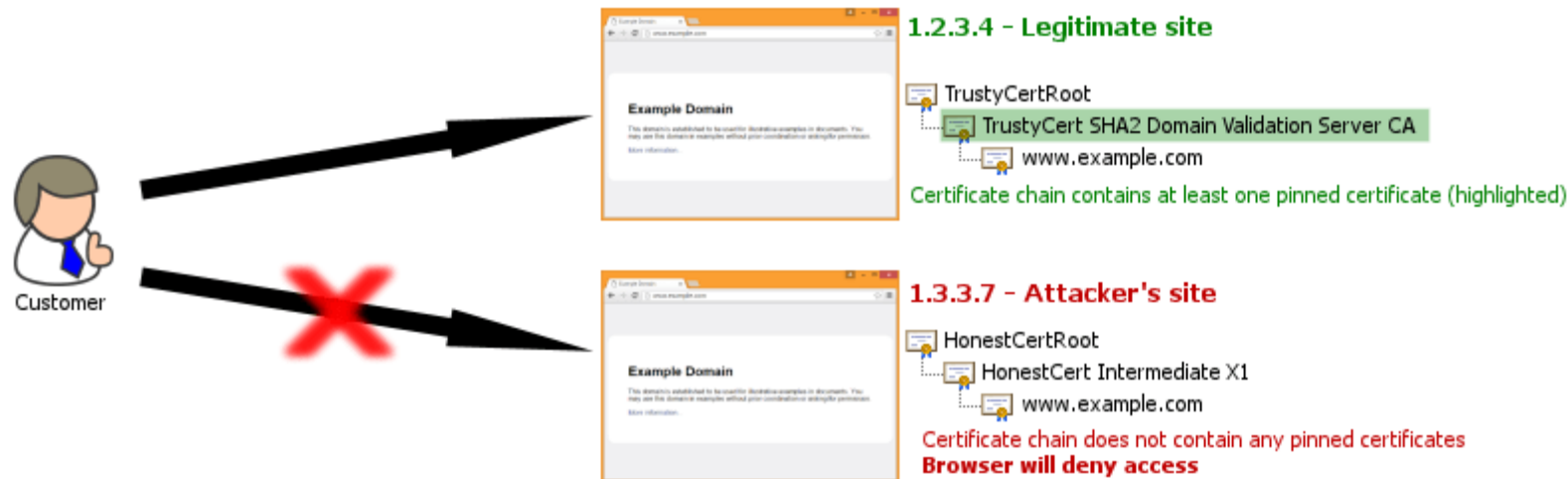
Androgoat

Bypass

Q & A

SSL Pinning

- SSL Pinning은 HTTP Public Key Pinning(HPKP) 라고도 불리며 SSL/TLS 암호화 통신에 취약한 중간자 공격을 보완하기 위해 사용된다.
- 공격자가 임의로 설정한 인증서를 이용한 공격을 방지하기 위해 미리 SSL/TLS 암호화 통신에서 사용할 인증서를 고정(Pinning)해 중간자 공격을 방지하는 방법



Ref : <https://rsec.kr/?p=346>

Androgoat

- Kotlin을 사용하여 의도적으로 개발된 오픈 소스 vulnerable/insecure한 애플리케이션
- 여러 항목 중 NETWORK INTERCEPTING 항목을 사용하여 진행
- <https://github.com/satishpatnayak/AndroGoat>

분석 환경

- Galaxy S7 (Rooting)
- Burp Suite Setting - Proxy

Androgoat


```
1  override fun onCreate(savedInstanceState: Bundle?) {
2      super.onCreate(savedInstanceState)
3      setContentView(R.layout.activity_traffic)
4      val txtString = findViewById<TextView>(R.id.result)
5      val HttpsButton = findViewById<Button>(R.id.httpsButton)
6      val PinningButton = findViewById<Button>(R.id.PinningButton)
7      val HttpButton = findViewById<Button>(R.id.httpButton)
8      HttpButton.setOnClickListener {
9          run(httpurl)
10     }
11     HttpsButton.setOnClickListener{
12         run(httpsurl)
13     }
14     PinningButton.setOnClickListener{
15         doPinning()
16     }
17 } //onCreate
```





Pinning

Androgoat

Bypass

Q & A

 SM-G930S

   100%  오후 4:58

Network Intercepting

Objectives:

1. Intercept HTTP Traffic
2. Intercept HTTPS Traffic in
 - 2.1 API 23 or below
 - 2.2 API 24 or above
 - 2.3 Understand network_security_config.xml
3. Understand and Bypass Certificate Pinning using different tools(Eg: JustTrustMe, TrustMeAlready, Frida, AppMon...etc)

HTTP

HTTPS

CERTIFICATE PINNING


```
1 fun run(url: String) {  
2     try {  
3         val request = Request.Builder()  
4             .url(url)  
5             .build()  
6         Toast.makeText(this, "Request sent to", Toast.LENGTH_SHORT).show()  
7  
8         client.newCall(request).enqueue(object : Callback {  
9             override fun onFailure(call: Call, t: Throwable) {  
10                override fun onResponse(call: Call, response: Response) {  
11                }}  
12            })  
13        catch (e: Exception)  
14        {  
15            e.printStackTrace()  
16        }  
17    }  
18 }
```

URL 이 외에도 다양한 설정을 할 수 있다.

```
interface Http {  
    fun url(method: String, url: String): Http  
    fun header(key: String, value: String): Http  
    fun form(key: String, value: String): Http  
    fun json(json: String): Http  
    fun file(key: String, filename: String, mine: String, file: ByteArray): Http  
    fun send(callback: (ResponseBody?, String?) -> Unit)  
}
```

```
1 fun doPinning() {  
2   doAsync {  
3     val url="owasp.org"  
4     try {  
5       val pinner1 = CertificatePinner.Builder()  
6         .add(url, "sha256/gdU/UHClHJBFbIdeKuyHm/Lq/aQvMLyuTtcvTEE/1JQ=")  
7         .add(url, "sha256/YLh1dUR9y6Kja30RrAn7JKnbQG/uEtLMkBgFF2Fuihg=")  
8         .add(url, "sha256/Vjs8r4z+80wjNcr1YKepWQboSIRi63WsWXhIMN+eWys=")  
9         .build()  
10      val client = OkHttpClient.Builder().certificatePinner(pinner1).build()  
11      val request = Request.Builder()  
12        .url("https://" + url)  
13        .build()
```

코드 처럼 URL에 해당하는 인증서의 해시 값을 설정하는 것으로 이후 SSL Handshake 과정에서 지정한 인증서를 사용하는지 확인한다.

Pratice

Repackaging

- 말 그대로 기존 애플리케이션에서 코드를 수정한 후 다시 컴파일하여 사용
- 신뢰하는 인증서를 추가하는 코드를 추가하거나 인증서 관련 코드를 수정하여 후회한다.
- 단점!
 - 안드로이드 7.0 이후로 사용자가 설치한 루트 인증서를 신뢰하지 않도록 변경
 - 애플리케이션 자체적으로 변조 탐지를 진행할 시 감지되어 사용 불가



Frida

- Ole가 개발한 Dynamic Binary Instrumentation 프레임워크
- 실행중인 애플리케이션에 코드 명령어를 삽입하여 프로세스를 추적, 분석, 디버깅할 수 있는 도구
- 파이썬 기반으로 동작하며 Windows, Linux, iOS, Android, macOS ... 등 다양한 OS에서 사용 가능



Bypass - Frida

[Pinning](#)[Androgoat](#)[Bypass](#)[Q & A](#)

```
156 internal fun check(hostname: String, cleanedPeerCertificatesFn: () -> List<X509Certificate>) {
157     val pins = findMatchingPins(hostname)
158     if (pins.isEmpty()) return
159
160     val peerCertificates = cleanedPeerCertificatesFn()
161
162     for (peerCertificate in peerCertificates) {
163         // Lazily compute the hashes for each certificate.
164         var sha1: ByteString? = null
165         var sha256: ByteString? = null
166
167         for (pin in pins) {
168             when (pin.hashAlgorithm) {
169                 "sha256" -> {
170                     if (sha256 == null) sha256 = peerCertificate.sha256Hash()
171                     if (pin.hash == sha256) return // Success!
172                 }
173                 "sha1" -> {
174                     if (sha1 == null) sha1 = peerCertificate.sha1Hash()
175                     if (pin.hash == sha1) return // Success!
176                 }
177             }
178         }
179
180         // If we couldn't find a matching pin, format a message.
181         // If we couldn't find a matching pin, format
182         val message = buildString {
183             append("Certificate pinning failure!")
184             append("\n Peer certificate chain:")
185             for (element in peerCertificates) {
186                 append("\n ")
187                 append(pin(element))
188                 append(": ")
189                 append(element.subjectDN.name)
190             }
191         }
```

<https://github.com/square/okhttp/blob/master/okhttp/src/main/kotlin/okhttp3/CertificatePinner.kt>

Pratice

Q & A