Protecting the network traffic of one billion people

Reverse-engineering proprietary cryptography in popular Chinese apps

Mona Wang, Jeffrey Knockel, and Zoë Reichert

Most downloaded apps in 2023?

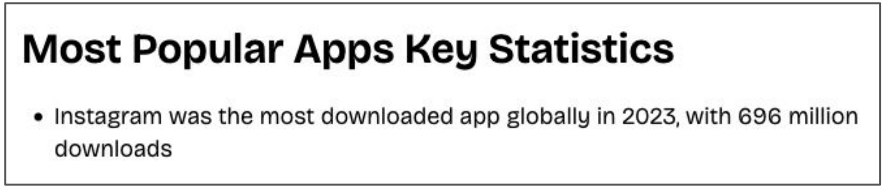
1		
2		
3		
4		
5		
6		

7		
8		
9		
10		
11		
12		

Source: https://www.businessofapps.com/data/most-popular-apps/ (1/30/2024)

Most downloaded apps in 2023?

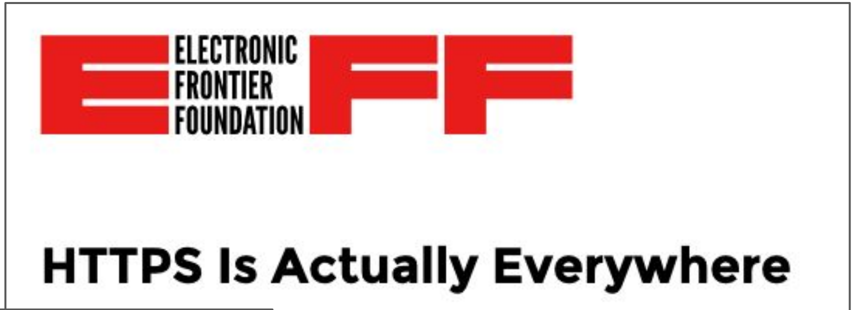
1	WeChat	1012
2	Alipay	901
3	Taobao	795
4	Pinduoduo	728
5	Instagram	696
6	Douyin	695
7	TikTok	654
8	QQ	583
9	Facebook	553
10	Baidu	491
11	Kuaishou	480
12	WhatsApp	475



Most Popular Apps Key Statistics

- Instagram was the most downloaded app globally in 2023, with 696 million downloads

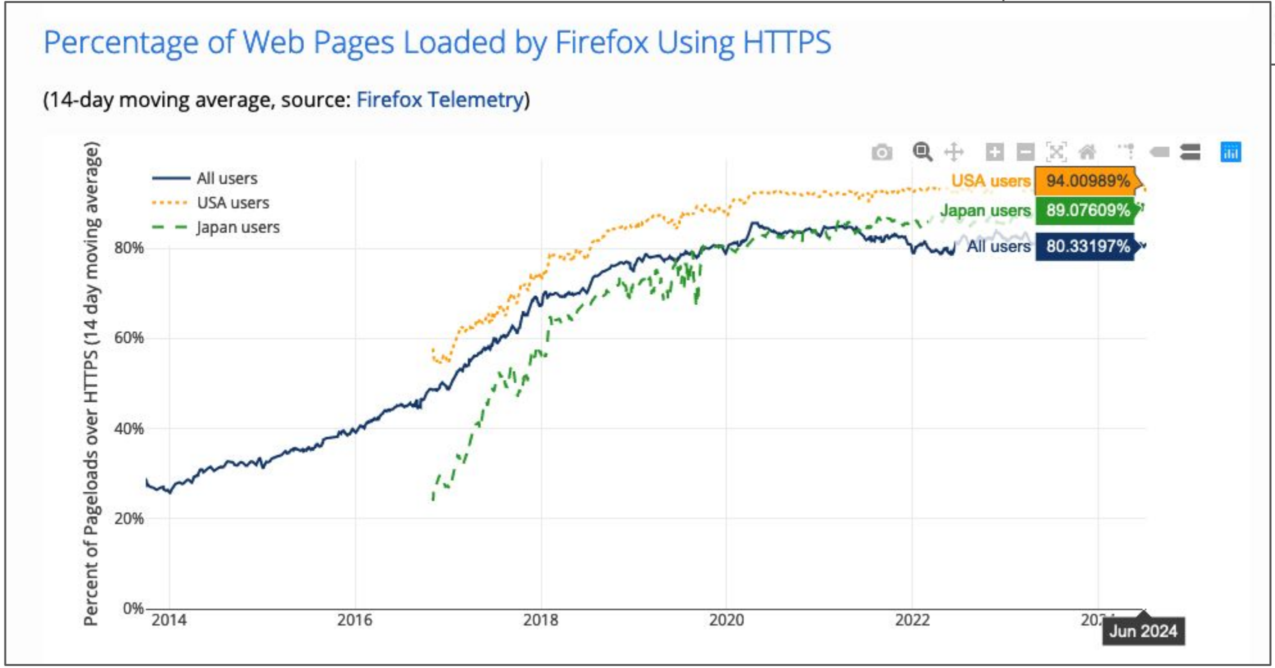
Source: https://www.businessofapps.com/data/most-popular-apps/ (1/30/2024)



ELECTRONIC FRONTIER FOUNDATION

HTTPS Is Actually Everywhere

SEPTEMBER 21, 2021



Percentage of Web Pages Loaded by Firefox Using HTTPS

(14-day moving average, source: Firefox Telemetry)

How many always use HTTPS/TLS?

	WeChat	1012.4
	Alipay	901
	Taobao	795.2
	Pinduoduo	728.2
	Instagram	696
	Douyin	694.9

	TikTok	654
	QQ	583.2
	Facebook	553
	Baidu	490.6
	Kuaishou	480.3
	WhatsApp	475

Source: https://www.businessofapps.com/data/most-popular-apps/ (1/30/2024)

How many always use HTTPS/TLS?

❌	WeChat	1012.4
❌	Alipay	901
❌	Taobao	795.2
❌	Pinduoduo	728.2
✅	Instagram	696
✅	Douyin	694.9

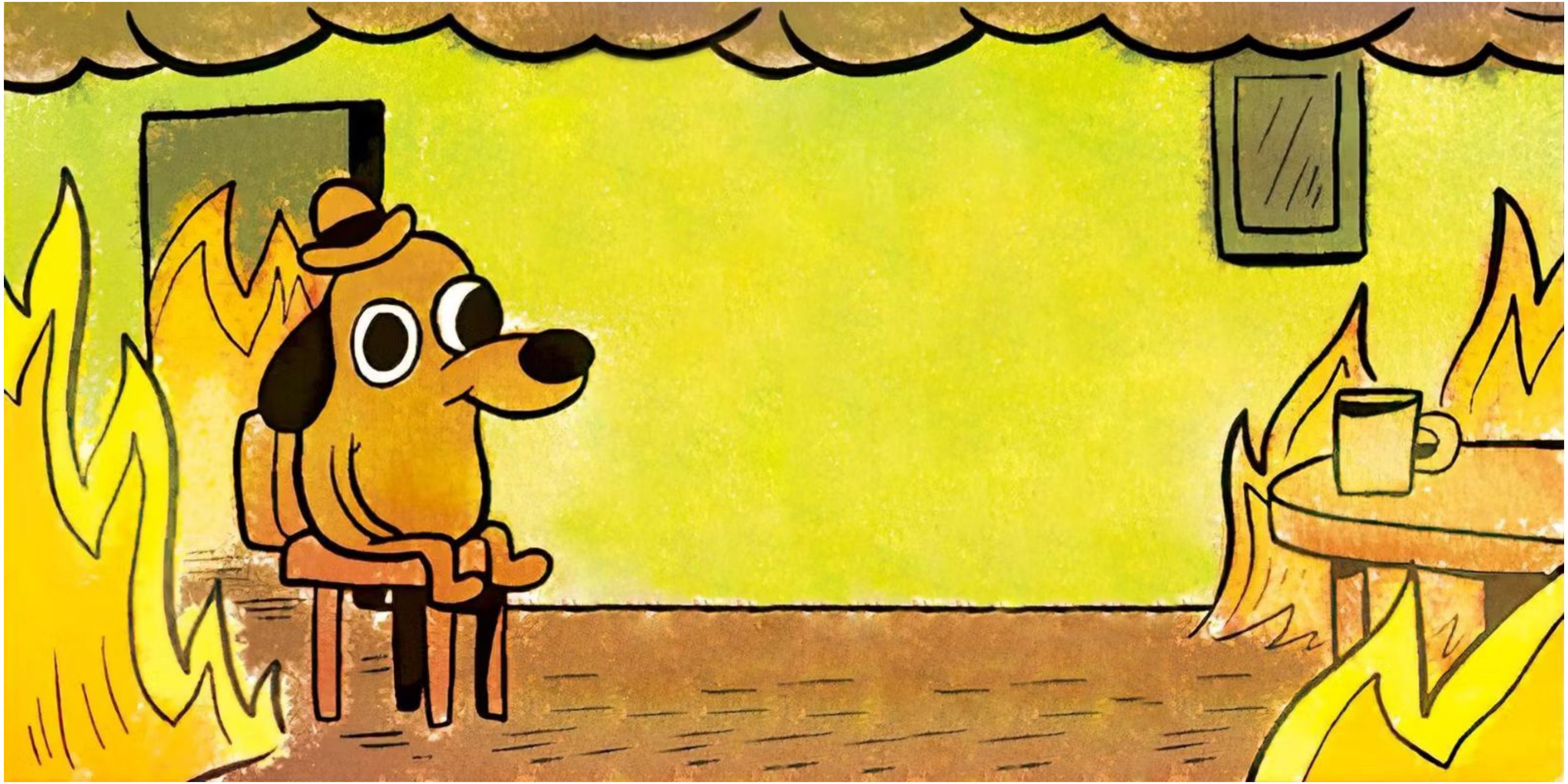
✅	TikTok	654
❌	QQ	583.2
✅	Facebook	553
❌	Baidu	490.6
❌	Kuaishou	480.3
✅	WhatsApp	475

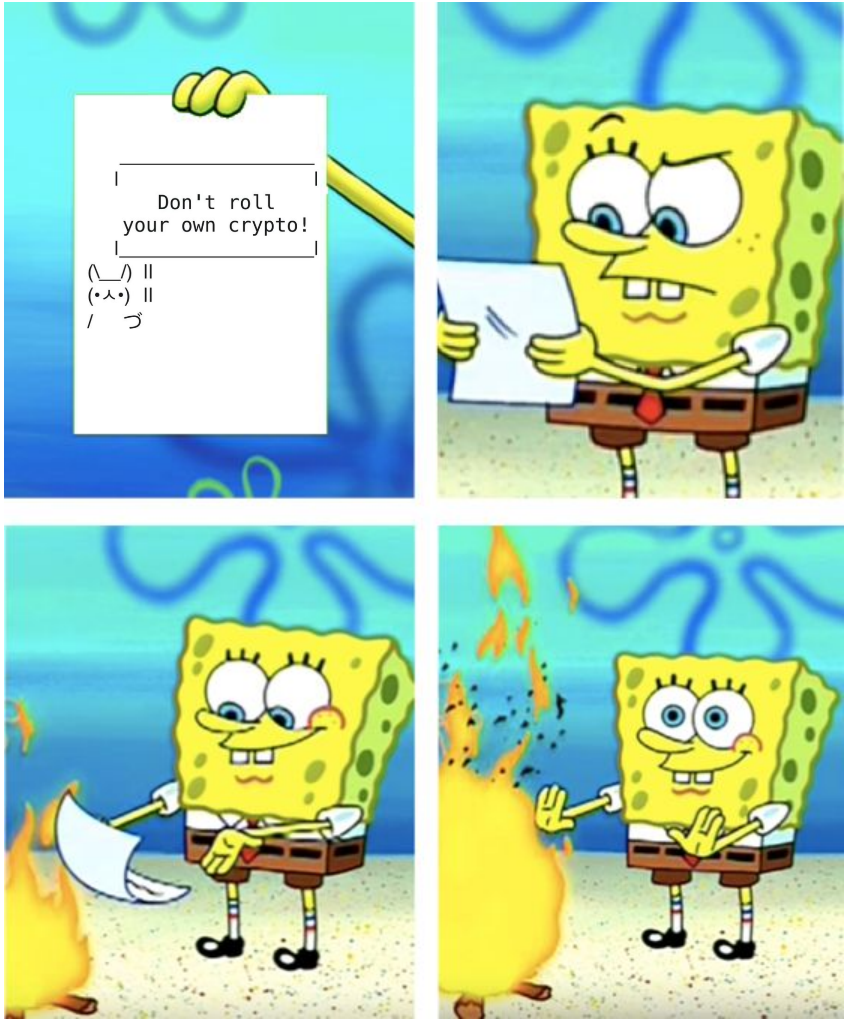
*but they're also **not not** encrypting…

many of them are **using proprietary cryptography**

Source: https://www.businessofapps.com/data/most-popular-apps/ (1/30/2024)

Uh-oh



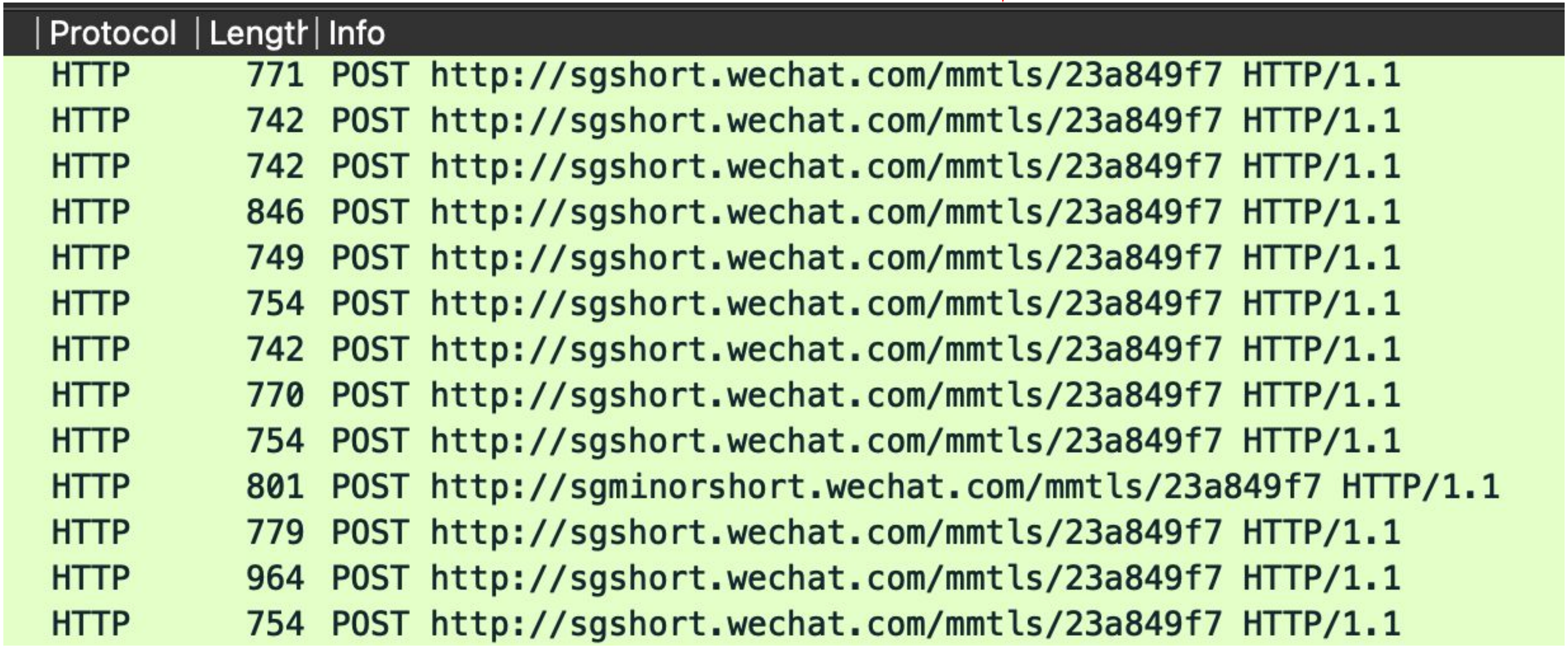

Don't roll your own crypto!
(__/) ||
(•ㅅ•) ||
/ 　 づ

Three case studies

1. WeChat (1 bill+ users) - Mona
2. Chinese-language keyboards (800 mill+ users) - Zoë
3. Browsers popular in China (400 mill+ users) - Jeff

WeChat

“MMTLS”?



```
| Protocol | Lengt | Info
  HTTP        771  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        742  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        742  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        846  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        749  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        754  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        742  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        770  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        754  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        801  POST http://sgminorshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        779  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        964  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
  HTTP        754  POST http://sgshort.wechat.com/mmtls/23a849f7 HTTP/1.1
```

1. Security: How secure is the cryptography?
2. Privacy: What sort of analytics data is WeChat collecting and sending over the network?

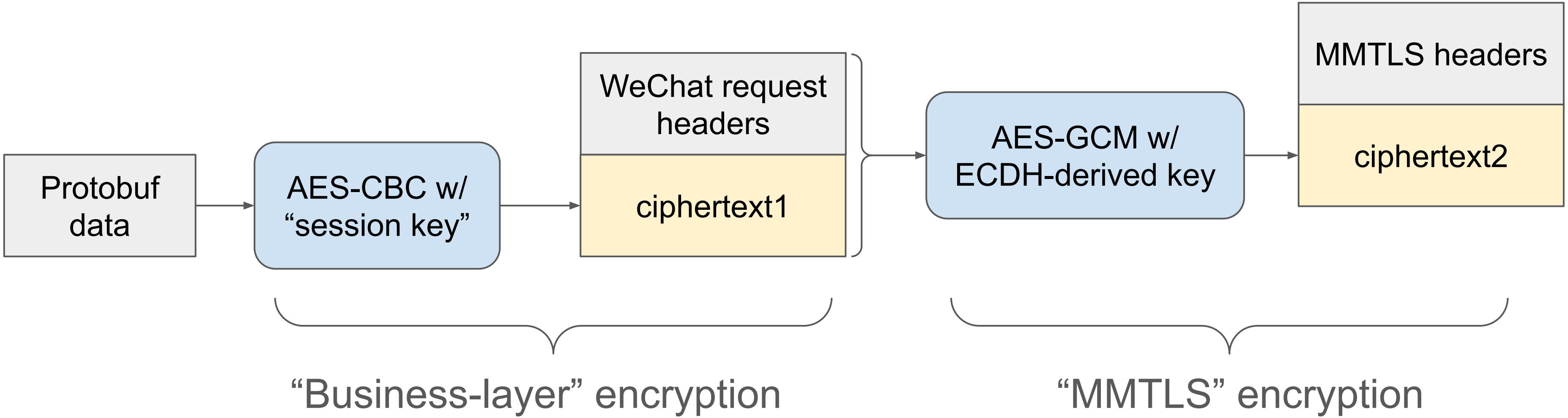
Security: How secure is the cryptography?

How does WeChat request encryption work?

How does WeChat request encryption work?

How does WeChat request encryption work?

- They're encrypted **twice**
  - (and also differently if you're logged-out)



Problems

Business-layer encryption

CBC-mode with “session key”

- No integrity/authenticity
  - “Signature” is forgeable
- Not CPA-secure (i.e. deterministic)
  - Key, IV re-use
- Key entropy measured around 100 bits
  - Server chooses session-key
- Long (key, IV) lifetime
  - As long as user has WeChat open
- AES-CBC
  - As long as user has WeChat open

MMTLS encryption

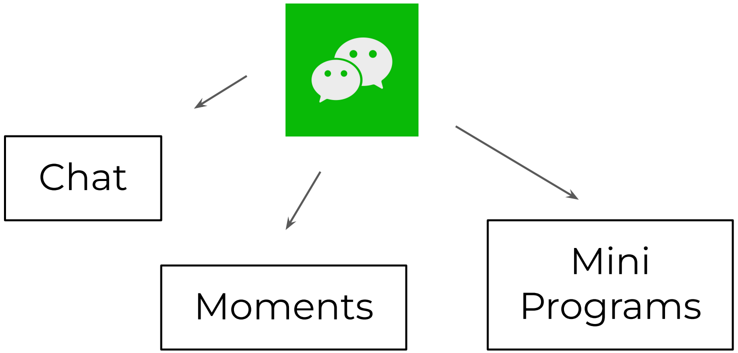
GCM-mode with ECDH-derived key

- Deterministic IV
  - GCM mode– can lead to accidental IV re-use
- Limited forward secrecy
  - Vast majority of requests are 0-RTT session resumptions
- No replay protections
  - Vast majority of requests are 0-RTT session resumptions

Privacy: What sort of analytics data is WeChat collecting and sending over the network?

We looked at **WeChat Mini Programs** as a case study

WeChat Mini Programs

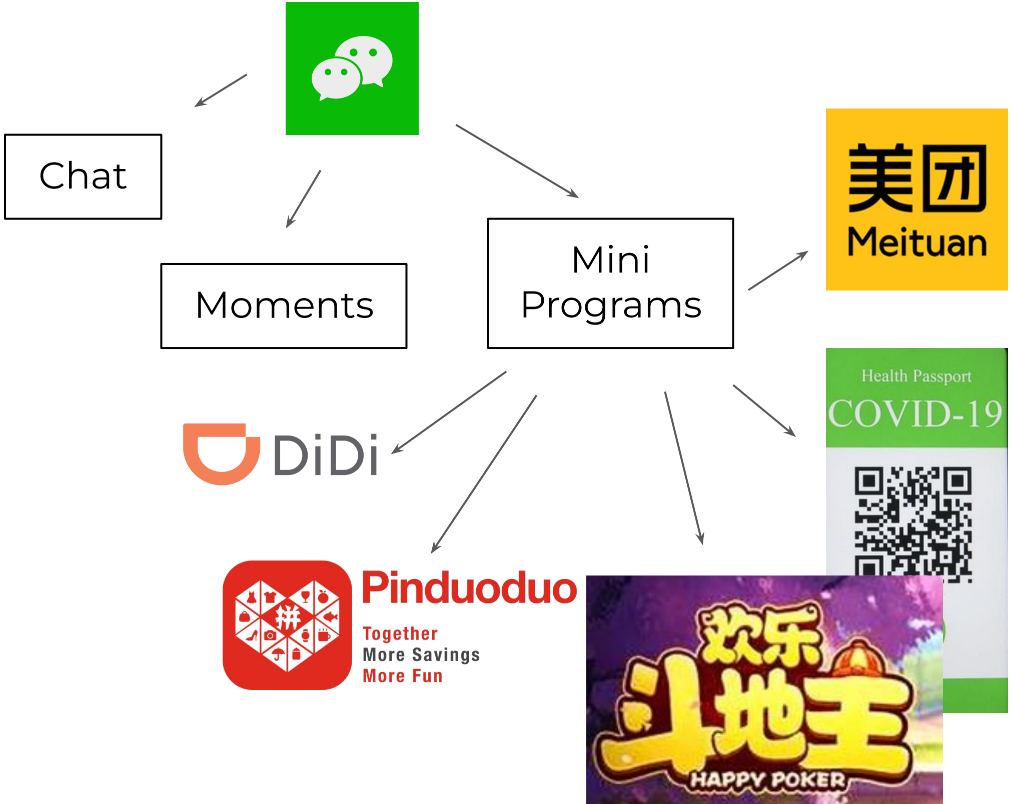


Chat

Moments

Mini Programs

WeChat Mini Programs

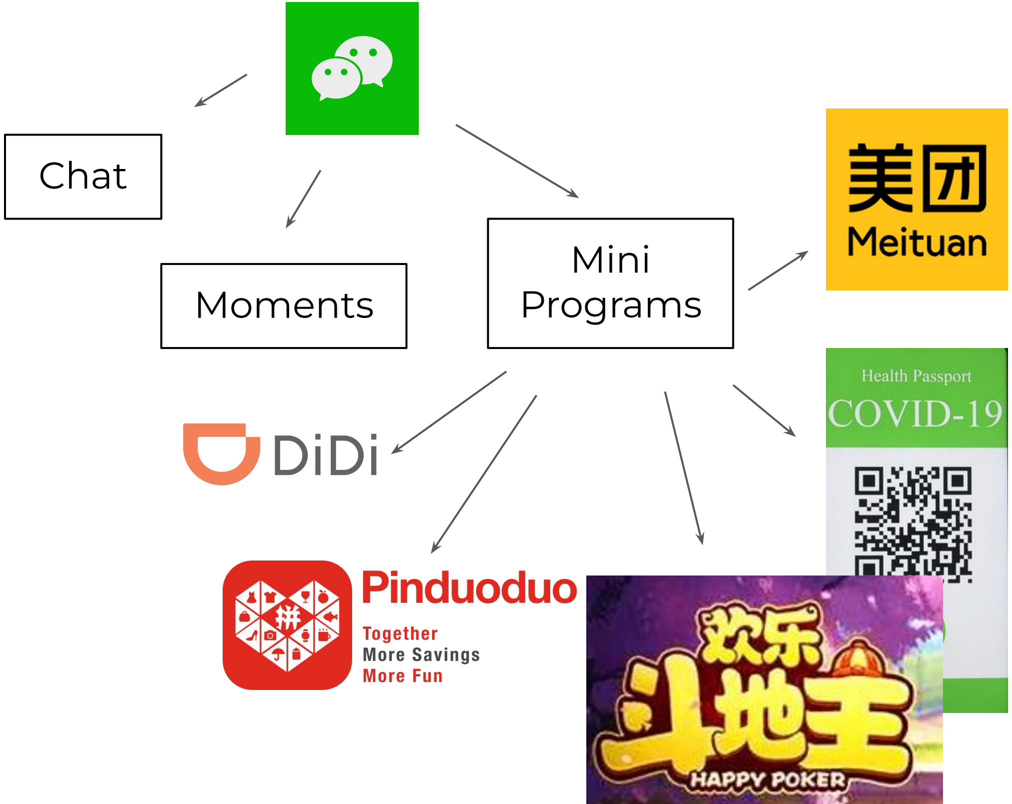


Chat

Moments

Mini Programs

WeChat Mini Programs



Chat

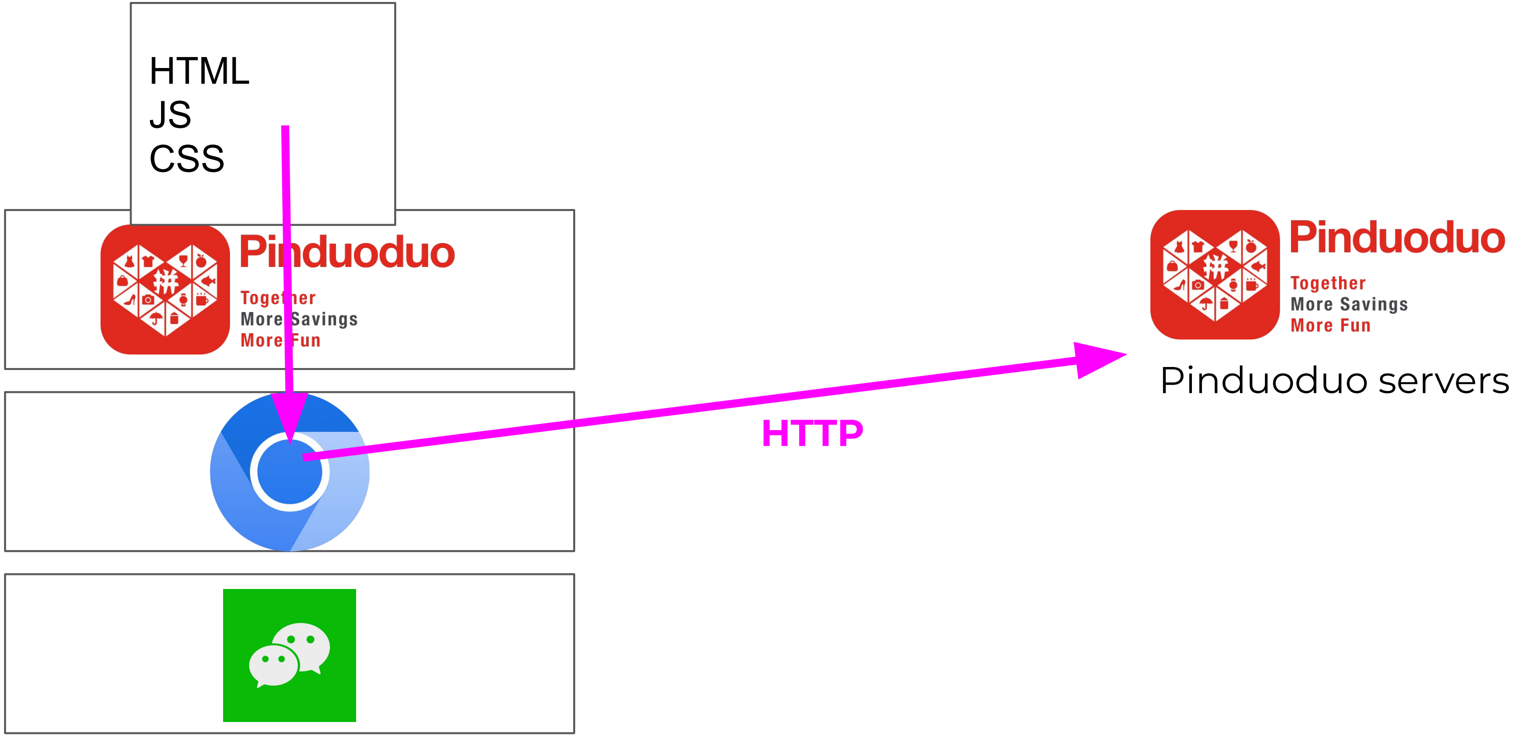
Moments

Mini Programs

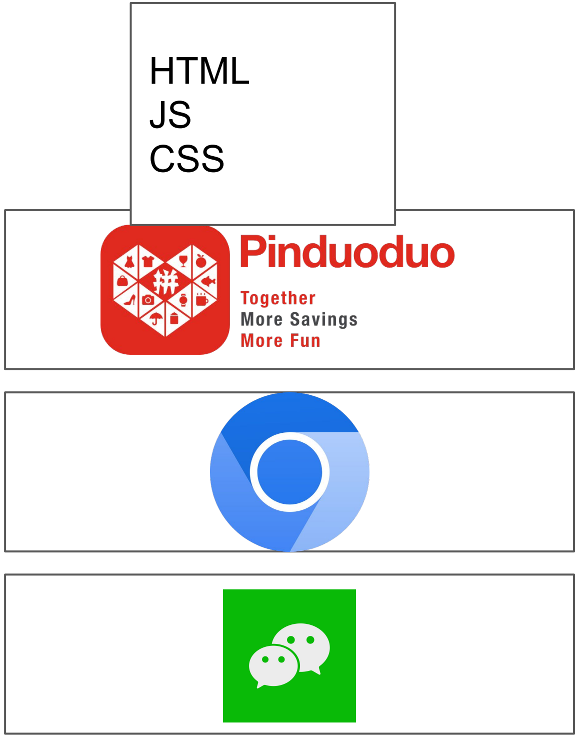
Google Play

App Store

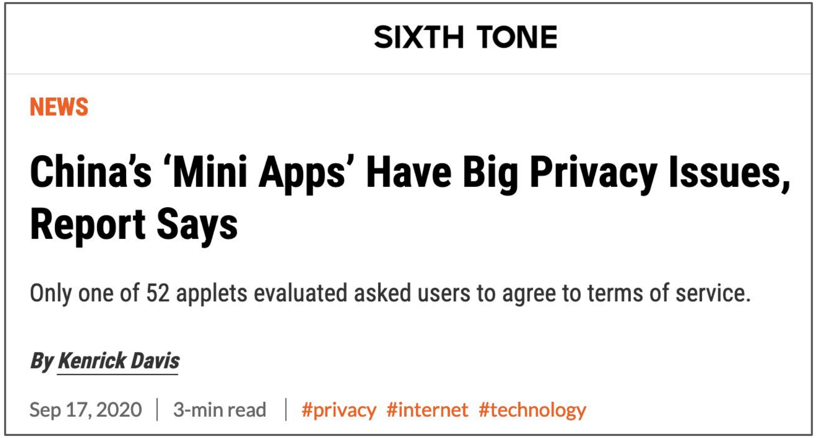
WeChat Mini Programs



WeChat Mini Programs



HTML
JS
CSS



SIXTH TONE

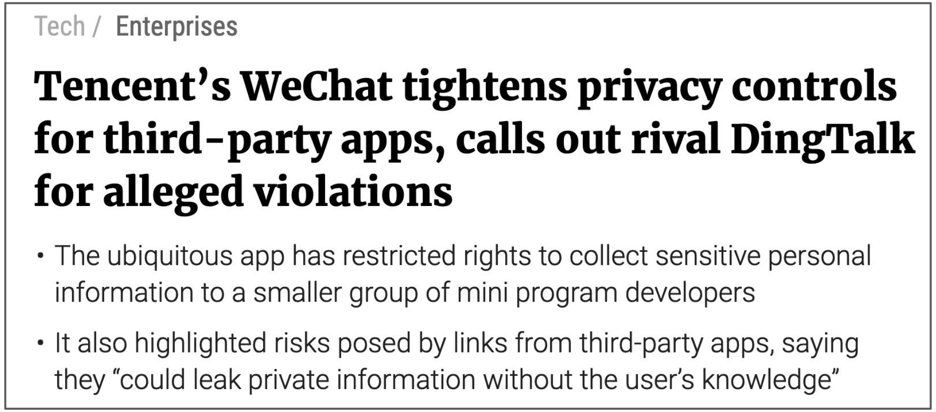
NEWS

China's 'Mini Apps' Have Big Privacy Issues, Report Says

Only one of 52 applets evaluated asked users to agree to terms of service.

By Kenrick Davis

Sep 17, 2020 | 3-min read | #privacy #internet #technology

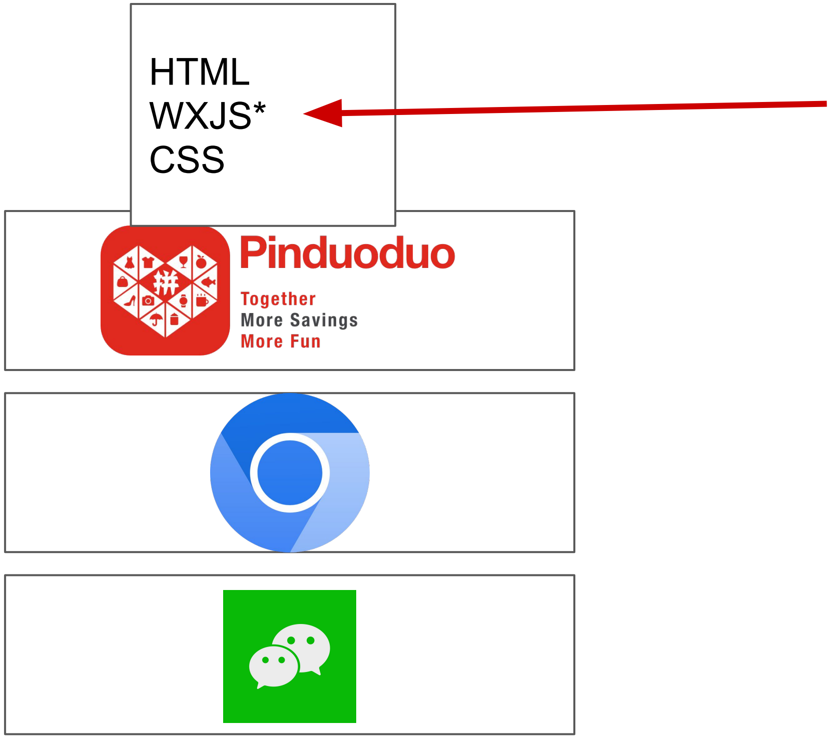


Tech / Enterprises

Tencent's WeChat tightens privacy controls for third-party apps, calls out rival DingTalk for alleged violations

- The ubiquitous app has restricted rights to collect sensitive personal information to a smaller group of mini program developers
- It also highlighted risks posed by links from third-party apps, saying they "could leak private information without the user's knowledge"

WeChat Mini Programs

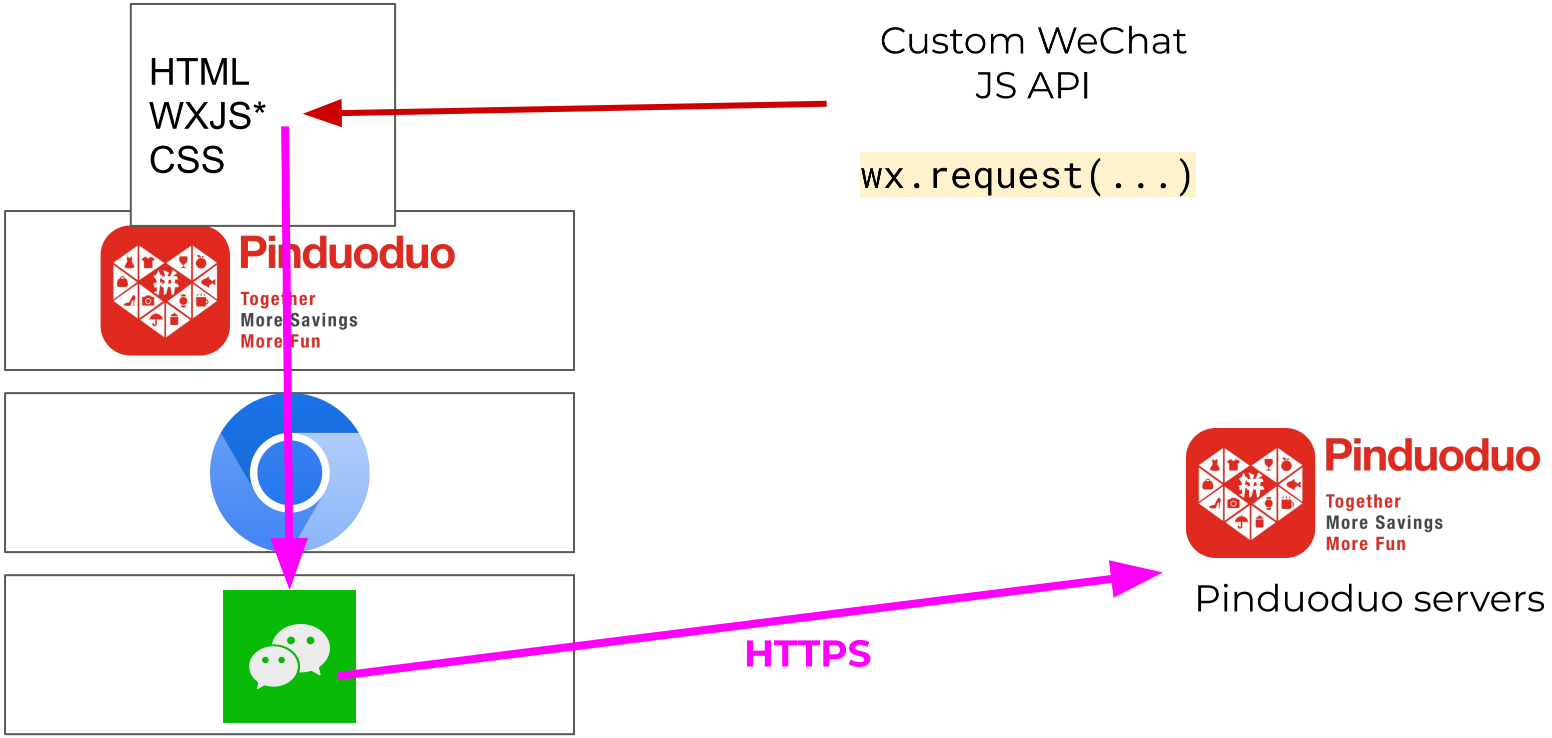


HTML
WXJS*
CSS

Custom WeChat
JS API

```
wx.request(...)
```

WeChat Mini Programs



HTML
WXJS*
CSS

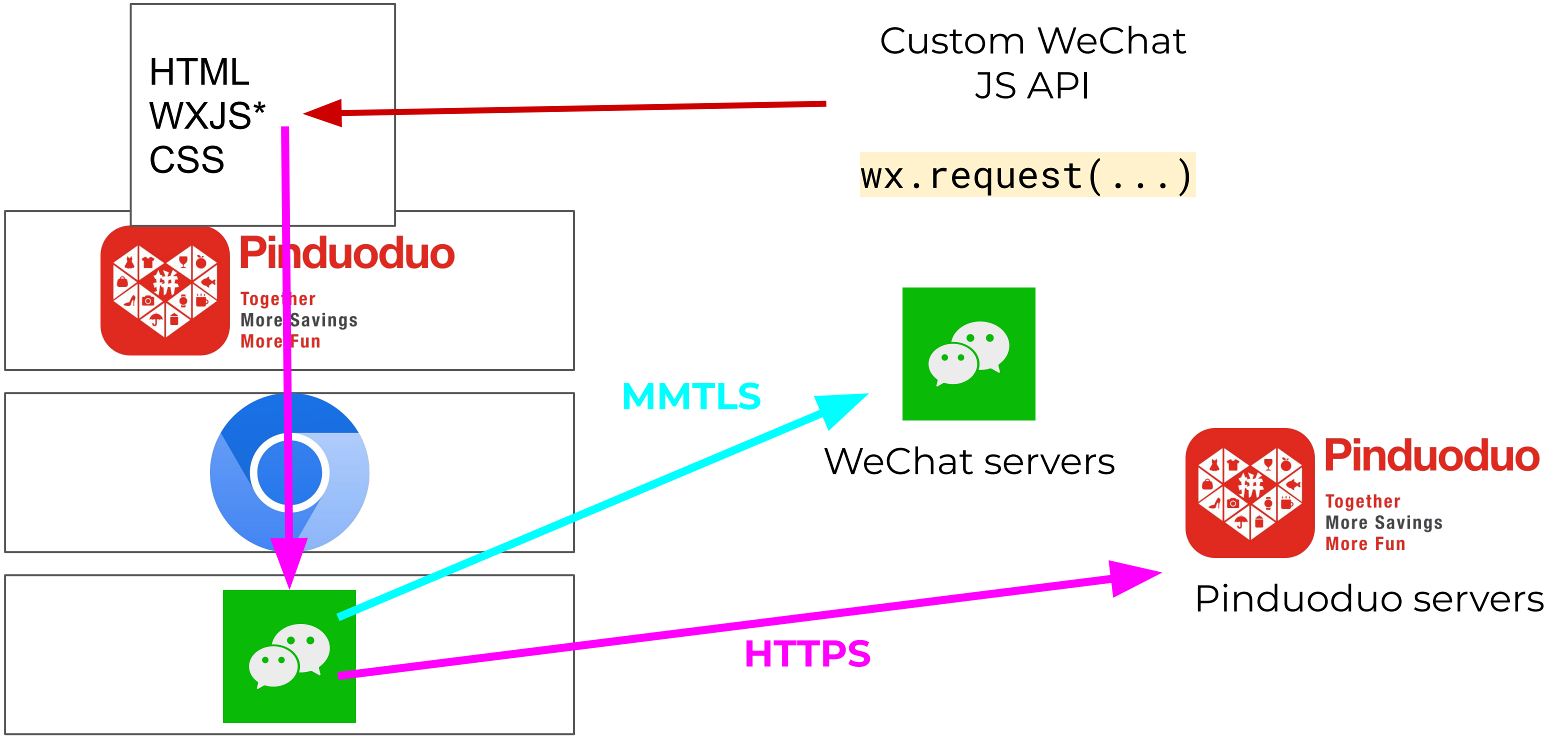
Custom WeChat
JS API

`wx.request(...)`

HTTPS

Pinduoduo servers

WeChat Mini Programs



HTML
WXJS*
CSS

Custom WeChat
JS API

`wx.request(...)`

MMTLS

WeChat servers

HTTPS

Pinduoduo servers

Decrypted MMTLS request during Mini Program usage

```
"pagepath": "pages/item/detail/detail.html",
"referpagepath": "pages/search_pg/index/index.html",
"query":
     "ss_pos_type=normal&referer=https://wq.jd.com/wxapp/pages/search_pg/index/index&
     __navVer=1&search_tab_result=0&csid=9527036b7fdef1f5619ce3823fa6da80_17067522670
     69_1_1706752267069&actid=&search_result=2&appCode=msc9ed9e31&ss_item_type=1
     &__pid=Pxatsdyboel0a&cover=https://img20.360buyimg.com/n1/jfs/t1/200392/11/34403
     /24690/6538dd03Fe7c3bb76d48516219c63a15a.jpg!q70.dpg&pps=&ss_tab_type=1&sf=14&po
     s=1&price=1675.00&name=斯凯奇（Skechers）男士运动休闲鞋 Segment The Search 经典潮流舒
     适低帮鞋Dark Brown 标准
     41.5/US8.5&sku=10089285173453&factory_goods=0&key=XXXXXX_SEARCH_DATA&navStart=17
     06752266671",
"clickTime": 1706752266696,
"reportTime": 1706752266767,
"anchorTargetRelatedText": "仅剩2件斯凯奇（Skechers）男士运动休闲鞋 Segment The Search 经
典潮流舒适低帮鞋 Dark Brown 标准41.5/US8.5
织物¥1675分期免息闪电退款包邮HEADED EAGLE海外专营店"
}
```

WeChat Mini Programs Analytics (We分析/WeData)

- Sends data about Mini Program usage/browsing back to WeChat over MMTLS
- Opt-in by default for all Mini Programs
- By default, all browsing activity is sent
    - Sends location, device metadata, page details, page path

	All MPs	Health	Shopping
Total analyzed	104	40	24
Profile update flows	51	19	12
% sent to WeChat	84.3%	68.4%	83.3%
Search flows	85	30	24
% sent to WeChat	72.9%	76.7%	70.8%
Browsing flows	96	39	22
% sent to WeChat	76.0%	89.7%	77.3%

Please help us continue this work!

- # MMTLS users is on a similar order of magnitude as # TLS users
  - More scrutiny from security researchers is needed; it deserves as much scrutiny as TLS!
- Studying these protocols enables future privacy studies such as this one

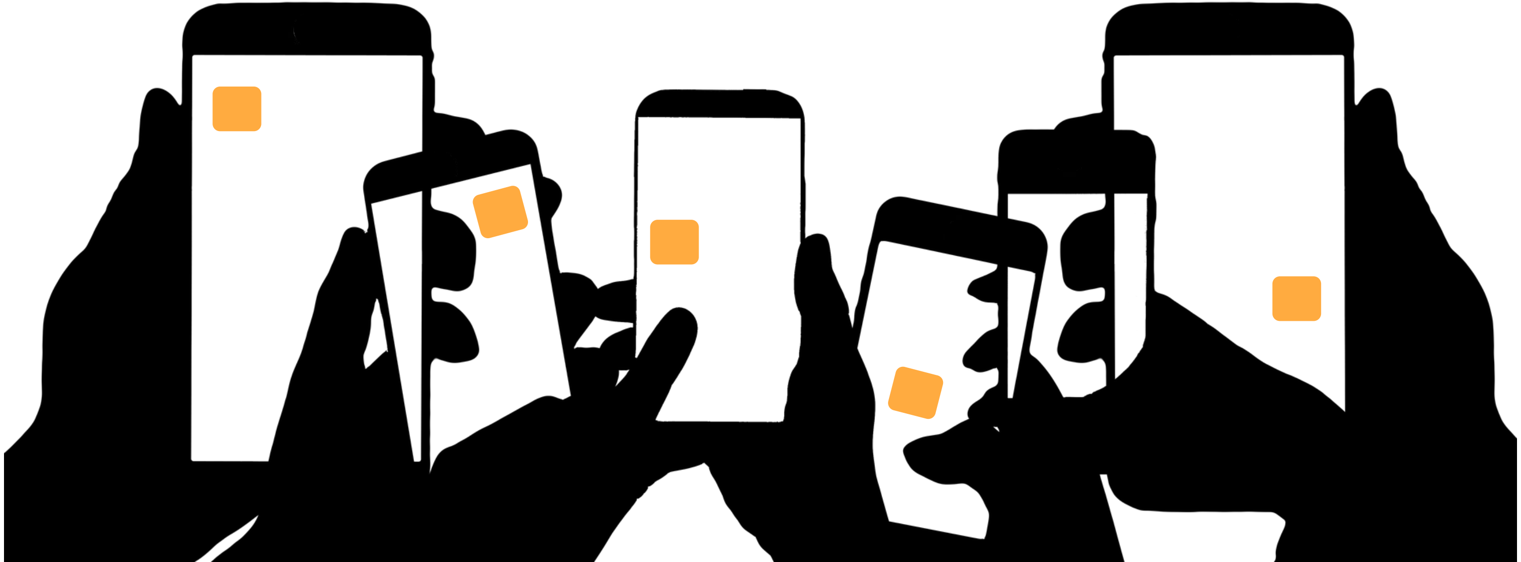
Vulnerabilities across popular Chinese-language keyboard apps

Imagine one billion people around the world typing, and as they're typing, a network eavesdropper could be reading every character they write...

including passwords, sensitive information, and private (even end-to-end encrypted) messages.

What do all of these people have in common?

They are all users of keyboard apps from
Baidu, Honor, Huawei*, iFlytek, OPPO, Samsung, Tencent, Vivo, and Xiaomi
— the most popular cloud-based keyboard apps in China.



We found that

8 out of 9

vendors’ keyboard apps revealed user keystrokes

As network eavesdroppers, we were able to **intercept** and **completely reveal** the contents of users’ keystrokes in transit.

Why do so many people need a “cloud-based” keyboard app?

Many of us may be used to typing out Latin script– using an alphabet of *only 26 letters*– but typing Chinese characters is a little bit trickier.

- There are **tens of thousands** of Chinese characters (used with varying frequency, of course).
- The **keyboard apps** we analyzed are installed on your phone or computer, allowing you to easily type in Chinese.
- Market research estimates that **nearly one billion users around the world use these apps!**

Case Study: What kinds of vulnerabilities did we find in Sogou Input Method?

In each version of the app:

iOS 11.21 → predictable AES key and IV, however, no known exploit

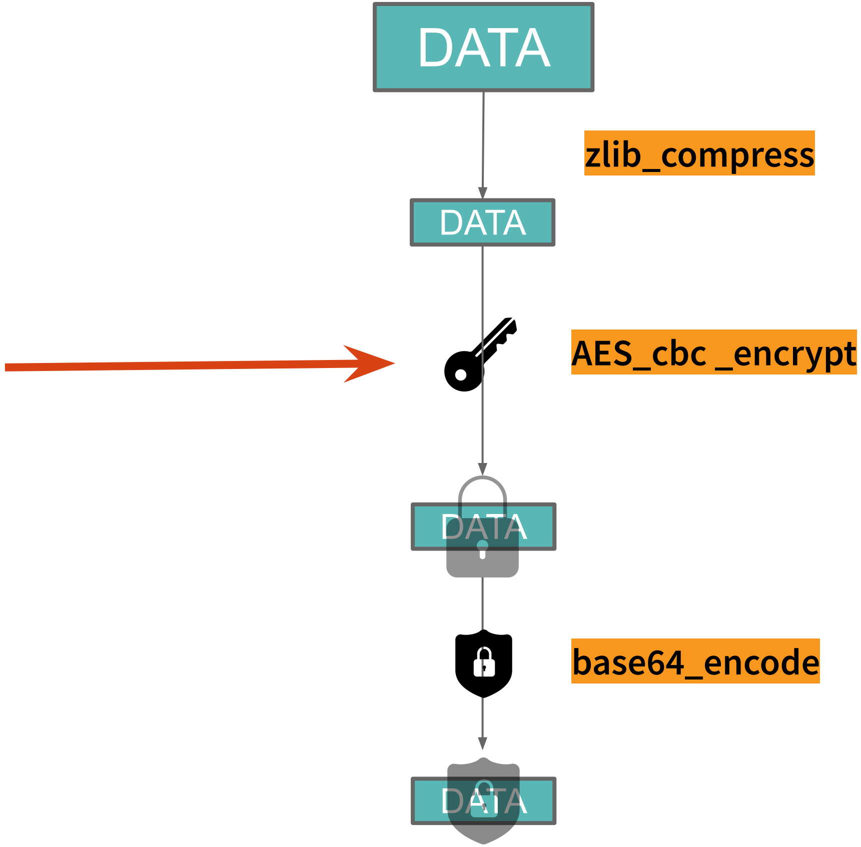
Windows 13.4 → vulnerable to a padding oracle attack

Android 11.20 → vulnerable to a padding oracle attack (with slight modifications)

Detailed explanation of Windows and Android exploits to follow…

Sogou’s EncryptWall

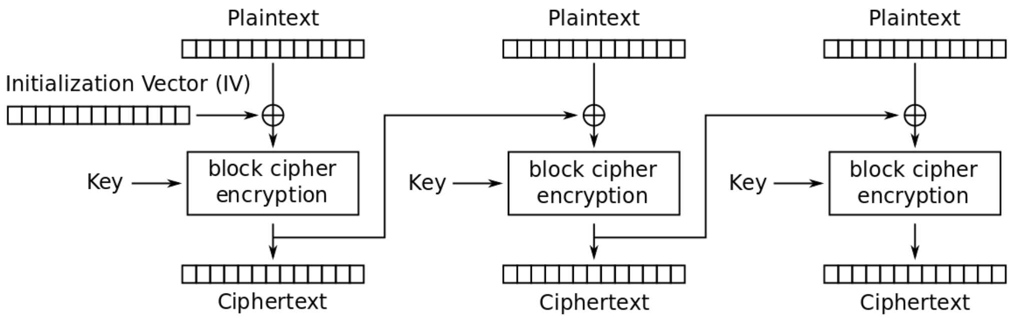
- EncryptWall request is sent as an **HTTP POST** request to a Sogou EncryptWall API endpoint
    - Contains the AES key and initialization vector (IV), which are used to encrypt the other data in the request
    - In most cases the AES key and IV are both generated randomly for each request
- As network eavesdroppers, we were able to intercept the **HTTP POST** requests and alter them ourselves



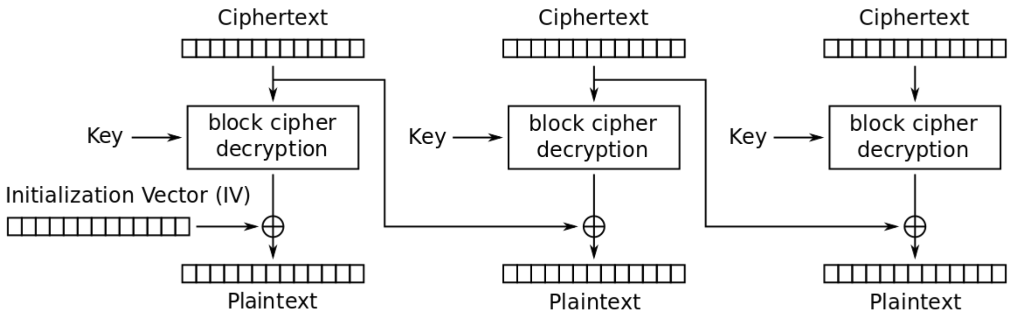
Padding Oracle Attack

The first CBC padding oracle attack was published in 2002, and Sogou’s use of CBC-mode left their encryption vulnerable to this type of attack (with some modifications).

AES-CBC encryption and decryption

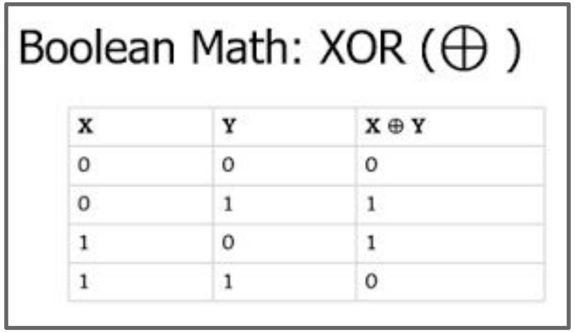


Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

⊕ is the XOR (“exclusive or”) operation. It works bit-by-bit!



Boolean Math: XOR (⊕)

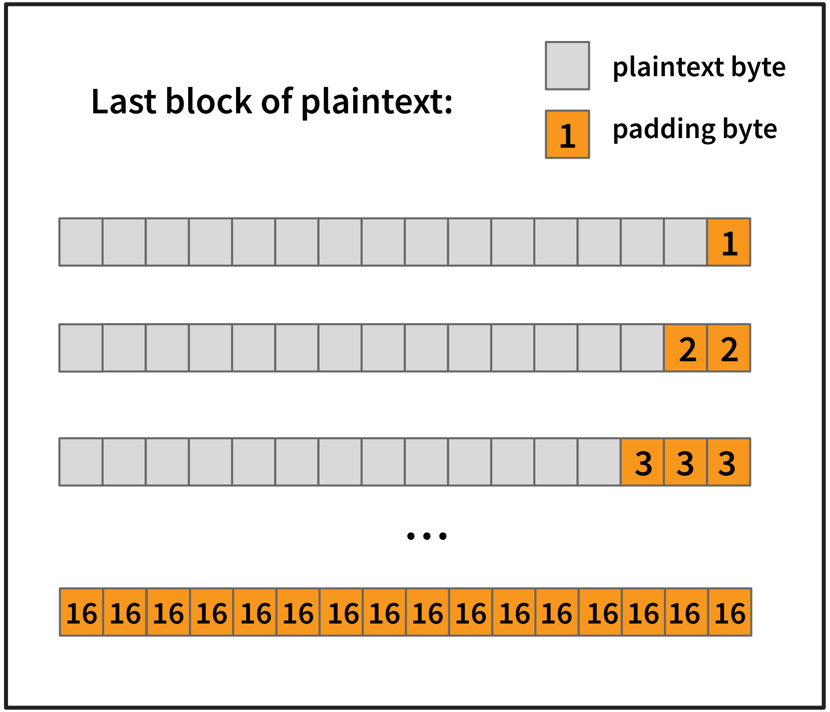
X	Y	X ⊕ Y
0	0	0
0	1	1
1	0	1
1	1	0

⊕ is easy to reverse: **the inverse operation of XOR is itself!!**

Padding Oracle Attack Part 1: PKCS7 Padding

One of the first things the server does after receiving and decrypting encrypted data is to check if the padding is correct.

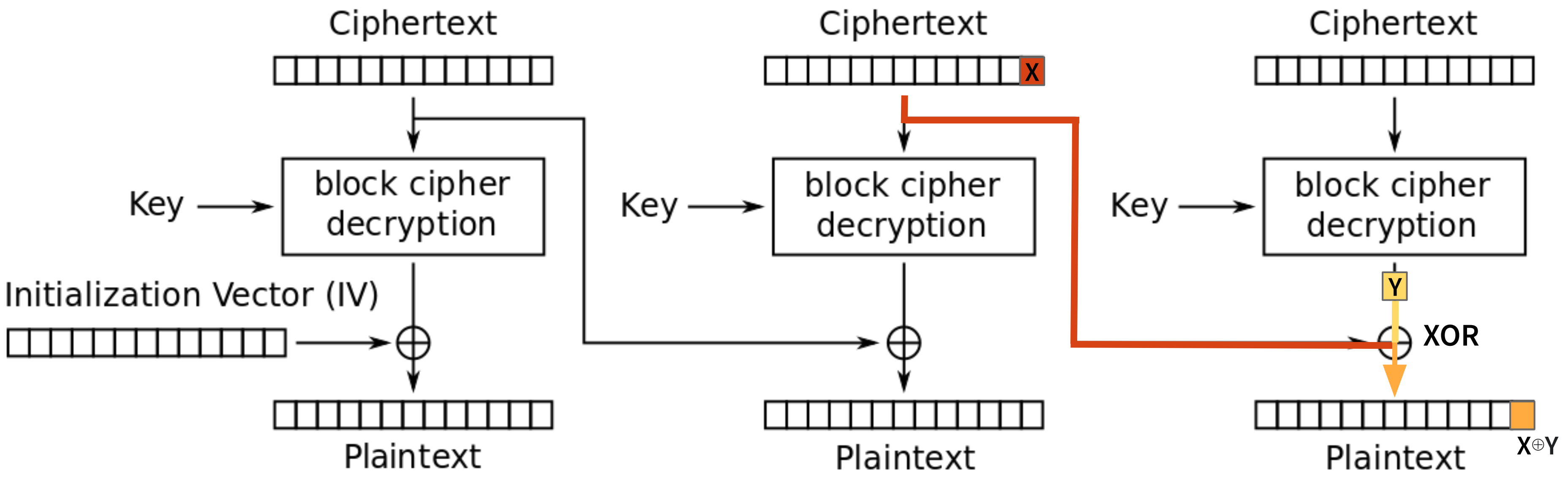
Sogou’s server returned unique HTTP status codes when the padding was incorrect versus when the padding was correct. This is our padding oracle.



Last block of plaintext:

□ plaintext byte

1 padding byte



Cipher Block Chaining (CBC) mode decryption

Padding Oracle Attack Part 2: changing the last byte of the penultimate ciphertext block changes the last byte of the final plaintext block.

There are only

256

possible bytes

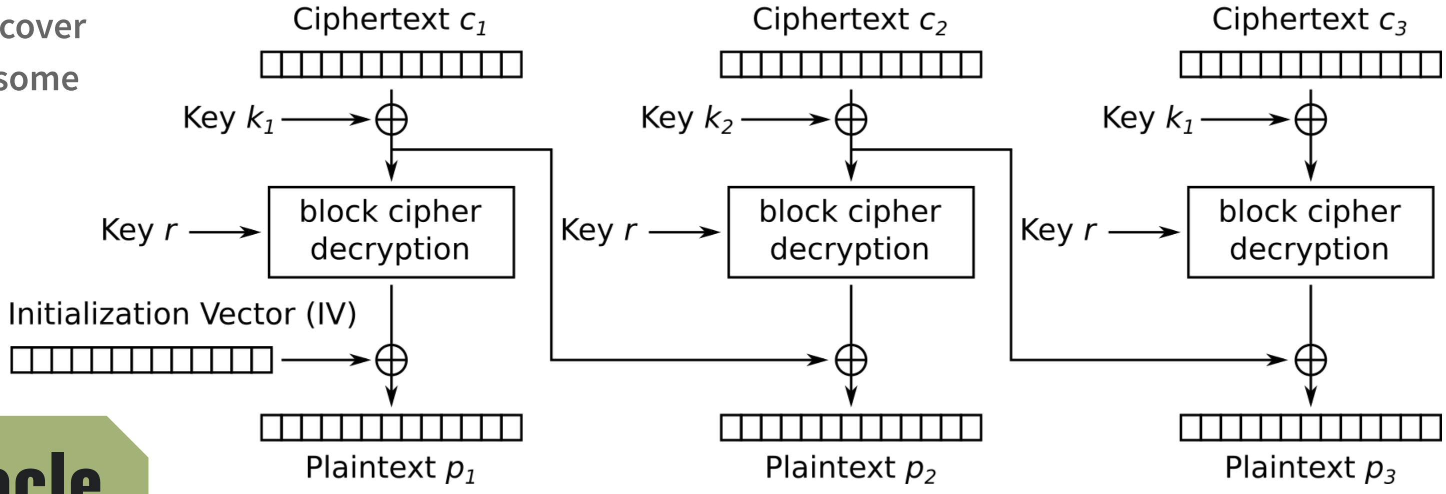
Windows

IV = ??? (encrypted)

But we were able to recover the decrypted IV with some creative XOR-ing!

Android

IV = “EscowDorisCarlos”



Padding Oracle Attack Part 3

This diagram shows decryption using Sogou’s EncryptWall in the Android version of the app

Padding Oracle Attack Part 4: More Vulnerabilities in the Android Version

Sogou sent their AES key *r* XORed with *k*, which means that if we had *k*, we could easily retrieve *r*.

$(k_2 \oplus r_2) \rightarrow$ unreadable $\qquad (k_2 \oplus r_2) \oplus k_2 \rightarrow r_2$

We were able to recover k_2 and therefore the second half of their AES key, *r*!

One of the things we decrypted in the Android version was a list of every app installed on the Android device…eek!

Sogou's Encryption System vs Other Keyboard Apps

Use of Sogou:

Sogou's EncryptWall was also used by:

- QQ Pinyin (also developed by Tencent)
- Samsung
- Huawei*
- Xiaomi
- OPPO
- Vivo

(Sogou's keyboard comes pre-installed on some phones. So do Baidu's and iFlytek's.)

Other apps:

Common vulnerabilities across apps:

- Static/predictable keys and IVs
- Outdated cipher systems (e.g. ECB mode, DES)
- Failing to follow modern cryptographic standards

For details on each keyboard app's vulnerabilities, read our report.

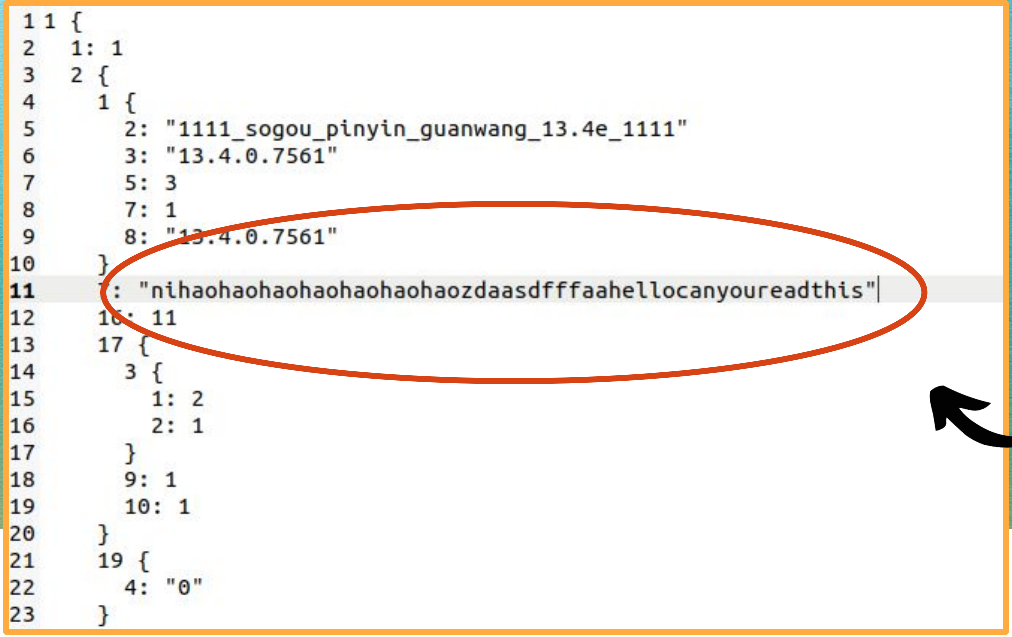
Use of Transport Layer Security:

Wrapping transmitted data in TLS blocks our side-channel attack.

Only Huawei used TLS before our disclosures.

Our solution to these issues:

We recommended to each vendor that they use TLS (a widely-respected, industry standard cryptographic protocol).



```
1 {
  1: 1
  2 {
    1 {
      2: "1111_sogou_pinyin_guanwang_13.4e_1111"
      3: "13.4.0.7561"
      5: 3
      7: 1
      8: "13.4.0.7561"
    }
    7: "nihaohaohaohaohaohaozdaasdfffaahellocanyoureadthis"
    16: 11
    17 {
      3 {
        1: 2
        2: 1
      }
      9: 1
      10: 1
    }
    19 {
      4: "0"
    }
```

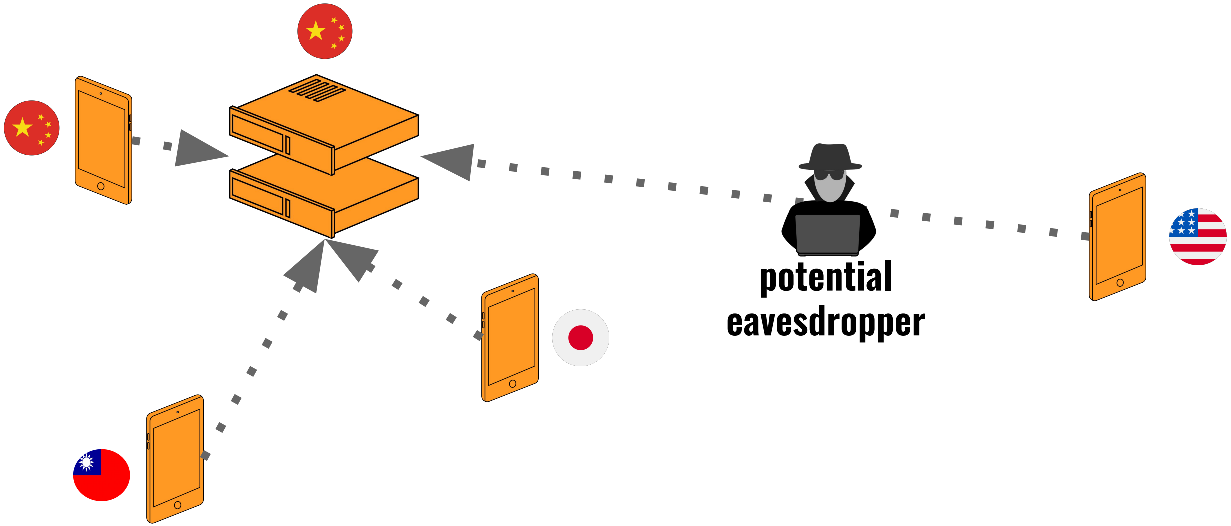
What did we decrypt?

→ URL, GET parameters, and raw POST data

That's weird… it looks like what we typed when we were testing!

What’s going on here?

All of these keyboard apps are sending **user keystrokes** to servers in China, and they are (for the most part) using very poor encryption to do so.

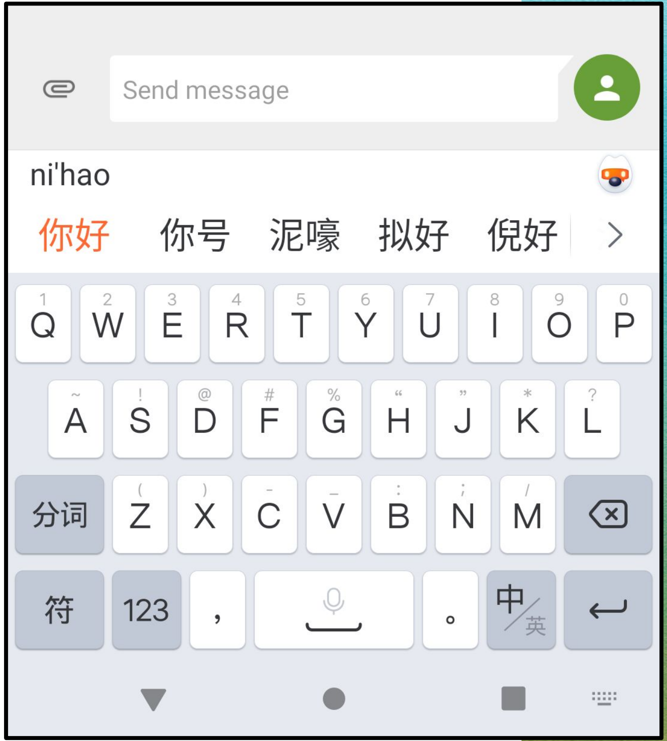


(Most users are in China, but market research shows that Taiwan, Japan, and the US host not-insignificant numbers of users as well)

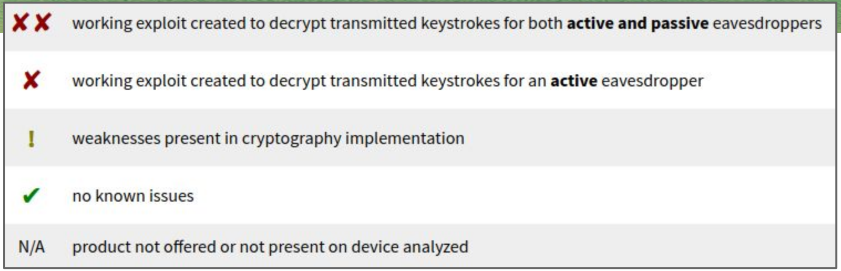
But Why?

These **keyboard apps** rely on a **cloud-based predictive text feature** (which can be enabled by the user) in order to suggest which character a user may want to type.

This is especially important when typing in Chinese because there are so many characters.



Results of Our Disclosures



✗✗	working exploit created to decrypt transmitted keystrokes for both **active and passive** eavesdroppers
✗	working exploit created to decrypt transmitted keystrokes for an **active** eavesdropper
!	weaknesses present in cryptography implementation
✓	no known issues
N/A	product not offered or not present on device analyzed

Before our disclosures*

Keyboard developer	Android	iOS	Windows
Tencent	✗	✓	✗
Baidu	!	!	✗✗
iFlytek	✗✗	✓	✓

Device manufacturer	Own	Sogou	Baidu	iFlytek	iOS	Windows
Samsung	✗✗	✓*	✗✗	N/A	N/A	N/A
Huawei	✓*	✓	N/A	N/A	N/A	N/A
Xiaomi	N/A	✗*	✗✗	✗✗	N/A	N/A
OPPO	N/A	✗	✗✗*	N/A	N/A	N/A
Vivo	✓*	✗	N/A	N/A	N/A	N/A
Honor	N/A	N/A	✗✗*	N/A	N/A	N/A

After our disclosures

Keyboard developer	Android	iOS	Windows
Tencent	✓	✓	✓
Baidu	!	!	!
iFlytek	✓	✓	✓

Device manufacturer	Own	Sogou	Baidu	iFlytek	iOS	Windows
Samsung	✓	✓*	!	N/A	N/A	N/A
Huawei	✓*	✓	N/A	N/A	N/A	N/A
Xiaomi	N/A	✓*	!	✓	N/A	N/A
OPPO	N/A	✓	!*	N/A	N/A	N/A
Vivo	✓*	✓	N/A	N/A	N/A	N/A
Honor	N/A	N/A	✗✗*	N/A	N/A	N/A

What does this mean for users?

1. Users of any **Baidu, iFlytek,** or **Sogou** keyboards, including the versions that are bundled or pre-installed on operating systems, should **update their apps and operating systems ASAP**.
2. Users of **Honor**'s pre-installed keyboard or **QQ Pinyin** should **switch keyboards immediately**.
3. However, high-risk users should be aware that **enabling cloud-based features in these apps means that the vendor (and whoever they choose to share the information with) will still be able to read everything that users type on their devices.**

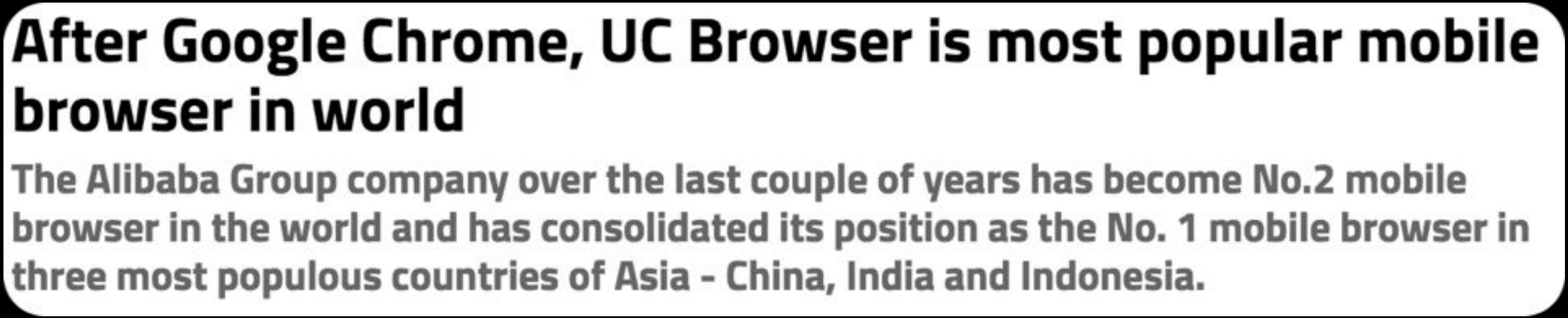
   These apps operate under Chinese jurisdiction and are subject to Chinese laws, which may be concerning to some users.

Privacy and Security Issues in BAT Web Browsers

What’s the most popular mobile web browser?

What’s the second most popular mobile web browser*?

(*In 2016 when we initially did this analysis.)



After Google Chrome, UC Browser is most popular mobile browser in world

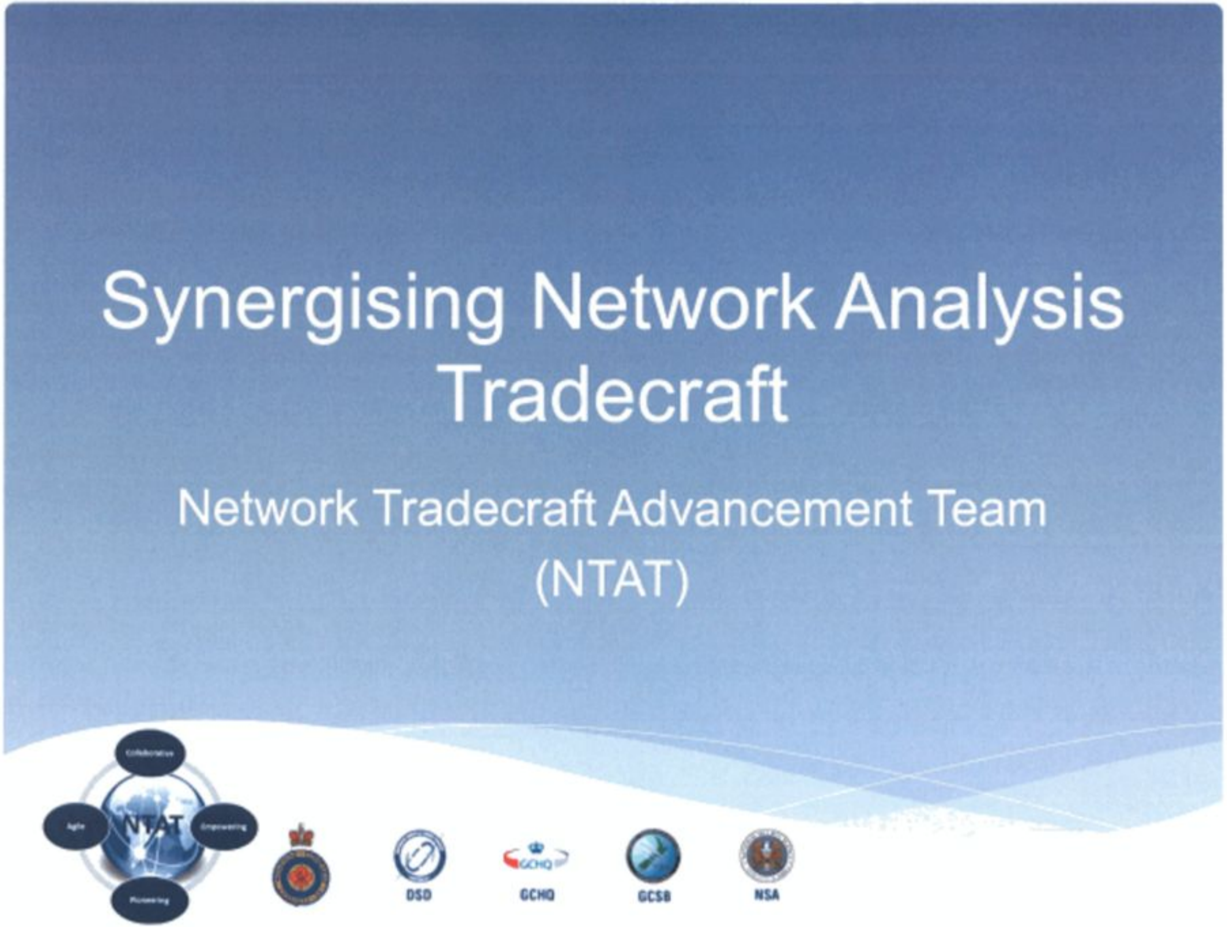
The Alibaba Group company over the last couple of years has become No.2 mobile browser in the world and has consolidated its position as the No. 1 mobile browser in three most populous countries of Asia - China, India and Indonesia.

BAT (Baidu Alibaba Tencent) Browsers

Baidu Browser
(百度浏览器)

UC Browser
(UC浏览器)

QQ Browser
(QQ浏览器)



Synergising Network Analysis Tradecraft

Network Tradecraft Advancement Team (NTAT)

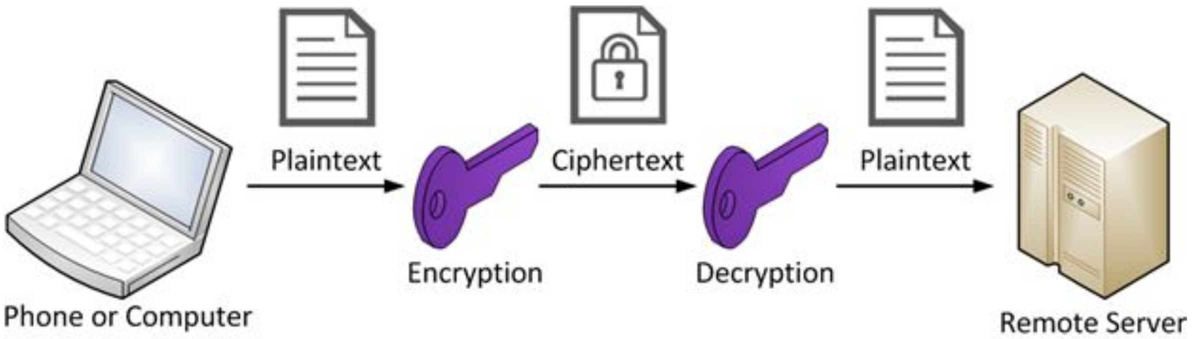
Collaborative
Agile
NTAT
Empowering
Pioneering
DSD
GCHQ
GCSB
NSA

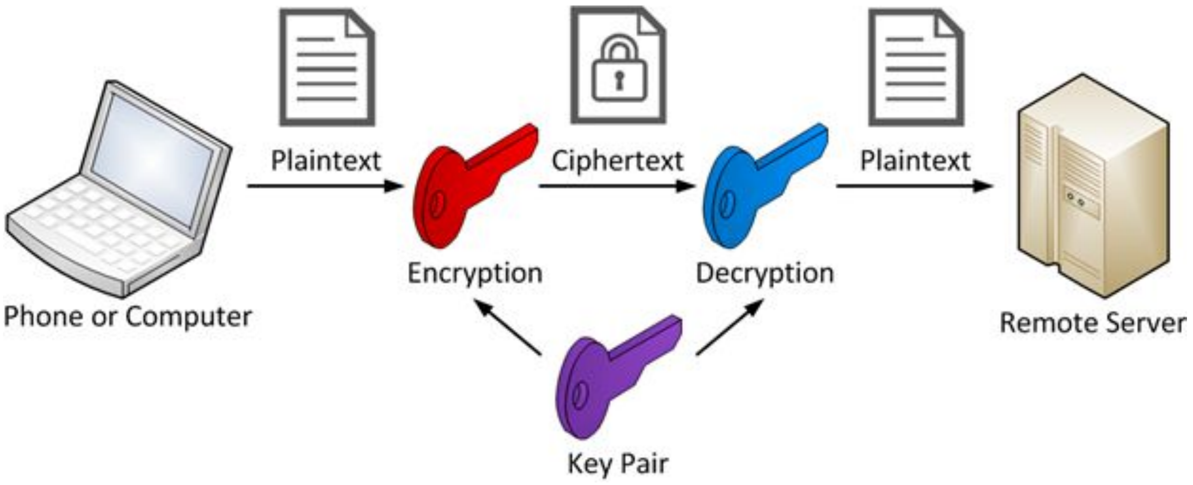
High level findings

- Reverse engineered Android & Windows versions
- Findings:
    - Found that each uses “easily decryptable” crypto (or sometimes no crypto) to transmit sensitive data
    - Found that most have insecure self-updating processes vulnerable to remote code execution

“Easily decryptable” crypto?

- Easily decryptable by anyone eavesdropping on the traffic who has reverse engineered the software
- *E.g.*, naive “home-rolled” crypto algorithms
- Symmetric crypto algorithms with hard-coded keys
- Asymmetric crypto with huge flaws


Phone or Computer
Plaintext
Encryption
Ciphertext
Decryption
Plaintext
Remote Server


Phone or Computer
Plaintext
Encryption
Ciphertext
Decryption
Plaintext
Remote Server
Key Pair

Kinds of sensitive data sent insecurely

- Personally identifiable
  - Hardware serial numbers
  - (Not so possible on mobile these days)
- Location
  - Geolocation coordinates
  - Active wireless access point
  - In-range wireless access points
- Activity
  - Search bar queries
  - URL of every page visited
  - Title of every page visited

Example transmission by UC Browser (encrypted)

```
m90.!.Ã#Ù.
GÚ}å.~%..7ÛÅC.\ ..§+xKû.,ý...%/@&..cq*.Í2äh:ÜÈ´Ü>ë..½.OL8."|.º±..¿Ü.ôýî. Ï¡°
_.Wß.p..dÄ·..¬à»®ðÕZìÁn..¶w.äb.!â.©Öà.&.J.Ë.ü7.5 w-.°,º.Ý$......0F.ß.#¶>.{$.
.CW[¿=.P.é.ôH.nþóTnM¸...ý.ËÙ+.îPÝû..u¦p.ãCËhìì!×¥ïæ 1Ï³¿.Þ@h.«Ww.X.u,¬W..å{.
H9ù·.Ä×#.S..@..!x.¢$w...¾;ýdt©Ì.öR.£(jY¦T|¸æsÐ~Ñö}.pOnJ$..M5E.ÃÅc.ÿãJç©.Ë©.¦
JzÄa/¥%jM.´Ê.ØÑ/r¾..çÃÁì|F-.G±:ºiíS¢¬òÏk8í$^6.p;.V¬é.YQ¡.ùÕ.ÿ+Í£..ÿ+V.##.5.Í
¯P.(ß¯h..O±ç¨.O>v2-äµ&r×À..dð.Ät;. ,©`×.Ñì..×.÷ªÕ¢å...O._Û¶.Át"ì´öZX.].ÑBùù.
Ìªf&cõ.ÓïW.ÒÙK.ßæ.°.W.ò.¿ñí3¯...è]G.Trg.¶»fKKb.ª.Ý.W B..B.oª.c#..ú..ÃÏ.Þ..¡µ
ê.+².2Å
```

How to decrypt this?

- Reverse engineer the software
- No asymmetric crypto :(
- Discover algorithm: home-rolled XOR
- Discover the key: "b59e216a8067d108"
- Write a python script to decrypt it

Example transmission by UC Browser (encrypted)

```
m90.!.Ã#Ù.
GÚ}å.~%..7ÛÅC.\ ..§+xKû.,ý...%/@&..cq*.Í2äh:ÜÈ´Ü>ë..½.OL8."|.º±..¿Ü.ôýî. Ï¡°
_.Wß.p..dÄ·..¬à»®ðÕZìÁn..¶w.äb.!â.©Öà.&.J.Ë.ü7.5 w-.°,º.Ý$......0F.ß.#¶>.{$.
.CW[¿=.P.é.ôH.nþóTnM¸...ý.ËÙ+.îPÝû..u¦p.ãCËhìì!×¥ïæ 1Ï³¿.Þ@h.«Ww.X.u,¬W..å{.
H9ù·.Ä×#.S..@..!x.¢$w...¾;ýdt©Ì.öR.£(jY¦T|¸æsÐ~Ñö}.pOnJ$..M5E.ÃÅc.ÿãJç©.Ë©.¦
JzÄa/¥%jM.´Ê.ØÑ/r¾..çÃÁì|F-.G±:ºiíS¢¬òÏk8í$^6.p;.V¬é.YQ¡.ùÕ.ÿ+Í£..ÿ+V.##.5.Í
¯P.(ß¯h..O±ç¨.O>v2-äµ&r×À..dð.Ät;. ,©`×.Ñì..×.÷ªÕ¢å...O._Û¶.Át"ì´öZX.].ÑBùù.
Ìªf&cõ.ÓïW.ÒÙK.ßæ.°.W.ò.¿ñí3¯...è]G.Trg.¶»fKKb.ª.Ý.W B..B.oª.c#..ú..ÃÏ.Þ..¡µ
ê.+².2Å
```

Decrypted

```
bluesky.1.5.1.1.10?cache=3102618000&ka=&kb=e2e63e260805aea910e1c2ce02b05211&
kc=3b5d366db90b1b60e22260a0278331f8v0000002e9952d46&firstpid=0501&bid=800&ve
r=5.5.10106.5&defalutbrowser=UCHTML.AssocFile.HTML&flashver=&hi=Intel(R) Cor
e(TM) i5-4300U CPU @ 1.90GHz&0&VB3bb90c33-fc547c89&searchaddress=google&sear
chbar=google&searchquick=google&openurltab=0&showsearch=1&showextension=1&ap
plyall=0&cloudspeed=0&autopage=0&autologin=0&theme_id=569&wallpaper_id=207&a
utoclearhistory=0&service=1&sis_fool=5.1.2600_SP3_x86&tch=0&ad_switch=10&lan
g=zh-CN
```

Example 2 encrypted

```
m90...._Õ.
÷.y.]¢=»ù¤Ìü<.Oò+DÛxh..Æj.¤]ß?;..u.Öá..7Ò.p`üPÐ·.O"c.ïoÔ¸$ Ä.Úm.¯.ø.¤Ñ.$"gÉ^
¿<kp8äL½.XgEÇ\0ïn...Ü5.F|¢?í.ª3..Ím5°.êó....ü÷Ö% 7a.`(þ/mXa¥nÁS...Õø.·.Ý.÷tÈ
Ø3'gÿ.j...ß±È.À0Bxä.Ù.8´î½û]üI3Ñe.O³¿G.Ö|.+½.ñpJÊÑ.+V.huÚ.È[~Ø.SG¨¶ÐLp`Ñ!.Þf
^4eåá.ç1s.ÈfdÐ>Öz÷v\6K.ÁÐ¥9.ýÈ~^...¥Í5.þ.st·U.Ó´®.dÄE[ñFÀ.ÎF²L..ýêth=.zãé¬;ë
=\nL..ØÔÕ¼..[+ÊÔÌ.¯Þ!!'alrÖ.0..qJ®\9Uë..¶Y.ýk·2Ñg¬DÚ5Á.ó%<qE.u.`ÿ.®â.2o.Ú½.÷
¤.Ô.]uùz.ø.ç.Å..Üú`ã (WäÓ.Ç.yà#:¶+ÝA9.µ3.:1!öf¬.XE.£.ð÷¬1ð.ÐCT.5/¿*ØHø~©P.ÉJ
 .L©Gq..`..OO9:.'ùïHÊG..úLÇ..Ï.¡.×öJ¶¤,ao+/.©.ËZ.Ø..ÚN....|.Ê8.æ.p.9¯F.ð`.Öô
áÆ©.ëXü.1©>W.§.X2Å.c..r¸{.Í°^.+î.y{.çáÀ..N®Ü,_ùR%.Æ%uµÍÉc£.7ù&.n..íH×Ë <¯P.Ö
ZðuÑ¥1.».mu.È. 7æÌ¶,Ý .Tj&×ýó£&.;´ä.á.ý÷÷...B..³.u[...).rïw¸;.èQ)W.e]Ü.:ÑôúU
.õ$óm-ûÔ}¡õÓ..@^b\..îâ%!Élq,ÅQPô..í sÓ..±....9ïNÉ¢mÆÍÍßéÁ.ý±r.÷§ö..q$.).§y5B
î.Q.Xôù.Ì^nÊKÒ.ðM·."t» «.ZÀ3mAØ¶Õ
```

Example 2 decrypted

```
bluesky.1.25.1.1.7?cache=3766412000&ka=&kb=e2e63e260805aea910e1c2ce02b05211&
kc=3b5d366db90b1b60e22260a0278331f8v0000002e9952d46&firstpid=0501&bid=800&ve
r=5.5.10106.5&type=1&ssl=1&bandwidth=29.63&target_ip=64.106.20.27&redirect_s
tart=0&redirect_duration=0&dns_start=0&dns_duration=218&connect_start=218&co
nnect_duration=251&request_start=469&request_duration=916&response_start=138
5&response_duration=1&dom_start=1386&dom_duration=268&dom_interactive=234&do
m_content_load_start=1420&dom_content_load_duration=0&load_event_start=1654&
load_event_duration=26&t0=1385&t1=1719&t2=1719&t3=1420&total_requests=2&requ
ests_via_network=2&cloud_acceleration_enabled=0&average_of_request_duration=
809&average_of_t2_duration=859&private_data=host=www.cs.unm.edu|url=https://
www.cs.unm.edu/~jeffk/&lang=zh-CN
```

Baidu Browser

- RC4 key "HR2ER"
- AES key "h9YLQoINGWyOBYYk"
- XOR mask (0x2D382324), bit rotations
- Base64 encoding with nonstandard alphabet:

```
qogjOuCRNkfil5p4SQ3LAmxGKZTdesvB6z_YPahMI9t80rJyHW1DEwFbc7nUVX2-
```

- Modified TEA crypto + non-standard block cipher mode, key “vb%,J^d@2B1l'Abn”
- …

UC Browser

- Home-rolled XOR-based algorithm with various keys ("b59e216a8067d108", "e19237a3a933f7eb", "aa171021f9438cb2")
- XOR mask "\xee\xb9\xe9\xb3\x81\x8e\x97\xa7"
- AES "key autonavi_amaploc"

QQ Browser

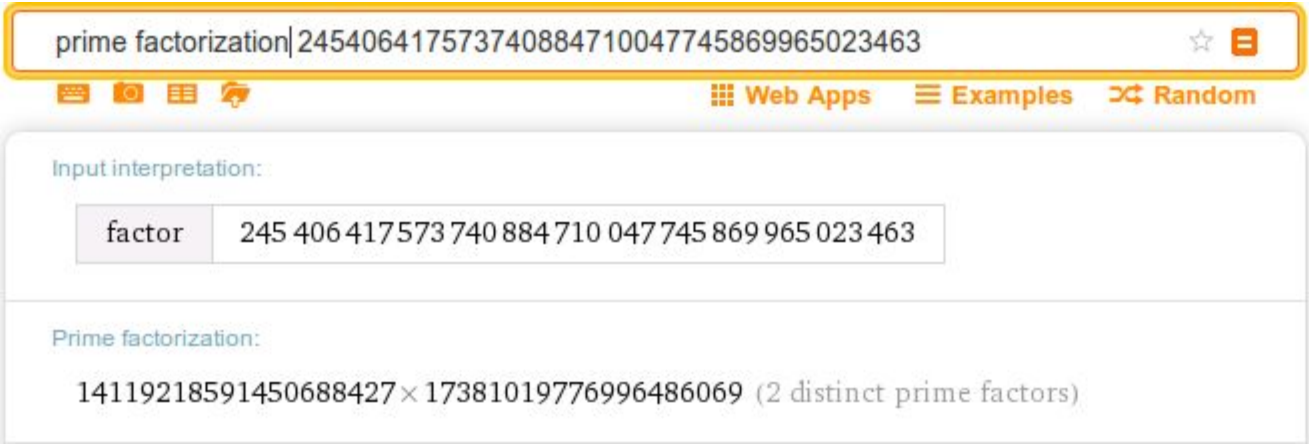
- RSA public key 245406417573740884710047745869965023463
- Remember: if we can factor this number, we can derive the private key

QQ Browser

- To factor it, we built our own quantum computer

QQ Browser

- RSA public key 245406417573740884710047745869965023463



prime factorization 245406417573740884710047745869965023463

Input interpretation:

factor	245 406 417 573 740 884 710 047 745 869 965 023 463

Prime factorization:

14119218591450688427 × 17381019776996486069 (2 distinct prime factors)

QQ Browser

Symmetric keys are generated poorly:

```
int i = 10000000 + new Random().nextInt(89999999);
int j = 10000000 + new Random().nextInt(89999999);
return (String.valueOf(i) + String.valueOf(j)).getBytes();
```

Entropy reduced from 2^{128} to $89999999^{2} < 2^{53}$.

QQ Browser

Symmetric keys are generated poorly:

```
Random random = new Random(System.currentTimeMillis());
byte[] bArr = new byte[8];
byte[] bArr2 = new byte[8];
random.nextBytes(bArr);
random.nextBytes(bArr2);
return new SecretKeySpec(ByteUtils.mergeByteData(bArr, bArr2), "AES");
```

If you know the time, you know the key…

Let C be the RSA encryption of 128-bit AES key k with RSA public key (n, e). Thus, we have

$$C \equiv k^e \pmod{n}$$

Now let C_b be the RSA encryption of the AES key

$$k_b = 2^b k$$

i.e., k bitshifted to the left by b bits. Thus, we have

$$C_b \equiv {k_b}^e \pmod{n}$$

We can compute C_b from only C and the public key, as

$$\begin{aligned} C_b &\equiv C(2^{be} \bmod n) \pmod{n} \\ &\equiv (k^e \bmod n)(2^{be} \bmod n) \pmod{n} \\ &\equiv k^e 2^{be} \pmod{n} \\ &\equiv (2^b k)^e \pmod{n} \\ &\equiv {k_b}^e \pmod{n} \end{aligned}$$

Vulnerable SDK

- Code leaking personally identifying and locational data in browser actually from a Baidu SDK
- Found SDK in hundreds of Google Play store apps (some very popular)
- ES File Explorer File Manager (com.estrongs.android.pop) has 100,000,000 – 500,000,000 installs

Vulnerabilities in update processes

- Remote code execution
- Vulnerabilities
  - Failing to check digital signatures (or protected with easily decryptable crypto)
    - Baidu Android, Baidu Windows, QQ Android, UC Windows
  - Failing to check version numbers → downgrade to vulnerable version
    - QQ Windows
  - Failing to check app name → sidegrade to vulnerable product
    - QQ Windows, UC Android

Disclosure

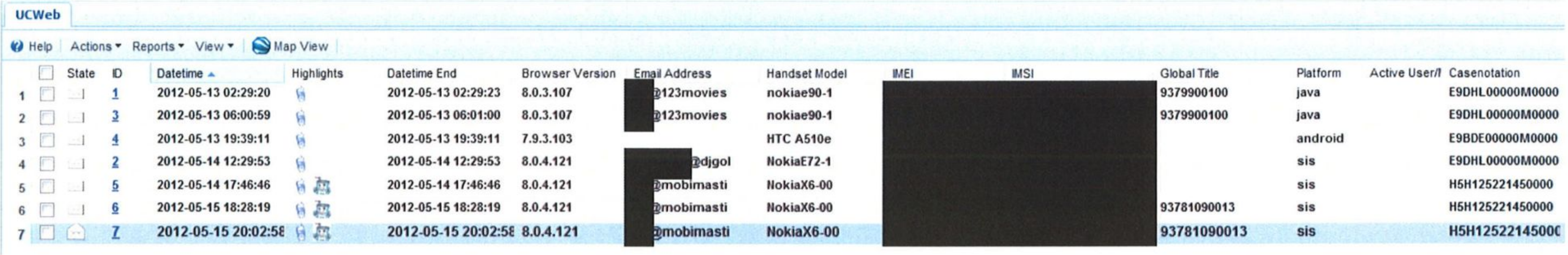
- We reported all of the vulnerabilities
- Most were fixed
- New ones have since been introduced

Success Stories

- UCWeb mobile browser identification
  - Discovered by GCHQ analyst during DSD workshop

  - Chinese mobile web browser – leaks IMSI, MSISDN, IMEI and device characteristics

UCWeb – XKS Microplugin



	□	State	ID	Datetime ▲	Highlights	Datetime End	Browser Version	Email Address	Handset Model	IMEI	IMSI	Global Title	Platform	Active User/I	Casenotation
1	□		1	2012-05-13 02:29:20		2012-05-13 02:29:23	8.0.3.107	[redacted]@123movies	nokiae90-1	[redacted]	[redacted]	9379900100	java		E9DHL00000M0000
2	□		3	2012-05-13 06:00:59		2012-05-13 06:01:00	8.0.3.107	[redacted]@123movies	nokiae90-1	[redacted]	[redacted]	9379900100	java		E9DHL00000M0000
3	□		4	2012-05-13 19:39:11		2012-05-13 19:39:11	7.9.3.103		HTC A510e	[redacted]	[redacted]		android		E9BDE00000M0000
4	□		2	2012-05-14 12:29:53		2012-05-14 12:29:53	8.0.4.121	[redacted]@djgol	NokiaE72-1	[redacted]	[redacted]		sis		E9DHL00000M0000
5	□		5	2012-05-14 17:46:46		2012-05-14 17:46:46	8.0.4.121	[redacted]@mobimasti	NokiaX6-00	[redacted]	[redacted]		sis		H5H125221450000
6	□		6	2012-05-15 18:28:19		2012-05-15 18:28:19	8.0.4.121	[redacted]@mobimasti	NokiaX6-00	[redacted]	[redacted]	93781090013	sis		H5H125221450000
7	□		7	2012-05-15 20:02:58		2012-05-15 20:02:58	8.0.4.121	[redacted]@mobimasti	NokiaX6-00	[redacted]	[redacted]	93781090013	sis		H5H12522145000

UCWeb

- Led to discovery of active comms channel from [redacted]

*(S//SI//REL TO USA, FVEY) The CONVERGENCE team helped discover an active communication channel originating from [redacted] that is associated with the [redacted] as they are known within the [redacted] hierarchy area of responsibility is for covert activities in Europe, North America, and South America. The customer [redacted] leveraged a **Convergence Discovery capability that enabled the discovery of a covert channel associated with smart phone browser activity in passive collection**. The covert channel originates from users who use UCBrowser (mobile phone compact web browser). **The covert channel leaks the IMSI, MSISDN, Device Characteristics, and IMEI back to server(s) in** [redacted] Initial investigation has determined that perhaps malware can be associated when the covert channel is established. [redacted] covert exfil activity identifies SIGINT opportunity where potentially none may have existed before. Target offices that have access to X-KEYSCORE can search within this type of traffic based on their IMSI or IMEI to determine target presence*

How did this happen?

Deliberate backdoors? No.

- Not a good backdoor
- CNCERT/CC is trying to improve domestic encryption
- Can get warrants for data

How did this happen?

Market Factors

- Highly competitive market
- Tight deadlines
- “Collect it all”

How did this happen?

Political factors

- Lack of access to Google Play
- Skepticism of “western” cryptography (e.g., Dual_EC_DRBG)

Takeaways

- We must pay more attention to apps from understudied ecosystems

Takeaways

- We must pay more attention to apps from understudied ecosystems
- Cost-benefit analysis: Huge user bases + major vulnerabilities + that are easy to find == high impact

Takeaways

- We must pay more attention to apps from understudied ecosystems
- Cost-benefit analysis: Huge user bases + major vulnerabilities + that are easy to find == high impact
- Finding vulnerabilities in popular North American browsers is becoming increasingly difficult

Takeaways

- Any researcher that even looked at this traffic in Wireshark would know there is a problem

Takeaways

- Any researcher that even looked at this traffic in Wireshark would know there is a problem
- If you know how to use Wireshark, you can get going analyzing understudied software

Takeaways

- Any researcher that even looked at this traffic in Wireshark would know there is a problem
- If you know how to use Wireshark, you can get going analyzing understudied software
- Please join us in studying software in understudied ecosystems