Demystifying the Apix Security Pattern Lock

Crystal R. Glasco

Introduction

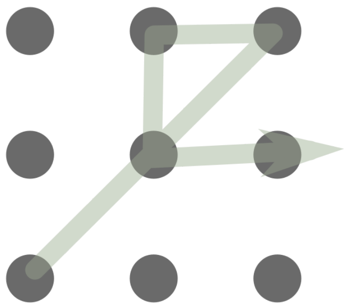
In this paper I will demonstrate how to defeat the Apix Security Pattern Lock (ASPL) that is included in the Apix Security Suite. I hope that by the end of this paper, you will be well-informed enough to never trust Apix Security for anything ever again.

How ASPL works

Like most popular pattern lock implementations, ASPL uses a 3x3 grid of points that the user can use to create a pattern. Each of the points on the grid can be identified by a number:

```
0 1 2
3 4 5
6 7 8
```

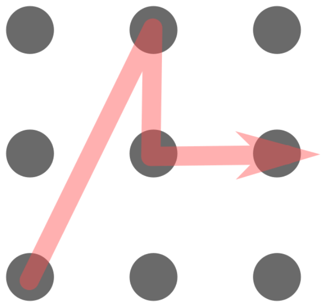
If, then, we have the following lock pattern:



We can write it numerically as `642145`.

Notice that, unlike most major pattern lock implementations, the swipe pattern is able to pass through a point multiple times. (The middle point is registered twice.) This is where Apix tries to separate itself from the competition: by allowing points to be registered multiple times, the number of possible combinations increases dramatically (up from the 389,112 combinations possible on competing lock implementations).

While the increase in combinations is no doubt an advantage, it is somewhat negated by the fact that each point in a pattern must be adjacent (or directly diagonal) to the previous point. For example,

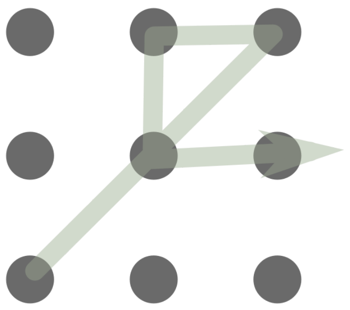


is not a valid combination since P1 is not connected to P6. This greatly reduces the number of combinations available, and if one or more points in the pattern are known, an attacker can eliminate a large number of invalid paths in a brute force approach. Even with this limitation, the large number of combinations possible with ASPL dramatically increases the amount of time required for a successful brute force attack. It can be (and has been) argued whether Apix Security's approach to increasing pattern combinations is actually beneficial.

Cracking the code

For security reasons, ASPL does not store the pattern data on the system in plaintext. Instead, when a user chooses a pattern, a "puzzle" is generated and saved on the device. Whenever the lock screen is displayed and a pattern supplied, the pattern is used as solution to the puzzle. Only valid solutions will unlock the device. This can be a sound approach--many asymmetrical cryptography solutions (RSA, for example) work this way. But in order to be successful, the puzzle must (1) have enough mathematical complexity to prevent reverse engineering or brute force, and (2) have a unique solution. As we will soon discover, ASPL has neither.

Let us use the example pattern from earlier:



After a user chooses a pattern, a configuration file `/sys/data/aspl_hsc.key` is stored on the device. When using the pattern above, the contents of `aspl_hsc.key` will be `011021100` every time. A different pattern will produce a different set of numbers. Armed with the knowledge that the contents of this file are deterministically created based on the pattern, the only challenge for the attacker is to figure out the pattern from code.

So what does the code mean? The astute reader will realize that the code contains nine digits, one for each of the possible points. If we break the given code into three groups, we have:

```
0 1 1
0 2 1
1 0 0
```

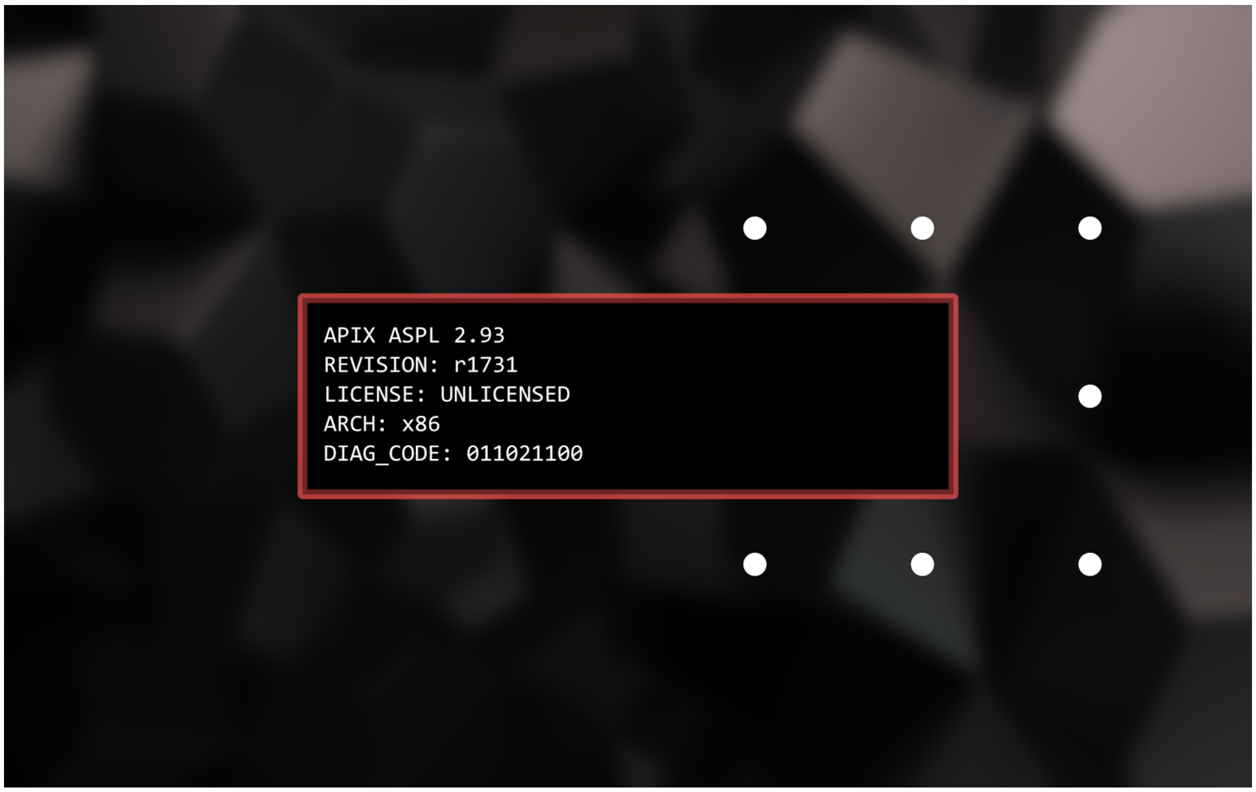
The numbers, then, represent the number of times a valid pattern passes through each point. Yes, it really is that simple. And while it may not be as straightforward of a pattern as the one above, if the code is known, it won't be hard for an attacker to determine a pattern that satisfies the code's conditions.

Notice, too, that there are multiple solutions for the code above. While our chosen pattern was 642145, other valid solutions are 645214, 452146, 541246, and several more. An attacker doesn't even have to know the original pattern: any of them will unlock the device!

It gets worse

As ridiculously simple as ASPL's "encryption" algorithm is, the approach is not too distant from approaches used in other popular pattern lock implementations. (Although even an unsalted hash is more secure than Apix's naive approach, since it's a one-way function.) And though it's incredibly easy to crack the ASPL if you know the code, on most devices, there is no direct access to `/sys/data/aspl_hsc.key`. If you do have access to that file, your device is probably rooted, and the lock screen probably isn't stopping you from getting what you want anyway.

But here's where Apix made their colossal screw-up. On the lock screen, if you simultaneously hold the four dots in each corner, a debug dialog will display. (This discovery is what started my investigation into the system: I noticed that the DIAG_CODE would only change when I changed my pattern.)



```
APIX ASPL 2.93
REVISION: r1731
LICENSE: UNLICENSED
ARCH: x86
DIAG_CODE: 011021100
```

Conclusion

I have no idea why Apix thought it would be a good idea to include that code in this dialog. Maybe they forgot to remove it during testing. Maybe they thought nobody would ever figure out how to access the dialog or what the code meant. Or maybe it was an intentional back door. Whatever the reason, Apix Security is either an incompetent company or a crooked one. They don’t deserve your money, and they sure as hell don’t deserve your trust.