Poster: Side-Channel Finder for AVR

Florian Dewald, Heiko Mantel, Alexandra Weber
Technische Universität Darmstadt
{dewald, mantel, weber}@mais.informatik.tu-darmstadt.de

I. Introduction

AVR microcontrollers [1] are widely used, e.g., in Internet of Things devices. Such devices are targets for attacks (e.g., the recent side-channel attack on smart lightbulbs [2]).

In a side-channel attack, an attacker observes execution characteristics, e.g., time, from which he deduces information about a secret. Timing-side-channel attacks are particularly dangerous because they can be mounted remotely [3].

On this poster, we present our results [4] and ongoing work on timing-side-channel detection for AVR microcontrollers.

II. Approach

We say that a program is secure against timing side channels if it satisfies timing-sensitive noninterference (TSNI). That is, two runs of the program that start in attacker-indistinguishable states must take the same amount of clock cycles and the final states must again be attacker-indistinguishable.

To check whether a program satisfies timing-sensitive noninterference, we developed a security type system. We assign security types (secret or public) to registers, memory, and stack. The type system defines which combinations of type-assignments are allowed. We have proven that the type system is sound, i.e., any typeable program satisfies TSNI.

We allow secret-dependent branching if both branches take the same amount of clock cycles. Moreover, we do not abstract from actual running times on the target platform. Our soundness proof is based on a formal semantics of AVR assembly, which faithfully captures the exact running times of instructions on the target platform as specified by the vendor [5].

III. Side-Channel Finder

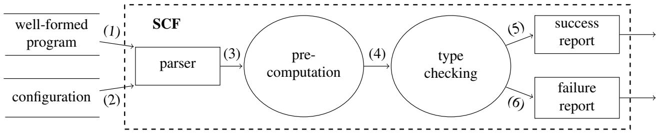
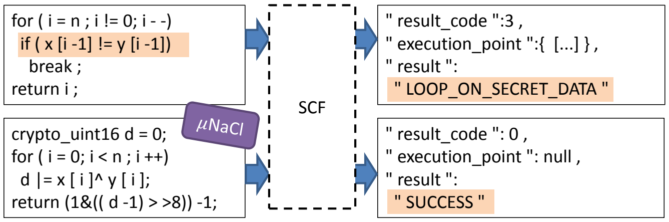


Fig. 1. Side-Channel Finder

We automated our approach in the tool Side-Channel Finder (SCF). SCF (Figure 1) takes an AVR program and a configuration specifying the secrets. SCF parses the inputs, precomputes control dependences, and applies our type system.

On typeable programs, SCF reports success, i.e., that the program is free of timing-side-channel vulnerabilities. Otherwise, SCF points to the location of the detected potential timing-side-channel vulnerability.



```
for ( i = n ; i != 0; i - -)
 if ( x [i -1] != y [i -1])
  break ;
return i ;
```

```
" result_code ":3 ,
" execution_point ":{ [...] } ,
" result ":
 " LOOP_ON_SECRET_DATA "
```

```
crypto_uint16 d = 0;
for ( i = 0; i < n ; i ++)
 d |= x [ i ]^ y [ i ];
return (1&(( d -1) > >8)) -1;
```

```
" result_code ": 0 ,
" execution_point ": null ,
" result ":
 " SUCCESS "
```

Fig. 2. Example Case Study

We have evaluated SCF on multiple case studies. We detected a leak in a simple implementation of string comparison and verified the security of the implementation of string comparison from the μNaCl [6] library (Figure 2). Moreover, we were able to verify the security of the μNaCl implementations of Salsa20, xSalsa20, and Poly1305.

Our ongoing work focuses on extending the scope of Side-Channel Finder to the full 8-bit AVR instruction set.

Acknowledgment

This work has been funded by the DFG as part of Project E3 within the CRC 1119 CROSSING.

References

[1] Microchip Technology Inc., "Microchip AVR® MCUs," 2018, https://www.microchip.com/design-centers/8-bit/avr-mcus, Accessed 28.06.18.

[2] E. Ronen, C. O'Flynn, A. Shamir, and A.-O. Weingarten, "IoT Goes Nuclear: Creating a ZigBee Chain Reaction," in *S&P*, 2017, pp. 195–212.

[3] B. B. Brumley and N. Tuveri, "Remote Timing Attacks Are Still Practical," in *ESORICS*, 2011, pp. 355–371.

[4] F. Dewald, H. Mantel, and A. Weber, "AVR Processors as a Platform for Language-Based Security," in *ESORICS*, 2017, pp. 427–445.

[5] Atmel Corporation, "Atmel AVR 8-bit Instruction Set: Instruction Set Manual," Rev. 0856KAVR05/2016, 2016.

[6] M. Hutter and P. Schwabe, "NaCl on 8-bit AVR Microcontrollers," in *AFRICACRYPT*, 2013, pp. 156–172.