HD-Sec: Holistic Design of Secure Systems on Capability Hardware

https://hd-sec.github.io

DSbD
Digital Security by Design

University of Southampton

Engineering and Physical Sciences Research Council

Colin Snook, Asieh Salehi, Thai Son Hoang, Robert Thorburn, Michael Butler, Leonardo Aniello, Vladimiro Sassone
University of Southampton, UK {cfs, a.salehi-fathabadi, , t.s.hoang, robert.thorburn , m.j.butler, l.aniello, vsassone}@soton.ac.uk

Project Overview:

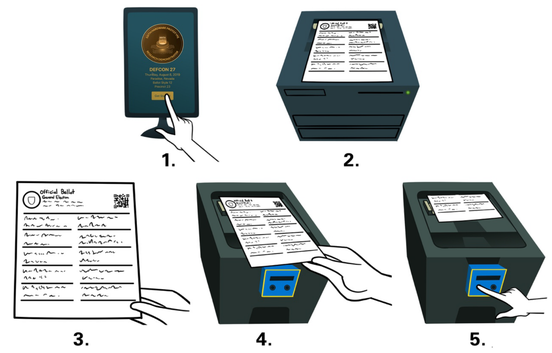
Transformation of security-critical software development

- From an expensive iterative test-and-fix approach
  - To a correctness-by-construction (CxC) approach

- The design of software from requirements to implementation
  - Formal modelling, Reusable formal abstractions
  - Verification
  - Model transformation
  - CxC tools and running on capability hardware

Case Study: Smart Ballot Box[1] (SBB)

- **Availability:** the voter should not be prevented from casting a ballot.
- **Confidentiality :** the voter's choices should be secret
- **Integrity:** the system should only accept valid ballots and reject invalid ballots.



Availability

- Availability is critical in many applications
  - Danger of facilitating DoS attacks

- Morello detects potential attacks
  - Does not provide recovery support
  - Some support for availability, e.g. CompartOS

Exception Handling/Recovery maintains availability

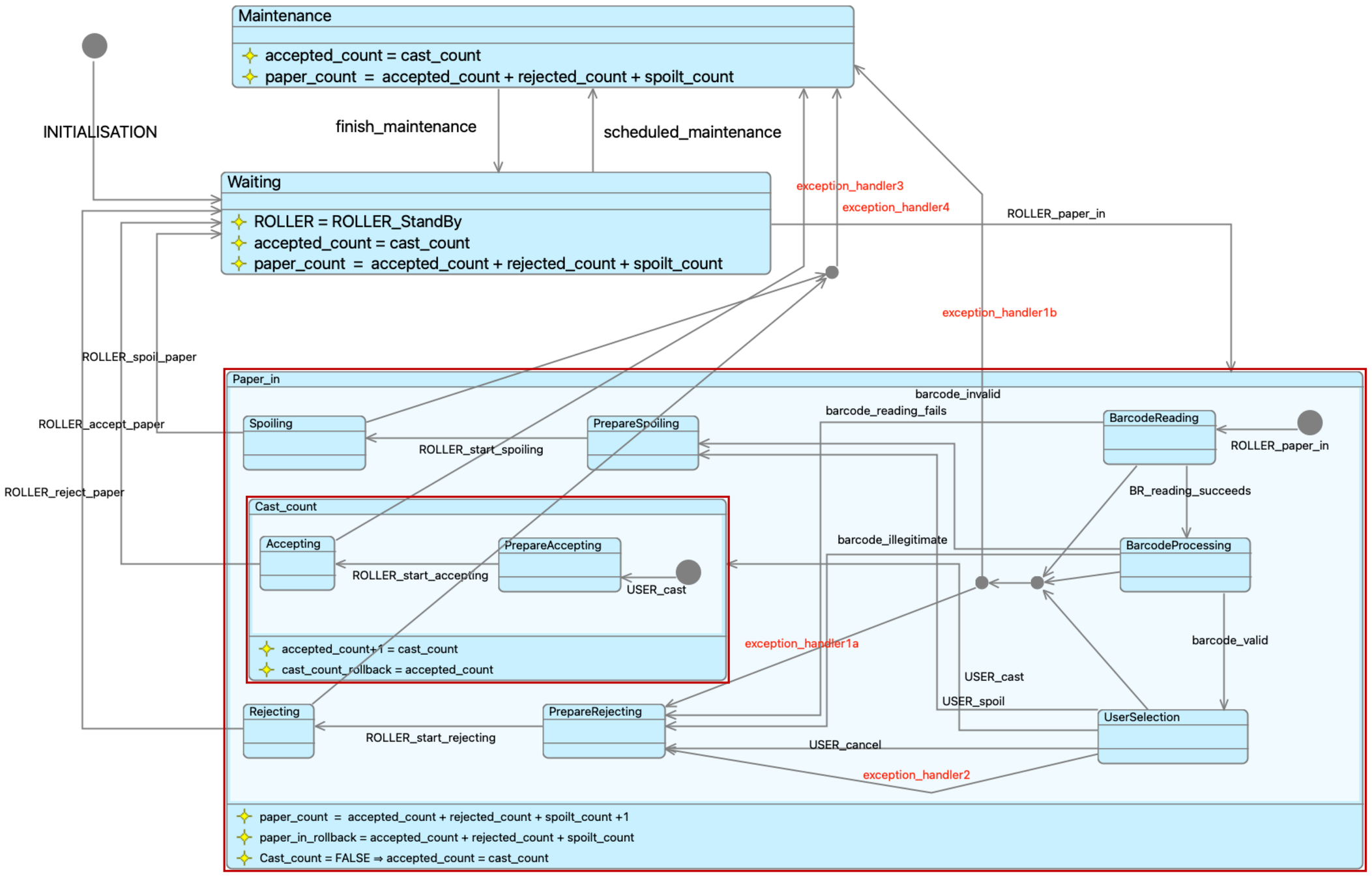
- Exception recovery allows system to continue
  - But is application dependent

- Must Preserve important functional properties
  - requires analysis at an abstract application level

Analysis using formal methods (Event-B/UML-B)

- Model normal behaviour
  - Verify (prove) important functional properties

- Add Exceptional behaviour to model
  - Verify (prove) that exception handling preserves functional properties



Translating formal model into C implementation for Morello

- State-machine translated to C code as the application
- For, illustration, C code also simulates:
  - the environment (e.g. roller, user, barcode library)
  - a capability violation during barcode reading/processing.
- Default behaviour on SIGPROT is to kill the process:
  - "In-address space security exception (core dumped)"
- With our exception handler, application recovers:
  - rejects ballot paper for first 2 attacks,
  - then goes to maintenance for third attack.

```
.
.
Barcode simulation .. Barcode = 67
Processing Barcode... ATTACK: reading from address 67

>>SBB Handler - sigtype: 34 while in state .. SBB state =  Paper_in -  BarcodeProcessing
..    si_signo =  34
..    si_errno =  0
..    si_code =  2
..    si_pid =  0
..    si_uid =  0
..    si_status =  0
>>> attack counter < attack limit
>>> inc attack counter to 1
>>> recover to: .. SBB state =  Paper_in -  PrepareRejecting
>>> aborting the step that caused the exception
.. SBB state =  Paper_in -  PrepareRejecting
.. ROLLER PaperIn
.
.
```

```
void SBB_setup_exception_handling(){
    //allow interrupts other than the ones we handle
    sigemptyset(&SBB_sa.sa_mask);
    sigaddset(&SBB_sa.sa_mask, SIGPROT);
    sigaddset(&SBB_sa.sa_mask, SIGALRM);
    sigaddset(&SBB_sa.sa_mask, SIGSEGV);
    //assign the exception handler routine
    SBB_sa.sa_handler = (void *)SBB_Handler;
    //assign handler to SIGALRM and SIGPROT and SIGSEGV
    sigaction(SIGALRM, &SBB_sa, &SBB_SIGALRM_oldsa);
    sigaction(SIGPROT, &SBB_sa, &SBB_SIGPROT_oldsa);
    sigaction(SIGSEGV, &SBB_sa, &SBB_SIGSEGV_oldsa);
}
```

```
if (sigsetjmp(SBB_abort_step,true) ==0) {   // TRY

    if (sbb_control.state==SBB_Null){
        printf("Something went wrong! SBB_state is Null\n");
    }else{
        alarm(getAlarm());      //set the timeout for the current state
        changedState = false;
        printCurrentState();
        do {
            ROLLER_step();                  //progress the roller simulation
            changedState = SBB_checkState(); //see if we can change the state
        } while (!changedState);            //repeat until a state change occurs
    }

}   //SBB_abort_step - exit point for handled exceptions
```

References

[1] Galois and Free & Fair. The BESSPIN Voting System (2019).
[2] Salehi Fathabadi, Asieh, Snook, Colin, Hoang, Thai Son, Thorburn, Robert, Butler, Michael, Aniello, Leonardo and Sassone, Vladimiro (2024)Designing Exception Handling using Event-B. In ABZ 2024 - 10th International Conference on Rigorous State Based Methods.
[3] Salehi Fathabadi, Asieh, Dghaym, Dana, Hoang, Thai Son, Butler, Michael and Snook, Colin (2022). Generating SPARK from Event-B, Providing Fundamental Safety and Security In DETECT.
[4] Thorburn, Robert, Sassone, Vladimiro, Salehi Fathabadi, Asieh, Aniello, Leonardo, Et al. (2022). A Lightweight Approach to the Concurrent Use and Integration of SysML and Formal Methods in Systems Design. In MODELS '22 .
[5] Hoang, Thai Son, Snook, Colin, Dghaym, Dana, Salehi Fathabadi, Asieh and Butler, Michael (2022). Building an Extensible Textual Framework for the Rodin Platform. In F-IDE – SEFM.
[6] D. Dghaym, T.S. Hoang, M. Butler, R. Hu, L. Aniello, V. Sassone (2021) Verifying System-level Security of a Smart Ballot Box. In ABZ 2021- 8th International Conference on rigorous State Based Methods.