

Role of Domain-Specific Techniques in Designed-in Security

John Launchbury

Galois, Inc

Designed-in Security



- ❖ Builds the capability to design, develop, and evolve high-assurance, software-intensive systems predictably and reliably while effectively managing risk, cost, schedule, quality, and complexity.
- ❖ Promotes tools and environments that enable the simultaneous development of cyber-secure systems and the associated assurance evidence necessary to prove the system's resistance to vulnerabilities, flaws, and attacks.
- ❖ Secure, best practices are built inside the system. Consequently, it becomes possible to evolve software-intensive systems more rapidly in response to changing requirements and environments.

Designed-in Security



- ❖ Builds the capability to **design, develop, and evolve** high-assurance, software-intensive systems predictably and reliably while effectively managing risk, cost, schedule, quality, and complexity.
- ❖ Promotes tools and environments that enable the simultaneous development of cyber-secure systems and the associated **assurance evidence** necessary to prove the system's resistance to vulnerabilities, flaws, and attacks.
- ❖ Secure, best practices are built inside the system. Consequently, it becomes possible to **evolve software-intensive** systems more rapidly in response to changing requirements and environments.

System Unreliabilities even in Critical Systems

- ❖ Malaysia Airlines Flight 124 (Boeing 777)
 - ❖ “Software anomaly”
- ❖ Qantas Airlines Flight 72 (Airbus A330)
 - ❖ Transient fault in the inertial reference unit
- ❖ Space Shuttle STS-124 aborted launch
 - ❖ Assumptions about distributed fault-tolerance
- ❖ Air France Flight 447, Airbus A330
 - ❖ Frozen pitot tubes, pilot error
 - ❖ 228 killed



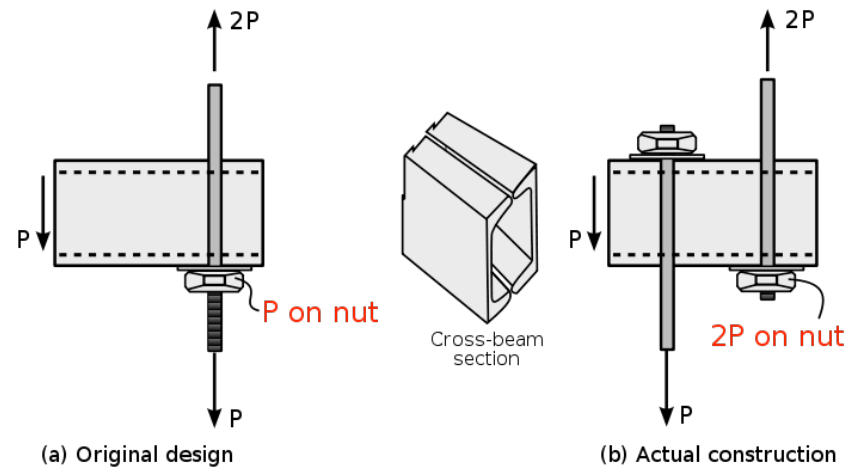
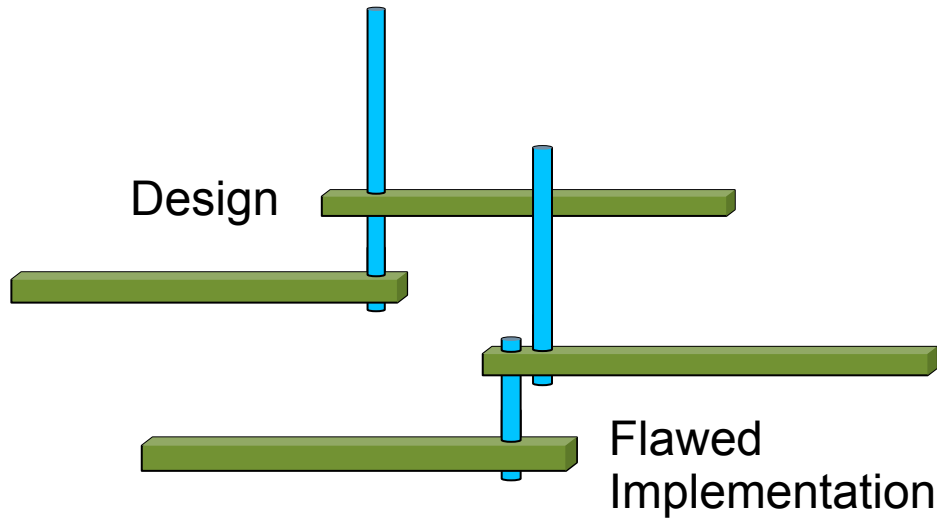
Hyatt Regency Walkway Collapse



1981, Kansas City

114 deaths, 213 injuries

Implementation didn't quite match the engineering specification



Functionality vs Security

❖ Functionality

- ❖ Enable things to happen
- ❖ Be as fast as possible
- ❖ Cost effective

❖ Security

- ❖ Prevent things from happening
- ❖ Be as thorough as possible
- ❖ Cost effective



Domain-Specific Languages

Application concepts:
temperature, velocity,
orientation, personality,
algebra, timing, visual
impact, calculation



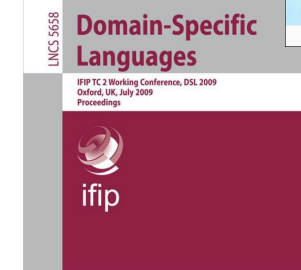
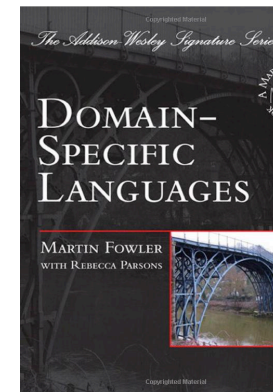
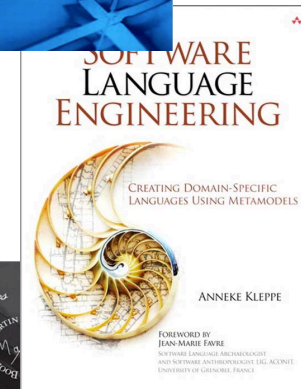
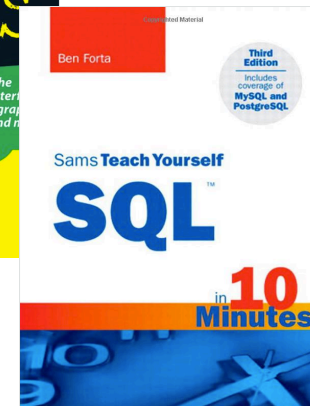
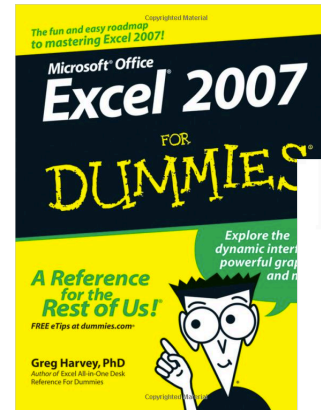
**Constrained
implementations**

Programming concepts:
procedure, assignment,
object, instruction,
memory, sequence,
concurrency, cache

**Unconstrained
implementations**

Value of DSLs

- ❖ Design-level programming
- ❖ 10x productivity increase over hand coding
- ❖ Broadening the programmer base
- ❖ Major flexibility in evolvability
 - ❖ Update the design; regenerate the code
- ❖ Natural maintenance of design documents
- ❖ Multiple interpretations
 - ❖ One specification, many uses

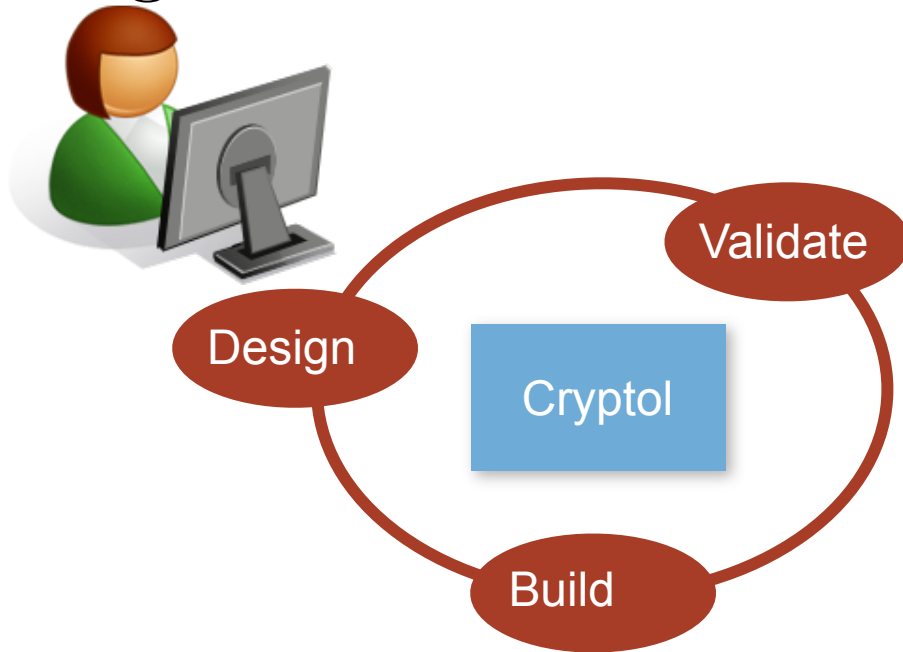


One Specification – Many Uses

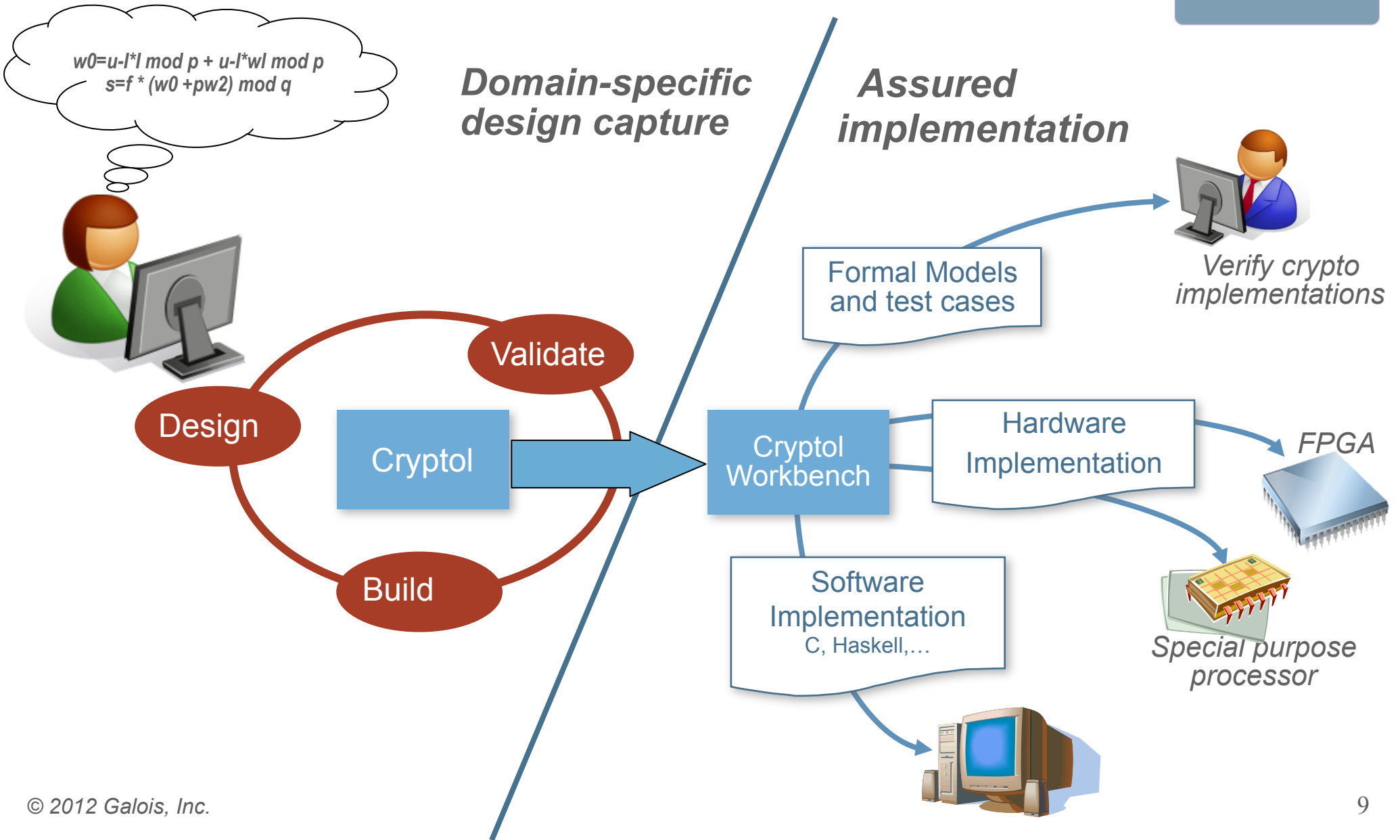


$$w_0 = u^{-1} \cdot l \pmod p + u^{-1} \cdot w_1 \pmod p$$
$$s = f * (w_0 + p \cdot w_2) \pmod q$$

*Domain-specific
design capture*



One Specification – Many Uses



Generating C



❖ Simple Example

```
coef : [8] -> [8];  
coef x = coeffs @ x  
  where coeffs = [0x3d 0xa7 0x3e 0x81];
```

```
coef2 : ([8], [8]) -> [8];  
coef2 (x, y) = coef x * coef y;
```

Ge

❖ {

```
inline Word32 cryptol_RotateLeft32(Word32 val, Word32 size, unsigned long mask, Word32 rotAmt)
{
    Word32 shiftAmt = (rotAmt < size) ? rotAmt : (rotAmt % size);
    Word32 result = (val << shiftAmt) | (val >> (size - shiftAmt));
    return (result & mask);
}
```

```
inline Word32 cryptol_RotateRight32(Word32 val, Word32 size, unsigned long mask, Word32 rotAmt)
{
    Word32 shiftAmt = (rotAmt < size) ? rotAmt : (rotAmt % size);
    Word32 result = (val >> shiftAmt) | (val << (size - shiftAmt));
    return (result & mask);
}
```

```
static const Word8 table0[] = {(Word8) 0x3d, (Word8) 0xa7,
                                (Word8) 0x3e, (Word8) 0x81};
```

```
void Coef2(const Word8 s0 /* [8] */,
           const Word8 s1 /* [8] */,
           Word8 *out0 /* [8] */)
{
    if(s0 >= (Word8) 4)
    {
        fprintf(stderr, "\"coef.cry\", line 2, col 17: index of ");
        fprintf(stderr, \"%d\", (Word32) s0);
        fprintf(stderr, " is out of bounds\n(valid range is 0 thru 3).");
        fprintf(stderr, "\n");
        exit(-1);
    }
    if(s1 >= (Word8) 4)
    {
        fprintf(stderr, "\"coef.cry\", line 2, col 17: index of ");
        fprintf(stderr, \"%d\", (Word32) s1);
        fprintf(stderr, " is out of bounds\n(valid range is 0 thru 3).");
        fprintf(stderr, "\n");
        exit(-1);
    }
    const Word8 s2 = (s0 < 4) ? table0[s0] : (Word8) 0;
    const Word8 s3 = (s1 < 4) ? table0[s1] : (Word8) 0;
    const Word8 s4 = s2 * s3;
```

```
*out0 = s4;
```

```
return;
```

ois |

Importance of Reducing Code Size



A line of code is a cost not an asset

“[...] expressed programmer productivity in terms of ‘number of lines of code produced’.

[...] I pointed out that a programmer should produce solutions, and that, therefore, we should not talk about the number of lines of code produced, but the number of lines used, and that this number ought to be booked on the other side of the ledger. ”

E.W. Dijkstra, Sept 1975







“Cost” applies to security as well as economics

High Speed Encryptor

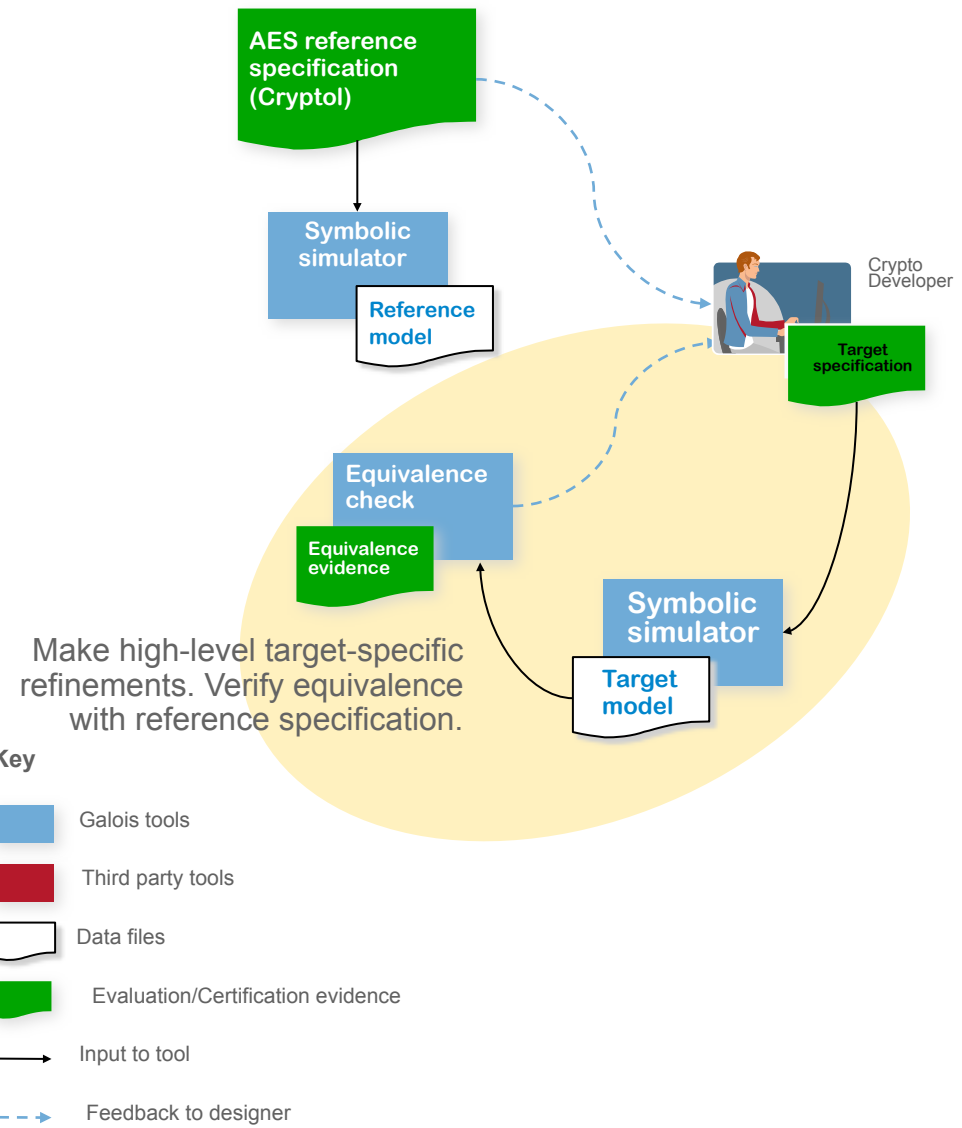


AES reference
specification
(Cryptol)

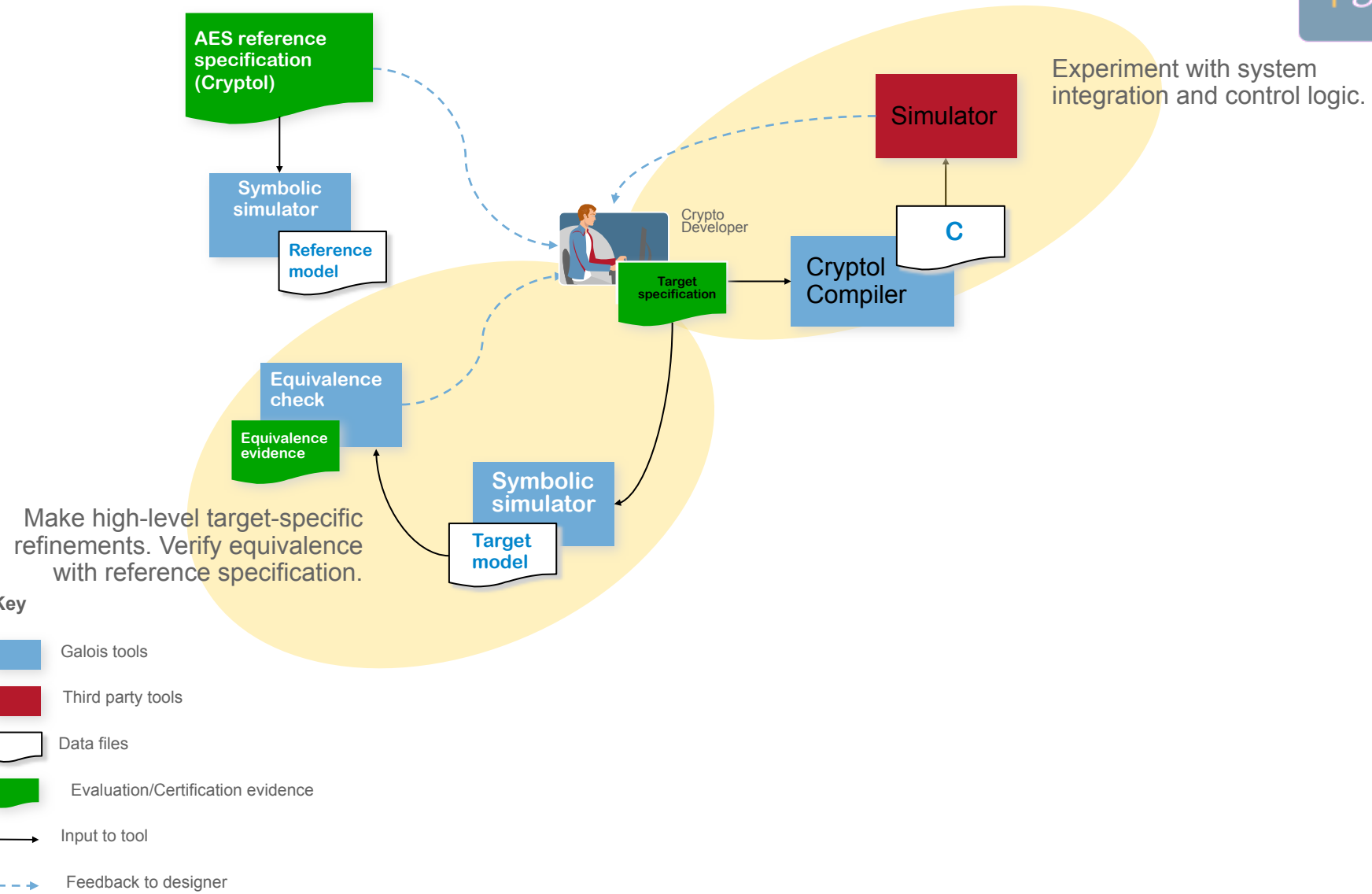
Key

-  Galois tools
-  Third party tools
-  Data files
-  Evaluation/Certification evidence
-  Input to tool
-  Feedback to designer

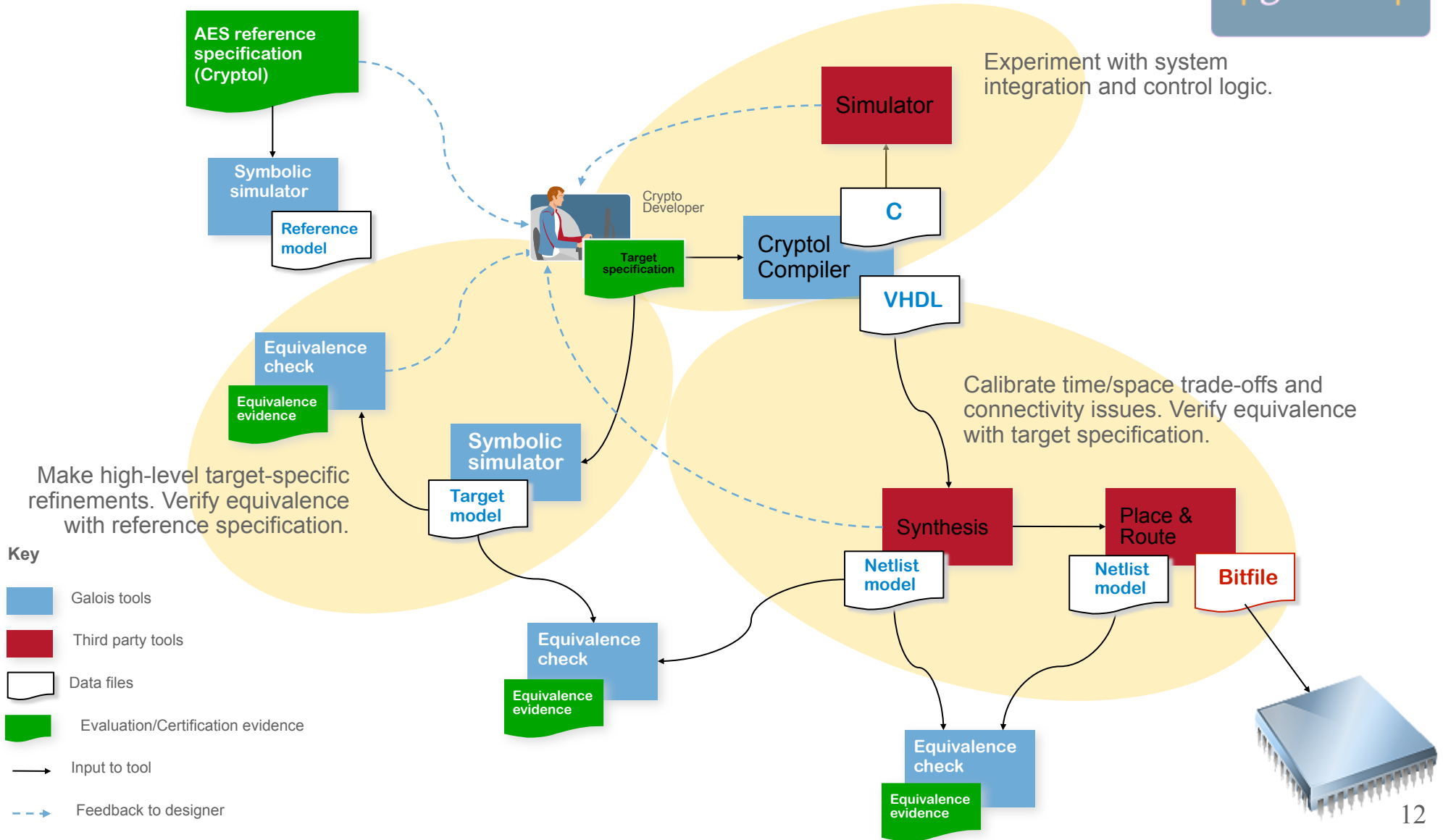
High Speed Encryptor



High Speed Encryptor

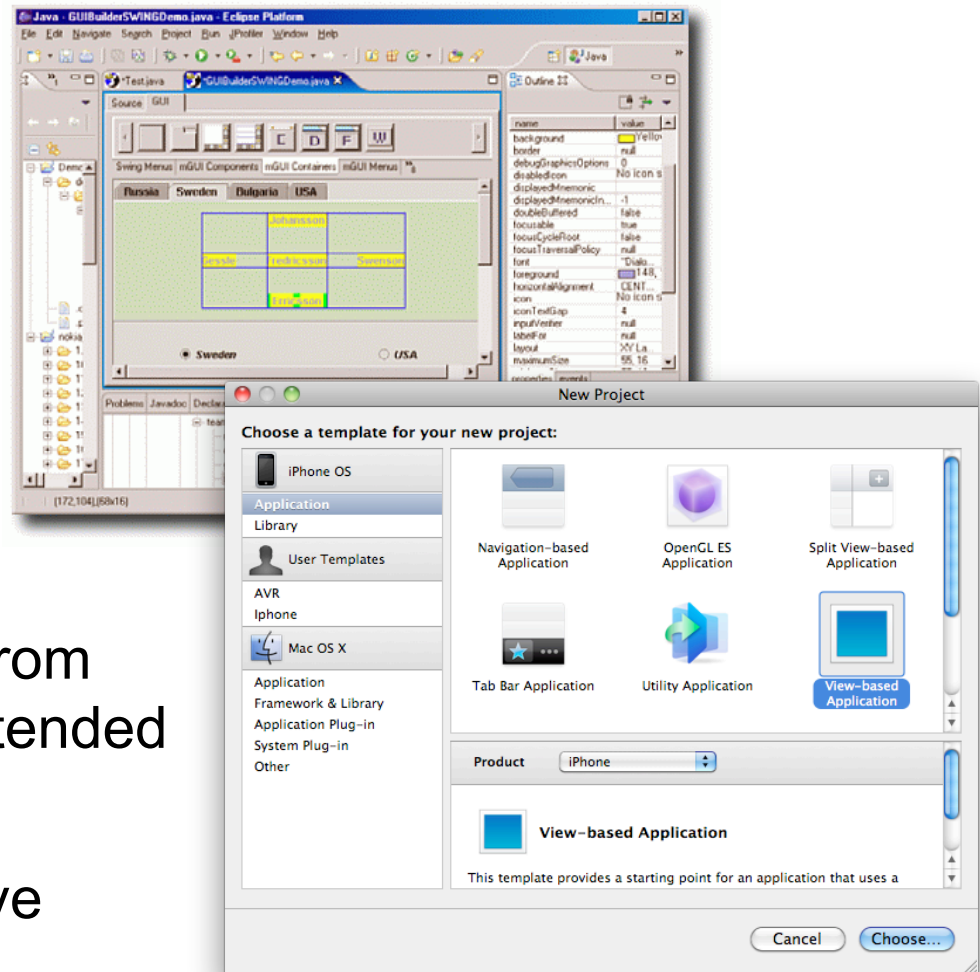


High Speed Encryptor



Where do DSLs come from?

- ❖ Existing domain notations
 - ❖ Textual
 - ❖ Graphical
 - ❖ Gestural, etc.
- ❖ Existing domain concepts
 - ❖ Often need to be extracted from domain expert through extended discussions
 - ❖ Domain experts may not have crystallized the concepts



SQL Injection Attack



server code

```
protected void btnSearch_Click(object sender, EventArgs e)
{
    String cmd = "SELECT [CustomerID], [CompanyName], [ContactName]
FROM [Customers] WHERE CompanyName ='" + txtCompanyName.Text
+ "'";
    SqlDataSource1.SelectCommand = cmd;
    GridView1.Visible = true;
}
```

- ❖ Easy to introduce SQL injection vulnerabilities into web apps
- ❖ 'Easy' to fix them *with proper diligence...*
- ❖ LINQ to SQL removes the possibility of SQL injection attacks

SQL Injection Attack

server code

```
protected void btnSearch_Click(object sender, EventArgs e)
{
    String cmd = "SELECT [CustomerID], [CompanyName], [ContactName]
FROM [Customers] WHERE CompanyName ='" + txtCompanyName.Text
+ "'";
    SqlDataSource1.SelectCommand = cmd;
    GridView1.Visible = true;
}
```

malicious input

```
John Launchbury' UNION SELECT CustomerID, ShipName, ShipAddress
FROM ORDERS--
```

- ❖ Easy to introduce SQL injection vulnerabilities into web apps
- ❖ 'Easy' to fix them *with proper diligence...*
- ❖ LINQ to SQL removes the possibility of SQL injection attacks

SQL Injection Attack



Corrupted query

```
server code
protected void Page_Load()
{
    String companyName = "John Launchbury";
    SqlCommand cmd = new SqlCommand("SELECT [CustomerID], [CompanyName], [ContactName] FROM [Customers] WHERE CompanyName = 'John Launchbury' UNION SELECT CustomerID, ShipName, ShipAddress FROM ORDERS--", conn);
    SqlDataReader reader = cmd.ExecuteReader();
    GridView1.DataSource = reader;
    GridView1.DataBind();
    GridView1.Visible = true;
}
```

malicious input

```
John Launchbury' UNION SELECT CustomerID, ShipName, ShipAddress FROM ORDERS--
```

- ❖ Easy to introduce SQL injection vulnerabilities into web apps
- ❖ 'Easy' to fix them *with proper diligence...*
- ❖ LINQ to SQL removes the possibility of SQL injection attacks

SQL Injection Attack

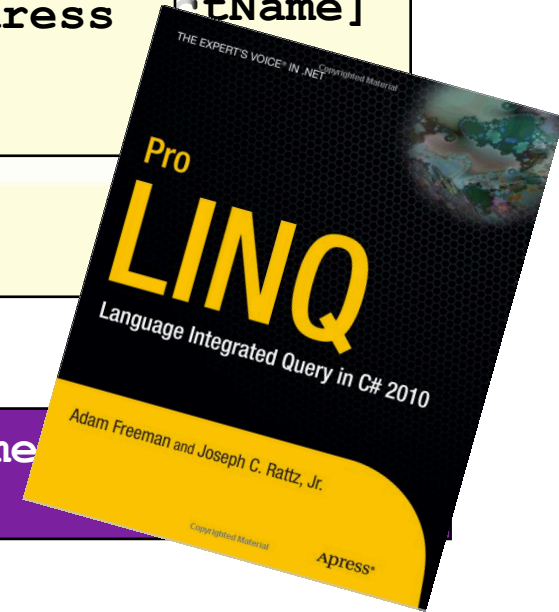


Corrupted query

```
server code
protected void Page_Load()
{
    String companyName = "John Launchbury";
    SqlCommand cmd = new SqlCommand("SELECT [CustomerID], [CompanyName], [ContactName] FROM [Customers] WHERE CompanyName = 'John Launchbury' UNION SELECT CustomerID, ShipName, ShipAddress FROM ORDERS--", conn);
    SqlDataReader reader = cmd.ExecuteReader();
    GridView1.DataSource = reader;
    GridView1.Visible = true;
}
```

malicious input

```
John Launchbury' UNION SELECT CustomerID, ShipName FROM ORDERS--
```



- ❖ Easy to introduce SQL injection vulnerabilities into web apps
- ❖ ‘Easy’ to fix them *with proper diligence...*
- ❖ LINQ to SQL removes the possibility of SQL injection attacks

Barrelfish

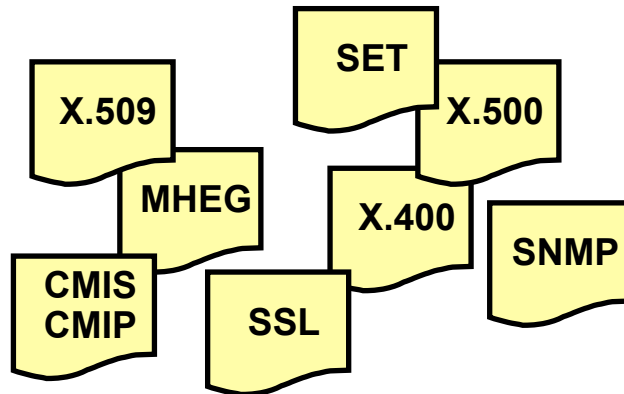
- ❖ Multikernel constructed with many DSLs
- ❖ E.g. Mackerel for specifying interaction with devices
- ❖ Produces high performance code



```
constants vdm "Vector delivery mode" {
    fixed    = 0b000 "Fixed";
lowest    = 0b001 "Lowest priority";
smi       = 0b010 "SMI";
nmi       = 0b100 "NMI";
init      = 0b101 "INIT";
startup   = 0b110 "Start Up";
extint    = 0b111 "ExtINT";
};
```

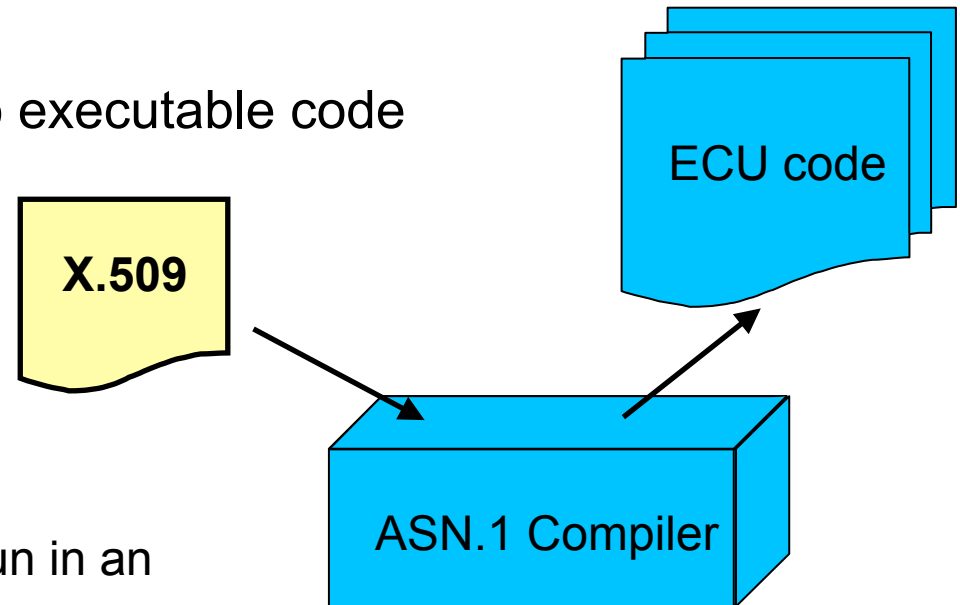
```
space std(idx) valuewise "Standard register space";
register ndx rw io(base, 0x70) "Standard index"
    type(uint8);
register target rw io(base, 0x71) "Standard target"
    type(uint8);
regarray standard rw std(0x00)[256] type(uint8);
```

ASN.1 Specifications



ASN.1 specifications are ubiquitous

- ❖ ASN.1 specifications are turned into executable code
 - ❖ By hand, or
 - ❖ By a compiler
- ❖ Compiler
 - ❖ Input: ASN.1 description
 - ❖ Output: encoder/decoder code to run in an application, e.g., on an ECU



ASN.1 Vulnerabilities (2009)

Search Results

There are **33** CVE entries or candidates that match your search.

CVE version: 20061101

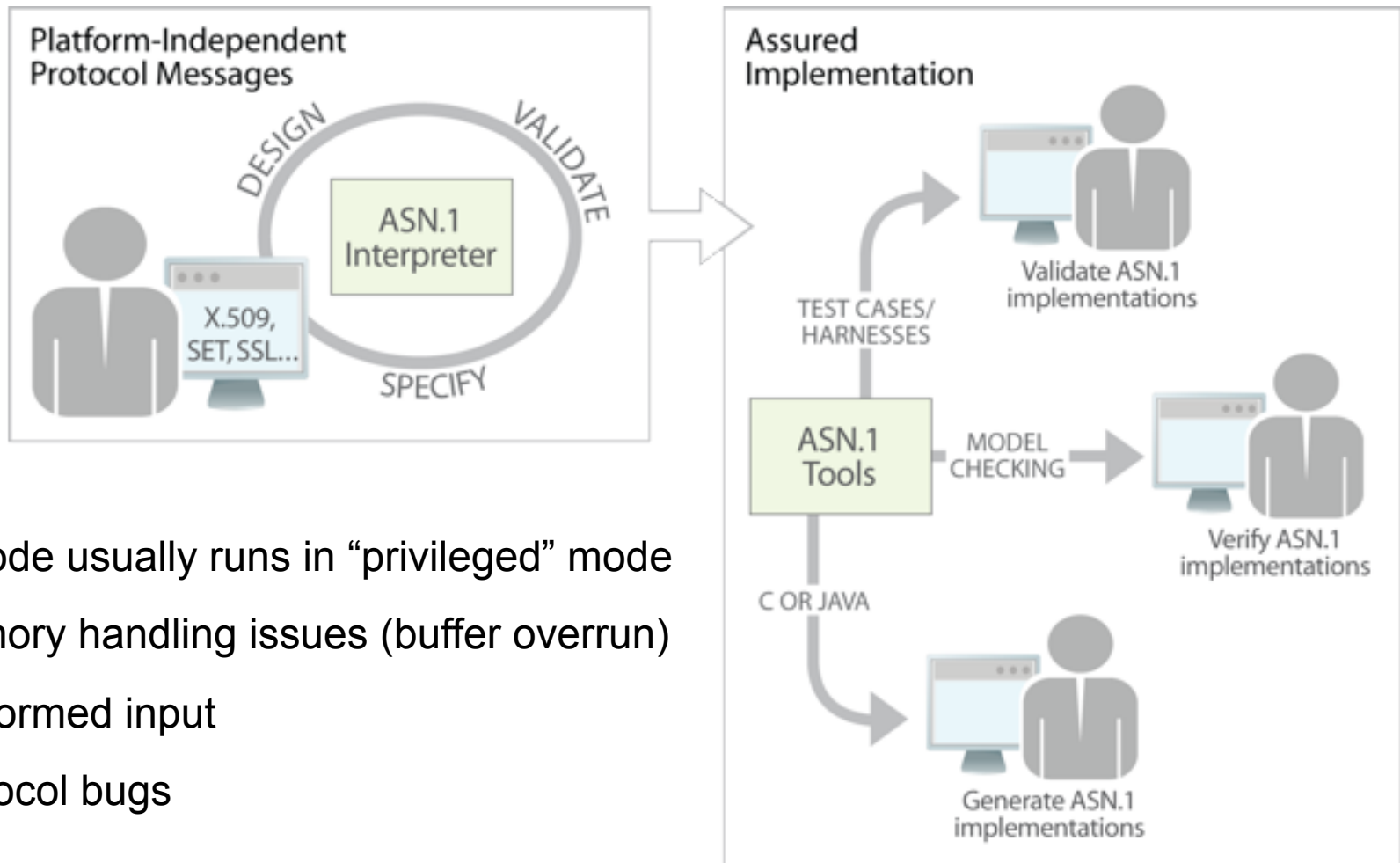
Name	Description
CVE-2009-0847	The <code>asn1buf_imbed</code> function in the ASN.1 decoder in MIT Kerberos 5 (aka krb5) 1.6.3, when PK-INIT is used, allows remote attackers to cause a denial of service (application crash) via a crafted length value that triggers an erroneous malloc call, related to incorrect calculations with pointer arithmetic.
CVE-2009-0846	The <code>asn1_decode_generaltime</code> function in <code>lib/krb5/asn.1/asn1_decode.c</code> in the ASN.1 GeneralizedTime decoder in MIT Kerberos 5 (aka krb5) before 1.6.4 allows remote attackers to cause a denial of service (daemon crash) or possibly execute arbitrary code via vectors involving an invalid DER encoding that triggers a free of an uninitialized pointer.
CVE-2009-0845	The <code>spnego_gss_accept_sec_context</code> function in <code>lib/gssapi/spnego/spnego_mech.c</code> in MIT Kerberos 5 (aka krb5) 1.5 through 1.6.3, when SPNEGO is used, allows remote attackers to cause a denial of service (NULL pointer dereference and daemon crash) via invalid ContextFlags data in the reqFlags field in a negTokenInit token.
CVE-2009-0844	The <code>get_input_token</code> function in the SPNEGO implementation in MIT Kerberos 5 (aka krb5) 1.5 through 1.6.3 allows remote attackers to cause a denial of service (daemon crash) and possibly obtain sensitive information via a crafted length value that triggers a buffer over-read.
CVE-2009-0789	OpenSSL before 0.9.8k on WIN64 and certain other platforms does not properly handle a malformed ASN.1 structure, which allows remote attackers to cause a denial of service (invalid memory access and application crash) by placing this structure in the public key of a certificate, as demonstrated by an RSA public key.
CVE-2008-2952	<code>liblber/io.c</code> in OpenLDAP 2.2.4 to 2.4.10 allows remote attackers to cause a denial of service (program termination) via crafted ASN.1 BER datagrams that trigger an assertion error.
CVE-2008-1673	The <code>asn1</code> implementation in (a) the Linux kernel 2.4 before 2.4.36.6 and 2.6 before 2.6.25.5, as used in the <code>cifs</code> and <code>ip_nat_snmp_basic</code> modules; and (b) the <code>gxsnmp</code> package; does not properly validate length values during decoding of ASN.1 BER data, which allows remote attackers to cause a denial of service (crash) or execute arbitrary code via (1) a length greater than the working buffer, which can lead to an unspecified overflow; (2) an oid length of zero, which can lead to an off-by-one error; or (3) an indefinite length for a primitive encoding.

The Challenge of ASN.1



- ❖ ASN.1 is a very large language
 - ❖ Grammar has ~700 productions (most languages < 100)
 - ❖ Constraints (X.680, X.682), information objects (X.681), parameterization (X.683)
- ❖ The ASN.1 definition is dense
 - ❖ Precise semantics of ASN.1 is difficult to extract
 - ❖ E.g., constraints and type equality are given in terms of concrete syntax
 - ❖ Features of the language interfere with each other
- ❖ ASN.1 executable code faces implementation challenges
 - ❖ Numerous opportunities for overflowing machine representations
 - ❖ E.g., arbitrarily long octet streams led to bug in a major ASN.1 library
 - ❖ Related concepts are handled quite differently
 - ❖ E.g., long tags vs. long lengths vs. long values

How to Gain High Assurance ASN.1



- ❖ ASN.1 code usually runs in “privileged” mode
 - ❖ Memory handling issues (buffer overrun)
 - ❖ Malformed input
 - ❖ Protocol bugs

Implementing DSLs



- ❖ Stand-alone implementation
 - ❖ Custom parsing and error messages
 - ❖ Many tools, integration with other systems
 - ❖ DSL has a tendency to grow...
- ❖ Embedded in a host language (EDSL)
 - ❖ Allows a tightly constrained domain-specific language with a rich and well-structured macro layer
 - ❖ Haskell and Java are popular host choices

Sample Copilot specification



If the majority of the three engine temperature probes has exceeded 250 degrees, then the cooler is engaged and remains engaged until the temperature of the majority of the probes drop to 250 degrees or less. Otherwise, trigger an immediate shutdown of the engine.

```
engineMonitor = do
  trigger "shutoff" (not ok) [arg maj]

where

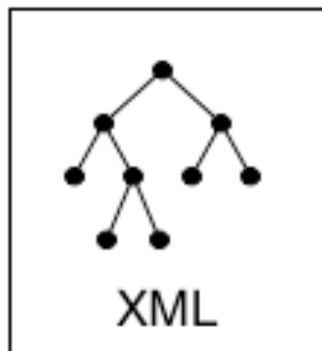
vals      = map externW8 ["tmp_probe_0", "tmp_probe_1", "tmp_probe_2"]
exceed    = map (< 250) vals
maj       = majority exceed
checkMaj  = aMajority exceed maj
ok        = alwaysBeen ((maj && checkMaj) ==> extern "cooler")
```

Key: library functions trigger macros

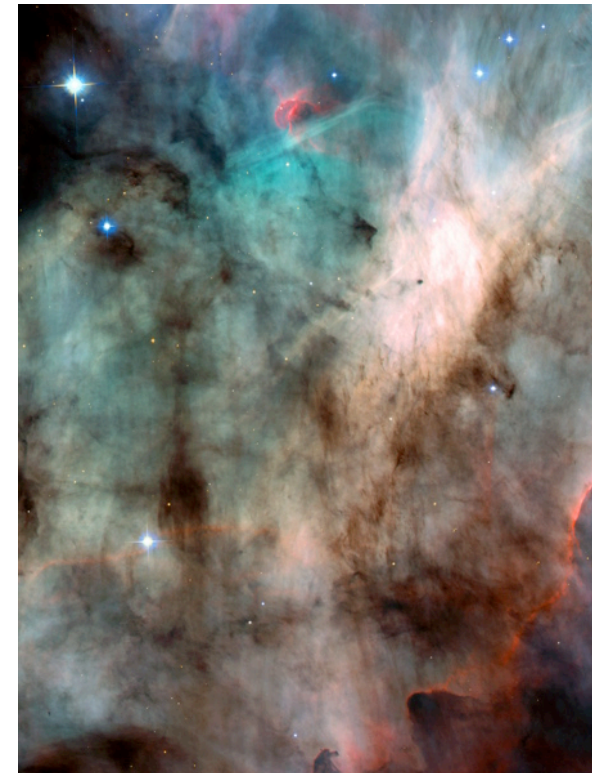
Data, data, everywhere!



Incredible amounts of data are stored in well-behaved formats:



Vast amounts of chaotic *ad hoc* data from logs, legacy systems, *etc.* exist as well.

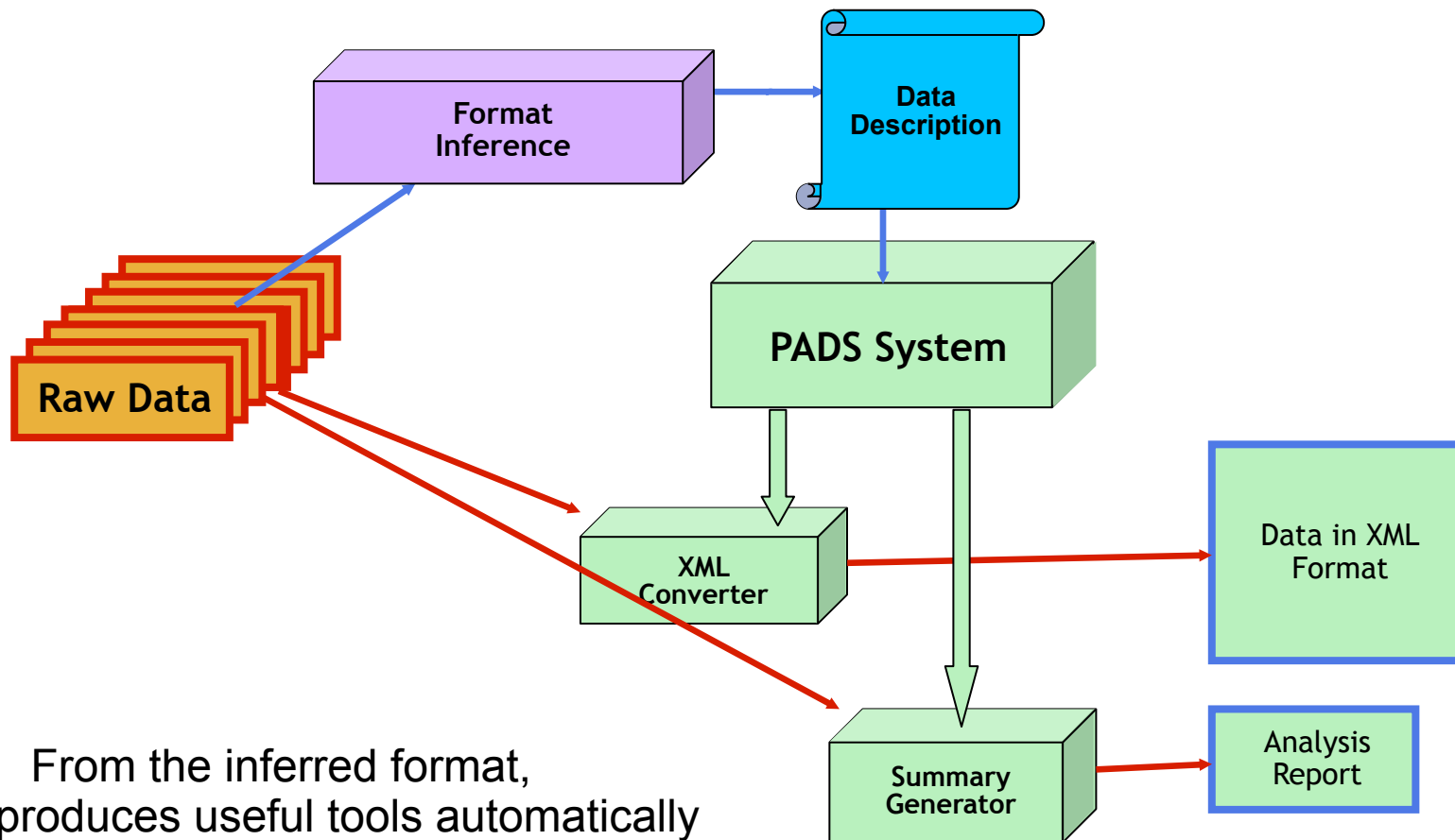


Lots of data is not in well-behaved formats, and has no ready tool support

Tools for Free



- ❖ From the data description, PADS automates many data tasks
 - ❖ Examples: converting raw data into XML; producing statistical reports; ...



PADS in Haskell

```
[pads|
type VCards = [VCard | EOR] terminator EOF

data VCard = VCard ("BEGIN:", vcardRE, EOR,
                  [Entry|EOR] terminator "END:", vcardRE)

data Entry = Entry { prefix    :: Maybe ("item", Int, '.'),
                   tag        :: Tag,
                   sep        :: StringME semicommaRE,
                   property   :: VCardProperty tag }

data Tag = VERSION | N | FN | NICKNAME | PHOTO | BDAY | ADR | LABEL
         | TEL | EMAIL | MAILER | TZ |GEO | TITLE | ROLE | LOGO | AGENT
         | ORG | CATEGORIES | NOTE | PRODID | REV | SORTSTRING "SORT-STRING"
         | SOUND | UID | URL | CLASS | KEY
         | EXTENSION ("X-", VCardString)
         | ITEM "item"

data VCardProperty (tag :: Tag) = case tag of

    VERSION  -> Version (Int, '.', Int)
  | N        -> Names IndividualNames
  | FN       -> FName VCardString
  | NICKNAME -> Nickname NameRs
  | PHOTO    -> Photo VCardData
  | BDAY     -> Birthday { bdayType :: Maybe ("value=date:", ())
                        , bdate     :: DateFSE <| ("%Y-%m-%d", RE "$") |>
                        }
  :
  :
```


PADS in Haskell



```
type VCards = VCard
```

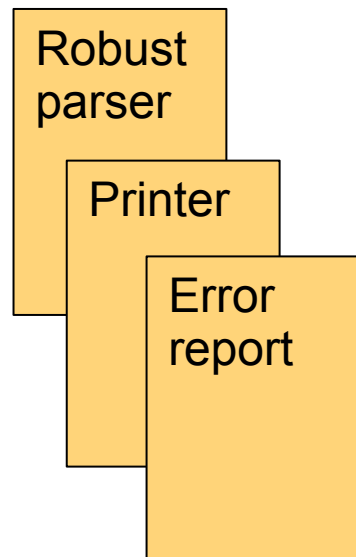
```
data VCard = VCard Entry
```

```
data Entry = Entry { prefix  :: Maybe Int,
                    tag      :: Tag,
                    sep       :: String,
                    property  :: VCardProperty }
```

```
data Tag = VERSION | N | FN | NICKNAME | PHOTO | BDAY | ADR | LABEL
         | TEL | EMAIL | MAILER | TZ | GEO | TITLE | ROLE | LOGO | AGENT
         | ORG | CATEGORIES | NOTE | PRODID | REV | SORTSTRING
         | SOUND | UID | URL | CLASS | KEY
         | EXTENSION VCardString
         | ITEM
```

```
data VCardProperty =
```

```
    Version (Int, Int)
  | Names IndividualNames
  | FName VCardString
  | Nickname NameRs
  | Photo VCardData
  | Birthday { bdayType :: Maybe ()
              , bdate   :: DateFSE String
              }
```



```
:
```

```
:
```

Conclusions



- ❖ Domain-specific techniques raise the level of abstraction
 - ❖ Provide opportunities for modeling, verification, code generation
 - ❖ Provide opportunities to systematically remove classes of vulnerabilities
 - ❖ A range of implementation techniques are available
- ❖ Questions remain
 - ❖ Composition of DSLs – how do you put them together?
 - ❖ How do we further leverage techniques across different DSLs?
 - ❖ How do we bring DSLs to new areas, e.g. modeling human behavior?