Attack tree of smart cards

Zoltán Kincses, kincses@sztaki.hu
Computer and Automation Research Institute, Hungarian Academy of Sciences,
Budapest – Hungary

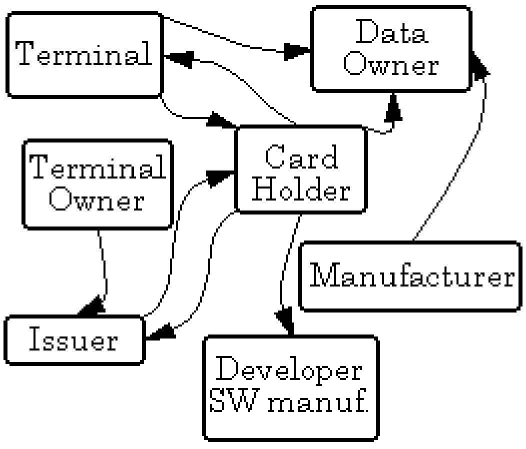
1. Introduction

The growth of smart card systems faces security threats to both the card and its environment [WR_OV]. Issues related to readers, protocol implementations, the smart card's hardware security features [ML_HW] or a combination of logical and physical attacks [KZ_LT] are of legitimate concern.

The attackers must be categorized by their expertise, goals and budget to discover the source of the possible attacks. State-sponsored spies, fanatical terrorists and researchers (with their students) have different goals, expertise and budget.

The smart card is not itself the goal. It is the tool for reaching goals and is only a small part of the system. In a complete system the card readers, the communication lines, and the interfaces between the elements, even the standards of each element and the compatibility between them must be considered.

All the elements of a smart card system have their own specific threats [DC_ES]. They could be attacked in various ways (see relevant chapters of [RA_SE]) or member parties could attack each other within a smart card system [BSAS_ST]. The possible "attack relations" are shown in the figure below:



Based on http://www.schneier.com/paper-smart-card-threats.html

Several publications mentioned above have organized the attacks into custom-built taxonomies. These taxonomies enumerate and discuss each type of attacks or attack classes, but the system analyst needs to draw the cumulative impact of the attacks in the case of a working smart card system. Moreover, the system changes during the design, development or even working phase, and these changes must be handled dynamically.

2. The basic concept

The "attack tree" concept [BS_AT] gives the opportunity not only to enumerate but also to organize in a manageable structure all possible attacks and many parameters of them. In this paper, the first version of *"Smart card attack tree"* is introduced.

The attack tree is an "and-or" tree where the goal to be reached is on the top (root of the tree) and the lines represent the ways where threats could come from. The 'AND' lines must both happen in order to reach the parent node, while in case of 'OR' lines it is sufficient that one of them is realized. Each node can have a cost, a probability of occurrence, or even a 'required tools' value associated with it. The lowest cost path can be determined to any intermediate node or to the root-node. It is also possible to identify the most probable way from any point in the tree to the root. New parameters (e.g. the relevance of the attack on a specific system) and its value range can be added. A detailed description with many samples about attack tree methodology can be found in [AM_ST].

Based on the available papers on this topic the top-level smart card attack types can be logical, physical, or social based on *type*; hardware, software or firmware by *target*; application, data or protocol by *level*; etc. Sometimes the attacks belongs to multiple types, therefore this classification method is not well suited to attack tree creation.

It is possible to build complex classifications where the elements produce a cross-connected net (such as in the figure above), but this is also not the best solution for a tree with multiple parameters, which must each have input values to serve different calculations.

The available papers help to enumerate and understand the possible attacks, and they can be put in place in a tree based on the three main threats against confidentiality, integrity and availability. Later this can be combined with the security controls (preventive, detective, corrective) to produce a PreDeCo/CIA matrix. Security auditors use this matrix and also a smart card system can be audited by such a concept to discover if there is any lack of control of main threats. A general matrix is shown below:

Threats / Controls	**Confidentiality** *Unauthorized access of sensitive information*	**Integrity** *Manipulation, replacement of data or application*	**Availability** *Denial of service, blocking, damage*
Preventive	Set access rights, encryption (encoding)	Encryption (hash, digital signature)	Multiplied resources, Reserves (stock)
Detective	Monitoring the access rights, analysing access logs (accountability)	Integrity checker, Data format and interpretation standards	Sensors
Corrective	Preventive controls for the future (near nothing for the past)	Rollback, Use of backup	New resources, Exclusion of illegal communication

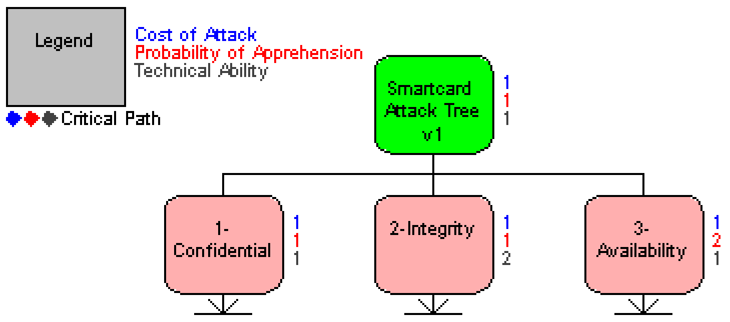
Summarizing the proposed protection methods discussed in the papers cited in the Introduction, it can be stated that:

- # in most cases, an encryption technique is proposed (preventive control) followed by proposing sensors and detectors (detective controls) [DC_ES];
- # there are expensive attacks, which are hard to defend against ("The ion beam will always get in." [RA_SE]), therefore, technical defences must be combined with legal defences;
- # in general, three approaches can be taken: *fewer splits* (e.g. data owner = cardholder, then the attack class is eliminated on the first figure), *greater transparency* (open publication helps to discover the possible weaknesses in time) and *design for security* (during the whole life-cycle of the smart card system).

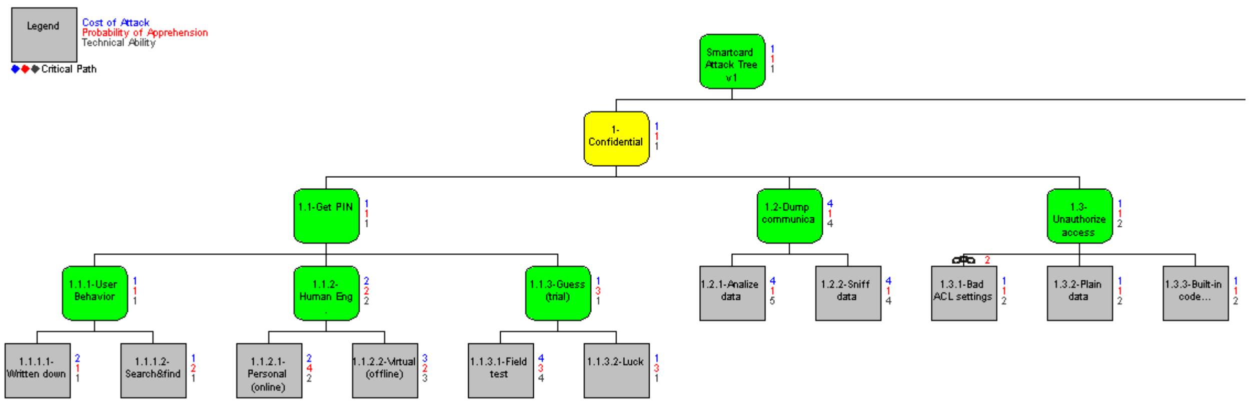
Going into details of this matrix is not the scope of this paper therefore I return to the discussion of the smart card attack tree.

3. The smart card attack tree

As mentioned above, the first level in the attack tree has three sub-nodes: confidentiality, integrity and availability (see also the legend information on the figure below). In this sample the nodes have three parameters (Cost of Attack, Probability of Apprehension[1] and Technical Ability) with a 1 to 5 value-range (e.g. 1 the cheapest cost level, 5 the most expensive cost level):



Taking into consideration the exact value of the ranges and the rules used of calculation, each node has its own value. These values can be recalculated if the tree, the calculations or the ranges are modified. Attack trees grow quickly as the builder goes deeper and deeper to the leaves. Therefore, in this paper, the graphical description of the lower levels is not included. A brief sub-tree of the 'Confidentiality' node is shown below:



Legend
Cost of Attack
Probability of Apprehension
Technical Ability
◆◆◆ Critical Path
Smartcard Attack Tree v1 1 1 1
1-Confidential 1 1 1
1.1-Get PIN 1 1 1
1.2-Dump communica 4 1 4
1.3-Unauthorize access 1 1 2
1.1.1-User Behavior 1 1 1
1.1.2-Human Eng 2 2 2
1.1.3-Guess (trial) 1 3 1
1.2.1-Analize data 4 1 5
1.2.2-Sniff data 4 1 4
2
1.3.1-Bad ACL settings 1 1 2
1.3.2-Plain data 1 1 2
1.3.3-Built-in code... 1 1 2
1.1.1.1-Written down 2 1 1
1.1.1.2-Search&find 1 2 1
1.1.2.1-Personal (online) 2 4 2
1.1.2.2-Virtual (offline) 3 2 3
1.1.3.1-Field test 4 3 4
1.1.3.2-Luck 1 3 1

The 1.3.1 node has a 'link' sign because the same attack type (Bad ACL settings) is used on another sub-node. The convention is to represent identical sub-trees as links.

Although the graphical format is illustrative, the text-based format is more easily printed and analysed:

[1] Usually the value is from 0 to 1, but in our case we use from 1 to 5.

```
Smartcard Attack Tree v1

1 <OR> Confidentiality
   1.1 <OR> Get PIN
      1.1.1 <OR> User Behaviour
         1.1.1.1  Written down
         1.1.1.2  Search & find
      1.1.2 <OR> Human Eng.
         1.1.2.1  Personal (online)
         1.1.2.2  Virtual (offline)
      1.1.3 <OR> Guess (trial)
         1.1.3.1  Field test
         1.1.3.2  Luck
   1.2 <OR> Dump communication
      1.2.1  Analyse data
      1.2.2  Sniff data
   1.3 <OR> Unauthorized access
      1.3.1  Bad ACL settings
      1.3.2  Plain data
      1.3.3  Built-in code...
2 <OR> Integrity
   2.1 <OR> Application/algorithm
      2.1.1  Bad RND
      2.1.2 <OR> Bad concept/development
         2.1.2.1  Bad RND
         2.1.2.2  Test code
         2.1.2.3  Bad code
      2.1.3 <OR> External influence
         2.1.3.1 <OR> HW + OS + silicon
            2.1.3.1.1  Differential Analysis
            2.1.3.1.2  Frequency manipulations
            2.1.3.1.3  Light attacks
            2.1.3.1.4  Cryptanalysis
         2.1.3.2 <OR> Reverse engineering
            2.1.3.2.1  Chip detach
            2.1.3.2.2  Optical analysis
         2.1.3.3  Fake CAD
   2.2 <OR> Data
      2.2.1  Bad ACL settings
      2.2.2  Bad codepage
      2.2.3  Cryptanalysis
   2.3 <OR> Protocol(s)
      2.3.1  Bad algorithm
      2.3.2  Bad design
      2.3.3  Bad implementation
3 <OR> Availability
   3.1 <AND> Block access
      3.1.1  Block PIN
      3.1.2  Block PIN2/PUK
   3.2 <OR> Denial of Service
      3.2.1  Extreme communication
   3.3 <OR> HW damage
      3.3.1 <OR> Card damage
         3.3.1.1  Physical damage
         3.3.1.2  Logical
      3.3.2 <OR> CAD damage
         3.3.2.1  Physical
         3.3.2.2  Logical
      3.3.3  CAD-holder damage
```

It can be seen that the tree has its contact points to other trees, like an operating system or cryptology attack trees. Some of the contacts can be interpreted as a sub-tree (e.g. under “2.1.3.3. Fake CAD” the attack tree of Card Acceptance Devices), some of them can be just connected as references (e.g. 2.1.1 and 2.1.2.1 Bad RND depending on definitions and attributes of bad random numbers). For example, if a smart card is used on a Solaris operating system for PGP-based digital signatures, then the attack tree of that version of the operating system and the attack tree of PGP (included in [BS_AT]) must be included in the attack tree of the smart card system.

It can be seen that in reality the attack tree resulting from the linked elements can be a graph, and the ‘attack forest’ of the attack trees can form a complex graph. This opens up other research possibilities about the features of such a graph and their behaviour in case of modifications (new elements, changes, deletion).

Stepping toward from general case, we can create an attack tree for such specific systems such as an electronic purse, a digital signature card, an ID card or a travel card. These trees can be created following the same concept as mentioned previously, but that is better for audit to discover the lack of controls. In a specific case, like a smart card for car parking, the elements of the system must be known. There are many questions before attack tree creation can be started, like (just a few samples):

- What are the system elements? E.g. can the parking fee be paid by mobile phone?
- Can be the card reloaded? Can it be reloaded illegally?
- The reader contains a SAM module? Could one stolen reader compromise the whole system?
- What kind of authentication exists between the reader and the card? Which one can be emulated or cloned to circumvent the other?

4. Conclusions, remarks

The advantage of an attack tree is that in the future every new attack type can be inserted or existing node values can be modified and the attack ways can be recounted. It is also possible to connect different attack trees. Smart card systems are used with biometry or cryptology many times, where both could have their own attack tree. The situation is the same true with the card readers or any other CAD (PCs, laptops or mobile phones), which have their own attack trees.

The creation of a smart card attack tree could involve other areas, – both directly and also indirectly influenced areas – producing an ‘attack forest’ of information technology.

Last but not least, such an attack tree could help the smart card project managers for planning their project (what are the risks and their cost?), helps the developers using the tree as a checklist when they design their applications (what must be considered?) and also helps auditors who check the security of a system or an implementation (what kind of controls have been applied?).

The work is also a research project and all the enthusiastic interested parties are welcome to add their own tree or small forest to turn the security jungle in a manageable oasis. In “Creating attack trees” part of [BS_AT] can be found: “*Give the tree to someone else, and have him think about the process and add any nodes he thinks of. Repeat as necessary, possibly over the course of several months. Of course there's always the chance that you forgot about an attack, but you'll get better with time. Like any security analysis, creating attack trees requires a certain mindset and takes practice.*”

5. Acknowledgements

Thanks to the Computer and Automation Research Institute of the Hungarian Academy of Sciences for their support during the project. Thanks to Amenaza Technologies Ltd. (especially to Terry Ingoldsby) for their continuous and selfless help with prolonged licenses to use their SecurITree™ product in creating and managing the attack tree of smart cards.

6. References

[AM_ST] Terrance R. Ingoldsby: Understanding Risk Through Attack Tree Analysis, CSI Computer Security Journal, Spring 2004, Volume XX, Number 2. pp 33-59
`http://www.amenaza.com/request_methodology.html`

[BS_AT] Bruce Schneier: (also appeared in his book 'Secrets and Lies') about attack trees:
`http://www.schneier.com/paper-attacktrees-ddj-ft.html`

[BSAS_ST] Bruce Schneier, Adam Schostack: Breaking Up Is Hard To Do: Modelling Security Threats for Smart Cards, Usenix Workshop on Smartcard Technology, February 1999.
`http://www.homeport.org/~adam/smart-card-threats.pdf`
`http://www.schneier.com/paper-smart-card-threats.pdf`

[DC_ES] David Corcoran: Security-related Exposures and Solutions in Smartcards, Information Security Bulletin, November 2000. pp. 13-22.

[KZ_LT] Zoltán Kincses: On avoidance of attacks against the pin error counter of smart cards, (CS)^2 – The Fourth Conference of PhD Students in Computer Science Szeged, Hungary, July 2004. Abstract on pp. 68:
`http://www.inf.u-szeged.hu/~cscs/cscs04.ps`

[ML_HW] Michael Lamla: Hardware attacks on smart cards – overview, Eurosmart Security Conference, 2000. Marseille, pp. 31-39.

[RA_SE] Ross Anderson: Security Engineering – A Guide to Building Dependable Distributed Systems, 2001. John Wiley & Sons Inc.
`http://www.cl.cam.ac.uk/~rja14/book.html`

[WR_OV] Wolfgang Rankl, Wolfgang Effing: Overview about Attacks on Smart Cards, 2003 Munich, from their own 'Smart Card Handbook' (John Wiley & Sons, ISBN: 0-470-85668-8)
`http://www.wrankl.de/SCH/Attacks.pdf`