Security modeling with BDMP: From theory to implementation

Ludovic Pièvre-Cambacédès (ludovic.pietre-cambacedes@edf.fr)*
Yann Deflesselle (yann.deflesselle@ensi-bourges.fr)†
Marc Bouissou (marc.bouissou@ecp.fr)*‡

Abstract: This paper discusses the implementation and use of the BDMP (Boolean logic Driven Markov Processes) formalism, recently adapted to graphical attack modeling. Theoretically, it offers an attractive trade-off between readability, scalability, modeling power and quantification capabilities. In practice, efficient model construction and security analysis need complementary tools and enhancements, which have been identified and developed only once the implementation has been realized and several security studies undertaken. In particular, attack sequence filtering based on attacker profiles and sensitivity analysis provide significant help for the analyst. Perspectives include the addition of a security pattern library or the connection with other modeling frameworks.

Keywords: Security, BDMP, attack modeling, risk analysis, sensitivity analysis.

1 Introduction

The BDMP (Boolean logic Driven Markov Processes) formalism has been recently adapted from the dependability area [BB03] to the security domain in order to model attack scenarios. Based on the theoretical framework of such an adaptation exposed in [PCB10c, PCB10a], we have extended the KB3 modeling software platform [Bou05] to let security analysts build and analyze security-oriented BDMP models. This paper presents this implementation. It also describes how such an implementation and first-hand experience have led to the development of complementary tools, enhancing the usability and utility of the models. The strengths and weaknesses of the approach are also discussed.

The paper is structured as follows. In Section 2, the main principles of BDMP applied to security are recalled. In Section 3, we describe our implementation and its technical background, before providing recommendations. Section 4 presents extensions that have been necessary to turn BDMP from a theoretical model into an operational formalism for attack scenario analysis. Section 5 focuses on perspectives and on-going work.

* Électricité de France R&D, 1 avenue du Général de Gaulle, 92141 Clamart, France
† ENSI de Bourges, 88 boulevard Lahitolle, 18000 Bourges, France
‡ École Centrale Paris, Grande Voie des Vignes, 92295 Châtenay-Malabry, France

2 BDMP and Security

2.1 An Attractive Trade-off in Graphical Attack Modeling

Graphical attack modeling formalisms are useful tools in red-team session preparations and risk analysis processes. The most common form is probably the attack tree [Sch99, MO05, Edg07], but there exist numerous other approaches, for instance based on Petri-nets [McD00, PML10] or on Bayesian networks [LM05, SEN09]. They all offer different levels of readability, scalability, modeling power and quantification capabilities. The adaptation of the BDMP formalism to attack modeling provides an attractive trade-off with respect to these four criteria [PCB10a]. Their visual aspect, close to attack trees, inherits their readability and ease of appropriation. Nevertheless, as opposed to attack trees, they allow capturing dynamic characteristics of the attack process such as sequences, detections and reactions. They also support more diverse results and quantifications: in particular, BDMPs support a whole spectrum of time-domain quantifications which cannot be obtained by classical attack trees. This includes the computation of the overall mean-time-to-success (MTTS), the probability of success in a given time and the list of possible attack success sequences, ordered by decreasing probability. Ref. [PCB10c] provides more detailed elements of comparison between BDMP, attack trees and Petri-net-based approaches for attack modeling.

2.2 BDMP for security in a nutshell

Fig. 1 represents a very simple BDMP modeling a two step attack with two alternatives for Step 1. The "trigger", represented by the dotted arrow, ensures that the leaf representing Step 2 is realizable only if Step 1 has been completed. The times needed for the realization of the leaves are defined by stochastic processes; their behaviors can be made dependent on other leaves by means of the triggers.

Tab. 1 shows the three kinds of leaves defined for security modeling. Their complete definitions can be found in [PCB10b].

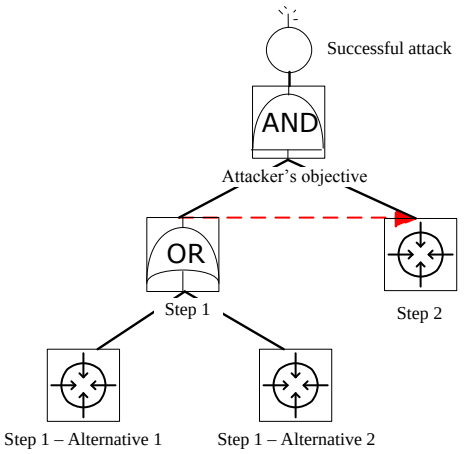
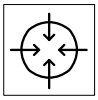


Fig. 1: A simple two-step attack model

Tab. 1: Basic BDMP leaves for security modeling

Representation	Modeled behavior
	The Attacker Action (AA) leaf models an attacker's step towards the realization of his/her objective. When activated (by means of triggers or as an initially active leaf), the time needed for its realization is modeled by an exponential distribution of parameter λ (implying a MTTS equal to $1/\lambda$).
	The Timed Security Event (TSE) leaf models an event the realization of which impacts the attacker's progress, but which is not under his direct control. The time needed is also exponentially distributed (MTTS=$1/\lambda$).
	The Instantaneous Security Event (ISE) leaf models an event that can happen instantaneously with a probability γ, when the leaf is activated.

In addition to the λ and γ parameters associated to the attack, detection and reaction parameters have been added, to cover defensive aspects. In [PCB10a], four types of detection have been defined for the AA and TSE leaves. These types are differentiated by the moment when the detection takes place. Type I (Initial) detections take place at the very start of the attacker's actions or of the modeled events; type O (On-going) detections take place during the attacker's attempts or during the modeled events; type F (Final) detections take place at the moment the attacker succeeds in an action or when an event is realized; type A (A posteriori) detections take place once an action or an event has been realized, based on the traces left by such an action or event. ISE leaves have been treated slightly differently with two types of detection.

2.3 Approach scalability and performance

In essence, BDMP can be seen as a compact and readable representation specifying potentially huge Markov chains. The main problem with Markov analysis is the combinatorial explosion of the state space that usually severely limits the use of matrix-based analytical methods. This led EDF to develop FigSeq [BL02], a tool designed and optimized to process large Markovian models by an original analytical method: the search and quantification of sequences leading the system to an undesirable state. This method allows making mastered approximations by limiting the sequence exploration to those having a probability greater than a given threshold; besides, it considerably helps in the model validation and in the detection of the most critical vulnerabilities of a system. For small models like those discussed in this article, exact calculations can be made by exploring exhaustively the sequences and using the closed form expressions for the probabilities of sequences given in [Har90]. The use of a Markovian model can of course be criticized for quantification aspects in security, as we have already discussed in [PCB10c]. But from a qualitative point of view, the sequences automatically found by FigSeq are correct under the weak assumption that the times involved in the modeled processes have all continuous distributions on $[0, +\infty[$.

The method used by FigSeq, even if it is very efficient on large Markov models, has its limits. BDMP bring further enhancement thanks to their native trimming mechanism. The combinatorial explosion in the processing needed to quantify a BDMP model is considerably limited by the use of the notion of relevant event, and the associated trimming mechanism [BB03]. For instance, as soon as one of the sons of an OR gate, modeling different attack techniques possibilities for a given intermediate objective, takes the value "true", the values of the other sons (and descendants) are not "relevant" anymore; the corresponding events are then inhibited. In fact, the trimming of irrelevant events generally corresponds to the inhibition of additional attack attempts of an already successful attack phase. This simplification provides both a better modeling and a very efficient reduction of the combinatorial explosion issues.

The ample experience gained from the processing of reliability models (the original application of BDMP) makes us confident in the scalability of the approach in security. Models with more than 100 leaves are routinely processed in reliability studies at EDF; indeed, we have been surprised by the size (sometimes exceeding 300 leaves) of some BDMP that FigSeq has been able to process in this area.

3 From Theory to Implementation

3.1 The KB3 Workbench

EDF started the development of its KB3 software suite in the 90's and has been using it since then to carry out its dependability studies [GPV99, PCPD04]. This software allows a fast and user-friendly definition of structural and behavioral models by graphical assembly of elementary components described in knowledge bases. Knowledge bases contain generic descriptions of the components typically encountered in a given kind of study. They enable expert knowledge re-usability and modularity. They are written in Figaro, an object-oriented language developed by EDF [BBBV91], specifically designed to build stochastic models. Fig. 2 shows the principles of the "KB3 workbench", made of KB3 (the model building tool) and various solvers.

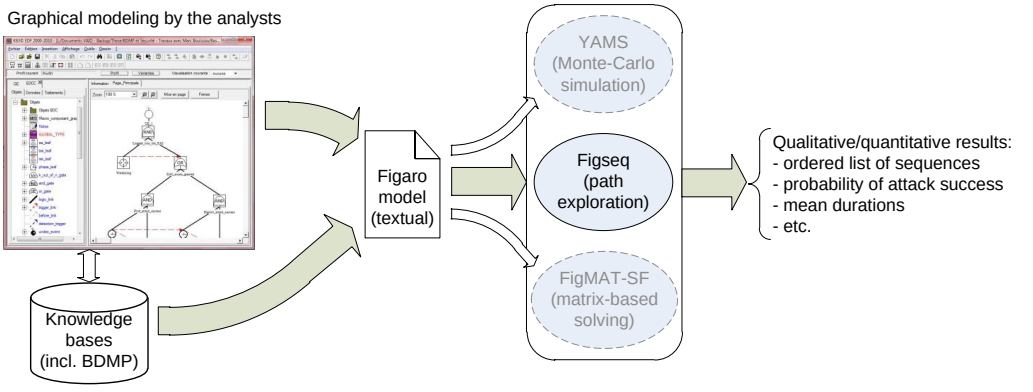


Fig. 2: The KB3 workbench overall architecture

3.2 A Specific Knowledge Base for Security

A specific knowledge base has been developed to implement the BDMP formalism for security modeling. It allows easy graphical modeling, interactive simulation and automatic processing for quantification. It is available on-line for testing[1]. This knowledge base is made of three parts: a text file written in the Figaro modeling language to describe the semantics (i.e. the behavior) of the graphical objects of a model, an XML file containing a customization of the generic GUI of KB3 (allowed/forbidden links between objects, color changes as a function of the state variables in an interactive simulation...), and the set of icons one can see in Tab. 1. The excerpts of the knowledge base in Fig. 3 are taken from the Figaro file, and thus give an idea of the language and its features. They are taken from the class `aa_leaf`, which corresponds to the Attacker Action leaf, briefly depicted in Section 2.1.

The first line is simply the declaration of the class, which inherits all the characteristics of its parent class `leaf`. Figaro is an object-oriented language, thus allowing the factorization of characteristics thanks to the mechanism of inheritance. The subsequent lines define a set of invariable characteristics of the class. Their value can belong to various domains: Boolean, integer, float or enumerated. Each of these constants has a default value defined in the knowledge base, but this value can be changed by the user in the description of a particular system. The code after the keyword `OCCURENCE` corresponds to the behaviorial part of the class description, with stochastic transition characteristics.

The behaviors of the leaves presented in Tab. 1 have been coded in Figaro following the formal definitions given in [PCB10b]. This work was performed using Visual Figaro [2], an Open-source application made to support the design of knowledge bases for KB3. The simplicity of Visual Figaro allowed the work to be completed in only a few weeks.

Once done, it has been possible to model diverse attack scenarios with KB3, including for instance the use-case depicted in Fig. 4. This use-case has been extensively described and analyzed in former publications [PCB10a, PCB10b]: it represents the attack of a password-protected file, a copy of which has been stolen. In this scenario, obtaining the password is the only way to access its contents, which is needed by the attacker within a week (this may for instance take place in a call for tender). Globally, the attacker will make use of three groups of techniques. He will of course try to crack the password locally with classical techniques, but in the meantime, he will also try social engineering techniques before trying to install a keylogger. In this third approach, the keylogger installation will first be attempted remotely, and if unsuccessful after a given time, the attacker will try to install it physically. The dynamics of the attack is captured by a set of triggers but also by phase leaves (represented by clocks): these leaves enable switching from one sub-tree to another sub-tree if the corresponding attack techniques have not succeeded in a given time. Detection and reaction parameters have also been taken into account in the use-case.

Once the model is parameterized, the FigSeq solver leads to the overall attack success probability for the considered timeframe, but also to a complete list of the sequences leading to the attack success, ordered by probability of occurrence. For the use-case of Fig. 4, there are more than four thousands attack sequences (4231); nevertheless, with the chosen parameters, the top 4, given in Table 2, represent more than 50 % of the success

[1] `http://sourceforge.net/projects/visualfigaro/files/Doc_and_examples/English/`
[2] `http://sourceforge.net/projects/visualfigaro/`

```
CLASS aa_leaf KIND_OF leaf;
(* This class corresponds to the Attacker Action leaf of Table 1 *)
  CONSTANT
    lambda_S_ND DOMAIN REAL DEFAULT 0.0001; ([...])
    skill DOMAIN '0' '1' '2' '3' '4' DEFAULT '2';
    insider DOMAIN BOOLEAN DEFAULT FALSE;
    cost DOMAIN REAL DEFAULT 0;
  ATTRIBUTE
    flagI DOMAIN BOOLEAN DEFAULT FALSE EDITION NOT VISIBLE;
    flagF DOMAIN BOOLEAN DEFAULT FALSE EDITION NOT VISIBLE;
  FAILURE
    (* This keyword reflects the reliability original orientation of Figaro *)
    aa_success LABEL "success of step %OBJECT" DEFAULT FALSE;
  OCCURRENCE
    success_undetected
      IF required AND (NOT detected) AND (NOT S)AND (relevant_evt OR NOT trimming OF OPTIONS)
      MAY_OCCUR
        FAULT aa_success LABEL "success undetected of step %OBJECT" DIST EXP (lambda_S_ND);
    success_detected
      IF required AND detected AND (NOT S)AND (relevant_evt OR NOT trimming OF OPTIONS)
      MAY_OCCUR
        FAULT aa_success LABEL "success detected of step %OBJECT" DIST EXP (lambda_S_D);
    initial_detection
      IF required AND (NOT detected) AND (NOT (detection='disabled'))
        AND (relevant_evt OR NOT trimming OF OPTIONS) AND (NOT flagI) AND (NOT S)
        AND (FOR_ANY x A triggered_by WE_HAVE final OF x)
        AND (FOR_ANY y A fathers WE_HAVE final OF y)
      MAY_OCCUR
    TRANSITION no_detectionI DIST INS(1-gamma_D_I) INDUCING flagI <-- TRUE
      OR_ELSE
        TRANSITION detectionI LABEL "initial detection of step %OBJECT"
        INDUCING detected <-- TRUE ; ([...])
```

Fig. 3: Excerpts of the Security BDMP knowledge base written in Figaro

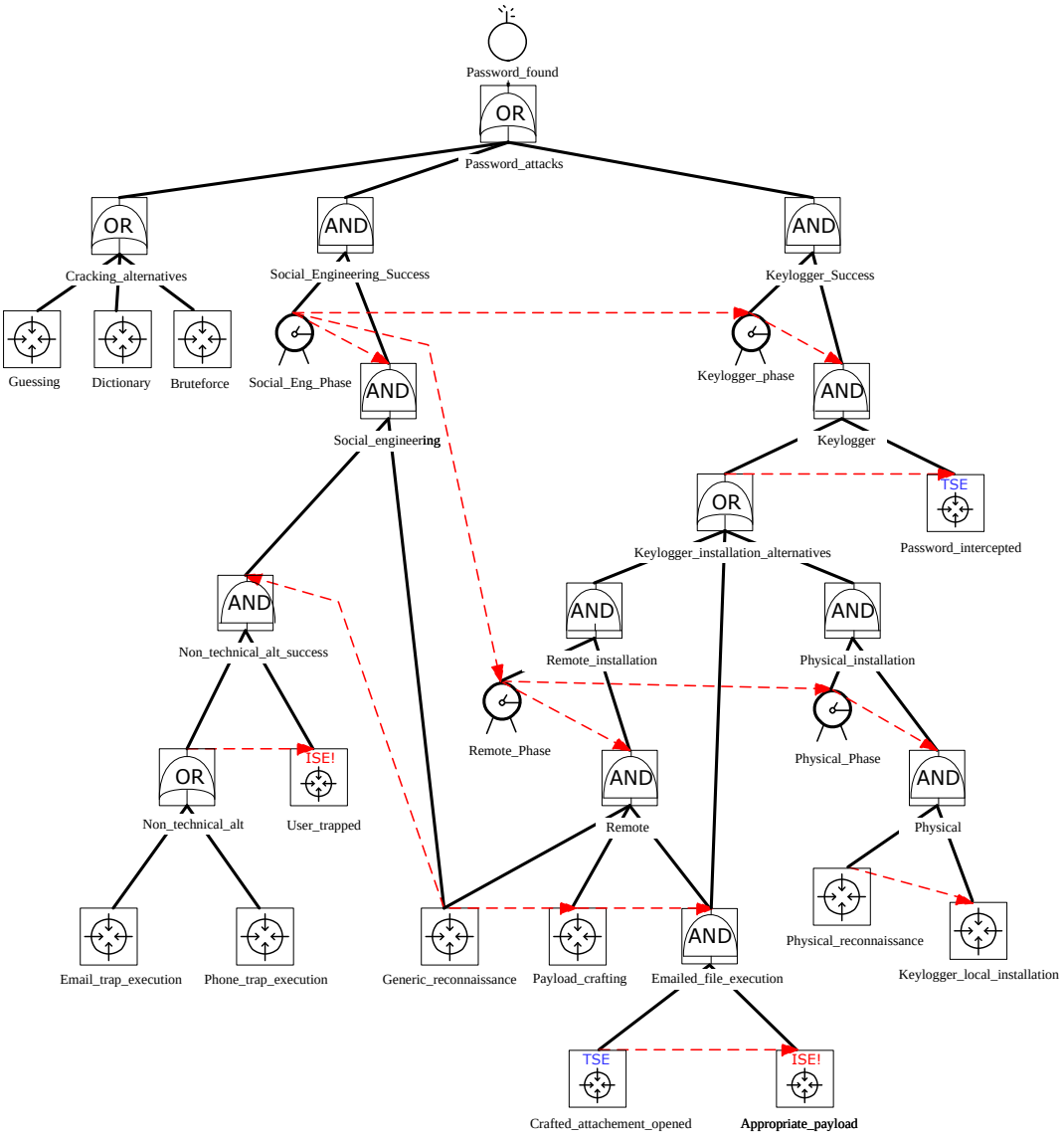


Fig. 4: Attack of a password-protected file (from [PCB10a])

probability. Thanks to the techniques mentioned in Section 2.3, the computation of such results is almost immediate.

Tab. 2: The top four successful attack sequences

	Sequences	Probability in a week	Average duration (h)	Contrib.
1	<Social Eng>Generic reconn., Email trap exec., User trapped	0.11	27.5	25.1%
2	<Social Eng>Generic reconn., Phone trap exec., User trapped	0.05	27.5	12.5%
3	Bruteforce	0.02	15.7	5.1%
4	<Social Eng></Social Eng><Keylogger><Remote></Remote> <Physical> Physical reconn., Keylogger local installation, Password intercepted	0.02	8.3	4.1%

3.3 Recommendations when building and analyzing models

3.3.1 Construction of the model structure.

The hierarchical structure of BDMP is a very powerful characteristic, but it is not always directly leveraged when building the models. Users tend to build directly detailed models whereas macroscopic ones should be preferred in the first stages: they are simpler to validate and can be extended progressively with the desired level of details.

Moreover, another common mistake is to try to turn attack trees directly into BDMP: BDMP models should be built from scratch, as their dynamic features, especially triggers and phase leaves, are specific and cannot be inferred from existing attack trees. Globally, the step-by-step simulation offered by KB3 is extremely useful to check at every stage of the model construction that it correctly captures the intended sequences and dependencies of the attack. On another level, the theoretical framework offers three kinds of security

leaves: AA, ISE and TSE leaves, as described in Tab. 1. In practice, if the use of AA leaves is straightforward, it is not the case for ISE and TSE leaves. Fig. 5 provides a simple example aiming at clarifying the rationale behind the use of each kind of leaf. The attack is based on a malicious email attachement; it is decomposed in Fig. 5 in three steps, each of them involving a specific kind of leaf:

- `Email preparation and sending` is naturally modeled by a AA leaf, as it is a timed and directly attacker-dependant step;
- `Attachement execution by the target` is also a timed step, but its duration does not depend on the attacker, but on the target behavior (does he consult his emails regularly? Is he careful with unknown attachements? etc.). The TSE leaf is useful to model such a situation. This process can begin only after the email has been sent: this dependence is represented by a trigger;
- `Successful payload execution` corresponds to an instantaneous event: when activated (i.e. when the attachement execution completes, thanks to the second trigger), the leaf models the fact that the malicious payload embedded in the attachement has a probability γ to be well suited to the technical environment of the target, and to be executed successfully. In this case, the ISE leaf is used and parameterized with an appropriate γ parameter.

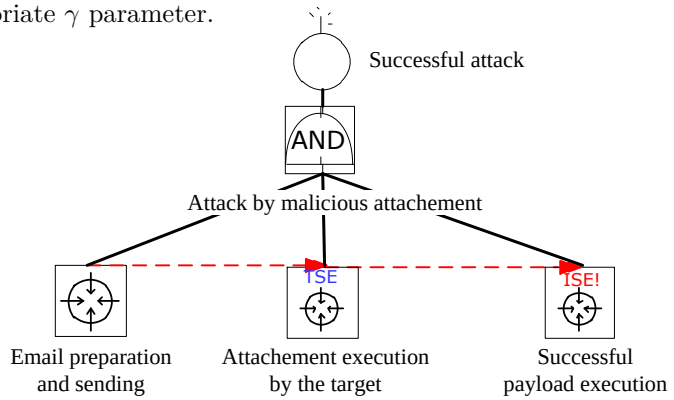


Fig. 5: A simple model to explain AA, TSE and ISE leaves

Finally, as a complement to the former discussion on the different types of leaves, the definition of a library of typical and ready-to-use BDMP security patterns would clearly improve both the appropriation of the formalism semantics, and the construction of the models. It is considered as a major perspective for future work (cf. Section 5).

3.3.2 *Parameter value assignment.*

The definition of the leaf parameter values is a delicate step in a BDMP security study. The main parameters are the λ values for AA and TSE leaves, and the γ values for ISE leaves. The former are assigned by reasoning in terms of mean-time-to-success (MTTS, corresponding to $1/\lambda$ in the Markovian framework of the BDMP), whereas the latter are evaluated directly. In both cases, contrary to safety and dependability studies, there are

generally no historical or statistical data on which such values can be based upon: they have to be based on expert judgments. However, such a limitation does not prevent us from obtaining meaningful results from a qualitative security risk assessment perspective. As a matter of fact, we do not claim that we are able to compute "true" values of sequence probabilities. The assigned numerical values should not be considered for themselves, but only as means for obtaining qualitative results like a hierarchy of the possible attack sequences. Such results can feed in the selection of appropriate security countermeasures.

To facilitate the evaluations of the different parameters, there are several principles that can be followed:

- Attacker's profile should be defined clearly, in particular in terms of resources and skills. It may be sometimes preferable to define several profiles, involving several series of results, rather than trying to characterize a "mean" attacker.

- When possible, inputs from the owners of the attacked components considered in the BDMP model should be obtained for the parameter evaluation. In particular, they should help characterizing the component vulnerabilities and level of protection, which are necessary to evaluate MTTS values, in addition to attacker profiles.

- There is no point in using excessively fine-grained values, as the parameters correspond *in fine* to subjective evaluations, and not to objective figures. For instance, regarding units, hours or days can usually be chosen instead of seconds for MTTS values; one decimal may be sufficient for γ parameters.

Regarding detection and reaction parameters, it is highly recommended to build the BDMP models first without any detection, and add such parameters in an advanced stage of the study. Moreover, the theoretical framework distinguished four types of detection (IOFA, cf. 2.2). However, in practice, new users of the tool should avoid trying to make the most of such a detailed decomposition; they should concentrate mainly on "On-going" and "A posteriori" detections, the most intuitive notions to handle.

Finally, it is also possible not to parameterize the BDMP at all: the models become only graphical representations of attack scenarios, without any quantitative elements. Nevertheless, they still visually capture a set of attack paths in a compact and readable form, which may prove valuable to document security cases and illustrate the coverage of the scenarios taken into account.

3.3.3 Analysis of the results.

The processing of BDMP models leads to overall quantitative values such as global MTTS or probability of success for the attack in a given time, but it also yields the ordered list of all the possible sequences leading to the attacker's objective, with their associated contribution to the overall probability of realization. Such a list is extremely rich in terms of qualitative and quantitative information, but it is also very difficult to exploit because of the high number of sequences to consider, and their uniform presentation. In fact, complementary tools are needed to support the users when analyzing the results obtained from the model treatment: the next section gives an overview of the efforts undertaken by the authors in this area.

4 From Implementation to Practical Tools

4.1 Sequence analysis

A first and major enhancement in the analysis of the produced list of sequences consists in a simple filtering mechanism, inspired by classical attack tree techniques [Sch99]. Each AA leaf is assigned static parameters such as the cost of the modeled action, a skill level needed on a discrete scale or Boolean indicators expressing specific requirements for the action (e.g., internal collaboration). A specific post-treatment tool has been developed to select the relevant sequences depending on attacker profiles defined with respect to the mentioned static parameters: only the sequences compliant with the corresponding thresholds and criteria are kept, which reduces significantly the scope of the analysis.Of course, the final results are still dynamic in nature, they are a selected subset of the overall sequence descriptions and associated time-domain quantifications, out of reach of static models like attack trees.

A second enhancement is related to the presentation of the results: visual conventions, including colors, symbols and font differentiation, have been specified and applied as a post-processing of FigSeq output list of sequences, discriminating the different types of events (success, failure, detection, start or end of phase, etc.). The combination of these two treatments have largely improved the exploitation of the raw ordered list of sequences produced by FigSeq, in a cost-effective manner as the model solver has not been modified.

4.2 Sensitivity Analysis

The main attack sequences, produced by FigSeq and modified by the post-processings, provide good guidance for identifying security optimizations which reduce the probability of success of the attacker. Nevertheless, identifying the most efficient approaches from this perspective, and in particular, which actions/security events should be made harder to realize or easier to detect, may not be obvious as the sequences are characterized without any consideration for their constitutive basic events. Substantial experience in security analysis and a multiple trials-errors approach are needed to get the most of the model and to result into a clear and relevant set of recommendations.

In order to ease such a task, we have developed a script allowing iterative model treatment while incrementing automatically selected leaf parameters. In particular, the evolution of the overall MTTS or probability of success in a given time can be traced with respect to a given parameter, characterizing the model sensitivity with respect to the chosen parameter. Such sensitivity analyses provide valuable support to select or discriminate attacker actions or security events on which the security effort should be concentrated. As an illustration, let's consider the use-case presented in Section 3.2, with the parameters listed in [PCB10b]. Fig. 6 and Fig. 7 allow a visual comparison of the impact of variations of the individual MTTS of the AA leaves on the overall probability of success for the attacker in a week (Note: using the value λ_0/λ, i.e. $\mathrm{MTTS}/\mathrm{MTTS}_0$, with λ_0 the λ parameter value initially chosen, ensures that the comparison is normalized).

Naturally, Fig. 6 shows the paramount importance of the Bruteforce leaf parameter with respect to the other leaves: such an impact was predictable as Bruteforce is in itself an attack success minimal sequence. But once the quality of the password ensured, it is now possible to zoom on the three other leaves, as depicted on Fig. 7. Such a zoom helps to target security efforts between the Keylogger installation leaf, the Payload crafting leaf

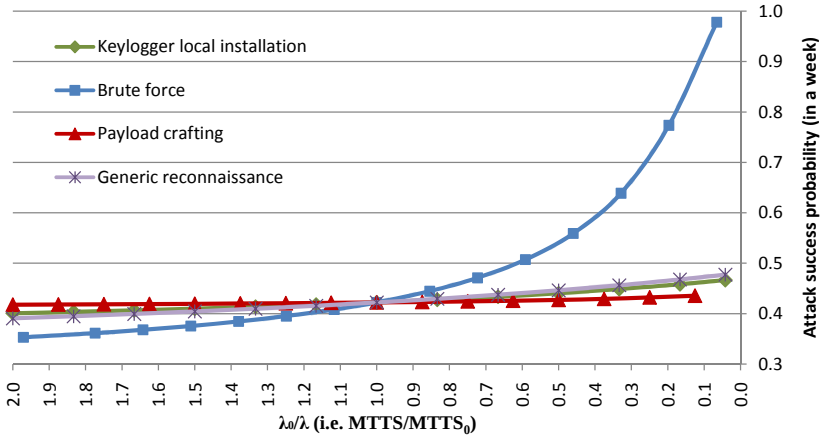


Fig. 6: Comparing the λ parameter impact of the AA leaves

and the Generic reconnaissance leaf. On the basis of this figure, with constrained security resources, efforts should be put on this last aspect rather than on the two others, as the impact on the overall probability of success is stronger.

The same kind of analysis can be made for the ISE leaves as illustrated in Fig. 8. Here, the figure shows that for the studied use-case, it is more effective to limit the probability of a user to be trapped by social engineering than to strengthen the system against malware execution.

5 Main Perspectives and On-going Work

The tools presented in Section 4 provide a valuable help in the analysis of attack scenarios with BDMP. This said, there are still several aspects that can be improved to facilitate model construction, parameterization and exploitation. In this section, we present the two main streams of work in this direction.

5.1 Attack pattern library

As mentioned in Section 3.3.1, the definition of a library of typical and ready-to-use BDMP security patterns could improve the appropriation of the formalism semantics and model construction. In fact, the construction of diverse models has led to identify recurrent patterns, coming back as similar sub-trees in different attack scenarios. A systematic identification and categorization of such patterns could support the creation of a library of small BDMP, modeling classical attack steps and ready to assemble when building complete models. Similar approaches are proposed in attack tree modeling software [Pul05, Ame] and could benefit from the literature on security patterns [sec04, SFBH+06].

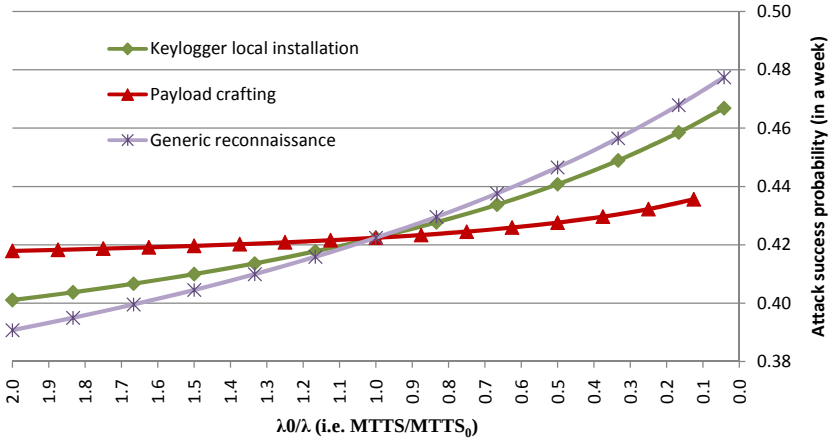


Fig. 7: Zooming on three AA leaves

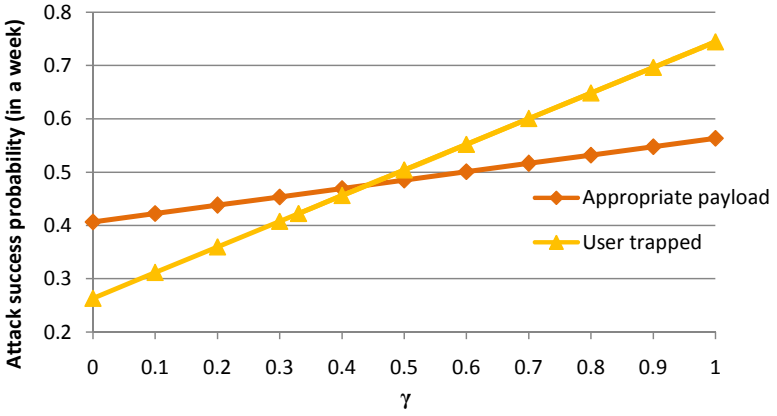


Fig. 8: Comparing the γ parameter impact of the ISE leaves

5.2 Parameter values and uncertainty handling

There are several tracks to investigate regarding the difficulties encountered in the attribution of the parameter values pointed in Section 3.3.2. A first perspective is to associate to the security BDMP patterns of the previous section, a selection of default values for their parameters, corresponding to predefined attacker profiles and levels of vulnerability. A second perspective is to integrate classical uncertainty handling approaches. For in-

stance, parameter values could be considered as random variables instead of crisp values: in that case, we would use Monte-Carlo simulation [MZ02] to assess the effect of uncertainty on both the global results and the ordering of sequences. Fuzzy sets could also be introduced [TFLT83], but the impact on the underlying model theory and on the end-user parameterization experience is still to be evaluated. Finally, a hybridization with other modeling frameworks such as Bayesian networks [SEN09] is presently considered: they are well adapted to capture incomplete and subjective evaluations, which can be refined by different sources and expert opinions.

6 Conclusion

The software implementation of the BDMP adaptation to attack modeling enables model construction, parameterization and processing. Thanks to the existence of the reliable and highly customizable modeling tool KB3 (originally developed for dependability studies), it has been possible to develop very rapidly the equivalent of a tool dedicated to security analysis, that would have had a considerable cost if it had been developed from scratch. The adopted development strategy avoided modifications of the main tools themselves (KB3 for building models, FigSeq for solving them) but could significantly enhance the available processings by the addition of facilities such as: filtering of sequences according to qualitative criteria (cost of attacker actions, needed skill level...), determination of minimal sequences, sensitivity analysis. The first concrete uses of these models have shown that the major remaining challenges in this approach are the parameter value assignment phase and the modeling of the related uncertainty. We intend to keep on making progress on BDMP security modeling following what has proved to be a virtuous circle: theoretical development - software implementation - concrete practice and experience; this approach has pointed out unexpected difficulties, tackled by new developments, and has led to identify new challenging theoretical and software extension perspectives.

References

[Ame] Amenaza. SecureITree. Amenaza website: `http://www.amenaza.com`.

[BB03] Marc Bouissou and Jean-Louis Bon. A new formalism that combines advantages of fault-trees and Markov models: Boolean logic driven Markov processes. *Reliability Engineering & System Safety*, 82(2):149–163, November 2003.

[BBBV91] M. Bouissou, H. Bouhadana, M. Bannelier, and N. Villatte. Knowledge modelling and reliability processing: Presentation of the FIGARO language and associated tools. In *Proceedings of SAFECOMP'91*, pages 69–75, Trondheim, Norway, November 1991.

[BL02] Marc Bouissou and Yannick Lefebvre. A path-based algorithm to evaluate asymptotic unavailability for large Markov models. In *Proceedings of the 48th Reliability and Maintainability Annual Symposium (RAMS'02)*, pages 32–39, Seattle, USA, 2002.

[Bou05] Marc Bouissou. Automated dependability analysis of complex systems with the KB3 workbench: the experience of EDF R&D. In *Proc. Int. Conf. on Energy and Environment (CIEM'05)*, Bucharest, Romania, October 2005.

[Edg07] Kenneth S. Edge. *A Framework for Analyzing and Mitigating the Vulnerabilities of Complex Systems via Attack and Protection Trees.* PhD thesis, Air Force Institute of Technology, July 2007.

[GPV99] M. Gallois, M. Pillière, and N. Villatte. Benfits expected from automatic studies with KB3 in PSAs at EDF. In *Proceedings of the International Topical Meeting on Probabilistic Safety Assessment (PSA'99)*, Washington, D.C., USA, August 1999.

[Har90] P. Harrison. Laplace transform inversion and passage time distributions in Markov processes. *Journal of Applied Probability*, 27(1):74–87, 1990.

[LM05] Yuan Liu and Hong Man. Network vulnerability assessment using Bayesian networks. In *Proceedings of SPIE Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security 2005*, volume 5812, pages 61–71, Orlando, FL, USA, March 2005.

[McD00] J. P. McDermott. Attack net penetration testing. In *Proceedings of the 2000 Workshop on New Security Paradigms (NSPW'00)*, pages 15–21, Cork, Ireland, September 2000.

[MO05] Sjouke Mauw and Martijn Oostdijk. Foundations of attack trees. In *Proceedings of the 8th International Conference on Information Security and Cryptology (ICISC'05)*, pages 186–198, Korea, December 2005.

[MZ02] Marzio Marseguerra and Enrico Zio. Basics of the Monte-Carlo method with application to system reliability. LiLoLe Publishing, Hagen, Germany, 2002.

[PCB10a] Ludovic Piètre-Cambacédès and Marc Bouissou. Attack and defense dynamic modeling with BDMP. In *Proceedings of the 5th International Conference on Mathematical Methods, Models, and Architectures for Computer Networks Security (MMM-ACNS-2010), LNCS 6258*, pages 86–101, St Petersburg, Russia, September 2010.

[PCB10b] Ludovic Piètre-Cambacédès and Marc Bouissou. Attack and defense dynamic modeling with BDMP (extended version). Tech. Report 2010D021, Telecom ParisTech, 2010.

[PCB10c] Ludovic Piètre-Cambacédès and Marc Bouissou. Beyond attack trees: dynamic security modeling with Boolean logic Driven Markov Processes (BDMP). In *Proceedings of the 8th European Dependable Computing Conference (EDCC-8)*, pages 199–208, Valencia, Spain, April 2010.

[PCPD04] J. Pestourie, P. Carer, P. Pamphile, and Cazaban D. Reliability and availability evaluation of cokeworks electrical network in a steel manufacturer site. In *International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pages 301–306, Ames, IA, USA, September 2004.

[PML10] Srdjan Pudar, G. Manimaran, and Chen-Ching Liu. PENET: a practical method and tool for integrated modeling of security attacks and countermeasures. *Computers & Security*, 28(8):754–771, May 2010.

[Pul05] Richard Pullen. Attacktree+, a computer tool for modeling attack scenarios. In *Proc. 29th ESReDA Seminar on Application of System Analysis and RAMS to Security of Complex System*, pages 333–343, Ispra, Italy, October 2005.

[Sch99] Bruce Schneier. Attack trees. *Dr. Dobb's Journal*, 12(24):21–29, 1999.

[sec04] Security design patterns: Open group technical guide. Open Group Security Forum, `http://www.opengroup.org/security/gsp.htm`, 2004.

[SEN09] Teodor Sommestad, Mathias Ekstedt, and Lars Nordström. Modeling security of power communication systems using defense graphs and influence diagrams. *IEEE Transactions on Power Delivery*, 24(4):1801–1808, October 2009.

[SFBH+06] Markus Schumacher, Eduardo Fernandez-Buglioni, Duane Hybertson, Frank Buschmann, and Peter Sommerlad. *Security Patterns: Integrating Security and Systems Engineering.* John Wiley & Sons, 2006.

[TFLT83] H. Tanaka, L.T. Fan, F.S. Lai, and K. Toguchi. Fault-tree analysis by fuzzy probability. *IEEE Transactions on Reliability*, 32(5):453–457, 1983.