Global Information Assurance Certification Paper

Framework for Innovative Security Decisions

GIAC (GCED) Gold Certification

Author: Ergash Karshiev, ergozn1@gmail.com

Advisor: Barbara L. Filkins

Accepted: September 29, 2015

Abstract

Remember the Periodic Table of chemical elements (Dayah, Dynamic Periodic Table, 1997)? It revolutionized chemistry and continues serving scientists daily. TRIZ is a similar resource for inventors and decision-makers. It is a Russian abbreviation translated as: "theory of inventive problem solving." It does so by consolidating common patterns of prior innovations and by solving contradictions. The goal of this paper is to combine principles of TRIZ, Science, and innovative thinking to propose its Information Security version: **Sec-TRIZ**. It is intended to support optimized decisions in security management and technology. Why is Sec-TRIZ a "secret weapon" of security innovations? Because the original TRIZ has done it in *material innovations* for decades. It is taught and applied in top innovative companies (Jana, 2006): Boeing, HP, IBM, Motorola, GE and many more. The provided Sec-TRIZ Framework is the beginning of the new innovative thinking research that will benefit readers of technical and management background.

1. Introduction

Information security is a highly innovative and fast-developing field. The growing need for better cyber security drives the investment into innovative approaches to solve both existing and new challenges. It is often said that "security is not a product, but a process" (Schneier, 2000). Therefore, daily security decisions (being part of the process) actually enable (or disable) businesses. *Specialized* security decision frameworks already exist in specific fields like mobile security decisions (US_CIO_Council, 2013) or cloud security decisions (Heiser, 2013). There is a need for a more *general* framework to aid in the complex multi-disciplined field of Information Security. Thus, this paper presents Sec-TRIZ with examples that support:

1. Tactical and strategic business decisions in security,
2. Designing macro- and micro-level security solutions for companies and vendors (from architectures to code and implementations),
3. Anticipating the future trends of the attacks and preparing defenses in advance.

The paper presents common patterns and classifications along with approaches for finding the "missing cells" in innovative security decisions space. The 40 principles of TRIZ are translated into Information Security terms. To prove the value of the new Sec-TRIZ framework, sample security decisions and proposals are presented.

That is a research effort itself perfect opportunity to show TRIZ in action: since it is designed to solve innovative contradictions. Therefore, the author did apply various TRIZ/Sec-TRIZ method to the research and writing of this paper. The solutions directly applied here are:

1. Use of visual tools to explain more (Sec-TRIZ methods: 17, 26, 24, 28, 32 – see *Section 4.2*)
2. Grouping challenges for mutual compensation (Sec-TRIZ methods: 8, 12, 13, and 22). The author did not have time and resources to find 40 images to illustrate the Sec-TRIZ methods, so instead one basic image (two overlapping

windows) was processed to make 40 illustrations. See *Appendix A* for the explanation of these methods.

TRIZ is not an automatic and complete problem-solver, it is a framework to guide innovators in more likely directions. Its creator, Genrich Altshuller, reviewed over 40,000 of innovation patents (San, 2009) to uncover re-occurring guiding principles. He also noticed that many prior innovations took years of blind trial-and-errors while having such framework in advance would eliminate those years of searching. The two key principles of TRIZ are:

1. Somebody has solved it already (find it)
2. Don't accept contradictions (solve them).

TRIZ was born out of analyzing very many *material* innovations (of physical products and tools). It is also written in the language of *material* innovations and science (Physics and Chemistry). Then how does it apply to cyber security? Fortunately, many of these principles can be translated into the terms of business management, computer science, algorithms, and more. An additional goal of this paper is to demonstrate this process of translating the TRIZ principles into actionable business and technology recommendations to improve security decisions.

One key question innovators often hear is: "OK, this a great idea. How can I put it to solve a specific problem with minimal money and labor?" Well, this is yet another *contradiction* (opposing requirements) that Sec-TRIZ is designed to solve. Not only will this process guide the reader to produce innovative proposals, it will also guide on solving implementation, management and financing challenges.

If this is true, why hasn't TRIZ "take over the World" yet? Because inventing is always hard and TRIZ takes learning too, though learning can be rewarding with powerful time-saving tools TRIZ provides. While being very successful in material innovations only in recent years business innovators started using TRIZ (Souchkov, 2014). Information Security practitioners are not yet widely applying TRIZ methodologies (nor software developers overall). One of the goals of this paper to bring

awareness of the power of TRIZ to the security community via the customized and translated adaptation of it: Sec-TRIZ.

2. Specifics of Security Decisions

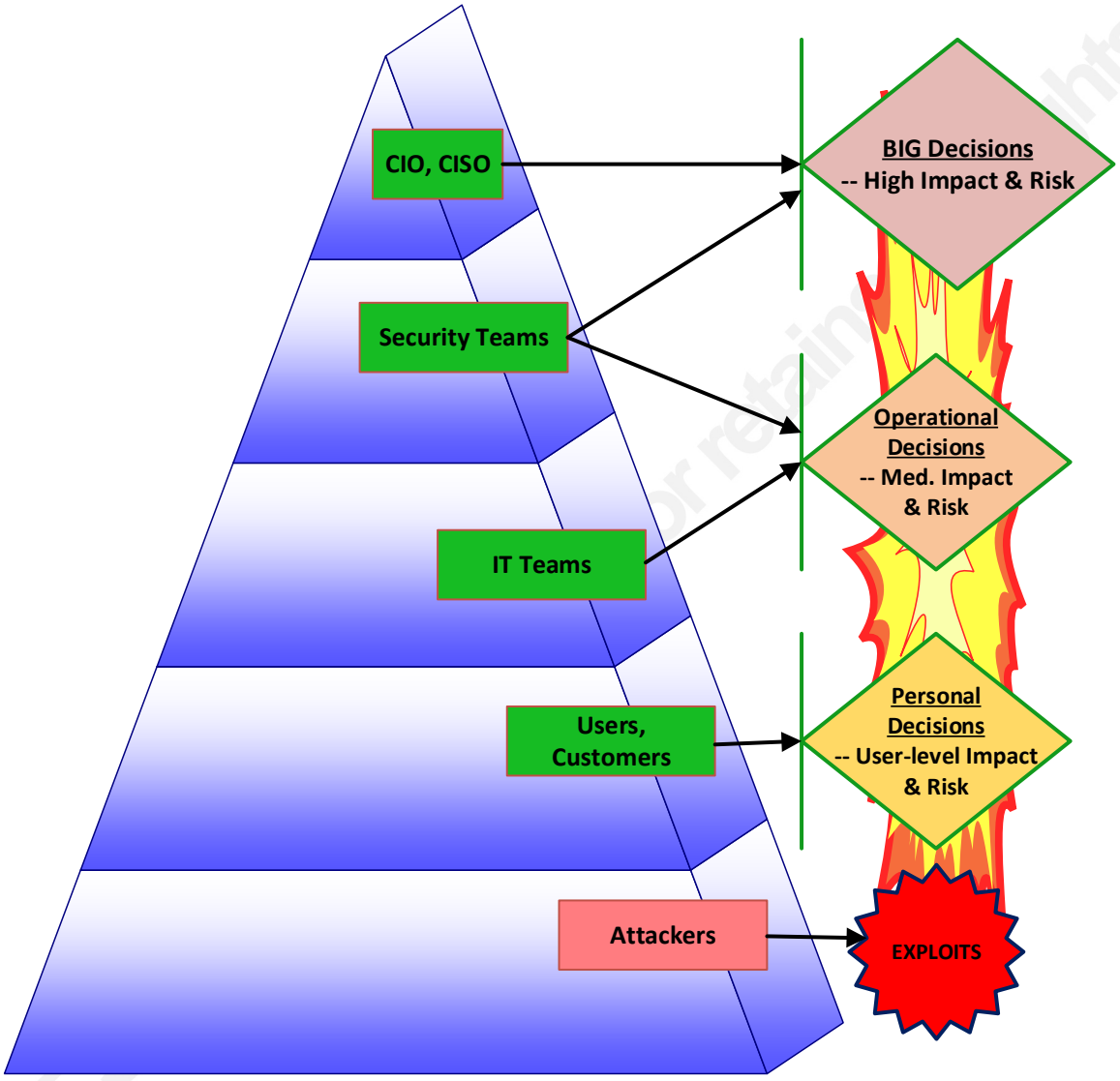
Decision-makers in charge of cyber security would agree on the complexity and diversity of this responsibility. Security settings and tools are often of a restrictive nature, so they *could* limit functionality of many users and devices while creating secondary unexpected effects as well. This is yet another *contradiction* to solve: enabling businesses by restricting them. Of course, security teams enable by preventing/remediating cyber-attacks, but the attacks are often invisible or misunderstood by the paying customers (the contradiction of financing security).

To add to this complexity and contradictions: an attacker can amplify any mistakes, exploit inputs and code weaknesses by optimally combining them. Complexity of modern systems increasingly adds two advantages to attackers:

1. Higher likelihood of bugs,
2. More opportunities for unexpected use (hacking).

Therefore, security decisions carry high impact, risks and difficulties: they can benefit from a solid framework used for decades to solve contradictions (TRIZ/Sec-TRIZ). The following diagram illustrates these challenges and a need for non-linear solutions (innovative shortcuts) to mitigate these non-linear impacts.

Security Decisions & Risks Pyramid:
f(Decision) = *Non-linear* Impact



Ordinary decisions *can* have big impact:
Due to the amplification by attackers

Ergash Karshiev, ergozn1@gmail.com

3. TRIZ for Security: Sec-TRIZ

Now it is time to start exploring Sec-TRIZ and its use in Information Security decisions. The following two sections explain the *Sec-TRIZ Decision Flow,* the process, and the *Sec-TRIZ Matrix – Navigation Map,* the summarized core of the Sec-TRIZ methods.

3.1. Sec-TRIZ Decision Flow

The diagrams on the following pages will demonstrate the *Sec-TRIZ Decision Flow* for technical and business solutions.

Side Note: *The following discussions and diagrams are written in second person style by design. This is intended to directly inspire new thoughts in a friendly conversation format.*

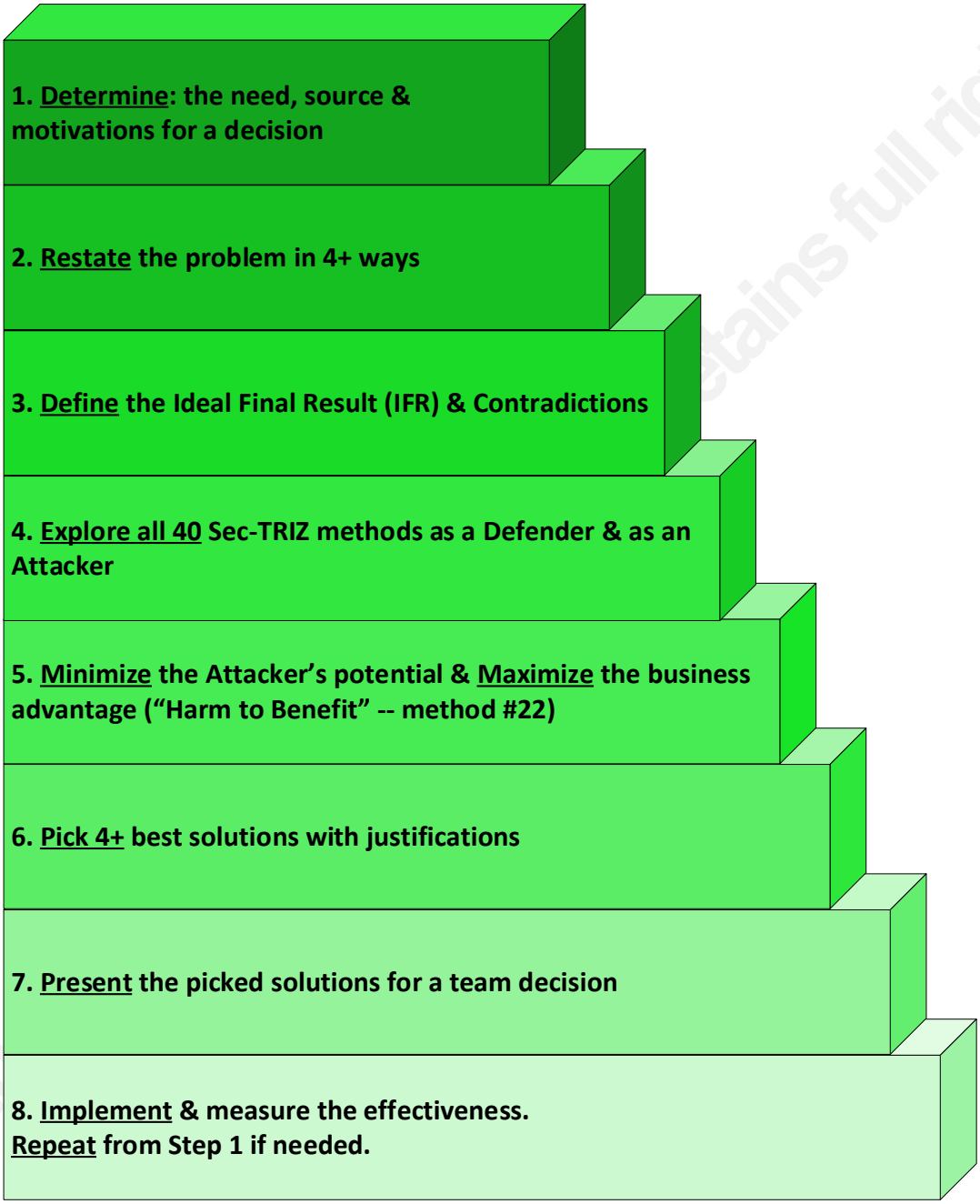
The following diagram shows the recommended steps on using the Sec-TRIZ methodology. The readers can try applying and modifying this process as part of the creative exercise of learning this framework. Here are the *key points* supporting the following process diagram:

- **Formulate:** Define the problem in writing: describe it in several ways, after explaining it verbally to somebody outside of your specialty.
- **Try All 40:** Consult the *Navigation Map* for the most closely matching *Sec-TRIZ Methods*.
- **Pick & Pilot:** Review the given problem through the lenses of the selected methods and repeat the process with other remaining methods. Pick several best solutions and pilot them.

It is important to try all methods (especially the seemingly *unlikely* ones). Writing thoughts with accompanying diagrams for each is an excellent way to learn the framework of Sec-TRIZ and to build your intuition. The future cycles of finding solutions will go increasingly faster.

Ergash Karshiev, ergozn1@gmail.com

Sec-TRIZ Decision Flow:
Recommended Approach



Ergash Karshiev, ergozn1@gmail.com

3.2. The Sec-TRIZ Matrix (Navigation Map)

The following *Sec-TRIZ Matrix (Navigation Map)* is the condensed version of the 40 methods, the core of this framework. The 4-by-10 grouping allows to easily map these methods to their detailed illustrations as shown in *Appendix A***.** Each diagram (on a separate page) covers 4 methods, corresponding to a line in the following Sec-TRIZ Matrix.

Use these examples as an overview of the methods you can explore for a given problem. It is useful to consider all 40 for each problem. The details on these methods are on the 10 pages of the *Appendix A*.

The wording of the Sec-TRIZ methods is very condensed (to preserve your thought patterns), so use it for your imaginative modelling and not as exact instructions. It is safe to skip any examples that don't immediately resonate with you. The goal of these building blocks is to give you visual options, so they are not "official terms". You may think of them as Zen-level cues. In fact, you would be more productive if your thoughts detach from the actual words, perceiving them instead as visual interacting models (i.e., moving and evolving pictures).

Sec-TRIZ Matrix: Navigation Map

1. Segmentation: Partition and Isolate to create: Zones, Layers, Onions *(increase degree of segmentation)*	**2. Extraction:** Filter, Distinguish, Squeeze out to Select and Retrieve *(most useful first, catch & amplify)*	**3. Local Quality:** Increase diversity for optimal specialized use *(natural differences of parts used ideally)*	**4. Asymmetry:** Shift from symmetry to higher degrees of asymmetry *(Custom Symmetries, Shifted Balance)*
5. Consolidation: Merge similar items in Space and in Time *(do it once at once, handle many with one, similar stay near)*	**6. Universality:** Do All-in-One for waste of none ***(max** # of useful functions to **min** # of objects to handle)*	**7. Nesting:** Embedded Smart Layers inside and throughout *(multi-shields + multi-tools, all here)*	**8. Counterweight:** Ride/join compensating forces by combining the opposites *(balance by clashing opposites)*
9. Prior Counteraction: Preloaded Energy = canned potential *(Jack-in-the-Box for every "surprise")*	**10. Prior Action:** Prepared Objects = ready NOW! *(zero-wait action from the best spot)*	**11. Cushion in Advance:** Pre-cushioned landing = soft pillow *(always carry parachute and padding)*	**12. Equipotentiality:** Level Naturally for self-balancing *(not too hot, not too cold)*
13. Do It in Reverse: Moving in reverse + around the problem *(flip upside-down, switch the static & moving parts)*	**14. Spheroidality:** Bend flats and squares into curves, spirals and balls *(use smooth rotation)*	**15. Dynamicity:** Mobility for flexible transformation of roles *(make all parts movable & play with the moves)*	**16. Partial/Excessive Action:** Good Enough is better than Perfect *(sufficient conditions that do the job)*
17. New Dimension: The other sides: add factors, facets & levels *(all-dimensional opportunities & degrees of freedom)*	**18. Vibrations & Waves:** Oscillate, change frequency and resonate *(change the nature of waves)*	**19. Periodic Action:** Impulse/ Pulse to Loop, Pause + Charge, Burst *(Continuous to Periodic: from drill to jack hammer)*	**20. Smooth Useful Action:** Being useful non-stop & all the way *(fully and continuously utilized capabilities)*
21. Rushing Through: Danger Zone quickly shrinks to Zero *(minimize risky exposures)*	**22. Harm to Benefit:** Inverted Evil for Good *("Harm" is the unused side of future Good)*	**23. Feedback:** Back-Feed – amplify yourself to create signal *(send Output back to Input to Generate)*	**24. Mediator:** X-in-the-Middle – the Super-Brokers *(the magic renewable Catalyst)*
25. Self-Service: Self-healing + turning waste to service *(cure from within with surplus energy & materials)*	**26. Copying:** Copy-Cat + Projections – easy replicas to alter & resize *(everything leaves a projected trail)*	**27. Dispose:** Replace-Ability + Free-Star – gold bare bone *(Trade without Trade-Offs)*	**28. Virtual Replacement:** Ghosts at Work – ideal natural and virtual emulations *(new twisted "Super-Model")*
29. Solid to Fluid/Gas: Shape-Less + Flow-More to transmit force & to flow around obstacles *(Zen: "Be like Water")*	**30. Flex. Membranes/Thin Films:** Un-Stiff to flex, filter & isolate *(nano-films for macro-separation)*	**31. Porous Structure:** Intake + Absorb to add useful emptiness *(expand while remaining the same, pre-fill the pores)*	**32. Color Change:** Colorize + Polarize to stand out *(Change: spectrum, transparency and glow)*
33. Homogeneity: Blend-in + Be-One – tools = resources *(use itself for pollution-free interactions)*	**34. Reject-Regenerate:** Ash back to Cash – remove to re-create *(continuously discarded and cast back into shape)*	**35. Transform Properties:** Trans-Form – move to another state of being *(Chameleon of shapes & configurations)*	**36. Phase Transition:** Phase-Zen – "Freeze-Melt-Evaporate" for desired effects *(ride natural states)*
37. Expansion: Riding Internal Forces – the environment dictates capacity & features *(harmonize the with World)*	**38. Fast Chemistry:** Art of Burning – Physics & Chemistry to challenge digital hacking *(watching from the outside)*	**39. Inert Environment:** Not-Reacting – skill of not responding *(neutrality for balanced service)*	**40. Composite Structure:** Optimal-Mix – many flavors: for new features *(sum is different than its parts)*

Ergash Karshiev, ergozn1@gmail.com

4. Example of Sec-TRIZ use

Now it is time to demonstrate how this methodology can be used via a detailed example. Sec-TRIZ can solve contradictions by re-wiring opposing factors. There are new and rare examples of applying TRIZ variations for innovative *information security* solutions, such as password and physical security (Fulbright R. , 2010). TRIZ patterns in LDAP innovations (based on US patents) are described in paper Inventions on LDAP Security (Mishra, 2006). These TRIZ use cases inspired the following detailed examples.

4.1. Detailed Example-1: Executable Data

Data security is the core of Information Security efforts. Data is treated as a passive element (being worked on). Why not the empower data with its own capabilities to authenticate, authorize, and to defend itself? This is the idea of the *Executable Data* example.

Detailed example of Sec-TRIZ use #1:

Executable Data
(security inside)

Needs:

- Data needs to defend, manage and expire itself.
- Data needs awareness of itself and of its users/attackers.
- Data should have many levels. It should phone home.

Contradictions:

- Data is NOT a program to act.
- Data need to authenticate its users independently.
- Awareness of its use/misuse built within.

Run to Read (Data)

Architecture	Logic
1. Self-encrypted executable file. 2. The decryption key is stored outside: it needs to phone home to authenticate, to report and to access the key. 3. It decrypts only the authorized portions per specific user. 4. It fakes/corrupts the data and self-destroys with signs of attack. It tires to phone home.	1. It treats every access as a potential intrusion (until proven otherwise). 2. As a software running in hostile environments is has hidden law enforcement capabilities. 3. It is designed to be undesirable to steal/ break. 4. it is completely safe (and verifiable) for authorized users.
Authentication	**Defenses**
1. It contains internal encrypted credentials of authorized users (for offline mode), but phones home for validation by default. 2. It detects password guessing and self-destroys at pre-defined number of attempts. 3. It tracks who accessed it and when (and reports home). 4. It scans itself for malware using trusted cloud services. It protects self-integrity.	Exe-DATA: 1. Runs before you can **Read it**, 2. Activates internal security **Defenses**, 3. Reports its use by **Phoning Home**, 4. **Self-destroys** upon expiration or misuse.

Ergash Karshiev, ergozn1@gmail.com

4.2. Detailed Example-2: Users ARE Software

The revolution in software and computing is continuing at accelerating pace. The user capabilities and software skills grow much slower. This example provides steps to bridge that gap and to bring users and their software closer:

- By modeling users in software,
- By software modeling the users' thinking.

Detailed example of Sec-TRIZ use #2:

Users ARE Software

(AI and I)

Needs:

- People and virtualized services are equally flexible.
- People think and work at the speed of software.
- Human talent supplements the software.

Contradictions:

- Competition vs. Resistance to Change.
- Exponential computing power vs. the same people.
- Need to be AI to manage AI.

Security Software-self

Shared Mind + RAM

1. Software sees your Mind – and you see the states of Software (in your Mind): mutual transparency.
2. Software sees your struggles and automatically removes them: re-explains.
3. Software models your brain, you practice the logic of software: you meet in the middle.
4. Offload part of your thinking and memory to RAM.

Upgradable Thinking

1. Thinking Hats: motivate you to deeply focus on specific subjects (and switch).
2. Emotional attachment to Software to know it better (game, relationship).
3. Speak like Software: run and verbally comment the commands, responses and results.
4. Deconstruct leading security software to learn its wisdom.

Working in a Security Game-space

1. Learning security games in virtual worlds: slowly replaced by real jobs in virtual space.
2. Treating the pressure and stress of work as a game.
3. Layered Games-in-Depth to avoid critical production failures (team support).
4. Mixing Drills, Work and Games for consistent great results and for quality assurance.

Codified Security Workers

I am Software:

1. My work is **Recorded**,
2. All my steps are **Coded**,
3. I help to debug and **Perfect that Code**,
4. Software and I become that **AI**.

Ergash Karshiev, ergozn1@gmail.com

4.3. Detailed Example-3: Security Manager of a Cloud-Hosted Team

This example provides steps for effectively managing security teams working in a cloud-hosted environment. There are parallels with innovative management styles and functioning of security products.

Detailed example of Sec-TRIZ use #3:

Security Manager in the Cloud

(managing cloud-hosted security business)

Questions:
* What is there to manage?
* Where are my People and Data?
* How do I protect and advance?

Challenges:
* You are a guest here: follow the rules.
* How to experiment & invent here?
* Everything is observed: little room for errors.

Winning in the Cloud

Collaboration (101*100)

1. Study workflow, link people and enable self-networking.
2. Promote the *Chain Reaction* effect: reward group work (easy to observe in the Cloud).
3. Rotate functions and roles (for cross-training and resilience).
4. Share the profits from savings of new Cloud efficiencies with the staff.

Measure *Everything*

1. Enable cost-efficient metrics collection and storage.
2. Save key metrics (and potential ones) with maximum precision.
3. Increase the rate of sampling (to measure micro-fluctuations).
4. Measure seemingly unrelated useful metrics together (with precise timing).

Global Analytics

1. Utilize advanced Data Analytics and use the results to feed new and better metrics.
2. Compare yourself/your team against other Cloud tenants and global competitors/ partners.
3. Question and challenge the statistics.
4. Find correlations and manage them for better outcomes.

Manager Super-Star

Manager in the Cloud is:

1. The Group Mind's **Catalyst**,
2. Team work **Router**,
3. Workflow **Switch**,
4. Guardian **Firewall & IPS**.

Ergash Karshiev, ergozn1@gmail.com

4.4. Case Study: Cloud First, Cloud Now – Securely

The *Cloud First* strategy is a new innovative strategy adopted by Microsoft (Endler, 2014) and others: it includes "ubiquitous computing" – flexible/scalable network services available anywhere. Therefore, a company makes a strategic decision: all new solutions should be cloud-based. The given company has a need for an urgent *secure* solution as the complex and slow transition to the cloud has created a backlog of projects. The steps corresponding to the *Sec-TRIZ Decision Flow* of the Section 3.1 follow.

Step 1-3: Formulate the problem: New distributed external approach to running IT business. There are risks of: delays, network (availability & latency) dependencies, sensitive data exposures and unverifiable vendor claims. By explaining the problem to others, new details/options have emerged:

a. It can be a private cloud (hosted in-house)

b. It can be a hybrid cloud (distributed between the vendor and the in-house clouds)

c. Security controls should be embedded in new ways (*read further*).

The following hidden contradictions (designated as C# where # is a number) surfaced:

a. **C1:** Cloud vendors promise better security, but don't provide visibility and transparency into their services (especially during security incidents: self-incriminating actions).

b. **C2:** Not having control of the vendor cloud, it is hard to implement security controls.

c. **C3:** Knowing who accesses the data is critical. The data does not "know" when it is being stolen or misused.

Step 4-5: Use all 40 Sec-TRIZ methods: Scanning the Navigation Map these methods showed promise: 13. Do it in Reverse, 15. Dynamicity, 17. New Dimension, 22. Harm to Benefit, 24. Mediator. In the next iteration phase these were added: 25. Self-Service, 26. Copying, 27. Dispose, 28. Virtual Replacement. Let's review the resulting possible solution and pick the best.

Step 6-8: Pick and pilot solutions: Our selections with corresponding solutions are:

13. Do it in Reverse: You don't have to copy sensitive data to the vendor cloud, you can bring the remote cloud in-house: borrow their virtual CPUs/services (via a fast remote network) to work in a cloud you own.

15. Dynamicity: Cloud allows for mobility and savings in other areas via cloud-cloud integrations. Use these savings to fund more complex migration projects.

17. New Dimension: Turn it on a side: Let's buy this cloud vendor (or form a joint venture). This way you can have complete security visibility.

22. Harm to Benefit: You could host less sensitive security projects in the cloud: Honeypots, Open Source Intelligence gathering (OSINT) and security labs for training.

24. Mediator: Let's pay another (specialized) company to do the cloud migration and integration for us. Let's use their infrastructure in the middle as another layer of security and a legal buffer.

25. Self-Service: You can reduce the costs by using the Cloud vendor's self-service feature and save once more.

26. Copying: Let's not copy sensitive data (probably 2% total), but everything else. Therefore, 98% of the data enjoys the cloud benefits without high risk. This is the "more useful cheap copy" of Sec-TRIZ.

27. Dispose: Let's destroy and recreate the virtual machines in the cloud every 5 minutes (while the data gracefully fails over). The estimate is that it is not enough time for attackers to exploit and take data from the virtual machine.

28. Virtual Replacement: Let's add layers of virtualization to add security and specialization. Attackers would have to escape/jump these layers. The total number and design of layer is unknown to most people in the company.

Now, specifically addressing the identified contradictions via combined solutions:

a. **C1:** Visibility and transparency: Build your cloud inside of the vendor's cloud via an embedded layer of virtualization: anything you created inside you completely control, log and see. Treat the vendor environment as "hostile" by default (Methods: 7, 17, 28, 30).

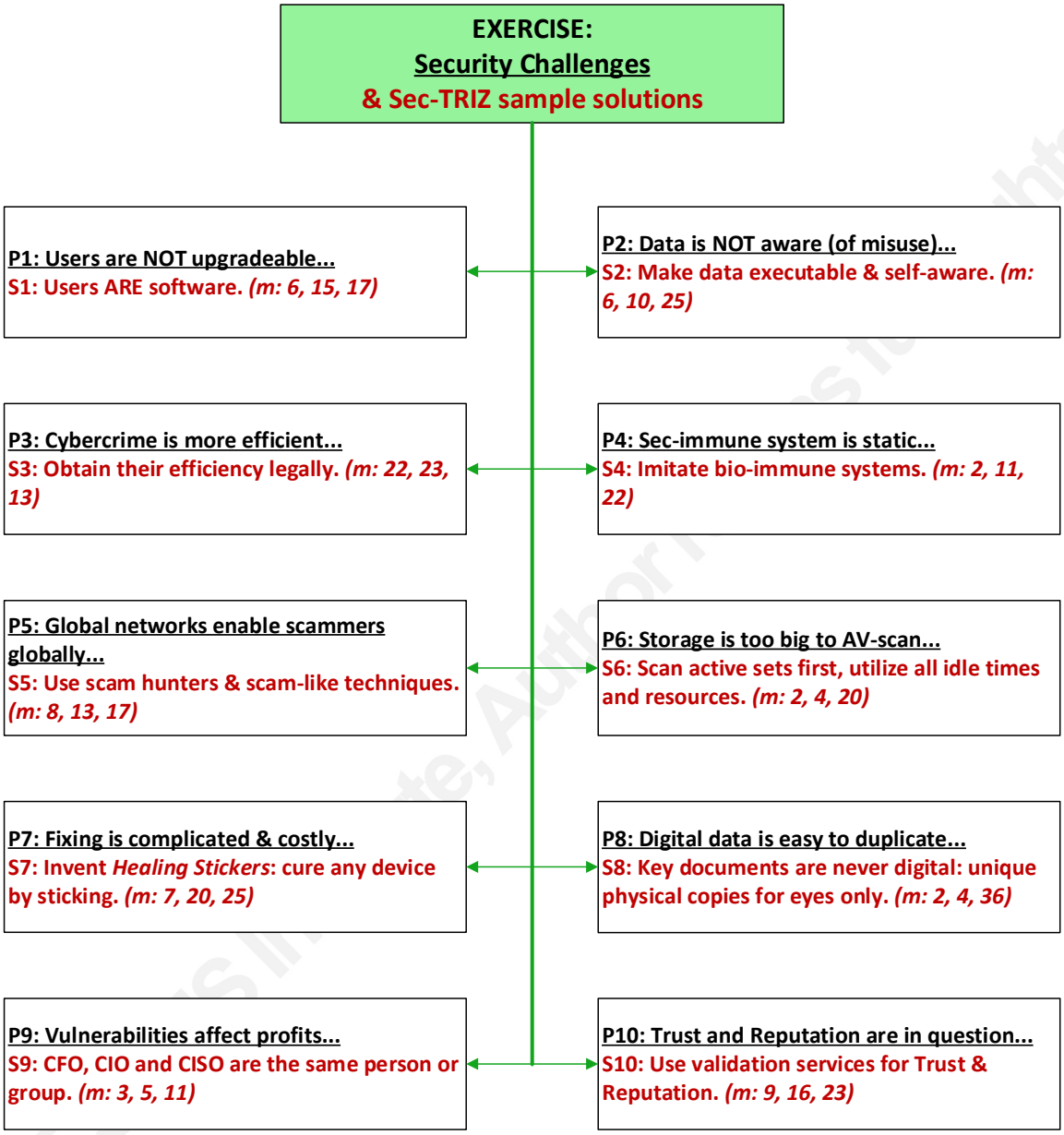
b. **C2:** Security controls: Controls are inseparable from applications and data: they are embedded invisibly. Nothing outside of the controls is trusted. Nothing is touched directly, but via a protective layer/field. Any "cheating" by vendors or attackers is flagged (Methods: 7, 24, 25, 29, 30).

c. **C3:** The data does not "know": Make data executable and self-aware: it can defend itself, authenticate autonomously, phone home (to report status) and self-destroy. The data runs elements of Artificial Intelligence, Machine Learning and its company "DNA": a biological-like code for secure interactions, evolution and immune-system-like defenses (Methods: 9, 10, 11, 23).

Therefore, this example show the capabilities of Sec-TRIZ in generating new ideas and picking optimized solutions for Information Security decisions. This is because the original TRIZ already pre-analyzed and grouped such patterns extracted from many thousands of successful inventions.

4.5. Innovation Exercise

Here are the brief examples of solving security challenges in the form of innovation exercises. Readers are encouraged to review them in order to find the correlations with the listed Sec-TRIZ methods.



Ergash Karshiev, ergozn1@gmail.com

5. Conclusion

Hopefully, you have enjoyed this quick journey into the new paradoxical way of innovative thinking coined as Sec-TRIZ. The author's role was to translate the highly successful TRIZ framework into the field of Information Security and to demonstrate its usefulness via examples. Hopefully by now you agree on the following:

1. Sec-TRIZ thinking methodology is universally applicable to many fields of Information Security.
2. It is a combination of Art and Science as opposed to a set of strict instructions: more creative control and more thinking time.
3. It is a new approach in Information Security and much more research is needed.

Further works is needed to:

1. Translate the *TRIZ Contradiction Matrix* (SolidCreativity, 2014) to the field of Information Security, generating a map of optimized shortcuts to the use of the 40 Sec-TRIZ methods.
2. More formalized use of the concept called *Ideal Final Result* (IFR).
3. Propose solutions for automating Sec-TRIZ.

Ergash Karshiev, ergozn1@gmail.com

6. References

Cisco. (2015). *Cisco MRS-2015 Report.* 2015: Cisco Corporation. Retrieved from http://www.cisco.com/web/offers/pdfs/cisco-msr-2015.pdf

Dayah, M. (1997, October 1). *Dynamic Periodic Table*. Retrieved from Dynamic Periodic Table: http://www.ptable.com/

Endler, M. (2014, April 28). *Microsoft's Mobile First, Cloud First Strategy, Explained.* Retrieved from Microsoft's Mobile First, Cloud First Strategy, Explained: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

Frank Buytendijk, Dan Sommer, Thomas W. Oestreich. (2014, September 26). *Maverick* Research: We Analyze Too Much, and Synthesize Too Little.* Retrieved from http://www.gartner.com/document/2856518

Fulbright, D. R. (2012, 07 19). *ITRIZ and Computer Security.* Retrieved from ITRIZ and Computer Security: http://www.uscupstate.edu/uploadedFiles/academics/arts_sciences/Informatics/ITRIZ%20and%20Computer%20Security.pdf

Fulbright, R. (2010, 4 13). *Using I-TRIZ for Computer Security Innovation.* Retrieved from Using I-TRIZ for Computer Security Innovation: https://www.uscupstate.edu/uploadedFiles/academics/arts_sciences/Informatics/ITRIZ%20for%20Computer%20Security%20-%20Fulbright.pdf

Heiser, J. (2013, June 7). *Toolkit: SaaS Security Decision Framework.* Retrieved from Toolkit: SaaS Security Decision Framework: http://www.gartner.com/document/2510115

Jana, R. (2006, May 30). *The World According to TRIZ*. Retrieved from The World According to TRIZ: http://www.bloomberg.com/bw/stories/2006-05-30/the-world-according-to-triz

Mishra, U. (2006, 8). *Inventions on LDAP security - A TRIZ based.* Retrieved from Inventions on LDAP security - A TRIZ based:

Ergash Karshiev, ergozn1@gmail.com

http://works.bepress.com/cgi/viewcontent.cgi?article=1109&context=umakant_mishra

Peters, T. (2004, 9 19). *PEP 20 -- The Zen of Python.* Retrieved from PEP 20 -- The Zen of Python: https://www.python.org/dev/peps/pep-0020/

San, Y. T. (2009). *TRIZ - Systematic Innovation in Manufacturing.* Firstfruits Sdn Bhd.

Schneier, B. (2000, April). *The Process of Security*. Retrieved from The Process of Security: https://www.schneier.com/essays/archives/2000/04/the_process_of_secur.html

SolidCreativity. (2014). *TRIZ contradictions table*. Retrieved from TRIZ contradictions table: http://www.triz40.com/TRIZ_GB.php

Souchkov, V. (2014). *BREAKTHROUGH THINKING WITH TRIZ FOR BUSINESS.* Retrieved from BREAKTHROUGH THINKING WITH TRIZ FOR BUSINESS: http://www.xtriz.com/TRIZforBusinessAndManagement.pdf

TRIZ Journal. (2015). *TRIZ Journal*. Retrieved from TRIZ Journal: http://www.triz-journal.com/

US_CIO_Council. (2013, May 23). *Mobile Computing Decision Framework*. Retrieved from Mobile Computing Decision Framework: https://cio.gov/wp-content/uploads/downloads/2013/05/Mobile-Security-Decision-Framework.pdf

Wikipedia_Contributors. (2015, August 20). *Periodic Table*. Retrieved from Periodic Table: https://en.wikipedia.org/wiki/Periodic_table

7. APPENDIX A: The 40 Sec-TRIZ Methods

The following ten pages illustrate the 40 Sec-TRIZ methods in more details with examples. Threat them as starting points of your problem-solving, not as dogmas or strict instructions. Use the provided visuals as imaginative clues.

7.1. Conventions in the Diagrams

In all of the following diagrams there is a 3-part structure: **NN. Name of the Method:** Method explanation Key Words *(key example)*.

Take the box containing "Segmentation" on the following page:

Name of the Method: **Segmentation**

Method explanation: Partition and Isolate to create

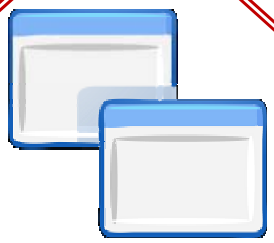
Key word: Zones, Layers, Onions

Key example: *(increase degree of segmentation)*

A capitalized WORD in circles of the Methods is a uniting concept of all 4 methods. On the following page it is the word "**SHAPE**".

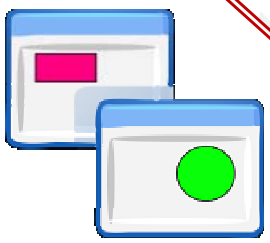
Ergash Karshiev, ergozn1@gmail.com

40 Sec-TRIZ Methods: 1-4



- Network security zones
- Hardware Abstraction Layers
- 7 OSI Network Layers
- Defense in Depth segments

(Model project milestones after OOP)



- Define and refine the differences
- Suppress and compress the "noise"
- Tag data units to track and amplify
- Keep "trouble" locked inside itself

(Radio Astronomy for IDS/AV…)

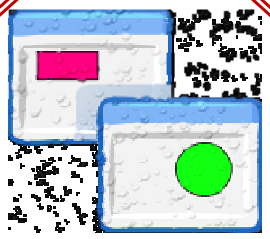
1. Segmentation: Partition and Isolate to create: Zones, Layers, Onions *(increase degree of segmentation)*

2. Extraction: Filter, Distinguish, Squeeze out to Select and Retrieve *(most useful first, catch & amplify)*

3. Local Quality: Increase diversity for optimal specialized use *(natural differences of parts used ideally)*

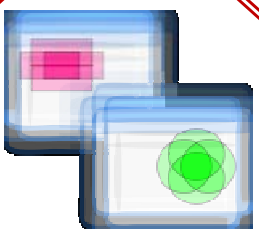
4. Asymmetry: Shift from symmetry to higher degrees of asymmetry *(Custom Symmetries, Shifted Balance)*

SHAPE



- Never repeat yourself (script & loop)
- Don't buy redundant stuff
- Combine different talents and tools
- Seek, increase & best fit uniqueness

(SWAT/Special Forces in IT)

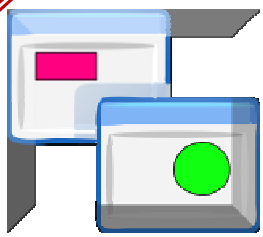


- Building asymmetric defenses
- Unexpected responses to attacks
- Fast resource-shifting as needed
- Non-symmetric: teams & projects

(Staff by needs, not by "common" org-charts)

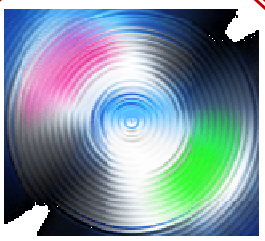
Ergash Karshiev, ergozn1@gmail.com

40 Sec-TRIZ Methods: 5-8



- Utilize all empty cycles
- Batch processes & scripts
- Similar items stay near for group use
- Just-in-Time methodology

(Architectural mantra: similar ones work & stay together)



- Multifunction securely
- Multi-task with less objects
- Standards enable Universality
- Redundant must not be jobless

(Justify the Cost of Existence by being Many-in-One)

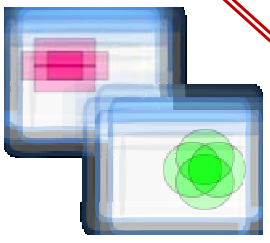
5. Consolidation: Merge similar items in Space and in Time *(do it once at once, handle many with one, similar stay near)*

6. Universality: Do All-in-One for waste of none (***max** # of useful functions to **min** # of objects to handle)*

7. Nesting: Embedded Smart Layers inside and throughout *(multi-shields + multi-tools, all here)*

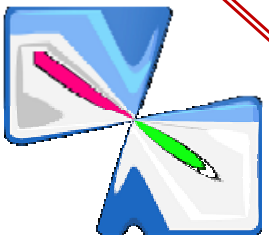
8. Counterweight: Ride/join compensating forces by combining the opposites *(balance by clashing opposites)*

STRUCTURE



- Nested permissions & inheritance
- Micro-devices in all things
- Defense-in-Depth onions
- Layers are stronger than each one

(guarding the inside with many inter-"faces")

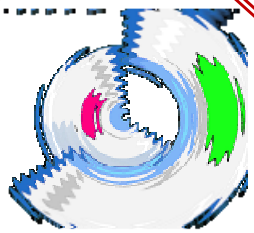


- Minimized energy via opposing
- Move energies vs. burning them
- Actions + Absence of for detections
- Compensate with money or learning

(For every problem there is an anti-problem)

Ergash Karshiev, ergozn1@gmail.com

40 Sec-TRIZ Methods: 9-12



- Scale on the spot
- Offset excess with preparedness
- Incident readiness enables response
- Prepare potential energy

(low potential resources is death, not "savings")

- One-push DR/Incident readiness
- Script (& test) key functions
- Design all code malware-ready
- All "plugs" fit, all is ready

(act as if an incident is one minute away – to be ready every minute)

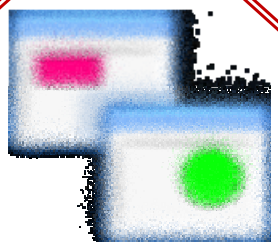
9. Prior Counteraction: Preloaded Energy = canned potential *(Jack-in-the-Box for every "surprise")*

10. Prior Action: Prepared Objects = ready NOW! *(zero-wait action from the best spot)*

11. Cushion in Advance: Pre-cushioned landing = soft pillow *(always carry parachute and padding)*

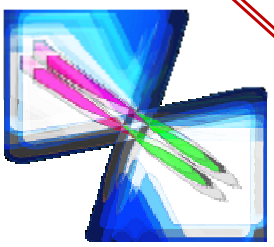
12. Equipotentiality: Level Naturally for self-balancing *(not too hot, not too cold)*

READY



- Users access safe functions only
- Every code has error-tolerance
- All inputs are known and safe
- All fragile is cushioned

(absorb the impact early to soft-recover)

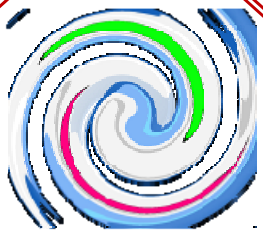


- Optimize system defaults
- Standardize all settings & protocols
- Proper "voltage" of all things
- Modular architecture & code

(all is in expected places and levels)

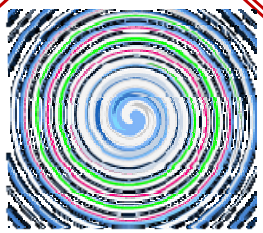
Ergash Karshiev, ergozn1@gmail.com

40 Sec-TRIZ Methods: 13-16



- "Yes, and...": accept and reverse
- Use Emptiness & Negative Space
- Go opposite from the obvious
- Double-reverse (-1 x -1 = +1)

(Nail the problem variable and change the constant instead)



- Infinite/smooth granularity settings
- Harmonize with big waves
- Zero-friction by turning to rotation
- Be more analog, less digital (1/0)

(fluid processes & technologies: smooth & reversible)

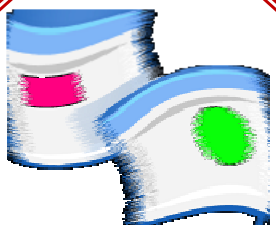
13. Do It in Reverse: Moving in reverse + around the problem *(flip upside-down, switch the static & moving parts)*

14. Spheroidality: Bend flats and squares into curves, spirals and balls *(use smooth rotation)*

15. Dynamicity: Mobility for flexible transformation of roles *(make all parts movable & play with the moves)*

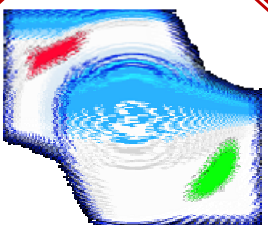
16. Partial/Excessive Action: Good Enough is better than Perfect *(sufficient conditions that do the job)*

MOVEMENT



- Decoupling & dynamic flexibility
- Movable technologies & services
- Business API: imitate software
- Move people, rotate approaches

(Use cloud services that add dynamic elasticity)

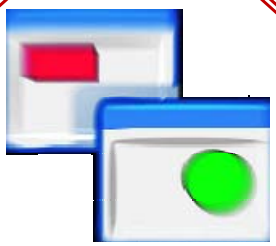


- "Perfect" carries the precision cost
- Overkill/under-shoot wisely
- Reimage infected machines
- Buy new vs. upgrade old

(Avoid costly victories, save via effective non-perfection)

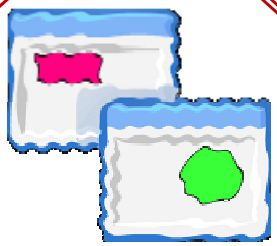
Ergash Karshiev, ergozn1@gmail.com

40 Sec-TRIZ Methods: 17-20



- Use all sides & flip them
- Review problems in 3D
- Remember to use the opposite side
- Use all levels and scales

(Add dimensions: Time, Space, Virtual. Explore the Matrix)



- Eliminate the "mass" & latency
- Respond on attackers' frequency
- Neutrality to ride the waves
- Resonate with business changes

(Maintain capability to run at ANY frequency)

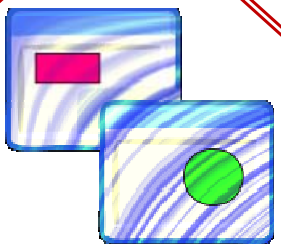
17. New Dimension: The other sides: add factors, facets & levels *(all-dimensional opportunities & degrees of freedom)*

18. Vibrations & Waves: Oscillate, change frequency and resonate *(change the nature of waves)*

19. Periodic Action: Impulse/Pulse to Loop, Pause + Charge, Burst *(Continuous to Periodic: from drill to jack hammer)*

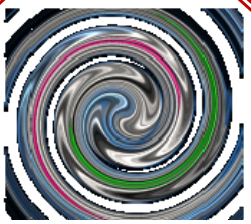
20. Smooth Useful Action: Being useful non-stop & all the way *(fully and continuously utilized capabilities)*

WAVES



- Sync business and IT cycles
- Use Rest cycles to recharge
- Use periodic bursts, if sufficient
- Frequent micro-rest in a work day

(scan periodically vs. monitoring continuously)

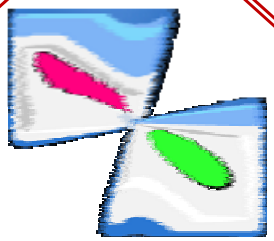


- Make a copy and keep analyzing
- Infinite levels of heuristic analysis
- Use all the licenses (and move)
- Maximize all the equipment use

(Utilize global time zones for continuous work. Optimally use free CPU cycles)

Ergash Karshiev, ergozn1@gmail.com

40 Sec-TRIZ Methods: 21-24



- Super-fast/speed + closing ports
- Invisibly fast –> undetectable
- Minimize "toxic" exposure times
- Dose dangerous, but needed steps

(Systems' risky exposure counters, like Geiger radiation counters)

- Software improves on each failure
- Conflicts to solve contradictions
- Infected machines –> lab resources
- Breaches to funded re-architecture

(Rivalry of Attackers teaches Defenders)

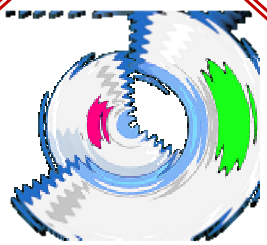
21. Rushing Through: Danger Zone quickly shrinks to Zero *(minimize risky exposures)*

22. Harm to Benefit: Inverted Evil for Good *("Harm" is the unused side of future Good)*

23. Feedback: Back-Feed – amplify yourself to create signal *(send Output back to Input to Generate)*

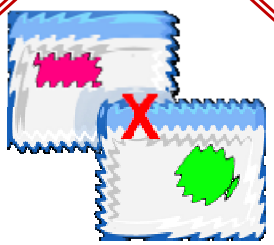
24. Mediator: X-in-the-Middle – the Super-Brokers *(the magic renewable Catalyst)*

GOLDMINE



- Loop-back to amplify detections
- Software execution feedback
- Use feedback to evolve
- Emulate Generators in processes

(Listen to all for feedback: customers, employees and enemies)

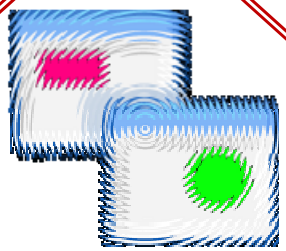


- MITM attack's idea for business
- Protect leadership with proxies
- Middle-Men with Middle Trust
- The Middle that fails safely

(Middleware as universal Security Brokers)

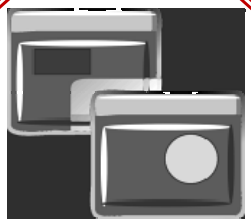
Ergash Karshiev, ergozn1@gmail.com

40 Sec-TRIZ Methods: 25-28



- Make your own "medicine"
- Built-in self-recovery mode
- Undo malware by being self-aware
- User self-help skills & survival

(No energy & materials go wasted: they are extra resources for extra benefits)



- Use "photos", not originals
- Zoom, project & color as needed
- Copy what matters, with focus
- Change the vision spectrum

(Copies are infinite, cheap & ideal: leave originals at home)

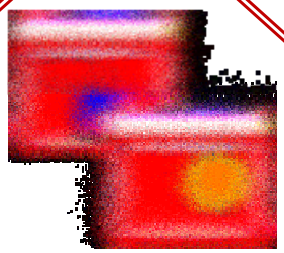
25. Self-Service: Self-healing + turning waste to service *(cure from within with surplus energy & materials)*

26. Copying: Copy-Cat + Projections – easy replicas to alter & resize *(everything leaves a projected trail)*

27. Dispose: Replace-Ability + Free-Star – gold bare bone *(Trade without Trade-Offs)*

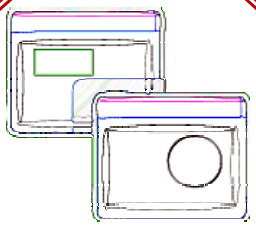
28. Virtual Replacement: Ghosts at Work – ideal natural and virtual emulations *(new twisted "Super-Model")*

GHOSTS



- Kill legacy early, keep gold essentials
- All is temporary, disposable & cheap
- "Irreplaceable" must die, so we live
- Be global-friendly by design

(Disposable security model: cheap, easy & transparent)

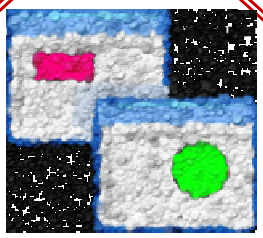


- Manual & Mechanical to software
- Don't touch: use fields & forces
- Replace fields with their opposites
- Play attracting & opposing forces

(The World is just Models: change them as needed)

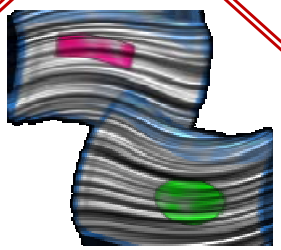
Ergash Karshiev, ergozn1@gmail.com

40 Sec-TRIZ Methods: 29-32



- "Owners" is a fluid concept
- Floating resources fill the vacuum
- Inflate & transfer force
- Cloud is a Gas: scale & cushion

(Cloud & People's elasticity by non-attachment & fluidity)



- Nano-tech security & design
- Quantum effects in macro world
- Super filters & rigid nano-containers
- Thin technology layers

(Use all your atoms & molecules: bridge the Micro- and Macro-Worlds)

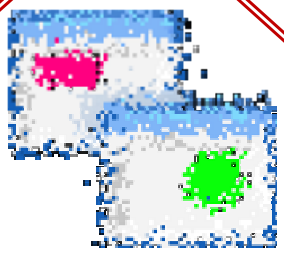
29. Solid to Fluid/Gas: Shape-Less + Flow-More to transmit force & to flow around obstacles *(Zen: "Be like Water")*

30. Flex. Membranes/Thin Films: Un-Stiff to flex, filter & isolate *(nano-films for macro-separation)*

31. Porous Structure: Intake + Absorb to add useful emptiness *(expand while remaining the same, pre-fill the pores)*

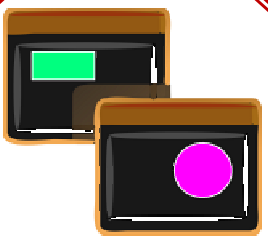
32. Color Change: Colorize + Polarize to stand out *(Change: spectrum, transparency and glow)*

FLOW



- "Porous" code for security & growth
- "NOP Sled" in exploits & analogies
- Bubbles of Negative Space
- Pores carry "fuel" or "medicine"

(Add Breathing Times for employees to boost productivity)

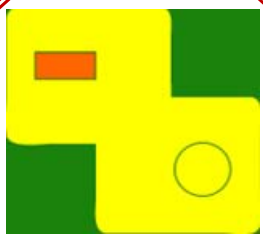


- Tag & Track: via high-res cameras
- Colors of secrecy & authentication
- See-Through Network Layers
- Glowing IDS for all

(The invisible is visible in a different light)

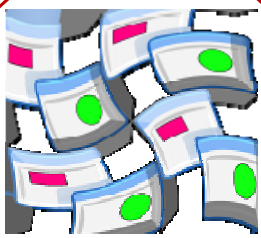
Ergash Karshiev, ergozn1@gmail.com

40 Sec-TRIZ Methods: 33-36



- Dissect code with what created it
- Money grows inside money
- One substance = no cleaning
- Cut diamonds with diamonds

(Cheap & clean defense shield made of suck bullets: the more the better)



- Reuse fragments of old code
- Internal Discard & Rebuild "factory"
- Training security-immune system
- Destroy shape – reuse substance

(Reuse people – redefine their jobs)

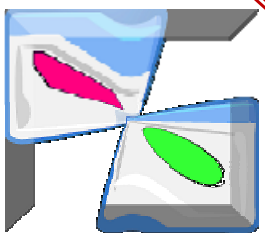
33. Homogeneity: Blend-in + Be-One – tools = resources *(use itself for pollution-free interactions)*

34. Reject-Regenerate: Ash back to Cash – remove to re-create *(continuously discarded and cast back into shape)*

35. Transform Properties: Trans-Form – move to another state of being *(Chameleon of shapes & configurations)*

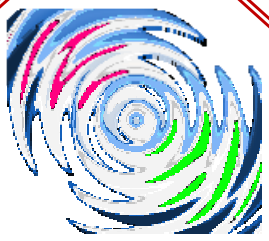
36. Phase Transition: Phase-Zen – "Freeze-Melt-Evaporate" for desired effects *(ride natural states)*

CHANGE



- Super-configurable software
- Fluid shaping of technology
- Change: densities, degrees & states
- Modify: flexibilities and abstractions

(Change state of "Being" by different logical "Viewing")

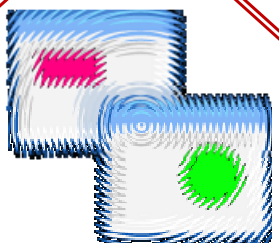


- Solid, Liquid & Gas – security modes
- Harden & Soften instantly per needs
- Evaporation = no garbage
- Freeze-Dry: use blended effects

(Be 3-states-in-1, just change the "temperature")

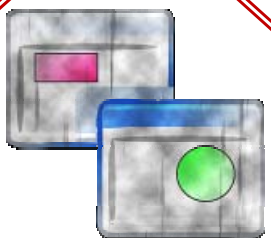
Ergash Karshiev, ergozn1@gmail.com

40 Sec-TRIZ Methods: 37-40



- Optimally match the ecosystem
- Blend expansions for new results
- Follow the natural "breathing"
- Security changes by the "weather"

(Capacity scales to meet demands)



- Secure-erase by natural oxidation
- True oxidation clock (for evidence)
- Physical snapshot of system states
- Unlimited bio-storage and tracker

(Mold on the motherboard tells the truth)

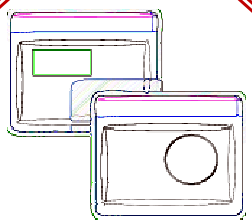
37. Expansion: Riding Internal Forces – the environment dictates capacity & features *(harmonize with the World)*

38. Fast Chemistry: Art of Burning – Physics & Chemistry to challenge digital hacking *(watching from the outside)*

39. Inert Environment: Not-Reacting – skill of not responding *(neutrality for balanced service)*

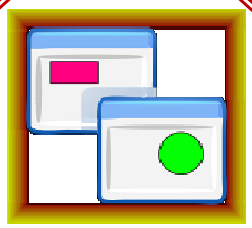
40. Composite Structure: Optimal-Mix – many flavors: for new features *(sum is different than its parts)*

REACTION



- Neutral substances: fill & not-feel
- Air-gap insecure devices
- See all in neutral light
- Born in a vacuum: malware-clean

(Inert filler + needed additives = clean interaction)



- Small composite teams
- Integration of humans & machines
- Universal glue to connect all
- Physically feel the IDS alerts

(Mix the un-mixable with technologies and techniques)

Ergash Karshiev, ergozn1@gmail.com