Intelligent Biological Security Testing Agents

Ishbel MacDonald Duncan
School of Computer Science,
University of St Andrews
Scotland
Ishbel.Duncan@st-andrews.ac.uk

Keywords- security testing models metrics test adequacy

I. THE PROBLEM

The fields of security testing and security metrics are in their infancy. Research has stated to the necessity of testing allegedly secure systems and to measuring system status in order to determine how secure that system is.

Security testing involves taking into consideration temporal aspects; a system tested at time T_1 is unlikely to be as secure, as robust, as reliable, as the same system tested at time T_0. There is a presumption that new vulnerabilities have been encountered and patched for, or that users, processes, subjects or resources have been added, updated, patched or even deleted from the system. Further, to get an image of the system state is an extremely complex and resource intensive task and may not be indicative of any future problem.

Given that a security oriented process must "Prevent, Detect and React", any developed system should be autonomous in its actions of searching, probing and reporting or fixing. Therefore a goal would be to develop an automated security testing system, or suite, that is applied to pre release and working systems to monitor the states of key processes via interaction with a test or audit monitor.

Given the distributed, or Cloud, based systems currently in use it would be advisable to design a system that could be extended to be free-ranging and roaming and not necessarily contained within a small network of related servers. However, to create any roaming, or even a static protection system, one needs to be aware of security metrics and levels of adequacy. The two main questions are "When is a tested system, tested enough?" and "When is a system protected enough?" Possibly both questions are unanswerable but the problem remains, that we must protect and we must monitor, even if we are not sure at which metric level of a particular system facet we can state that we have achieved something concrete.

II. PROTECTION SYSTEMS

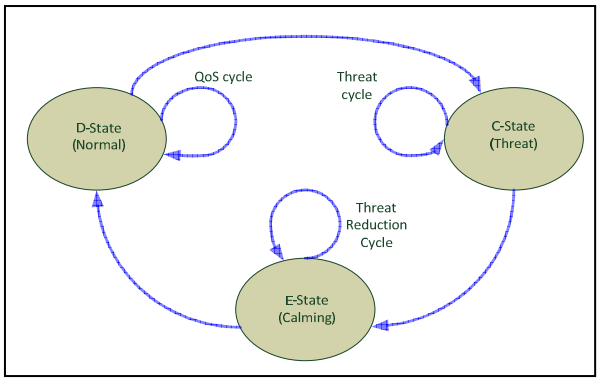
History tells us that protection comes in many layers; from strong walls, vallums, portcullis and mazes to the modern computer honeytraps, intrusion detection systems, firewalls, multi-level authentication, intruder jailing and lock-outs.

A secure system should be multiply protected with active, gauged responses depending on the state of the system. Ergo, one must monitor to gauge the state and the divergence from the operational norm, and the response must be engaged in the background with daemons or in the foreground with die-time routines taking over the system if absolutely necessary.

Psychology tells us that the brain has different mechanisms for producing the fight-or-flight response via an adrenalin rush or in soothing us with endorphins, and allowing us to perform tasks. In humans, our experiences of emotions emerge from the patterns created in our brains and bodies. The affect regulation system reacts to experiences to give rise to appropriate feelings and responses. Our self-protection system is affected by the levels of cortisol which plays a role in our sensitivity to threats. Our incentive system , our drives, are affected by dopamine and lastly, our soothing system is controlled, or affected by, endorphins. These three components affect our brain and our bodies and if they get out of control, each can cause differing problems for us; too much cortisol and we become very sensitive to threats, too many endorphins and we are too calm to respond to threats efficiently and effectively.

Here, it is theorized that just as our complex brains require at least these three regulators to work together, so too does a complex computer system. We need to monitor threats via IDS, authentication routines, trusted computing bases etc., the equivalent of our human threat system. We need to make a system efficient and well connected, operating as the human incentive system, but only if it is not under threat. The re-balancing between a threat system and a quality system, in terms of the software engineering quality goals of availability, reliability, robustness etc. requires dedicated security routines and resources, the equivalent of the human endorphin calming chemicals.

Therefore to maintain a balanced and balancing system we require at least two related and off-setting regulators, the c-regulator (cortisol) and the e-regulator (endorphin). The regulators work as a system, co-operating to increase the threat state or return to the calm state. The third state, the d-state (dopamine), can be considered as the initial or operating system state where a system could improve its quality, or operational envelope, in short burst cycles under the guidance of directive metrics. A threat pushes the system into the c-state only returning via the e-state once a threat has been mitigated. The e-state performs the transitions required to return the system to the quality driven cycle (see the following figure).



III. ANALYSIS

In an analysis of a basic, small, well contained system, within a protecting firewall, we can perceive the system to start at the d-state. Updating the system or improving its quality of service, should keep the system in the d-state. However, once the system becomes under threat, the system enters the c-state where the defence mechanisms are tuned up and the system perhaps reduces its user process throughput or degrades services. Once the threat is deemed to be have passed, the system updates its threat profiling and moves into the e-state where the user quality is once again improved and the protecting processes move once again into the background. This is obviously a simplistic view of a system. The problem domain is a much more complex test site including multiple layers of operating system, networking, data files, processes, user profiles and even system architecture.

However, if there is a layered approach to protection, one can presume layered descriptions of systems. There can be many levels of complexity of systems but, for argument, it is presumed at least three different systems structures:

- Base level – a low coupled system, with cooperating processes in a contained system with a single connection to the world, perhaps a small user system.
- Intermediate level – a mixed system with distributed processes over a range of servers, multiple connections to the world, perhaps intra-company with multiple users.
- High level – a highly complex system within a distributed environment or Cloud, with multiple VMs, multiple users, multiple connections and multiple companies and users.

A model may work well for one level of complexity but fail at higher levels; a simplistic biological model may work well for a base level system because there are few co-operating or competing processes. Taking the intermediate level system, a biological model may well be applicable because the co-operating processes that instantiate the three states are essentially governed by one superuser or Security Officer. The system may be slower and more prone to user process deterioration because of the number of threats that occur, but essentially the model remains unchanged. The major adaptation would have to come in consideration of the high level system in which multiple users over many sites essentially breaks the single management structure of the biological testing model. However, in returning to the human body, we know that although parts of the brain and nervous system control our reaction to events, the biological effects are distributed around the body. So it must be with a computer system. It is not enough to mitigate risk on one part of a distributed system, the effects must be duplicated and transmitted across networks, architectures and processes.

IV. MEASUREMENT

A secondary aspect here, as mentioned before, is one of measurement. Security metrics are very topical and many involve simple counts of attacks per time span or threat mitigated per number of attacks. This may be applicable within base and intermediate level systems, but in a very highly complex distributed system one question to be asked is “How much do we need to know these counts?” or “What do they tell us anyway?” Measuring is useful intra-network and intra-company but there is no merit in assigning resources when the end result tells us little. Let us not fall down the trap of many metrics programs, but let us examine what we really need to know to guide a security test or a security model. We need to know what is successful and what is not, and we need to know the where and the how. We can reasonably assume that the answer to when is “always” and the who has to be an automated system at its most optimal.

System security knowledge is undoubtedly made up of, at least, vulnerabilities, associated costs, failures, quality (metrics) and asset fragility. The latter could be subdivided into protection measures, threat metrics, costs of protection and again, the domain of protection measures could be subdivided into the usual temporal, physical, human and cryptographic. There are many ways of subdividing a system but the result is to define measures, costs and levels at which cut-offs may be applied. This information, however derived, should be used within the protection system and the biological model allows a way of using information to feed into cycles of protection and easing back into “normal” modes. By applying our adequacy measures, gained from trials, to the biological state guards, we can see that the system could, theoretically be self-adjusting.

However, as mentioned above, metrics may be applicable within smaller scale systems but may well fail within large, distributed systems. Any security monitoring system will need to collate information across many domains and networks.

V. SYSTEM INTELLIGENCE

The final part of this biological security testing model is to take each of the above problems and to add intelligence to ensure applicability to large scale systems. By using ideas taken from artificially intelligent agents, we can consider that testing becomes a function of roaming AI agents, destined to wander, perhaps constrained, networks, with each network policed by its own testing agents. Testing agents could be trained to look for specific vulnerabilities, or system facets, and to report back to their test controller at specified intervals. It is the controller agent that reports the measurements within domains gained from system interactions and thus there is a separation of duties and separation of concerns.

Agents would have to be verified onto networks, or into interactions with users or processes, and therefore specialist guard agents would be required to validate and verify.