who’s watching the watch dogs?

kowsik@mudynamics.com

http://labs.mudynamics.com

agenda

- ✓ rant on the state of affairs
- ✓ winds of change
- ✓ test driven development
- ✓ new perspectives
- ✓ summary

the early days

- “close this port”
- morphed into
  - omg! ftp doesn’t work
- along came proxies and ips
  - protocol dissectors to detect protocol bugs
- and we now have…

layered [in] security

- ✓ anti-spam
- ✓ anti-spyware
- ✓ anti-phishing
- ✓ anti-virus
- ✓ network/application firewalls
- ✓ stateful/deep inspection and ips
- ✓ ssl/ipsec vpn
- ✓ data leak detection
- ✓ network access control
- ✓ …

security software, not secure software

- software wrapped in aluminum
- as vulnerable as the targets they protect
- software flaws at multiple levels
  - configuration
  - protocols
  - file formats
- don’t forget centralized management
  - typically the weakest link

winds of change

- "*routers no longer route*"
- networks are ever more application aware
- applications are acting like infrastructure
  - machine to machine
  - broken up into services and components
- *perimeter* is blurring fast
- happy hour at the confluence

time to unask the question?

mainframes

- monolithic
- all parts came from the same vendor
- minimal attack surface
- minimal dependencies to other systems
- typically tested for
  - reliability
  - availability
  - serviceability

services

- ✓ huge attack surface and interdependencies
- ✓ speed mismatch between rollouts and testing
- ✓ problems are *punted* to incident management

test driven development

a brief detour

unit testing

- key aspect of TDD
- 5 steps to TDD
  - add a test
  - run all tests and see the new one fail
  - write some code
  - run the *automated* tests and see them succeed
  - refactor code

interfaces, objects and methods

- method invocation
  - arguments and return values
- assertions
  - positive and negative
  - cause and effect
- automated tests accelerates innovation
  - you know exactly what changed and what broke

negative testing

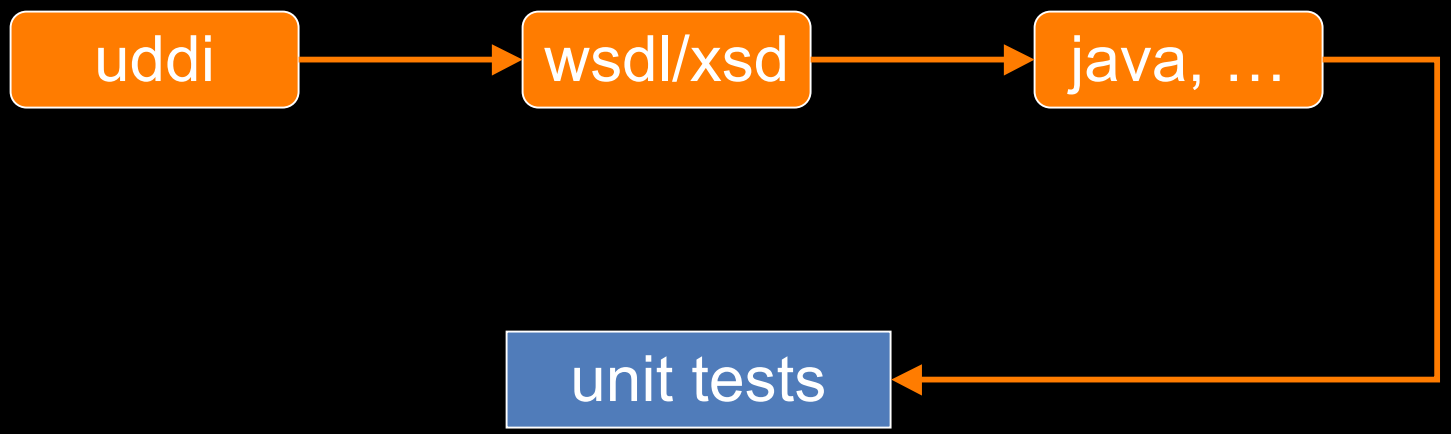
- has its roots with the origins of the Internet
  - "*where wizards stay up late*"
- is about boundary conditions
  - ability to handle exceptions
  - unanticipated input
  - fuzzing is one type of negative testing
- security testing is inherently negative
  - "*hacking is outsourced QA*"
- automation is a must-have
  - test case generation
  - test case execution

interface-based applications

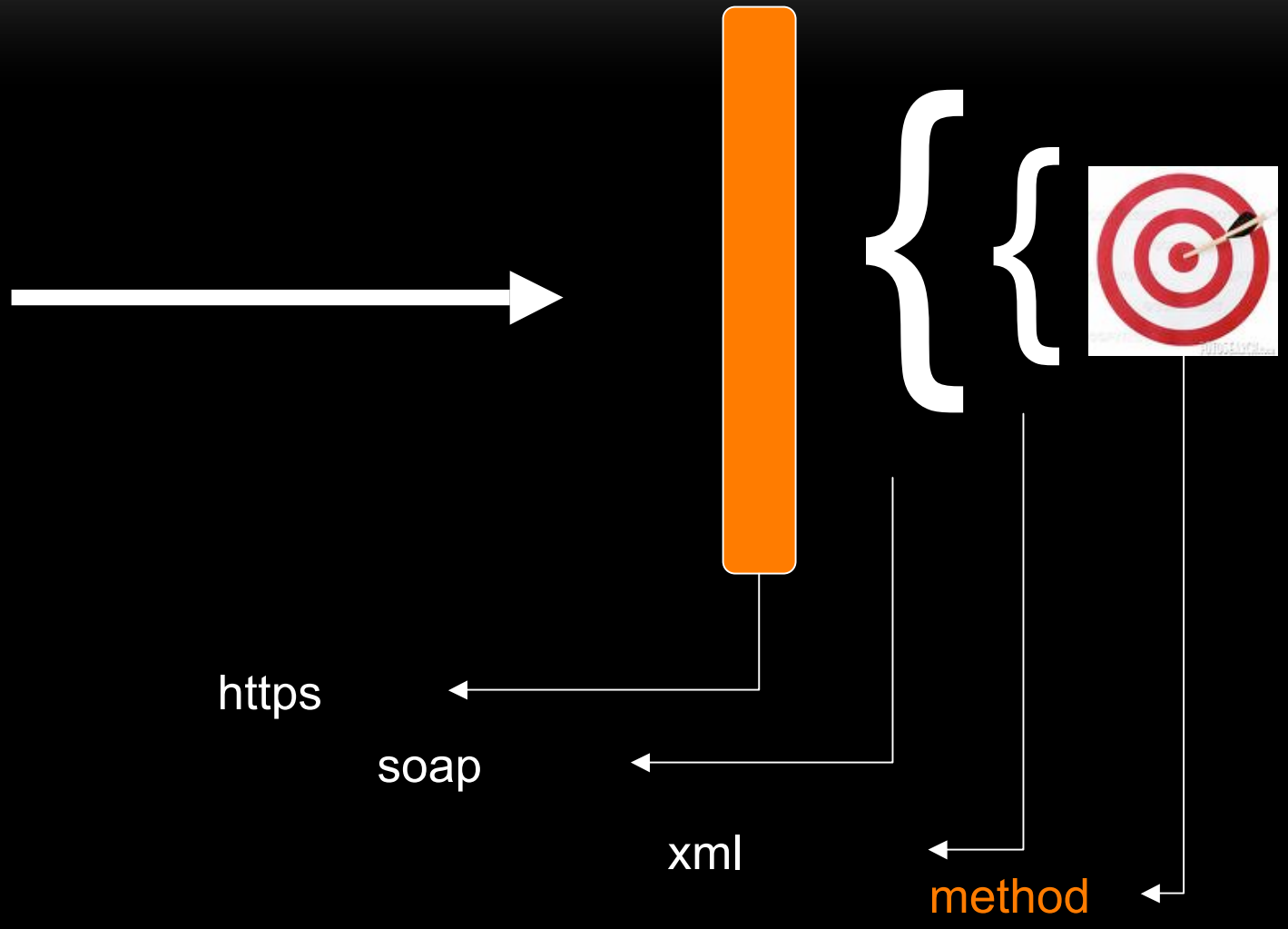
service oriented applications

- in essence XML-RPC
  - REST
  - SOAP
- machine to machine
- well-defined interfaces
- code generateable
  - but remoted
- application as an API
- can we unit test them?

unit testing soa



what are we testing?



attack surface

- is *not* just the method
- exposure is from the
  - method
  - encoding
  - message
  - protocol
  - channel
- and all the pieces of infrastructure in front of it!

are we doomed?

- cannot test applications in isolation
- cannot change infrastructure without affecting applications
- and it's not about
  - known vulnerabilities
  - incident management
  - log correlation
  - and patching
- can we unit test a service?
  - for their capabilities and dependencies
  - to anticipate and detect failures

testing 2.0

new perspectives

next generation services

- VoIP, IMS, IPTV
  - applications or infrastructure?
- characteristics
  - complex
  - highly interconnected
  - real-time
  - high rate of change
- before we talk about security…

some insights…

- critical services on standard OS’
- minimal to no hardware acceleration
  - higher order application protocols
- just valid traffic alone leads to crashes
  - interoperability or security?
- highly susceptible to dos
- functional and load testing no longer sufficient

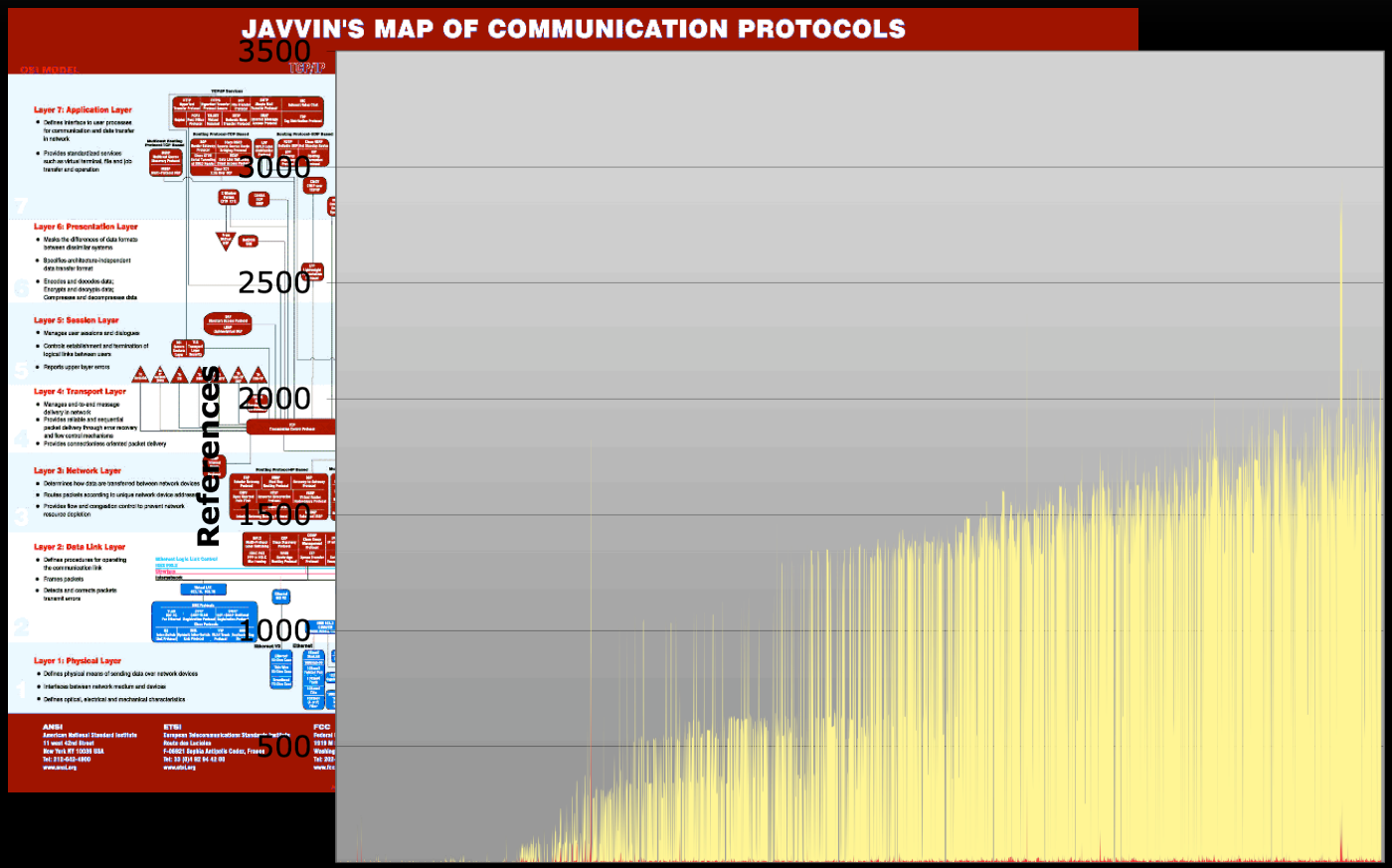
r.a.s

- spin on what mainframes were tested for
  - reliability
  - availability
  - security
- but takes into account the interconnectedness
  - protocols are key
- can we test them in a unified way?

protocols

- ✓ are nothing like each other
- ✓ seem adhoc with structures and encodings
- ✓ arbitrarily complex
- ✓ no canonical form to operate on
- ✓ not necessarily machine parsable
- ✓ or are they?

kevin bacon and six degrees



rfc's

six degrees of protocols

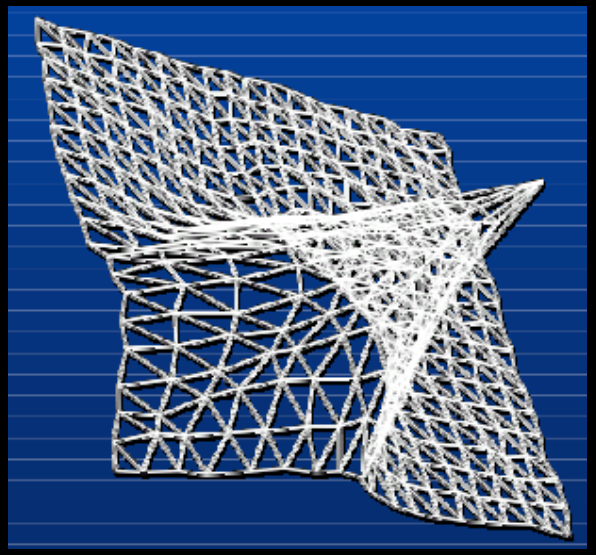
- SIP uses LDAP DN's
  - which use ASN
    - which are in X.509 certificates
      - which is used in TLS/SSL
        - which contains Name/Value pairs
        - that's used in iCal format
- DHCP has NetBIOS names
  - which is used in CIFS
    - which uses Kerberos
      - which uses ASN
        - which …

abstracting protocols

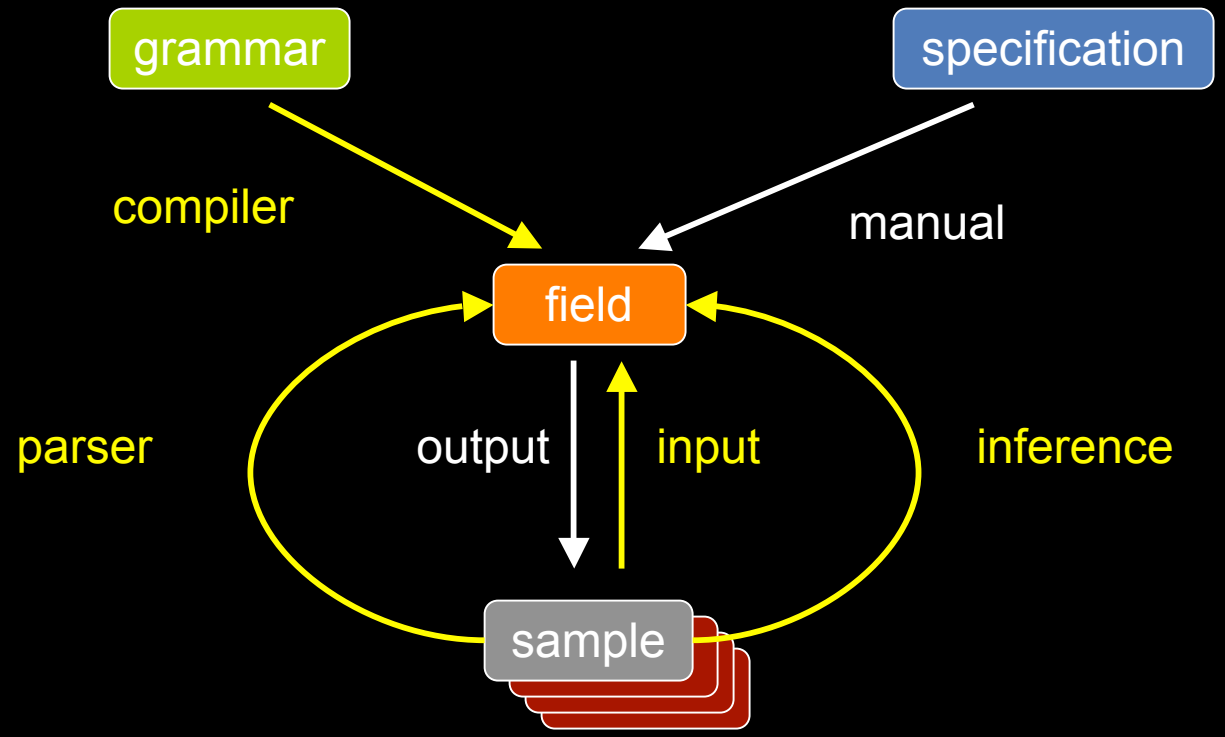
- state, structure, semantics and constraints
  - a semantic DOM
  - with associated vulnerability patterns
- io/delivery mechanism (channels)
  - sockets (raw, v4, v6, tcp, udp, ssl, sctp, …)
  - interactive channels (telnet, ssh, console, …)
  - bluetooth, wireless, usb, firewire
  - ioctl's
  - files

fuzzing

- is really about semantic data structures
  - free form deformation
  - dependency propagation
  - constraint violation



unification



http://labs.mudynamics.com/2008/03/28/cansecwest-slides/

dos

- channel abuse
  - not just layer 2/3
  - stateless for best effect
  - 20,000 packets/sec more than sufficient
- so many tools, so much redundancy
  - is there a pattern here?
  - can we characterize systems subject to dos?

characteristics

- unsolicited packets
  - mgcp notification
  - isakmp notifcation
  - rtp flood
- *lack of* rate limiting for responses
  - icmp ping's
- *incomplete* session setup
  - sip invite/register
  - syn floods
  - sctp init
  - dhcp discover

uniqueness

- not enough to spoof src-ip/src-mac
- application dos
  - has unique regions inside payloads
  - has references to l3/l4 header
- packet has to be sufficiently valid
  - force target to allocate resources

breaking up dos

- underlying transport
  - ethernet, ipv4, ipv6, udp, tcp
- payload with update regions
  - references and random
- traffic pattern
- service monitors
  - stateful transactions

dos'ing SIP

INVITE sip:bob@example.com SIP/2.0

Via: SIP/2.0/UDP client.example.com:5060;branch=z9hG4bKa1b2c3d4;rport

To: "Bob" <sip:bob@example.com>

From: "Alice" <sip:alice@example.com>;tag=x1y2z3

Call-ID: abcd1234@192.168.1.1

CSeq: 1 INVITE

Contact: <sip:alice@client.example.com>

Max-Forwards: 70

Content-Type: application/sdp

Content-Length: 0

update regions

INVITE sip:bob@example.com SIP/2.0

Via: SIP/2.0/UDP client.example.com:5060;branch=z9hG4bKa1b2c3d4;rport

To: "Bob" <sip:bob@example.com>

From: "Alice" <sip:alice@example.com>;tag=x1y2z3

Call-ID: abcd1234@192.168.1.1

CSeq: 1 INVITE

Contact: <sip:alice@client.example.com>

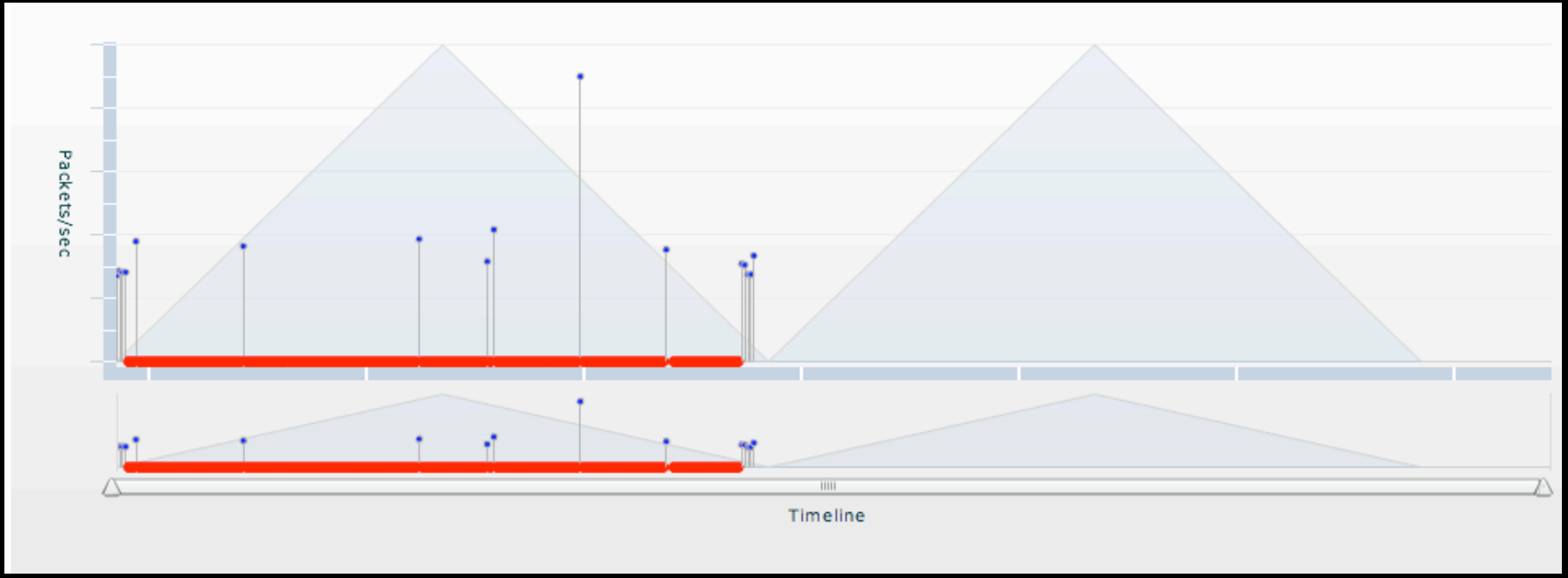
Max-Forwards: 70

Content-Type: application/sdp

Content-Length: 0

results

- ✓ INVITE dos with OPTIONS monitor
- ✓ multiple src-ip's with payload randomization
- ✓ 5000 packets/sec



summary

- watch dogs are just software
  - as susceptible as the targets
- functional and load testing no longer sufficient
- testing 2.0 is proactive
  - a concrete automated way to measure r.a.s.
  - a prerequisite for NG services

questions?

kowsik@mudynamics.com

http://labs.mudynamics.com