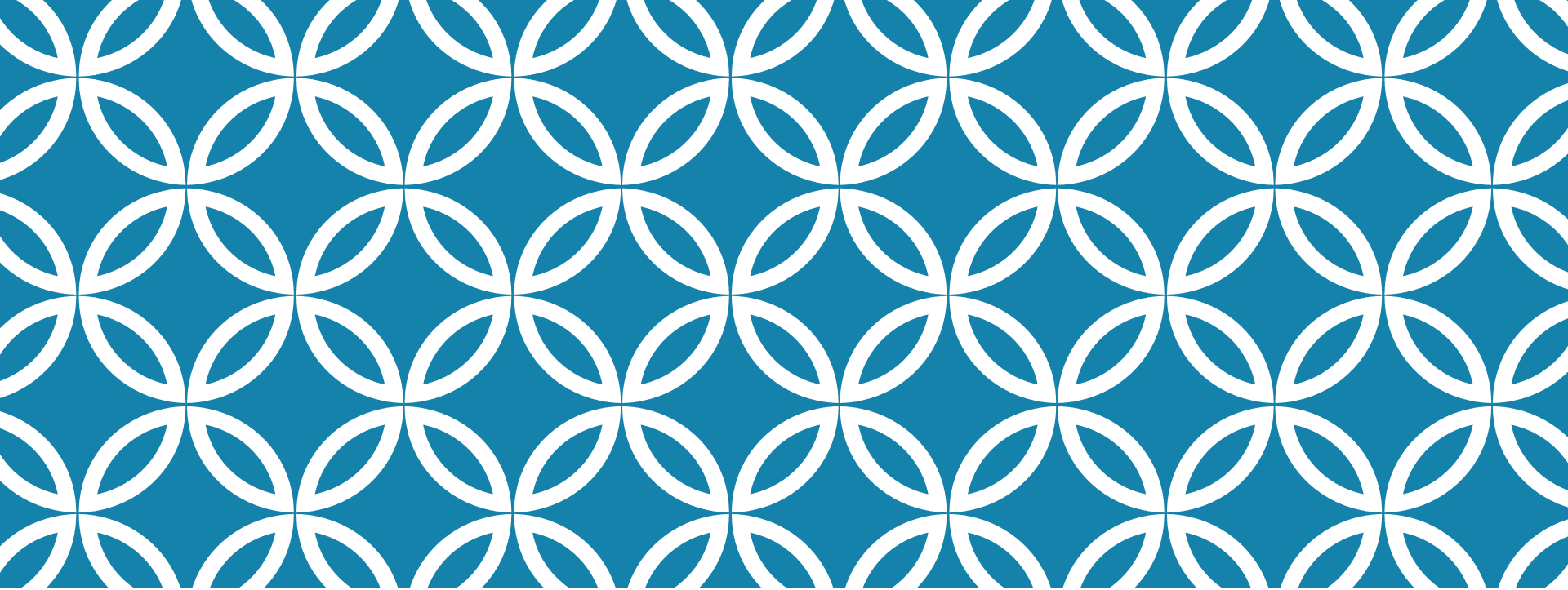


WIMSE AND AGENTIC AI

Yaron Sheffer and Dmitry Izumskiy, IETF-123

Why Agentic AI

It is being adopted by real companies for real applications

- It enables text-related applications with better accuracy than previous-gen LLM apps
- Natural language “programming” is easier – or perceived as easier

Agentic AI is not well defined, but the main properties are:

- LLM-based application
- Iterative: multiple “runs” through the LLM until a goal is reached
- Autonomously selects its own execution flow
- Uses tools, i.e. outside interfaces: to other agents, to traditional APIs, to internal modules
- The agent manages persistence or long-term memory
- Composite: agents often consist of multiple smaller interacting agents, let’s call them “micro-agents”

And Why Not…

LLM-based applications are subject to vulnerabilities that are hard or impossible to mitigate

- Mostly variants of prompt injection and indirect prompt injection
- The applications are also non-deterministic, thus hard to test and secure

Agentic AI amplifies the risk

- More complexity
- The control flow is dynamic and can be manipulated
- Agents are often defined with a very large scope (through tools) and therefore broad permissions

Protocols in This Space

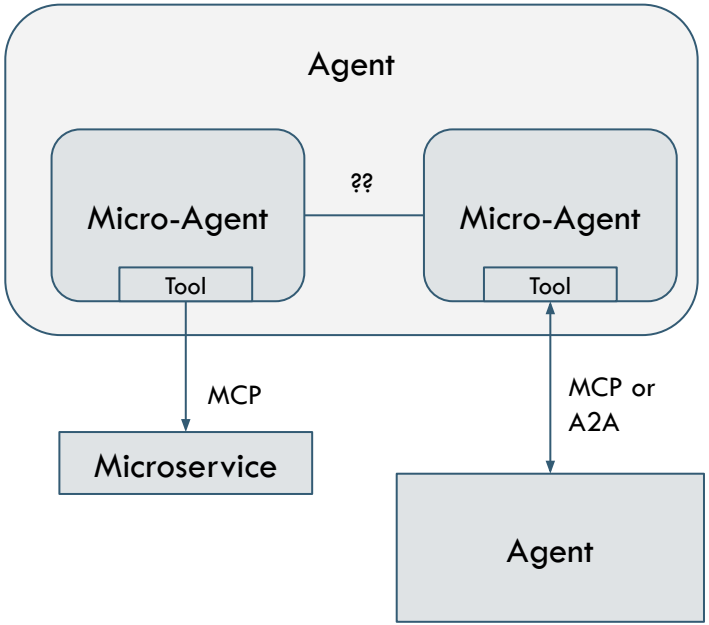
MCP: agent-to-tool protocol, mostly consists of a free text description and a JSON-RPC schema of the tool. Very widely adopted.

- MCP is often used across organizations, OAuth is appropriate and is used in practice
- The MCP "server" is often a container deployed on the *client* side, resulting in major security concerns

A2A: agent-to-agent protocol, developed by Google. More powerful and less adopted.

- I don't know enough about A2A to discuss its security properties

Modeling The Agent



Things are changing fast - Modeling and terminology are to be taken with a grain of salt

Are Agents “Workloads”?

Working definition: **Agents** are AI systems that can autonomously plan and execute complex tasks

The WIMSE architecture has a very loose definition of “workload”

In particular, one workload can consist of several “internal” workloads

A typical micro-agent activation involves some computation, one LLM call and possibly one or more tool calls

Very much like a microservice

→Both the larger agent and the micro-agents can be seen as WIMSE workloads

So Can We Declare Success?

No: transport, identity, attestation and authorization

WIMSE has focused on transport security (the workload-to-workload protocol), punted on identity, attestation and authorization

We may need to reconsider each one

Transport

WIMSE has so far focused on REST APIs

MCP may not be directly applicable to communication between micro-agents, but we should learn from it

- Some combination of structured input/output (a.k.a. JSON) and streaming, potentially streaming of free text
- Yep, this goes beyond REST

Workload Identity

Agent identity/context has the full complexity of user identity

- Associated with a human
- Organization
- Human-within-organization
- Human acting on behalf of any of the above
- Etc.

We should beware of popular literature painting AI agents as ”human-like”, implying an identity on par with humans for the agent itself

No HR department for agents!

Attestation

The rapid emergence of agents is leading to many companies looking at agent marketplaces

The agent's code may be developed by org X, verified by marketplace Y and run by org Z

All of which may be relevant for authz decisions

Do we have a way to denote these properties? Should we recommend one?

Authorization

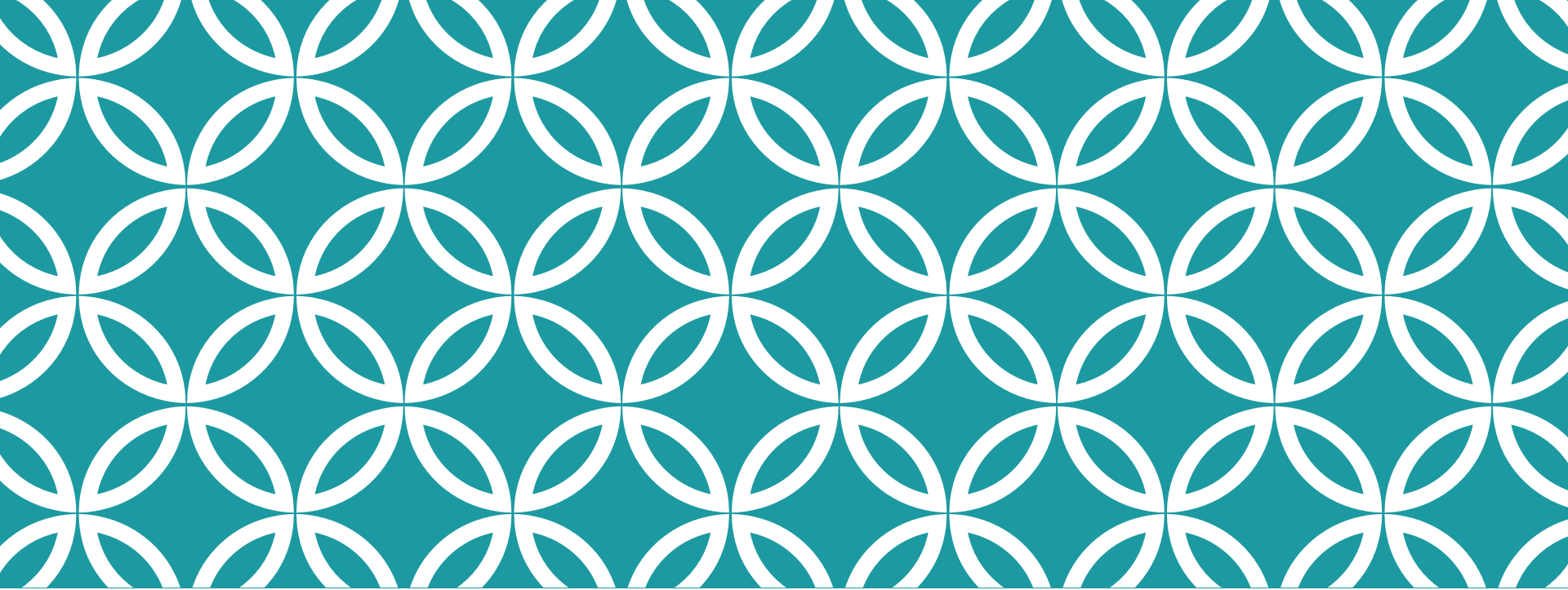
In the old world, each microservice has a relatively small blast radius

We could get by with just “service A can call service B” authorization

With Agentic AI, teams are pushed to develop very high-level agents (“an investment banker”) with very broad permissions

- This can only be mitigated by per micro-agent permissions
- As well as dynamic authz management

WIMSE should at the very least support authz meta-syntax (RAR) and possibly support/recommend specific authz languages



Thank You!

yaronf.ietf@gmail.com